NXP Semiconductors
Reference Manual

Document identifier: MCXW71RM
Rev. 3, September 2024

MCX W71 Reference Manual

Supports MCXW71xx

Contents

Chapter 1
About this manual

1.1 Audience

This reference manual is intended for system software and hardware developers and applications programmers who want to develop products with this device. It assumes that the reader understands operating systems, microprocessor system design, and basic principles of software and hardware.

1.2 Organization

This manual begins with a global introduction of the chip, followed by chapters organized into *functional groups* that detail particular areas of functionality, such as system control, clocking, and timers. Each functional group can have two main types of chapters:

- *System-level* chapters contain information that applies to the components (modules) within the group.
- *Module-level* chapters contain technical descriptions of individual modules within the group.

Note that application-specific groups (such as timers) may only contain module-level chapters.

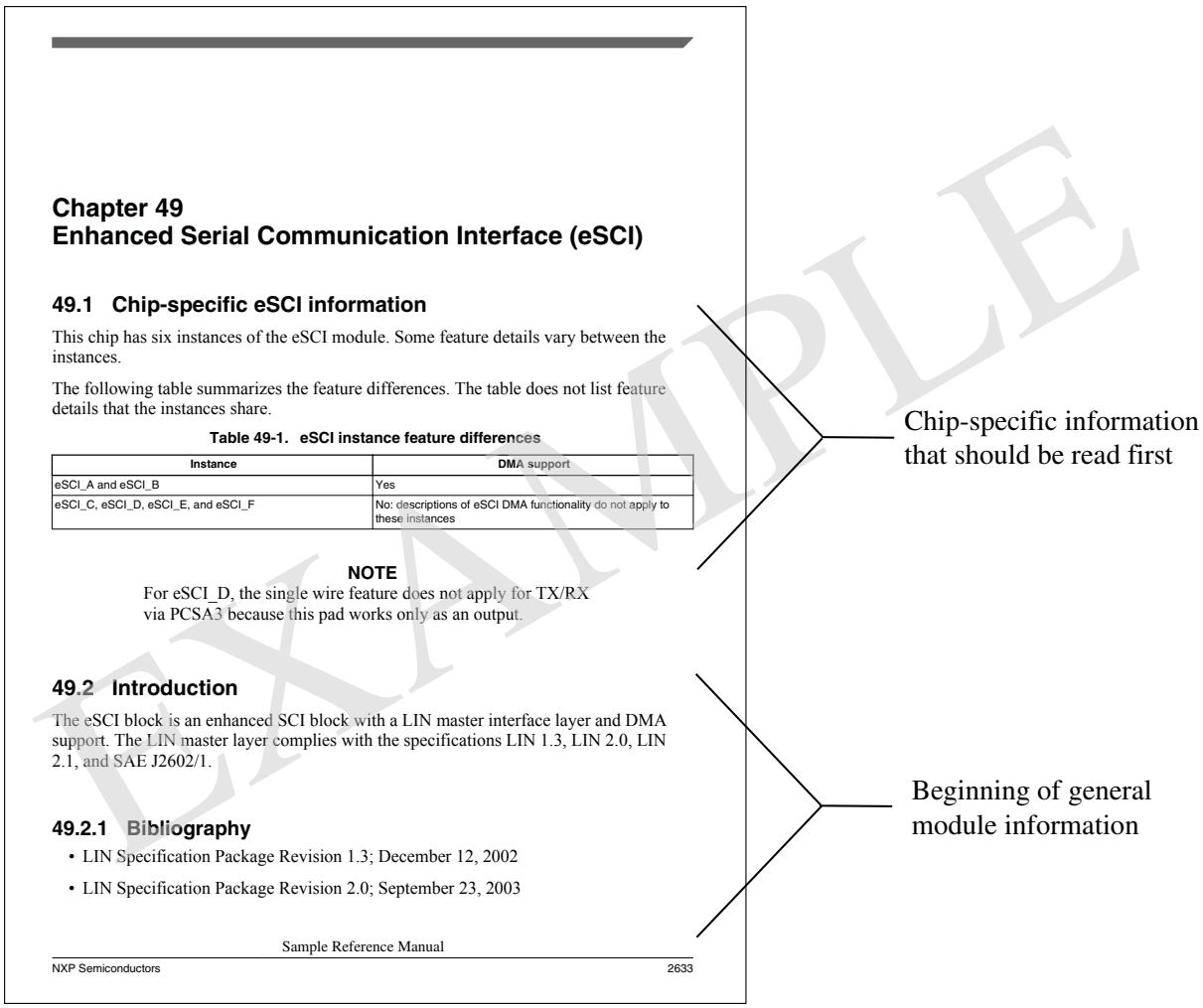
1.3 Module descriptions

Each module chapter has two main parts:

- **Chip-specific:** The first section, *Chip-specific [module name] information*, includes the number of module instances on the chip and possible implementation differences between the module instances, such as differences in FIFO depths or the number of channels supported. It may also include functional connections between the module instances and other modules. Read this section *first* because its content is crucial to understanding the information in other sections of the chapter.
- **General:** The subsequent sections provide general information about the module, including its signals, registers, and functional description.

NOTE

If there is a conflict between the chip-specific module information (first section) and the general module information (subsequent sections), the chip-specific information supersedes the general information.



Chapter 49
Enhanced Serial Communication Interface (eSCI)

49.1 Chip-specific eSCI information

This chip has six instances of the eSCI module. Some feature details vary between the instances.

The following table summarizes the feature differences. The table does not list feature details that the instances share.

Table 49-1. eSCI instance feature differences

Instance	DMA support
eSCI_A and eSCI_B	Yes
eSCI_C, eSCI_D, eSCI_E, and eSCI_F	No: descriptions of eSCI DMA functionality do not apply to these instances

NOTE
For eSCI_D, the single wire feature does not apply for TX/RX via PCSA3 because this pad works only as an output.

49.2 Introduction

The eSCI block is an enhanced SCI block with a LIN master interface layer and DMA support. The LIN master layer complies with the specifications LIN 1.3, LIN 2.0, LIN 2.1, and SAE J2602/1.

49.2.1 Bibliography

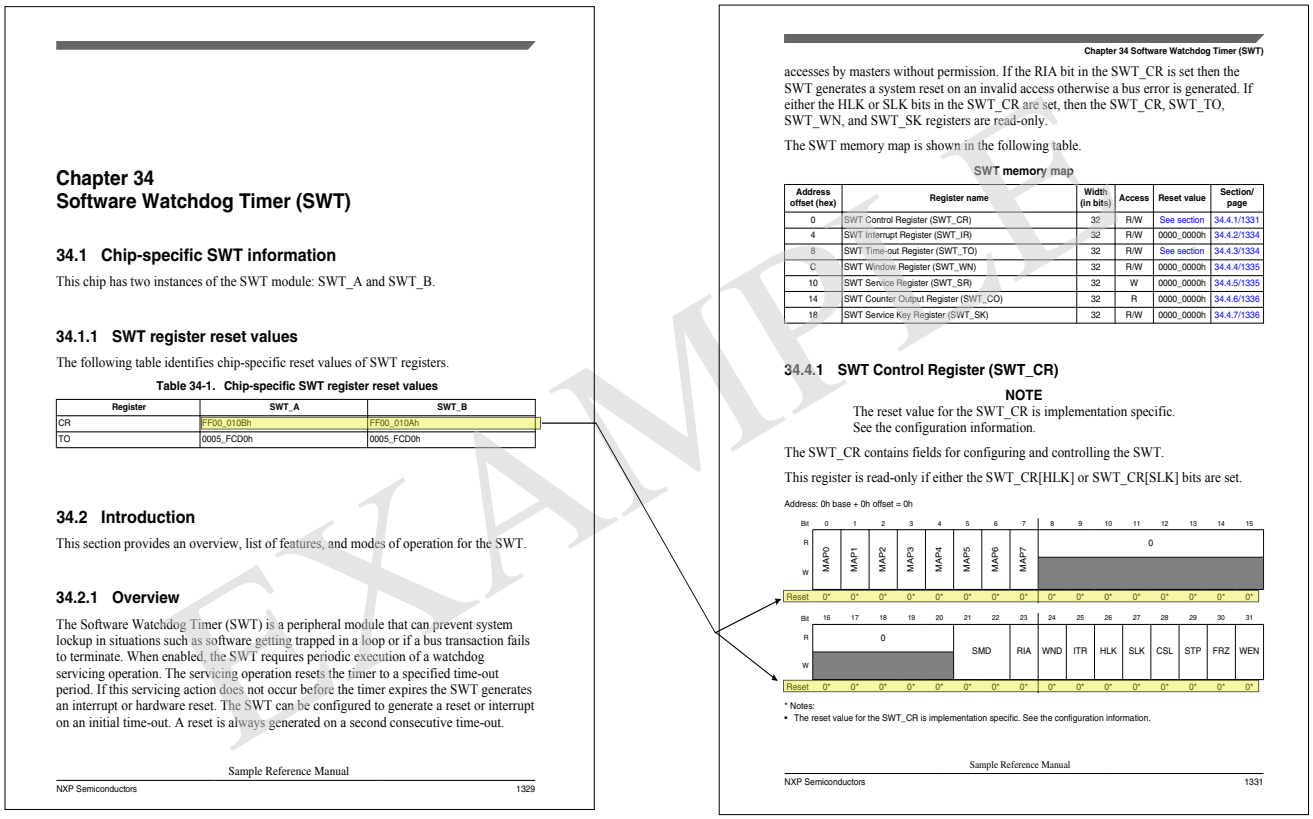
- LIN Specification Package Revision 1.3; December 12, 2002
- LIN Specification Package Revision 2.0; September 23, 2003

Sample Reference Manual
NXP Semiconductors 2633

Figure 1. Example: chapter chip-specific information and general module information

1.3.1 Example: chip-specific information that supersedes content in the same chapter

The example below shows chip-specific information that supersedes general module information presented later in the chapter. In this case, the chip-specific register reset values supersede the reset values that appear in the register diagram.



Chapter 34
Software Watchdog Timer (SWT)

34.1 Chip-specific SWT information

This chip has two instances of the SWT module: SWT_A and SWT_B.

34.1.1 SWT register reset values

The following table identifies chip-specific reset values of SWT registers.

Table 34-1. Chip-specific SWT register reset values

Register	SWT_A	SWT_B
CR	FF00_010Bh	FF00_010Ah
TO	0005_FCD0h	0005_FCD0h

34.2 Introduction

This section provides an overview, list of features, and modes of operation for the SWT.

34.2.1 Overview

The Software Watchdog Timer (SWT) is a peripheral module that can prevent system lockup in situations such as software getting trapped in a loop or if a bus transaction fails to terminate. When enabled, the SWT requires periodic execution of a watchdog servicing operation. The servicing operation resets the timer to a specified time-out period. If this servicing action does not occur before the timer expires the SWT generates an interrupt or hardware reset. The SWT can be configured to generate a reset or interrupt on an initial time-out. A reset is always generated on a second consecutive time-out.

Sample Reference Manual
NXP Semiconductors 1329

Chapter 34 Software Watchdog Timer (SWT)

accesses by masters without permission. If the RIA bit in the SWT_CR is set then the SWT generates a system reset on an invalid access otherwise a bus error is generated. If either the HLK or SLK bits in the SWT_CR are set, then the SWT_CR, SWT_TO, SWT_WN, and SWT_SK registers are read-only.

The SWT memory map is shown in the following table.

SWT memory map

Address offset (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	SWT Control Register (SWT_CR)	32	R/W	See section	34.4.1/1331
4	SWT Interrupt Register (SWT_IR)	32	R/W	0000_0000h	34.4.2/1334
8	SWT Time-out Register (SWT_TO)	32	R/W	See section	34.4.3/1334
C	SWT Window Register (SWT_WN)	32	R/W	0000_0000h	34.4.4/1335
10	SWT Service Register (SWT_SR)	32	W	0000_0000h	34.4.5/1335
14	SWT Counter Output Register (SWT_CO)	32	R	0000_0000h	34.4.6/1336
18	SWT Service Key Register (SWT_SK)	32	R/W	0000_0000h	34.4.7/1336

34.4.1 SWT Control Register (SWT_CR)

NOTE
The reset value for the SWT_CR is implementation specific.
See the configuration information.

The SWT_CR contains fields for configuring and controlling the SWT.

This register is read-only if either the SWT_CR[HLK] or SWT_CR[SLK] bits are set.

Address: 0h base + 0h offset = 0h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MAP0	MAP1	MAP2	MAP3	MAP4	MAP5	MAP6	MAP7	0							
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0					SMD	RIA	WND	ITR	HLK	SLK	CSL	STP	FRZ	WEN	
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- The reset value for the SWT_CR is implementation specific. See the configuration information.

Sample Reference Manual
NXP Semiconductors 1331

Figure 2. Example: chip-specific information that supersedes content in the same chapter

1.4 Register descriptions

Module chapters present register information in:

- Memory maps including:
  - Addresses
  - The name and acronym/abbreviation of each register
  - The width of each register (in bits)
  - Each register's reset value
  - The page number on which each register is described
- Register figures
- Field-description tables
- Associated text

The register figures show the field structure using the conventions in the following figure.

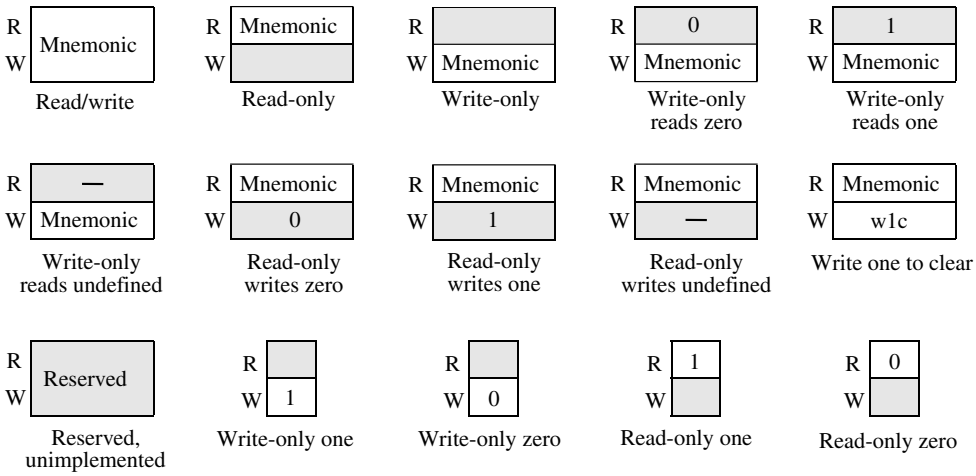


Figure 3. Register figure conventions

1.5 Conventions

1.5.1 Numbering systems

The following suffixes identify different numbering systems:

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is written 101b. In some cases, binary numbers are shown with the prefix *0b*.
d	Decimal number. Decimal numbers are followed by this suffix only when the possibility of confusion exists. In general, decimal numbers are shown without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is written 3Ch. In some cases, hexadecimal numbers are shown with the prefix *0x*.

1.5.2 Typographic notation

The following typographic notation is used throughout this document:

Example	Description
placeholder, x	Items in italics are placeholders for information that you provide. Italicized text is also used for the titles of publications and for emphasis. Plain lowercase letters are also used as placeholders for single letters and numbers.
`code`	Fixed-width type indicates text that must be typed exactly as shown. It is used for instruction mnemonics, directives, symbols, subcommands, parameters, and operators. Fixed-width type is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.
SR[SCM]	A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets and separated by a colon represent either:

Table continues on the next page...

Table continued from the previous page...

Example	Description
	• A subset of a register's named field For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus.

1.5.3 Special terms

The following terms have special meanings:

Term	Meaning
asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
reserved	Refers to a memory space, register, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. • Do not modify the default value of a reserved programming setting, such as the reset value of a reserved register field. • Consider undefined locations in memory to be reserved.
w1c	Write 1 to clear: Refers to a register bitfield that must be written as 1 to be "cleared."

Chapter 2
Introduction

2.1 Introduction

The MCX W71 microcontroller is a low-power, highly secure, Arm® Cortex-M33-based wireless device targeting these applications:

- Smart Home IoT
  - Smart Home environmental, occupancy and security sensors
  - Home Gateways and Bridges
  - Smart Lighting
  - Smart Plugs
  - Access Control
  - HVACs and Thermostats
  - Window Shades
- Industrial/IoT
  - Positioning/Localization
  - Building Control and Monitoring
  - Process/Factory Automation
  - Access Control

The device includes these key features:

Highly secure	• Arm Trustzone-M • Trusted resource domain controller (TRDC) providing access policies for embedded memory and peripherals • NXP's EdgeLock® Secure Enclave providing secure key management and storage plus hardware acceleration of cryptographic algorithms • Internal flash memory with on-the-fly encryption and decryption using a PRINCE XEX block cipher mode • Factory Root of Trust programming • Secure boot, debug and firmware updates
Power-efficient operating modes	• Less than 35 µA/MHz active current at 96 MHz • Less than 2 µA in Power Down mode with FRO32K active and 32 KB SRAM retention • Less than 1 µA in Deep Power Down mode with FRO32K active

2.2 Functional overview

The following figure shows a top-level organization of the modules within the device organized by functional category.

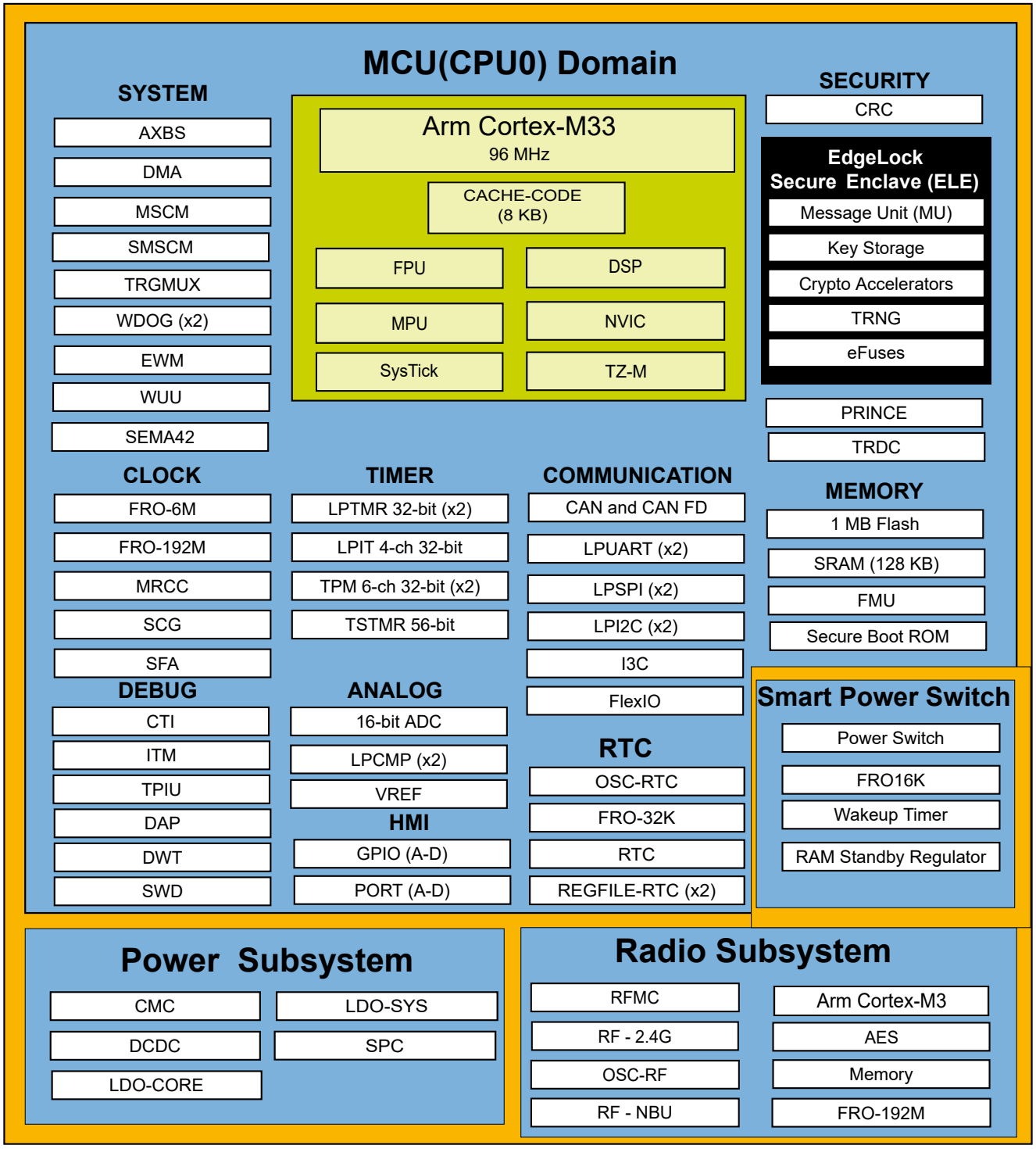


Figure 4. Features block diagram

Table 1. Module functional categories

Module category	Description
Core	The Arm® Cortex®-M33 is a member of the Cortex® M Series of processors targeting microcontroller cores focused on cost sensitive, deterministic, interrupt driven environments. The Cortex® M33 processor is based on the Arm®v8-M Architecture and ThumbR-2 ISA and is upward compatible with the Cortex® M7, M4, M3, M1 and M0/M0+.
System	• AHB Cross-Bar Switch (AXBS) • Core Mode Controller (CMC) • Direct Current / Direct Current Converter - Low Voltage (DCDC-LV) • Direct Memory Access (DMA) controller • External Watchdog Monitor (EWM) • Low Drop Out Regulator - Core (LDO_CORE) • Low Drop Out Regulator - System (LDO_SYS) • Miscellaneous System Control Module (MSCM) • Secure Miscellaneous System Control Module (SMSCM) • Semaphore Module (SEMA42) • System Power Controller (SPC) • Trigger Multiplexer (TRGMUX) • Wake-Up Unit (WUU) • Watchdog (WDOG)
Memory	• Flash Management Unit (FMU) • Read Only Memory - Boot (ROM_BOOT) • Register File - REGFILE • Static Random Access Memory (SRAM)
Clock	• Crystal Oscillator - Real Time Clock (OSC_RTC) • Free Running Oscillator - 32 kHz (FRO_32K) • Free Running Oscillator - 16 kHz (FRO_16K) • Free Running Oscillator - 6 MHz (FRO_6M) • Free Running Oscillator - 192 MHz (FRO_192M) • Radio Free Running Oscillator - 192 MHz (RFRO_192M) • Module Reset and Clock Control (MRCC) • System Clock Generator (SCG) • Signal Frequency Analyzer (SFA)
Security	• EdgeLock Secure Enclave

Table continues on the next page...

Table 1. Module functional categories (continued)

Module category	Description
	• Trusted Resource Domain Controller (TRDC) • Cyclic Redundancy Check (CRC) • Non-Volatile Memory PRINCE XEX (NPX) encryption/decryption • Up to 4 digital tamper pins
Timer	• Real Time Clock (RTC) • Low-Power Timer (LPTMR) • Low Power Periodic Interrupt Timers (LPIT) • Time Stamp Timer (TSTMR) • Timer/PWM Module (TPM)
Communication	• Controller Area Network with Flexible Data rate (CAN FD) • Flexible Input/Output (FlexIO) • Low Power Inter-Integrated Circuit (LPI2C) • Improved Inter-Integrated Circuit (I3C) • Low Power Serial Peripheral Interface (LPSPI) • Low Power Universal Asynchronous Receive/Transmit (LPUART)
HMI	• General Purpose Input/Output (GPIO) • Port Control (PORT)
Analog	• Analog-to-Digital Converter - General Purpose (ADC) • Low Power Comparator (LPCMP) • Voltage Reference (VREF)
Radio	• Radio Frequency - 2.4 GHz (RF-2.4G)

2.2.1 Core

The following core modules are available on this device.

Table 2. Core modules

Module	Description
CPU0	CPU0 is an Arm® Cortex®-M33, a member of the Cortex® M Series of processors targeting microcontroller cores focused on cost-sensitive, deterministic, interrupt-driven environments.
Code Cache Memory (CACHE-CODE)	The CACHE-CODE is an 8-KB cache for memory attached to the Code bus.
Digital Signal Processing Extensions (DSP)	The Cortex-M33 processor features extended single-cycle Multiply Accumulate (MAC) instructions, optimized Single Instruction Multiple Data (SIMD) arithmetic, and saturating arithmetic instructions.

Table continues on the next page...

Table 2. Core modules (continued)

Module	Description
Floating Point Unit (FPU)	The FPU provides single-precision floating point computation compliant to the IEEE Standard for Floating-Point Arithmetic (IEEE 754).
Memory Protection Unit (MPU)	The MPU provides support for eight unified protection regions, overlapping protection regions with ascending region priority, and access permissions. MPU mismatches and permission violations invoke the HardFault handler. See the Armv8-M Architecture Reference Manual for more information.
Nested Vectored Interrupt Controller (NVIC)	The Armv8-M exception model and Nested-Vectored Interrupt Controller (NVIC) implement a relocatable vector table supporting many external interrupts, a single non-maskable interrupt (NMI), and priority levels.
System Timer (SysTick)	See the Armv8-M Architecture Reference Manual for more information about this system timer.
TrustZone - M (TZM)	The TZM is an Arm security feature related to hardware-enforced access control mechanisms.

2.2.2 Debug

The following Debug (DBG) modules are available on this device:

Table 3. Debug modules

Module	Description
Cross Trigger Interface (CTI)	The CTI enables the device's debug logic to interact with (cross trigger) each other.
Data Watchpoint and Trace (DWT)	The DWT is a generic name for several modules that allow debug access of the Cortex-M33. The DWT is composed of the Debug Watchpoint and Trace (DWT) module and the Flash Patch and Breakpoint (FPB) unit.
Debug Access Port (DAP)	The DAP enables real-time access to the device registers from an external debugger without halting the processor cores.
Instruction Trace Macrocell (ITM)	The ITM provides a memory-mapped register interface that applications can use to write logging or event words for profiling software.
Serial Wire Debug (SWD)	The SWDv2 is a serial communication interface used for debugging devices with multiple cores while only requiring a single external interface.
Trace Port Interface Unit (TPIU)	The TPIU acts as a bridge between the on-chip trace data from the modules such as the Embedded Trace Macrocell (ETM) or the Instrumentation Trace Macrocell (ITM), which have separate system IDs, to the external world.

2.2.3 System

The following System (SYS) modules are available on this device:

Table 4. System modules

Module	Description
AHB Cross-Bar Switch (AXBS)	The AXBS connects bus masters and bus slaves, allowing all bus masters to access different bus slaves simultaneously and providing arbitration among the bus masters when they access the same slave.
Core Mode Controller (CMC)	The CMC provides control and protection on entry and exit to each power mode, control for the System Power Controller (SPC), and reset entry and exit for the complete device.
Direct Current / Direct Current Converter - Low Voltage (DCDC-LV)	An efficient voltage converter for generating the low voltage supply for internal Low Drop Out (LDO) regulators.
Enhanced Direct Memory Access (eDMA)	The eDMA controller provides programmable channels with transfer control descriptors for data movement via dual-address transfers for 8-, 16-, 32- and 128-bit data values.
External Watchdog Monitor (EWM)	Similar to the Watchdog timer, the EWM monitors internal and external system operations and forces a reset in case of failure.
Low Drop Out Regulator - Core (LDO-CORE)	A voltage regulator for generating the core voltage for the device.
Low Drop Out Regulator - System (LDO-SYS)	A voltage regulator for generating the system voltage for the device.
Miscellaneous System Control Module (MSCM)	The MSCM contains CPU configuration registers and on-chip memory controller registers.
Secure Miscellaneous System Control Module (SMSCM)	The SMSCM contains configuration registers related to security.
Semaphore Module (SEMA42)	The SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.
System Power Controller (SPC)	The SPC provides control over the operation and configuration of the system power generation modules to optimize power consumption for the level of functionality needed.
Trigger Multiplexer (TRGMUX)	The TRGMUX allows the trigger output of one peripheral to be connected to the trigger input of a second peripheral.
Wake-Up Unit (WUU)	The WUU provides a mechanism for external or internal sources to initiate the exit of certain low-power modes.
Watchdog (WDOG)	The WDOG monitors internal system operation and forces a reset in case of failure. It can run from multiple clocks with a programmable refresh window to detect deviations in program flow or system frequency.

2.2.4 Clocks

The following Clock (CLK) modules are available on this device:

Table 5. Clock sources

Module	Description
Crystal Oscillator - Radio Frequency (OSC-RF)	The OSC-RF, in conjunction with an external crystal or resonator, generates a reference clock for the Radio that can also be used by the device.
Crystal Oscillator - Real Time Clock (OSC-RTC)	The OSC-RTC oscillator, in conjunction with an external 32 kHz crystal, generates a 32 kHz reference clock for the RTC that can also be used by the radio and other modules of the device.
Free Running Oscillator - 32 kHz (FRO-32K)	The FRO-32K is an internal clock source that generates a 32 kHz reference clock for the RTC that can also be used by the radio and other modules of the device.
Free Running Oscillator - 16 kHz (FRO-16K)	The FRO-16K is an internal clock source that generates a 16 kHz reference clock that is used by the Smart Power Switch domain.
Free Running Oscillator - 6 MHz (FRO-6M)	The FRO-6M is an internal clock source that generates a 6 MHz frequency for use by the radio and other modules of the device.
Free Running Oscillator - 192 MHz (FRO-192M)	The FRO-192M is an internal clock source that generates a 192 MHz frequency for use by the modules other than the radio of the device.
Radio Free Running Oscillator - 192 MHz (RFRO-192M)	The RFRO-192M is an internal clock source that generates a 192 MHz frequency for use by the radio module of the device.
Module Reset and Clock Control (MRCC)	The MRCC provides the user with access to the configuration control registers for the peripherals of the device.
System Clock Generator (SCG)	The SCG provides the user with access to the configuration control registers for the system level clock sources.
Signal Frequency Analyzer (SFA)	The SFA provides facilities for measurement of clock period/frequency as well as time between triggers.

2.2.5 Memory

The following Memory and Memory Interface (MEM) modules are available on this device:

Table 6. Memory modules

Module	Description
Flash Memory	Programmable flash memory — non-volatile flash memory that can store executable program code or data.
Flash Management Unit (FMU)	The FMU manages the interface between the device and the on-chip flash memory.
Read Only Memory - Boot (ROM-BOOT)	Provisions the internal flash with an embedded firmware image during manufacturing or at any time during the life of the device.
Register File - REGFILE	32-byte register file that can be retained in all power modes.

Table continues on the next page...

Table 6. Memory modules (continued)

Module	Description
Static Random Access Memory (SRAM)	Internal system SRAM memory. Each individual block of SRAM can be configured to be retained in low-power modes.

2.2.6 Security

The following Security modules are available on this device:

Table 7. Security modules

Module	Description
EdgeLock Secure Enclave	The EdgeLock Secure Enclave is a security sub system providing essential cryptographic services to the host system within a single microcontroller specifically in two areas: secret key storage and management, plus execution of symmetric and public key cryptographic services, for example, AES, DES/3DES, SHA-256, RSA and ECC, based on requests from the host
Cyclic Redundancy Check (CRC)	Hardware CRC generator provides error detection for all single, double, and many multi-bit errors.
Non-Volatile Memory PRINCE XEX (NPX)	Provides on-the-fly encryption/decryption of the embedded flash contents using a PRINCE XEX block cipher.
Trusted Resource Domain Controller (TRDC)	Provides programmable access control mechanisms for independent processing domains. Unique isolation and access mechanisms for memory and peripheral resources are configured on a domain basis.

2.2.7 Timers

The following Timer (TMR) modules are available on this device:

Table 8. Timer modules

Module	Description
Real Time Clock (RTC)	32-bit seconds counter with 32-bit Alarm Independent power supply
Low-Power Timer (LPTMR)	32-bit time or pulse counter with compare feature
Low Power Periodic Interrupt Timers (LPIT)	Four general purpose interrupt timers with 32-bit counter resolution and DMA support
Time Stamp Timer (TSTMR)	Montonic counter used for long term program debug 56-bit counter resolution
Timer/PWM Module (TPM)	Six channel timer with DMA support for electric motor and power management applications. Clocked by an asynchronous clock that can remain enabled in low power modes.

2.2.8 Communications

The following Communication (COM) modules are available on this device:

Table 9. Communication modules

Module	Description
Controller Area Network (CAN)	Supports the full implementation of the CAN Specification Version 2.0, Part B. FD Support.
Flexible Input/Output (FlexIO)	Functions in Sleep/Deep Sleep modes. DMA support
Low Power Inter-Integrated Circuit (LPI2C)	Allows communication between a number of devices. Also supports the System Management Bus (SMBus) Specification, version 2.
Improved Inter-Integrated Circuit (I3C)	I3C is an extension of the I2C bus protocol supporting higher speeds.
Low Power Serial Peripheral Interface (LPSPI)	Synchronous serial bus for communication to an external device.
Low Power Universal Asynchronous Receive/ Transmit (LPUART)	Asynchronous serial bus communication interface with programmable 8- or 9-bit data format.

2.2.9 Radio

The following are Radio Frequency (RF) modules.

Table 10. Radio modules

Module	Description
Radio Frequency - 2.4 GHz (RF-2.4G)	The 2.4 GHz band radio covers 2.36-2.4835 GHz and is built to support multiple protocols: • Bluetooth Low Energy V5.3 • Generic 2-level FSK/GFSK/MSK/GMSK • IEEE 802.15.4-2015

2.2.10 Human Machine Interface

The following Human Machine Interfaces (HMI) modules are available on this device:

Table 11. HMI modules

Module	Description
General Purpose Input/Output (GPIO)	All General Purpose Input / Output (GPIO) pins are capable of interrupt and DMA request generation.
Port Control	The Port Control (PORT) module provides support for pad control functions.

2.2.11 Analog

The following Analog (ANA) modules are available on this device:

Table 12. Analog modules

Module	Description
Dual Single End Analog-to-Digital Converter (ADC)	A Dual 16-bit Single End Successive-Approximation Register (SAR) Analog-to-Digital Converter (ADC)
Low Power Comparator (LPCMP)	Compares two analog input voltages across the full range of the supply voltage.
Voltage Reference (VREF)	The VREF module supplies an accurate voltage output that can be used by internal or external peripherals.

2.3 Block diagram

The following figure shows a detailed block diagram of the device.

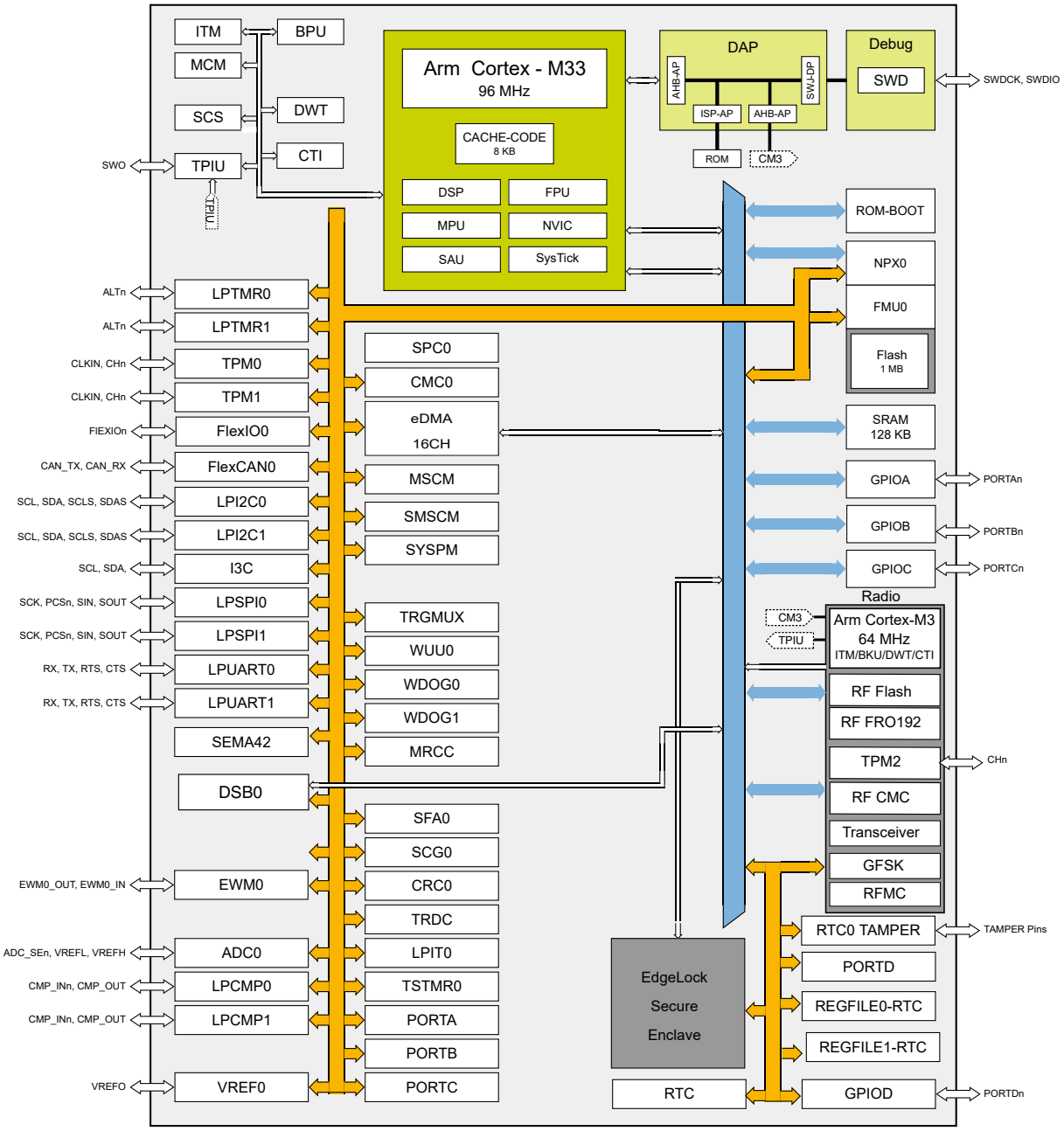


Figure 5. Bus architecture block diagram

2.4 Ordering information

The following table summarizes the part numbers of the devices covered by this document.

Table 13. Ordering information

Part number	Flash (KB)	SRAM (KB)	GPIOs	Pin Count	Package	CAN	802.15.4	Bluetooth 5.3	Qualification	Temperature	Packaging type
MCXW716CMFTAT	1024	128	29	48	HVQFN	Yes	Yes	Yes	Industrial	–40 °C to 125 °C	Tray
MCXW716CMFTAR	1024	128	29	48	HVQFN	Yes	Yes	Yes	Industrial	–40 °C to 125 °C	Tape and Reel
MCXW716AMFTAT	1024	128	29	48	HVQFN	No	Yes	Yes	Industrial	–40 °C to 125 °C	Tray
MCXW716AMFTAR	1024	128	29	48	HVQFN	No	Yes	Yes	Industrial	–40 °C to 125 °C	Tape and Reel
MCXW716CMFPAT	1024	128	22	40	HVQFN	Yes	Yes	Yes	Industrial	–40 °C to 125 °C	Tray
MCXW716CMFPAR	1024	128	22	40	HVQFN	Yes	Yes	Yes	Industrial	–40 °C to 125 °C	Tape and Reel
MCXW716AMFPAT	1024	128	22	40	HVQFN	No	Yes	Yes	Industrial	–40 °C to 125 °C	Tray
MCXW716AMFPAR	1024	128	22	40	HVQFN	No	Yes	Yes	Industrial	–40 °C to 125 °C	Tape and Reel

Chapter 3
Core Overview

3.1 Introduction

This section covers the core modules included in this device.

3.2 Cortex-M33 Code and System Buses

The Cortex-M33 processor implements a modified Harvard memory architecture using two 32-bit bus interfaces: the Code and System buses. The bus interfaces are activated by address range and can include both instruction fetches and operand data references on a given bus port. (A traditional Harvard architecture strictly separates instruction fetches and operand data references onto specific bus ports regardless of access address.)

The Code bus is typically used for instruction fetching and data accesses of PC-relative data, while the system bus is typically used for operand data references to the on- and off-chip memories and peripheral accesses. The bus structure fully supports concurrent instruction fetch and data accesses, but the Cortex-M33 implementations can generate both types of references on each bus.

NOTE

It is recommended that performance critical code be located such that it fetches from the Code bus interface as defined by addresses < 0x2000_0000.

3.2.1 Code Bus access

Code Bus accesses are routed to the Code TCM if they are mapped to that space. All other Code Bus accesses are routed to the Code Cache Controller. This controller then processes the cacheable accesses as needed, while bypassing the non-cacheable accesses or forwarding the cache write-through and cache miss accesses to the downstream memories through the master port of this cache controller.

3.2.2 System bus access

System Bus accesses are routed to the System TCM if they are mapped to that space. All other System Bus accesses are routed to the target address in destination memories through S1 port.

3.2.3 Access control

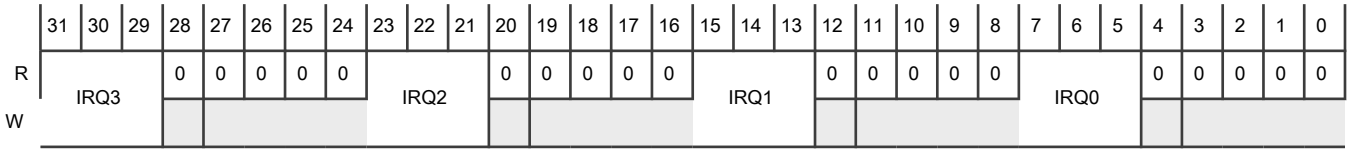
All core Code and System Bus accesses are checked by the core access control logic, that is, IDAU/SAU and MPU. All requests that miss or bypass the cache are checked by downstream TRDC logic. The caches include protection control signals (HPROT[3:0]) and processing domain bits as part of the tags. If a fetch address hits the cache but the protection control and/or domain bits are different, the cache controller forces a miss with the allocate location the same as the address hit location in the cache. This policy allows all the downstream checks to take place, and this new miss is loaded in the cache with the updated protection control and domain bits overwriting the line with the same address. This keeps the cache coherent while always checking accesses that need to see the downstream checks.

All requests are checked by normal platform slave bus checks in the corresponding Memory Block Controller (MBC).

3.3 Nested Vectored Interrupt Controller (NVIC)

3.3.1 Interrupt priority levels

This device supports 8 priority levels for interrupts. Therefore, in the NVIC each source in the IPR registers contains 3 bits. For example, IPR0 is shown below:



3.3.2 Non-Maskable Interrupt (NMI) configuration

The Non-Maskable Interrupt (NMI) request to the NVIC is controlled by the external NMI_b signal. The pin the NMI_b signal is multiplexed on, must be configured for the Non-Maskable Interrupt function to generate the non-maskable interrupt request.

3.3.3 Interrupt channel assignments

The interrupt source assignments are defined in the following table.

- Vector number — the value stored on the stack when an interrupt is serviced.
- IRQ number — non-core interrupt source count, which is the vector number minus 16.

The IRQ number is used within Arm's NVIC documentation.

Table 15. Interrupt Vector Assignments

Address	Vector	IRQ	NVIC non-IPR register number	NVIC IPR register number	Module	Source Description
0000_0000	0	-	-	-	Cortex-M33	Initial Stack Pointer
0000_0004	1	-	-	-	Cortex-M33	Initial Program Counter
0000_0008	2	-	-	-	Cortex-M33	Non-Maskable Interrupt (NMI)
0000_000C	3	-	-	-	Cortex-M33	Hard Fault
0000_0010	4	-	-	-	Cortex-M33	MemManage Fault
0000_0014	5	-	-	-	Cortex-M33	Bus Fault / TRDC
0000_0018	6	-	-	-	Cortex-M33	Usage Fault
0000_001C	7	-	-	-	Cortex-M33	Secure Fault
0000_0020	8	-	-	-	Reserved	-
0000_0024	9	-	-	-	Reserved	-
0000_0028	10	-	-	-	Reserved	-
0000_002C	11	-	-	-	Cortex-M33	Supervisor Call (SVCall)
0000_0030	12	-	-	-	Cortex-M33	Debug Monitor
0000_0034	13	-	-	-	Reserved	-
0000_0038	14	-	-	-	Cortex-M33	Pendable request for system service (Penable SrvReq)
0000_003C	15	-	-	-	Cortex-M33	System Tick Timer
0000_0040	16	0	0	0	Cortex-M33	Cross Trigger Interface interrupt

Table continues on the next page...

Table 15. Interrupt Vector Assignments (continued)

Address	Vector	IRQ	NVIC non-IPR register number	NVIC IPR register number	Module	Source Description
0000_0044	17	1	0	0	CMC0	Core Mode Controller interrupt
0000_0048	18	2	0	0	eDMA	eDMA channel 0 error or transfer complete
0000_004C	19	3	0	0	eDMA	eDMA channel 1 error or transfer complete
0000_0050	20	4	0	1	eDMA	eDMA channel 2 error or transfer complete
0000_0054	21	5	0	1	eDMA	eDMA channel 3 error or transfer complete
0000_0058	22	6	0	1	eDMA	eDMA channel 4 error or transfer complete
0000_005C	23	7	0	1	eDMA	eDMA channel 5 error or transfer complete
0000_0060	24	8	0	2	eDMA	eDMA channel 6 error or transfer complete
0000_0064	25	9	0	2	eDMA	eDMA channel 7 error or transfer complete
0000_0068	26	10	0	2	eDMA	eDMA channel 8 error or transfer complete
0000_006C	27	11	0	2	eDMA	eDMA channel 9 error or transfer complete
0000_0070	28	12	0	3	eDMA	eDMA channel 10 error or transfer complete
0000_0074	29	13	0	3	eDMA	eDMA channel 11 error or transfer complete
0000_0078	30	14	0	3	eDMA	eDMA channel 12 error or transfer complete
0000_007C	31	15	0	3	eDMA	eDMA channel 13 error or transfer complete
0000_0080	32	16	0	4	eDMA	eDMA channel 14 error or transfer complete
0000_0084	33	17	0	4	eDMA	eDMA channel 15 error or transfer complete
0000_0088	34	18	0	4	EWM0	External Watchdog Monitor 0 interrupt
0000_008C	35	19	0	4	MCM	Miscellaneous Control Module interrupt
0000_0090	36	20	0	5	MSCM	Miscellaneous System Control Module interrupt
0000_0094	37	21	0	5	SPC0	System Power Controller 0 interrupt
0000_0098	38	22	0	5	WUU0	Wake-Up Unit 0 interrupt
0000_009C	39	23	0	5	WDOG0	Watchdog Timer 0 interrupt
0000_00A0	40	24	0	6	WDOG1	Watchdog Timer 1 interrupt
0000_00A4	41	25	0	6	SCG0	System Clock Generator 0 interrupt
0000_00A8	42	26	0	6	SFA0	Singal Frequency Analyzer 0 interrupt
0000_00AC	43	27	0	6	FMU0	Flash Memory Unit 0 interrupt
0000_00B0	44	28	0	7	ELE	EdgeLock enclave command interface interrupt
0000_00B4	45	29	0	7	ELE	EdgeLock enclave interrupt
0000_00B8	46	30	0	7	ELE	EdgeLock enclave non-secure interrupt

Table continues on the next page...

Table 15. Interrupt Vector Assignments (continued)

Address	Vector	IRQ	NVIC non-IPR register number	NVIC IPR register number	Module	Source Description
0000_00BC	47	31	0	7	TRDC0	Trusted Resource Domain Controller 0 interrupt
0000_00C0	48	32	1	8	RTC0	Real Time Clock 0 alarm interrupt
0000_00C4	49	33	1	8	RTC0	Real Time Clock 0 seconds interrupt
0000_00C8	50	34	1	8	LPTMR0	Low-Power Timer0 interrupt
0000_00CC	51	35	1	8	LPTMR1	Low-Power Timer1 interrupt
0000_00D0	52	36	1	9	LPIT0	Low-Power Periodic Interrupt Timer 0 interrupt
0000_00D4	53	37	1	9	TPM0	Timer / PWM Module 0 interrupt
0000_00D8	54	38	1	9	TPM1	Timer / PWM Module 1 interrupt
0000_00DC	55	39	1	9	LPI2C0	Low-Power Inter Integrated Circuit 0 interrupt
0000_00E0	56	40	1	10	LPI2C1	Low-Power Inter Integrated Circuit 1 interrupt
0000_00E4	57	41	1	10	I3C0	Improved Inter-Integrated Circuit 0 interrupt
0000_00E8	58	42	1	10	LPSPI0	Low-Power Serial Peripheral Interface 0 interrupt
0000_00EC	59	43	1	10	LPSPI1	Low-Power Serial Peripheral Interface 1 interrupt
0000_00F0	60	44	1	11	LPUART0	Low-Power Universal Asynchronous Receiver/ Transmitter 0 interrupt
0000_00F4	61	45	1	11	LPUART1	Low-Power Universal Asynchronous Receiver/ Transmitter 1 interrupt
0000_00F8	62	46	1	11	FLEXIO0	Flexible Input/Output 0 interrupt
0000_00FC	63	47	1	11	CAN0	Controller Area Network 0 interrupt
0000_0100	64	48	1	12	RF-2.4G	Radio IMU interrupt 0 (msg_rdy_imu)
0000_0104	65	49	1	12	RF-2.4G	Radio IMU interrupt 1(msg_space_avail_imu)
0000_0108	66	50	1	12	RF-2.4G	Radio NBU timeout interrupt
0000_010C	67	51	1	12	RF-2.4G	Radio FMU interrupt
0000_0110	68	52	1	13	RF-2.4G	Radio WOR RX FAIL interrupt
0000_0114	69	53	1	13	RF-2.4G	Radio Frequency 2.4GHz - 802.15.4 Link Layer interrupt
0000_0118	70	54	1	13	RF-2.4G	Radio Frequency 2.4 GHz - Generic Link Layer interrupt
0000_011C	71	55	1	13	RF-2.4G	Radio Frequency 2.4 GHz - BRIC interrupt
0000_0120	72	56	1	14	RF-2.4G	Radio Transceiver - Radio LANT_SW interrupt
0000_0124	73	57	1	14	RF-2.4G	RFMC interrupt
0000_0128	74	58	1	14	DSB	Data Stream Buffer interrupt

Table continues on the next page...

Table 15. Interrupt Vector Assignments (continued)

Address	Vector	IRQ	NVIC non-IPR register number	NVIC IPR register number	Module	Source Description
0000_012C	75	59	1	14	GPIOA	General Purpose Input/Output A interrupt 0
0000_0130	76	60	1	15	GPIOA	General Purpose Input/Output A interrupt 1
0000_0134	77	61	1	15	GPIOB	General Purpose Input/Output B interrupt 0
0000_0138	78	62	1	15	GPIOB	General Purpose Input/Output B interrupt 1
0000_013C	79	63	1	15	GPIOC	General Purpose Input/Output C interrupt 0
0000_0140	80	64	2	16	GPIOC	General Purpose Input/Output C interrupt 1
0000_0144	81	65	2	16	GPIOD	General Purpose Input/Output D interrupt 0
0000_0148	82	66	2	16	GPIOD	General Purpose Input/Output D interrupt 1
0000_014C	83	67	2	16	PORTA	Port A EFT interrupt
0000_0150	84	68	2	17	PORTB	Port B EFT interrupt
0000_0154	85	69	2	17	PORTC	Port C EFT interrupt
0000_0158	86	70	2	17	PORTD	Port D EFT interrupt
0000_015C	87	71	2	17	ADC0	Analog-to-Digital Converter 0 interrupt
0000_0160	88	72	2	18	LPCMP0	Low-Power Comparator 0 interrupt
0000_0164	89	73	2	18	LPCMP1	Low-Power Comparator 1 interrupt
0000_0168	90	74	2	18	VBAT	Smart Power Switch Domain interrupt
0000_016C	91	75	2	18	Reserved	-

3.4 System memory map

The following table shows the device's high-level memory map. Additionally, the various sections are split into "Secure" and "Non-Secure" apertures for each IP. Although these apertures address the same IP, access through the "Secure" aperture can only be performed by code with the appropriate security settings.

Table 16. CPU0 memory map

Start address	Default [1]	Description	Alias	Size	Access	Cached
0000_0000	Non-Secure	Program Flash	FLASH	1024 KB	All Masters except data stream and RF	Code Cache
0010_0000		Reserved	—	—	—	—
0100_0000		Flash Logical Window	FLW	1024 KB	All Masters except data stream and RF	Code Cache

Table continues on the next page...

Table 16. CPU0 memory map (continued)

Start address	Default [1]	Description	Alias	Size	Access	Cached
0110_0000		Reserved	—	—	—	—
0200_0000		IFR0	IFR0	32 KB	All Masters except data stream and RF	No
0200_8000		Reserved	—	—	—	—
0210_0000		IFR1	IFR1	8 KB	All Masters except data stream and RF	No
0210_2000		Reserved	—	—	—	—
0400_0000		Tightly Coupled Memory - Code	CTCM	16 KB	All Masters except data stream and RF	No
0400_4000		Reserved	—	—	—	—
0480_0000		Read Only Memory - Boot	ROM-BOOT	96 KB	All Masters except data stream and RF	No
0481_8000		Reserved	—	—	—	—
1000_0000	Secure	Program Flash	FLASH	1024 KB	All Masters except data stream and RF	Code Cache
1010_0000		Reserved	—	—	—	—
1100_0000		Flash Logical Window	FLW	1024 KB	All Masters except data stream and RF	Code Cache
1110_0000		Reserved	—	—	—	—
1200_0000		IFR0	IFR0	32 KB	All Masters except data stream and RF	No
1200_8000		Reserved	—	—	—	—
1210_0000		IFR1	IFR1	8 KB	All Masters except data stream and RF	No

Table continues on the next page...

Table 16. CPU0 memory map (continued)

Start address	Default [1]	Description	Alias	Size	Access	Cached
1210_2000		Reserved	—	—	—	—
1400_0000		Tightly Coupled Memory - Code	CTCM	16 KB	All Masters except data stream and RF	No
1400_4000		Reserved	—	—	—	—
1480_0000		Read Only Memory - Boot	ROM-BOOT	96 KB	All Masters except data stream and RF	No
1481_8000		Reserved	—	—	—	—
2000_0000	Non-Secure	Tightly Coupled Memory - System	STCM	112 KB	All Masters except RF	No
2001_C000		Reserved	—	—	—	—
3000_0000	Secure	Tightly Coupled Memory - System	STCM	112 KB	All Masters except RF	No
3001_C000		Reserved	—	—	—	—
4000_0000	Non-Secure	Peripheral Bridge2	PBRIDGE2	512 KB	All Masters	No
4008_0000		Reserved	—	—	—	—
4800_0000		Fast Peripherals 0	FASTP0	8 MB	All Masters except RF	No
4880_0000		Fast Peripherals 1	FASTP1	8 MB	All Masters except data stream and RF	No
4900_0000		Reserved	—	—	—	—
5000_0000	Secure	Peripheral Bridge2	PBRIDGE2	512 KB	All Masters	No
5008_0000		Reserved	—	—	—	—
5800_0000		Fast Peripherals 0	FASTP0	8 MB	All Masters except RF	No
5880_0000		Fast Peripherals 1	FASTP1	8 MB	All Masters except data stream and RF	No
5900_0000		Reserved	—	—	—	—
6000_0000	Non-Secure	Reserved	—	—	—	—
7000_0000	Secure	Reserved	—	—	—	—

Table continues on the next page...

Table 16. CPU0 memory map (continued)

Start address	Default [1]	Description	Alias	Size	Access	Cached
8000_0000	Non-Secure	Reserved	—	—	—	—
9000_0000	Secure	Reserved	—	—	—	—
A000_0000	Non-Secure	Reserved	—	—	—	—
B000_0000	Secure	Reserved	—	—	—	—
C000_0000	Non-Secure	Reserved	—	—	—	—
D000_0000	Secure	Reserved	—	—	—	—
E000_0000	Mixed	Private Peripheral Bus - Internal	PPB	256 KB	Core	No
E004_0000		Private Peripheral Bus - External	EPPB	768 KB	Core	No
E010_0000	Non-Secure	Reserved	—	—	—	—

1. This is the default Targetted Security Attribute for the memory region at reset. It is set by the IDAU and the SAU configuration. The attribute listed is the intended use case for users of TrustZone-M (TZM) security function.

3.5 Peripheral bridge2 (PBRIDGE2)

The Peripheral Bridge2 (PBRIDGE2) is the portion of the bus fabric that connects the peripherals to the processor elements. Each peripheral has a base address where the processor elements can access them. The following sections provide the memory map of the peripherals connected to the PBRIDGE2.

3.5.1 Peripheral Bridge2 (PBRIDGE2) memory map

Table 17. Peripheral Bridge2

Base address (non-secure)	Base address (secure)	Slot	Module	
System modules				
4000_0000	5000_0000	0	AHB Crossbar Switch 0	AXBS0
4000_1000	5000_1000	1	Core Mode Controller 0	CMC0
4000_2000	5000_2000	2	Enhanced Direct Memory Access - Management Page 0	eDMA-MP
4000_3000	5000_3000	3	Enhanced Direct Memory Access - Channel 0	eDMA-TCD-CH0
4000_4000	5000_4000	4	Enhanced Direct Memory Access - Channel 1	eDMA-TCD-CH1
4000_5000	5000_5000	5	Enhanced Direct Memory Access - Channel 2	eDMA-TCD-CH2
4000_6000	5000_6000	6	Enhanced Direct Memory Access - Channel 3	eDMA-TCD-CH3

Table continues on the next page...

Table 17. Peripheral Bridge2 (continued)

Base address (non-secure)	Base address (secure)	Slot	Module	
4000_7000	5000_7000	7	Enhanced Direct Memory Access - Channel 4	eDMA-TCD-CH4
4000_8000	5000_8000	8	Enhanced Direct Memory Access - Channel 5	eDMA-TCD-CH5
4000_9000	5000_9000	9	Enhanced Direct Memory Access - Channel 6	eDMA-TCD-CH6
4000_A000	5000_A000	10	Enhanced Direct Memory Access - Channel 7	eDMA-TCD-CH7
4000_B000	5000_B000	11	Enhanced Direct Memory Access - Channel 8	eDMA-TCD-CH8
4000_C000	5000_C000	12	Enhanced Direct Memory Access - Channel 9	eDMA-TCD-CH9
4000_D000	5000_D000	13	Enhanced Direct Memory Access - Channel 10	eDMA-TCD-CH10
4000_E000	5000_E000	14	Enhanced Direct Memory Access - Channel 11	eDMA-TCD-CH11
4000_F000	5000_F000	15	Enhanced Direct Memory Access - Channel 12	eDMA-TCD-CH12
4001_0000	5001_0000	16	Enhanced Direct Memory Access - Channel 13	eDMA-TCD-CH13
4001_1000	5001_1000	17	Enhanced Direct Memory Access - Channel 14	eDMA-TCD-CH14
4001_2000	5001_2000	18	Enhanced Direct Memory Access - Channel 15	eDMA-TCD-CH15
4001_3000	5001_3000	19	External Watchdog Monitor 0	EWM0
4001_4000	5001_4000	20	Miscellaneous System Control Module	MSCM
4001_5000	5001_5000	21	Secure Miscellaneous System Control Module	SMSCM
4001_6000	5001_6000	22	System Power Controller 0	SPC0
4001_7000	5001_7000	23	System Performance Monitor	SYSPM
4001_8000	5001_8000	24	Trigger Multiplexer 0	TRGMUX0
4001_9000	5001_9000	25	Wake-Up Unit 0	WUU0
4001_A000	5001_A000	26	Watchdog 0	WDOG0
4001_B000	5001_B000	27	Watchdog 1	WDOG1
Clock modules				
4001_C000	5001_C000	28	Module Reset and Clock Control Module 0	MRCC0
4001_D000	5001_D000	29	Signal Frequency Analyzer 0	SFA0

Table continues on the next page...

Table 17. Peripheral Bridge2 (continued)

Base address (non-secure)	Base address (secure)	Slot	Module	
4001_E000	5001_E000	30	System Clock Generator 0	SCG0
4001_F000	5001_F000	31	32 kHz Clock Control Module	CCM32K
Memory modules				
4002_0000	5002_0000	32	Flash Memory Unit 0	FMU0
4002_1000	5002_1000	33	Register File 0	REGFILE0
4002_2000	5002_2000	34	Register File 1	REGFILE1
Security modules				
4002_3000	5002_3000	35	Cyclic Redundancy Check 0	CRC0
4002_4000	5002_4000	36	Messaging Unit	ELEMU
4002_5000	5002_5000	37	FMC with Non-Volatile Memory Prince XOR-Encrypt-XOR	NPX
4002_6000	5002_6000	38	Trusted Resource Domain Controller - Manager	TRDC-MGR
4002_7000	5002_7000	39	Trusted Resource Domain Controller - MBC 0	TRDC-MBC0
4002_8000	5002_8000	40	Trusted Resource Domain Controller - MBC 1	TRDC-MBC1
4002_9000	5002_9000	41	Trusted Resource Domain Controller - MBC 2	TRDC-MBC2
4002_A000	5002_A000	42	Trusted Resource Domain Controller - Memory Region Control 0	TRDC-MRC0
System modules				
4002_B000	5002_B000	43	Smart Power Switch Domain Controller 0	VBAT0
Timer modules				
4002_C000	5002_C000	44	Real Time Clock	RTC
4002_D000	5002_D000	45	Low-Power Timer 0	LPTMR0
4002_E000	5002_E000	46	Low-Power Timer 1	LPTMR1
4002_F000	5002_F000	47	Low-Power Periodic Interrupt Timer 0	LPIT0
4003_0000	5003_0000	48	Time Stamp Timer 0	TSTMR0
4003_1000	5003_1000	49	Timer/PWM Module 0	TPM0
4003_2000	5003_2000	50	Timer/PWM Module 1	TPM1
Communication modules				
4003_3000	5003_3000	51	Low-Power Inter-Integrated Circuit 0	LPI2C0
4003_4000	5003_4000	52	Low-Power Inter-Integrated Circuit 1	LPI2C1
4003_5000	5003_5000	53	Improved Inter-Integrated Circuit	I3C
4003_6000	5003_6000	54	Low-Power Serial Peripheral Interface 0	LPSPI0
4003_7000	5003_7000	55	Low-Power Serial Peripheral Interface 1	LPSPI1

Table continues on the next page...

Table 17. Peripheral Bridge2 (continued)

Base address (non-secure)	Base address (secure)	Slot	Module	
4003_8000	5003_8000	56	Low-Power Universal Asynchronous Receiver/ Transmitter 0	LPUART0
4003_9000	5003_9000	57	Low-Power Universal Asynchronous Receiver/ Transmitter 1	LPUART1
4003_A000	5003_A000	58	Flexible Input/Output 0	FlexIO0
4003_B000	5003_B000	59 - 62	Controller Area Network 0	CAN0
4003_F000	5003_F000	63	Semaphore2	SEMA42
Radio modules				
4004_0000	5004_0000	64	Radio Mode Controller	RFMC
4004_1000	5004_1000	65	Data Stream Buffer 0	DSB0
HMI modules				
4004_2000	5004_2000	66	Port Control Module A	PORTA
4004_3000	5004_3000	67	Port Control Module B	PORTB
4004_4000	5004_4000	68	Port Control Module C	PORTC
4004_5000	5004_5000	69	Port Control Module D	PORTD
4004_6000	5004_6000	70	General Purpose Input/Output D	GPIOD
Analog modules				
4004_7000	5004_7000	71	Analog-to-Digital Converter 0	ADC0
4004_8000	5004_8000	72	Low-Power Comparator 0	LPCMP0
4004_9000	5004_9000	73	Low-Power Comparator 1	LPCMP1
4004_A000	5004_A000	74	Voltage Reference 0	VREF0

3.6 Fast peripherals

3.6.1 Fast peripherals 0 memory map

The following table shows the memory map of the fast peripherals in this device.

Table 18. Fast Peripherals 0

Base address (non-secure)	Base address (secure)	Slot	Module	
DBG Modules				
4800_0000	5800_0000	0	ISP-AP Debug Module 0	ISP-AP0
HMI Modules				
4801_0000	5801_0000	1	General Purpose Input/Output A	GPIOA

Table continues on the next page...

Table 18. Fast Peripherals 0 (continued)

Base address (non-secure)	Base address (secure)	Slot	Module	
4802_0000	5802_0000	2	General Purpose Input/Output B	GPIOB
4803_0000	5803_0000	3	General Purpose Input/Output C	GPIOC
4804_0000	5804_0000	4-127	Reserved	—

3.6.2 Fast peripherals 1 memory map

The following table shows the memory map of the fast peripherals in this device.

Table 19. Fast Peripherals 1

Base address (non-secure)	Base address (secure)	Modules	
RF Modules			
4880_0000	5880_0000	Radio Flash/IFR	RF-Flash
4890_0000	5890_0000	Reserved	—
4894_8000	5894_8000	Radio NBU CIU2	RF-CIU2
4898_0000	5898_0000	Radio FRO192M	RF-FRO192M
4898_1000	5898_1000	Radio Flash Memory Unit	RF-FMU
4898_2000	5898_2000	Radio FMC CFG	RF-FMCCFG
4898_3000	5898_3000	Radio CMC	RF-CMC
4898_4000	5898_4000	Radio Low-Power Timer	TPM2
4898_5000	5898_5000	Radio Flash Memory Unit - Test	RF-FMU-TST
4898_6000	5898_6000	Reserved	—
489C_0000	589C_0000	Radio NBU SMU2 0	RF-NBU-SMU2
48A0_0000	58A0_0000	Reserved	—
48A0_1000	58A0_1000	Radio Zigbee Link Layer	RF-ZigbeeLL
48A0_2000	58A0_2000	Radio Generic LinkLayer	RF-GENLL
48A0_3000	58A0_3000	Radio Generic LinkLayer Remapping 0	RF-GENLL-REMAP0
48A0_4000	58A0_4000	Radio Generic LinkLayer Remapping 1	RF-GENLL-REMAP1
48A0_5000	58A0_5000	Radio Generic LinkLayer Remapping 2	RF-GENLL-REMAP2
48A0_6000	58A0_6000	Rdio Control	RF-CTRL
48A0_7000	58A0_7000	Radio Transceiver	RF-XCVR
48A0_8000	58A0_8000	Radio Packet RAM	RF-Packet-RAM

Table continues on the next page...

Table 19. Fast Peripherals 1 (continued)

Base address (non-secure)	Base address (secure)	Modules	
48A0_A000	48A0_A000	Reserved	—
48A1_0000	48A1_0000	Reserved	—

3.7 PPB memory map

The following table shows the memory map of the Private Peripheral Bus (PPB) in this device.

Table 20. Private Peripheral Bus - Internal

Base	Slot	Module	Alias
DBG Modules			
E000_0000	0	Instrumentation Trace Macrocell 0	ITM
E000_1000	1	Data Watchpoint and Trace 0	DWT
E000_2000	2	Breakpoint Unit 0	BPU
E000_3000	3-13	Reserved	—
E000_E000	14	Secure System Control Space 0	Secure SCS
E000_F000	15-45	Reserved	—
E002_E000	46	Non-secure System Control Space 0	Non-secure SCS
E002_F000	47-63	Reserved	—

Table 21. Private Peripheral Bus - External

Base	Slot	Module	Alias
DBG Modules			
E004_0000	0	Trace Port Interface Unit 0	TPIU
E004_1000	1	Reserved	—
E004_2000	2	Cross Trigger Interface	CTI
E004_3000	3	Reserved	—
E004_4000	4	Funnel	FUNNEL
E004_5000	5-63	Reserved	—
E008_0000	64	Miscellaneous Control Module 0	MCM
E008_1000	65-190	Reserved	—
E00F_F000	191	ROM Table 0	ROM

Chapter 4
Low Power Cache Controller (LPCAC)

4.1 Chip-specific LPCAC information

Table 22. Reference links to related information

Topic	Related module	Reference
Full description	LPCAC	LPCAC
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing
Additional Cache Control and Status	MCM SMSCM MSCM	Miscellaneous Control Module (MCM) Secure Miscellaneous System Control Module (SMSCM) Miscellaneous System Control Module (MSCM)

4.1.1 Module instances

This device has one instance of the LPCAC module.

4.1.2 Additional cache control and status

In this device, the primary controls for the cache from outside the LPCAC include :

- Miscellaneous Control Module (MCM):
  - ISCR[CWBEE] - Should always set as 0 to disable the interrupt for cache write buffer error
  - ISCR[CWBER] - This field indicates whether the cache write buffer error is received.
  - ISCR[CPEE] - This field enables cache parity error reporting.
  - ISCR[CPES] - This field indicates whether the cache parity error is detected.
  - FATR register - This register indicates the faulting attributes, when a properly-enabled cache write buffer error interrupt event is detected.
  - FADR register - This register captures the faulting address when a properly-enabled cache write buffer error interrupt event is detected.
  - FDR register - This register indicates the captured faulting data, when a properly-enabled cache write buffer error interrupt event is detected.
  - CPCR2 register - This register configures code cache features.
- Secure Miscellaneous System Control Module (SMSCM):
  - OCMDR0[OCMCF2] bitfield has settings related to cache
- Miscellaneous System Control Module (MSCM):
  - There are status registers related to cache

4.2 Overview

LPCAC provides low-latency access to instructions or data. This decouples processor performance from system memory performance, increasing bus availability for other modules and improving system performance.

LPCAC has a 32-bit datapath AMBA-AHB input bus and a 32-bit datapath AMBA-AHB output bus.

4.2.1 Block diagram

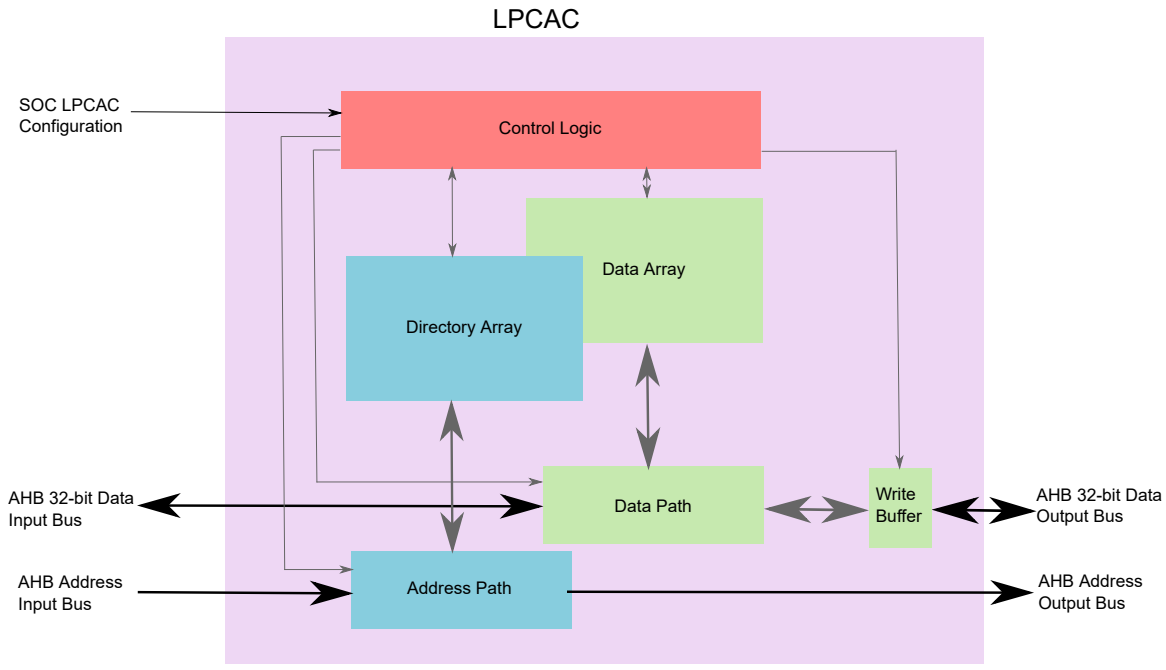


Figure 6. Block diagram

4.2.2 Features

LPCAC supports the following features:

- Nonblocking and write-through cache mode
- 8 KB total cache size
- Cache organization as an 8-way, 4-set-associative design based on 256-byte superpages
- Each way or superpage contains up to 16 sequential 16-byte pages
- Cache supports optional odd parity with one bit of odd parity per 4 bytes of cache data.
  - Cache can recover and scrub data parity errors by forcing miss on parity error and reloading data

4.3 Functional description

This section provides further information on the LPCAC module operation.

A reset signal clears and enables this module.

LPCAC examines every valid AHB input access. If the access hits the cache memory region and the cache is enabled, the access goes to the cache portion of the module. If the access is not cacheable, the access is passed directly to the AHB output bus.

The LPCAC cache has a 256-byte line subdivided into 16 16-byte sublines:

- Cache read accesses that are cache misses perform an aligned 16-byte subline-size burst cache read miss to the module's AHB output bus. Then the cache loads the needed data in the cache’s data storage. The cache miss uses a 4-beat (32 bits per beat), wrapped burst bus access to fetch the cache miss data.
- Cache read accesses that are cache hits return the desired read data from the cache.
- Cache write accesses that are cache misses perform the desired write operation only to the module's AHB output bus (for write-through mode accesses, AHB_LPCAC has a no allocation on write-miss policy).
- Cache write accesses that are cache hits perform the desired write operation to the cache and the module's AHB output bus.

LPCAC has a one entry write buffer. When enabled, cache write accesses with a bufferable attribute use the buffer. When enabled and available, the buffer allows write from the processor to receive an immediate (zero wait state) bus termination.

4.3.1 Cache functional description

The LPCAC cache is an 8-way, 4-set-associative, 256-byte per line write-through design. It supports a total cache capacity of 8 KB for a 32-bit wide cache miss datapath.

4.3.1.1 Cache controls

This module has controls to enable, disable, and clear the cache.

The cache control inputs are as follows:

- clr_lpcac: clear lpcac
- dis_lpcac: disable lpcac
- dis_lpcac_wtbf: disable write buffer
- lim_lpcac_wtbf: limit write buffer
- frc_no_alloc: force no allocation
- mode_ctl_hprot: ignore LPCAC Memory Regions
- parity_miss_en: enable parity, miss on parity error
- parity_fault_en: enable parity error reporting

See chip-specific section for details on how these signals are connected or controlled on a given device.

4.3.2 Clocks

The core clock domain defines the module's clock domain.

4.3.3 Reset

A reset signal clears and enables this module. See the chip-integration information for the resets that affect LPCAC.

4.3.4 Interrupts

This module has no interrupts.

4.4 Signal descriptions

This module has no external signals.

4.5 Memory regions and input control description

4.5.1 Memory regions

LPCAC decodes cacheable address regions. The cache memory region may be composed of one or more disjointed memory regions and the total size may be larger than the cache storage.

Any address not in the cacheable memory regions is in the pass-through memory region. All accesses to the pass-through memory regions are non-cacheable. For all accesses to the cacheable memory regions, the access is cacheable if the attributes of the access indicate that it can be cached, else the access is non-cacheable.

The following table defines the memory map.

Table 23. Memory map

Cache memory range	Cache memory range	Cache total memory address space
0000_0000h–07FF_FFFFh (128 MB)	1000_0000h–1FFF_FFFFh (256 MB)	384 MB

4.5.2 Input controls

The LPCAC does not have a module-resident programming model. An external block supplies the needed LPCAC control inputs.

Table 24. Control inputs for LPCAC

Function	Control input	Description
clear lpcac	clr_lpcac	One cycle active-high pulse clears the lpcac.
disable lpcac	dis_lpcac	• 0 = lpcac enabled (reset configuration) • 1 = lpcac disabled, all cacheable accesses pass from LPCAC input to output bus
disable write buffer	dis_lpcac_wtbf	• 0 = write buffer enabled • 1 = write buffer disabled
limit write buffer	lim_lpcac_wtbf	• 0 = If write buffer is enabled, buffer all writes to spaces that are bufferable • 1 = If write buffer is enabled, buffer all writes to spaces that are both bufferable and cacheable
force no allocation	frc_no_alloc/entry	When frc_no_alloc is asserted and the cache is enabled, all accesses to the cache search the cache. If they hit a valid entry that was loaded before frc_no_alloc was asserted, they operate normally. That is, read hits return cache data without going through the cache and write hits update cache data and write through the cache. If they miss, they bypass the cache (that is, even though a read access is to a cacheable space, it does not allocate on a cache miss). This control input is useful for debug. Say there is a software breakpoint, halting the processor. Then, using the debug port, cacheable memory as well as other address spaces are accessed. These debug accesses go through the processor, through the cache, and to the rest of the system. The idea is during the debug accesses the cache is placed on frc_no_alloc mode. In this way, debug access through the core and then through the cache do not disturb the state of the cache. Before restarting the processor, frc_no_alloc mode is negated. When the processor restarts, the cache state is the same or at least very similar as the cache state when the breakpoint was hit. • 0 = normal allocation on cache miss • 1 = no allocation on cache miss

Table continues on the next page...

Table 24. Control inputs for LPCAC (continued)

Function	Control input	Description
ignore CACHE_MAP	mode_ctl_hprot	• 0 = use LPCAC Memory Regions to determine if an access to a given address may be cached. Then, if the attributes of the access indicate it can be cached, the access is cached. • 1 = Use only the attributes of the access to indicate it can be cached.
parity miss enable	parity_miss_en	Enables miss on parity to scrub parity error and to reload the data line that had the parity error. • 0 = do not take a miss on a parity error • 1 = force a miss on a parity error
parity enable	parity_fault_en	Enables and disables parity error report. • 0 = disable • 1 = enable

Chapter 5
Peripheral Bridge2 (PBRIDGE2)

5.1 Chip-specific PBRIDGE2 information

Table 25. Reference links to related information

Topic	Related module	Reference
Full description	PBRIDGE	PBRIDGE
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

5.1.1 Module instances

This device has one PBRIDGE2.

5.2 Overview

Peripheral Bridge2 (PBRIDGE2) Controller converts an AMBA AHB interface to a peripheral interface. The peripheral interface has peripheral module enables allowing the controller to interface to multiple peripherals.

PBRIDGE2 has a configurable datapath width (32-bit or 64-bit) for the AMBA AHB input bus and 8 Bit, 16 Bit, and 32 Bit datapath peripheral output buses. Each peripheral bridge occupies a configurable amount of the address space, which is divided into either 128 4 Kbyte slots or 128 64 Kbyte slots. (It is possible that all the peripheral slots are not used. See the memory map chapter for details on slot assignments.)

5.2.1 Features

The PBRIDGE2 key features include:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width (See Memory Maps for details on slot datapath width assignments)
- Includes separate clock enable inputs for each of the slots to accommodate slower peripherals.
- Includes separate powered-down inputs for each of the slots to accommodate powering-down individual peripherals.

5.2.2 Number of Peripheral Bridges

This device contains 1 peripheral bridges.

5.3 Functional Description

This section provides further information on the PBRIDGE2 operation.

The peripheral bridge functions as a bus protocol translator between an AMBA AHB bus and the slave peripheral bus. The peripheral bridge manages all transactions destined for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

The PBRIDGE2 examines every valid AHB input access. If the access hits a peripheral slot that is configured, not powered-down, and the access width is less than or equal to the slot data width, the access goes to that slot peripheral bus. If the access completes error free on the slot peripheral bus, the AHB access completes error free. If the AHB access is to a peripheral slot that is not configured, is powered-down, or has width greater than the slot data width or the access to the slot completes with an error on the slot peripheral bus, the AHB access terminates with a bus error.

5.3.1 Access Support

Aligned 32-bit, 16-bit, and byte accesses are supported for 32-bit peripherals, aligned 16-bit and byte accesses are supported for 16-bit peripherals, and byte accesses are supported for 8-bit peripherals. All accesses are performed with a single transfer. All accesses to the peripheral slots must be sized less than or equal to the designated peripheral slot size. If an access is attempted that is larger than the targeted port, an AHB bus error response is generated.

5.4 Memory Maps

The peripheral bridges are used to access the registers of most of the peripheral modules on this device. See the memory map section for the memory slot assignment for each module.

Chapter 6
Crossbar Switch (AXBS)

6.1 Chip-specific AXBS information

Table 26. Reference links to related information

Topic	Related module	Reference
Full description	AXBS	AXBS
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

6.1.1 Module instances

This device has one instance of the AXBS module, AXBS0.

6.1.2 Crossbar switch assignment

The masters connected to the crossbar switch are assigned as follows:

Table 27. Crossbar master assignment

Master port number	Master module
M0	CM33 Code Cache
M1	CM33 Data Cache
M2	eDMA
M3	EdgeLock secure enclave
M4	Data steam
M5	NBU

Table 28. Crossbar slave assignment

Slave port number	Slave module
S0	FMC, FMU
S1	CTCM
S2	STCM2, STCM3
S3	STCM0, STCM1, STCM4
S4	STCM5
S5	PBRIDGE2
S6	GPIOA, GPIOB, GPIOC
S7	Radio

6.2 Overview

This section provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows all bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave. A variety of bus arbitration methods and attributes may be programmed on a slave-by-slave basis.

6.2.1 Features

- Symmetric crossbar bus switch implementation
  - Allows concurrent access from different masters to different slaves
  - Slave arbitration attributes configured on a slave-by-slave basis
- Single-clock 32-bit transfer
- Support for burst transfers of 64 bits of data
- Support for low-power park mode
- 32-bit AHB crossbar bus switch compatible with ARM's AMBA Specification v2.0

6.3 Memory map and register definition

Each slave port of the crossbar switch contains configuration registers. Read- and write transfers require two bus clock cycles. The registers can be read from and written to only in supervisor mode. Additionally, these registers can be read from or written to only by 32-bit accesses.

A bus error response is returned if an unimplemented location is accessed within the crossbar switch.

The CRS*n* and PRS*n* registers can be programmed as read-only to prevent changes to their configuration. After being read-only protected, future writes to them terminate with a data storage error.

NOTE

This section shows the registers for all eight master and slave ports. If a master or slave is not used on this particular chip, then unexpected results occur when writing to its registers. See the chip configuration details for the exact master and slave assignments for your chip.

All references to the crossbar switch registers are based on the physical port connections.

6.3.1 AXBS register descriptions

6.3.1.1 AXBS memory map

AXBS0 base address: 4000_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Priority Slave Registers (PRS0)	32	RW	0054_3210h
10h	Control Register (CRS0)	32	RW	0000_0000h
100h	Priority Slave Registers (PRS1)	32	RW	0054_3210h
110h	Control Register (CRS1)	32	RW	0000_0000h
200h	Priority Slave Registers (PRS2)	32	RW	0054_3210h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
210h	Control Register (CRS2)	32	RW	0000_0000h
300h	Priority Slave Registers (PRS3)	32	RW	0054_3210h
310h	Control Register (CRS3)	32	RW	0000_0000h
400h	Priority Slave Registers (PRS4)	32	RW	0054_3210h
410h	Control Register (CRS4)	32	RW	0000_0000h
500h	Priority Slave Registers (PRS5)	32	RW	0054_3210h
510h	Control Register (CRS5)	32	RW	0000_0000h
600h	Priority Slave Registers (PRS6)	32	RW	0054_3210h
610h	Control Register (CRS6)	32	RW	0000_0000h
700h	Priority Slave Registers (PRS7)	32	RW	0054_3210h
710h	Control Register (CRS7)	32	RW	0000_0000h
800h	Master General Purpose Control Register (MGPCR0)	32	RW	0000_0000h
900h	Master General Purpose Control Register (MGPCR1)	32	RW	0000_0000h
A00h	Master General Purpose Control Register (MGPCR2)	32	RW	0000_0000h
B00h	Master General Purpose Control Register (MGPCR3)	32	RW	0000_0000h
C00h	Master General Purpose Control Register (MGPCR4)	32	RW	0000_0000h
D00h	Master General Purpose Control Register (MGPCR5)	32	RW	0000_0000h

6.3.1.2 Priority Slave Registers (PRS0 - PRS7)

Offset

Register	Offset
PRS0	0h
PRS1	100h
PRS2	200h
PRS3	300h
PRS4	400h
PRS5	500h
PRS6	600h
PRS7	700h

Function

The priority slave registers(PRS*n*) set the priority of each master port on a per slave port basis and reside in each slave port. The priority register can be accessed only with 32-bit access. After the CRS*n*[RO] bit is set, the PRS*n* register can only be read; attempts to write to it have no effect on PRS*n* and result in a bus-error response to the master initiating the write.

Two available masters must not be programmed with the same priority level. Attempts to program two or more masters with the same priority level result in a bus-error response and the PRS*n* is not updated.

NOTE

Valid values for the M*n* priority fields depend on which masters are available on the chip. This information can be found in the chip-specific information for the crossbar.

- If the chip contains fewer than three masters, only one bit is valid.
- If the chip contains fewer than five masters, only two bits are valid.
- If five or more masters are present, all three bits of the priority field are used.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved			0	Reserved			0	M5			0	M4		
W		-				-										
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	M3			0	M2			0	M1			0	M0		
W																
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-24 —	Reserved
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22-20 M5	Master 5 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
19 —	Reserved
18-16 M4	Master 4 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
15 —	Reserved
14-12 M3	Master 3 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8the or lowest priority when accessing the slave port.
11 —	Reserved
10-8 M2	Master 2 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8the or lowest priority when accessing the slave port.
7 —	Reserved
6-4 M1	Master 1 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port. 001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or lowest priority when accessing the slave port.
3 —	Reserved
2-0 M0	Master 0 Priority This field sets the arbitration priority for this port on the associated slave port. 000b - This master has level 1 or highest priority when accessing the slave port.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - This master has level 2 priority when accessing the slave port. 010b - This master has level 3 priority when accessing the slave port. 011b - This master has level 4 priority when accessing the slave port. 100b - This master has level 5 priority when accessing the slave port. 101b - This master has level 6 priority when accessing the slave port. 110b - This master has level 7 priority when accessing the slave port. 111b - This master has level 8 or the lowest priority when accessing the slave port.

6.3.1.3 Control Register (CRS0 - CRS7)

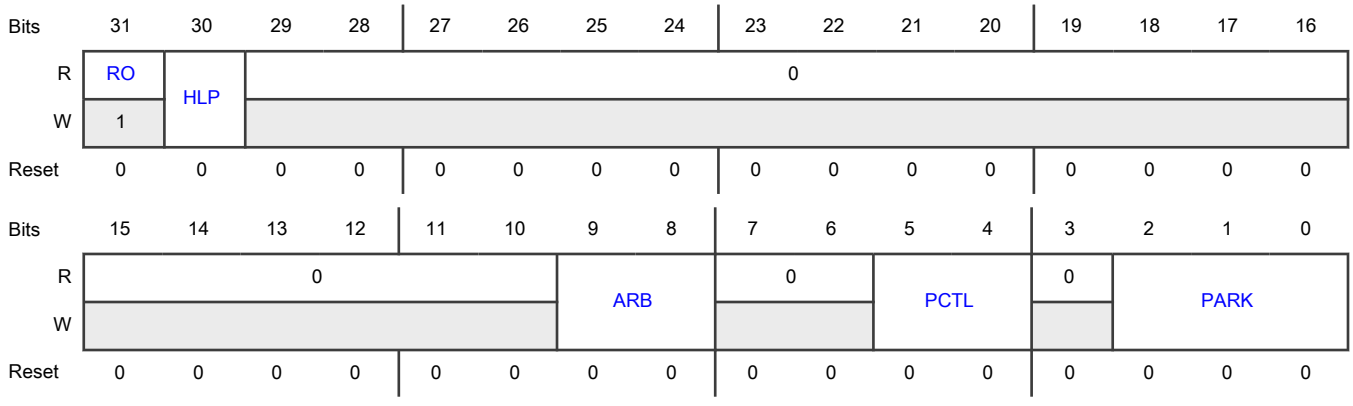
Offset

Register	Offset
CRS0	10h
CRS1	110h
CRS2	210h
CRS3	310h
CRS4	410h
CRS5	510h
CRS6	610h
CRS7	710h

Function

These registers control several features of each slave port and must be accessed using 32-bit accesses. After CRS*n*[RO] is set, the PRS*n* can only be read; attempts to write to it have no effect and resuresultsan error response.

Diagram



Fields

Field	Function
31 RO	Read Only This field forces the slave port's CSR*n* and PRS*n* registers to be read-only. Aftersettingt, only a hardware reset clears it. 0b - The slave port's registers are writeable. 1b - The slave port's registers are read-only and cannot be written. Attempted writes do not affect the registers and result in a bus error response.
30 HLP	Halt Low Priority This field sets the initial arbitration priority for low-power mode requests. Setting this bit will not affect the request for low-power mode from attaining the highest priority after it has control of the slave ports. 0b - The low-power mode request has the highest priority for arbitration on this slave port. 1b - The low-power mode request has the lowest initial priority for arbitration on this slave port.
29-10 —	Reserved
9-8 ARB	Arbitration Mode This field selects the arbitration policy for the slave port. 00b - Fixed priority 01b - Round-robin(RR) or rotating priority 10b - Reserved 11b - Reserved
7-6 —	Reserved
5-4 PCTL	Parking Control This field determines the slave port's parking control. The low-power park feature results in overall power saving if the slave port is not saturated. However, this forces an extra latency clock when a master tries to access the slave port when not in use because it is not parked on any master. 00b - When no master makes a request, the arbiter parks the slave port on the master port defined by the PARK field. 01b - When no master makes a request, the arbiter parks the slave port on the last master to be in control of the slave port. 10b - When no master makes a request, the slave port is not parked on a master and the arbiter drives all outputs to a constant safe state. 11b - Reserved
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 PARK	Park This field determines which master port the current slave port parks on when no masters are actively making requests and the PCTL bits are cleared. **NOTE** Select only those master ports that are present on the chip. Otherwise, undefined behavior may occur. 001b - Park on master port M1. 010b - Park on master port M2. 011b - Park on master port M3. 100b - Park on master port M4. 101b - Park on master port M5. 110b - Park on master port M6. 111b - Park on master port M0.

6.3.1.4 Master General Purpose Control Register (MGPCR0 - MGPCR5)

Offset

Register	Offset
MGPCR0	800h
MGPCR1	900h
MGPCR2	A00h
MGPCR3	B00h
MGPCR4	C00h
MGPCR5	D00h

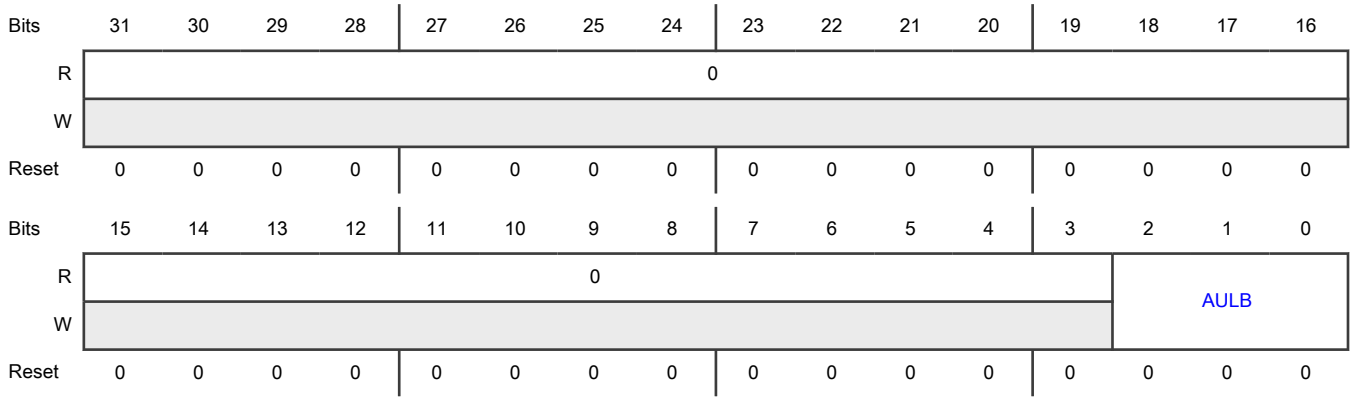
Function

The MGPCR controls only whether the master's undefined length burst accesses are allowed to complete uninterrupted or whether they can be broken by requests from higher priority masters. The MGPCR can be accessed only in supervisor mode with 32-bit accesses.

NOTE

If there are fewer than eight master ports, only the registers associated with those masters are present. Register addresses associated with master ports that are not present are reserved. See the module's chip-specific information for which master ports are present on this module.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 AULB	Arbitrates On Undefined Length Bursts This field determines whether, and when, the crossbar switch arbitrates away the slave port the master owns when the master is performing undefined length burst accesses. 000b - No arbitration is allowed during an undefined length burst. 001b - Arbitration is allowed at any time during an undefined length burst. 010b - Arbitration is allowed after four beats of an undefined length burst. 011b - Arbitration is allowed after eight beats of an undefined length burst. 100b - Arbitration is allowed after 16 beats of an undefined length burst. 101b - Reserved 110b - Reserved 111b - Reserved

6.4 Functional description

Information about general operation and arbitration are provided in this section.

6.4.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device does not know whether it owns the slave port it is targeting. The master waits while it does not have control of the slave port it is targeting.

After the master acquires control of the slave port, it controls the port until it relinquishes the port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port. However, if the master is running a fixed- or undefined-length burst transfer, it retains control of the slave port until that transfer completes. Based on MGPCR[AULB], the master either retains control of the slave port when doing undefined-length, incrementing burst transfers or loses the bus to a higher-priority master.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it can park the slave port on the master port indicated by CRS*n*[PARK]. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port. The slave port can also be put into low-power park mode to save power, by using CRS*n*[PCTL].

6.4.2 Register coherency

The operation of the crossbar is affected as soon as a register is written. The values of the registers do not track with slave-port-related master accesses, but instead track only with slave accesses.

The MGPCRx[AULB] bits are the exception to this rule. The update of these bits is only recognized when the master on that master port runs an idle cycle, even though the slave bus cycle to write them will have already terminated successfully. If the MGPCRx[AULB] bits are written between two burst accesses, the new AULB encodings do not take effect until an IDLE cycle is initiated by the master on that master port.

6.4.3 Arbitration

The crossbar switch supports the following arbitration algorithms:

- Fixed priority
- Round-robin

The arbitration scheme is independently programmable for each slave port.

6.4.3.1 Arbitration during undefined length bursts

Arbitration points during an undefined length burst are defined by the current master's MGPCR[AULB] field setting. When a defined length is imposed on the burst via the AULB bits, the undefined length burst is treated as a single or series of single back-to-back fixed-length burst accesses.

The following figure illustrates an example:

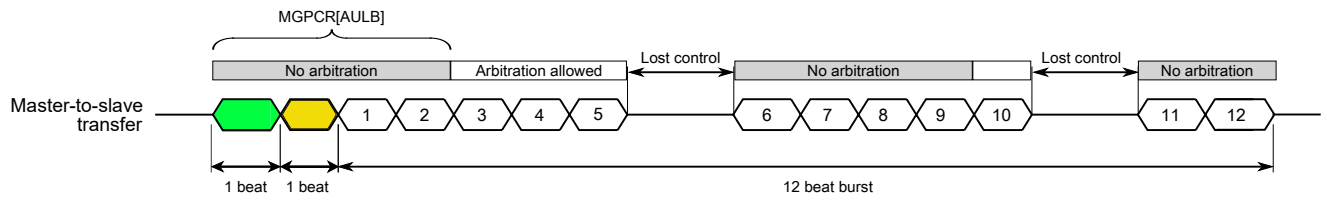


Figure 7. Undefined length burst example

In this example, a master runs an undefined length burst and the MGPCR[AULB] bits indicate arbitration occurs after the fourth beat of the burst. The master runs two sequential beats and then starts what will be a 12-beat undefined length burst access to a new address within the same slave port region as the previous access. The crossbar does not allow an arbitration point until the fourth overall access, or the second beat of the second burst. At that point, all remaining accesses are open for arbitration until the master loses control of the slave port.

Assume the master loses control of the slave port after the fifth beat of the second burst. After the master regains control of the slave port no arbitration point is available until after the master has run four more beats of its burst. After the fourth beat of the now continued burst, or the ninth beat of the second burst from the master's perspective, is taken, all beats of the burst are once again open for arbitration until the master loses control of the slave port.

Assume the master again loses control of the slave port on the fifth beat of the third now continued burst, or the 10th beat of the second burst from the master's perspective. After the master regains control of the slave port, it is allowed to complete its final two beats of its burst without facing arbitration.

NOTE

Fixed-length burst accesses are not affected by the AULB bits. All fixed-length burst accesses lockout arbitration until the last beat of the fixed-length burst.

6.4.3.2 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level in the priority registers (PRS*n*). If two masters request access to the same slave port, the master with the highest priority in the selected priority register gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing access from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port.

Table 29. Methods of how the crossbar switch grants control of a slave port to a master

When	Then the crossbar switch grants control to the requesting master
Both of the following are true: • The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	At the next clock edge
Both of the following are true: • The current master is running a fixed-length burst transfer or a locked transfer. • The requesting master's priority level is higher than that of the current master.	At the end of the burst transfer or locked transfer
Both of the following are true: • The current master is running an undefined length burst transfer. • The requesting master's priority level is higher than that of the current master.	At the next arbitration point for the undefined length burst transfer **NOTE** Arbitration points for an undefined length burst are defined in the MGPCR for each master.
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

6.4.3.3 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requests the master owns the slave bus at the next transfer boundary, accounting for locked and fixed-length burst transfers. Priority is based on how far ahead the ID of the requesting master is of the ID of the last master.

After a master is granted access to a slave port, a master may perform as many transfers as desired to that port until another master requests the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle, if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume that the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and masters 0, 4, and 5 make simultaneous requests, they are serviced in this order: 4,5, and then 0.

Parking may continue to be used in a round-robin mode, but does not affect the round-robin pointer unless the parked master performs a transfer. Handoff occurs to the next master in line after one cycle of arbitration. If the slave port is put into low-power park mode, the round-robin pointer is reset to point at master port 0, giving it the highest priority.

6.4.3.4 Clocking

This module has no clocking considerations.

6.4.3.5 Interrupts

This module has no interrupts.

6.4.3.6 Priority assignment

Each master port must be assigned a unique 3-bit priority level. If an attempt is made to program multiple master ports with the same priority level within the priority registers (PRS*n*), the crossbar switch responds with a bus error and the registers are not updated.

6.5 External signals

This module has no external signals.

6.6 Initialization/application information

No initialization is required for the crossbar switch.

Hardware reset ensures that all the register bits used by the crossbar switch are properly initialized to a valid state. However, the following settings and priorities may be programmed to achieve the maximum system performance:

- During the configuration of the crossbar switch, all other masters must be idle.
- To prevent reconfiguration of the crossbar switch, write 1 to CRS*n*[RO].

6.7 Glossary

AMBA	Advanced Microcontroller Bus Architecture
IDLE	A type of transfer that a master uses when it does not want to perform a data transfer
ID	Master port number

Chapter 7
Flash Memory Controller (FMC)

7.1 Chip-specific FMC information

Table 30. Reference links to related information

Topic	Related module	Reference
Full description	FMC	FMC
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

See SMSCM_OCMDR0 for details on flash speculation and flash cache control.

NOTE

The FMC module includes an optional PRINCE encryption/decryption features that is not described here. For details on this feature, refer to the Security RM.

7.1.1 Module instances

This device has one instance of the FMC module.

7.2 Overview

The Flash Memory Controller (FMC) is a memory interface and acceleration unit providing:

- An interface between the device and the nonvolatile memory
- Buffers that can accelerate flash memory transfers

The FMC manages the interface between the device and the flash memory. The FMC receives status information describing the configuration of the memory and uses this information to ensure a proper interface. The next table shows the supported read/write operations.

Flash memory type	Read	Write
Program, IFR, IFR1 flash memory	8-bit, 16-bit, and 32-bit reads	16-byte and 128-byte writes

The FMC provides quick access to flash memory using 3 separate acceleration mechanisms:

- A flash-phrase-sized buffer holds the most recently accessed flash phrase
- An optional flash-phrase-sized speculation buffer can prefetch the next flash phrase
- An optional 4-way, set-associative cache can store previously accessed flash phrases

The FMC is controlled by a programmer's model external to the FMC module; see the chip-specific FMC information for details. The FMC's programming model provides a very configurable, high performance flexible memory controller, which can be optimized for the runtime characteristics of specific applications.

NOTE

Program the FMC's controls only while the flash controller is idle. Changing the configuration settings while a flash access is in progress can cause non-deterministic, unpredictable behavior.

7.2.1 Features

- Interface between the device and the flash memory:
  - The FMC's input bus supports 8-bit, 16-bit, and 32-bit read operations to flash memory.
  - The FMC's flash memory interface fetches a 128-bit flash phrase.
  - For input read requests, the FMC fetches a flash phrase with the desired read data from flash memory.
  - The FMC's flash memory interface can write aligned 16-byte flash write phrases or aligned 128-byte flash write pages to flash memory.
  - The FMC has a 16-byte aligned write buffer, used once for aligned 16-byte flash write phrases or used 8 times for aligned 128-byte flash write pages.
  - The FMC's input bus supports 32-bit write operations for flash memory writes to fill the FMC's write buffer.
  - For input write requests, the FMC must receive the 4-word write of an aligned phrase in order.
- Acceleration of data transfer from flash memory to the device:
  - A flash-phrase-sized buffer that holds the current decrypted flash phrase fetched due to a FMC read request. Subsequent FMC read requests *that hit in the current buffer* return data with no wait states.
  - A flash-phrase-sized prefetch speculation buffer with controls for prefetching on instructions and/or data reads. When prefetching is enabled, idle FMC-to-flash interface cycles are used to fetch the next sequential flash phrase and hold it in the prefetch buffer. Subsequent FMC read requests *that hit in the speculation buffer* return data with no wait states.
  - Input controls:
    - to disable data type speculation
    - to disable all speculation
    - to invalidate the current and speculation buffers
  - The flash cache has input controls:
    - to disable instruction caching
    - to disable operand caching
    - to disable all caching
    - to clear the cache
  - The size of the flash cache in bytes is calculated as follows:

    flash cache size = [number of ways] x [number of sets] x [flash phrase size (in bytes)]

    For example, a flash cache with 4 ways, 1 set, and a 128-bit flash phrase (= 16 bytes) has a total flash cache size = 64 bytes

    (4 ways) x (1 set) x (16 bytes per flash phrase) = 64 bytes, the size of the flash cache

    NOTE

    Clear the speculation buffer and flash cache before accessing recently modified flash addresses. The flash cache has a specific clear control bit. To clear the speculation buffer, first disable then re-enable the speculation.

7.3 Functional description

The FMC is a flash interface and acceleration unit, with flexible buffers for user configuration.

- The FMC's input bus can operate faster than the flash memory.
- The FMC-to-flash interface has flow control to add wait states as needed (for input bus reads that need flash accesses).

- The FMC also contains various configurable buffers that hold recent flash accesses. If an input bus read hits a valid buffer, then that access will complete with no wait states.

7.3.1 Modes of operation

The FMC only operates when a bus master accesses the flash memory.

For any device power mode where the flash memory cannot be accessed, the FMC is disabled.

7.3.2 Default configuration

After system reset, the FMC is configured to provide a significant level of buffering for transfers from the flash memory. For all banks:

- The current and speculation buffers are cleared by reset.
- Prefetch support for data and instructions is enabled.
- The cache is cleared by reset.
- The cache is configured for data or instruction replacement.

7.3.3 Configuration options

The default configuration provides a high degree of flash acceleration, however, advanced users may want to customize the FMC buffer configurations, to maximize throughput for their use cases. When reconfiguring the FMC for custom use cases, do not program the FMC's control registers while the flash memory is being accessed. Instead, change the control registers with a routine executing from RAM in supervisor mode.

The FMC's cache and buffering controls allow the tuning of resources to suit specific application requirements. The cache and buffer are each controlled individually. The controls enable buffering and prefetching per access type (instruction fetch or data reference).

As an application example: if both instruction fetches and data references are accessing flash memory, then control is available to send instruction fetches, data references, or both to the cache or the single-entry buffer. Likewise, speculation can be enabled or disabled for either type of access.

For best performance, the FMC cache and speculation buffering should be enabled for instruction and/or data fetching. The following is recommended for best performance:

- If the speculation buffer is enabled for both instruction and data speculation, also enable the flash cache for both instruction and data caching.
- If the speculation buffer is enabled for instruction speculation only, also enable the flash cache for at least instruction caching.

7.3.4 Wait states

Because the core, crossbar switch, and bus masters can be clocked at a higher frequency than the flash clock, flash memory accesses that do not hit in the speculation buffer or cache usually require wait states.

All wait states and synchronization delays are handled automatically by the Flash Memory Controller. No direct user configuration is required (or even allowed) to set up the flash wait states.

7.3.5 Speculative reads

The FMC has a single buffer that reads ahead to the next phrase in the flash memory if there is an idle cycle. Speculative prefetching is programmable for instruction and/or data accesses. Because many code accesses are sequential, using the speculative prefetch buffer improves performance in most cases.

When speculative reads are enabled, the FMC immediately requests the next sequential phrase address after a read completes. By requesting the next phrase immediately, speculative reads can help to reduce or even eliminate wait states when accessing sequential code and/or data.

7.3.6 Interrupts

This module has no interrupts.

7.4 External signals

The FMC has no external signals.

7.5 Initialization and application information

The FMC does not require user initialization. Flash acceleration features are enabled by default.

The FMC has no visibility into flash memory erase and program cycles because the Flash Memory module manages them directly. As a result, if an application is executing flash memory commands, the FMC's current buffer, speculation buffer, and cache might need to be disabled and/or flushed to prevent the possibility of returning stale data.

Chapter 8
System Performance Monitor (SYSPM)

8.1 Chip-specific SYSPM information

Table 31. Reference links to related information

Topic	Related module	Reference
Full description	SYSPM	SYSPM
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

8.1.1 Module instances

This device has one instance of the SYSPM module.

8.2 Overview

System Performance Monitor (SYSPM) is a memory mapped peripheral that enables the user to monitor system and/or CPU events. The SYSPM consists of sixteen 256 Byte sub-slots (SS). The first subslot is the configuration slot. The registers in the configuration SS0 describe the number of monitors, types of monitors and number of counters per monitor. Subsequent sub-slots are occupied by an optional Programmable System Activity Monitor (PSAM), and Performance Monitors (PERFMON) registers.

8.2.1 Features

The SYSPM includes the following features:

- Three 40-bit event counters
- Programmable event select
- Countable events that include cache, branch fetches, and stalls

Memory mapped performance monitor that counts instructions, cache, branch, IPS, and TCM stats

8.2.2 Block diagram

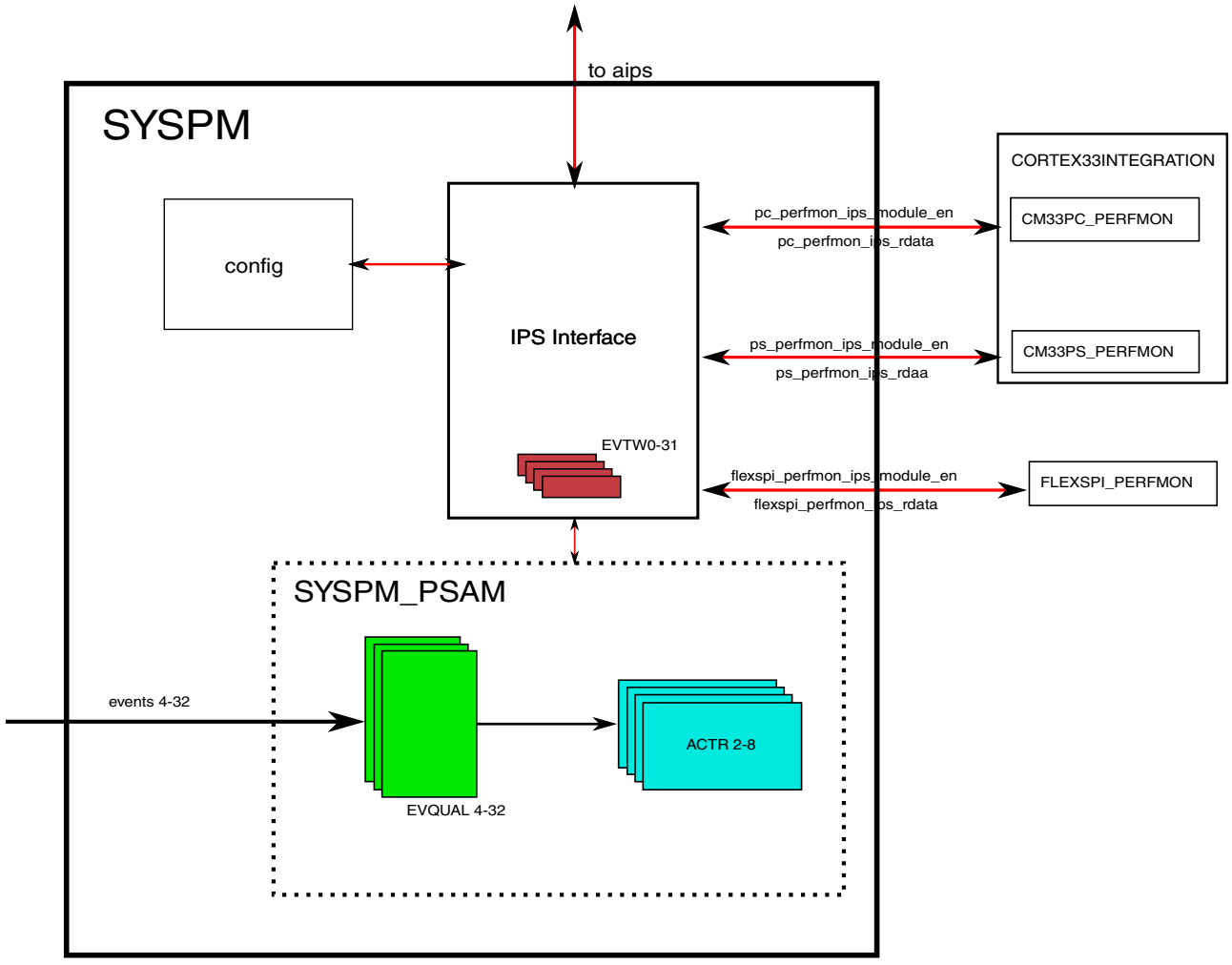


Figure 8. SYSPM Block Diagram

8.3 Functional description

8.3.1 Operating modes

There are no special operating modes for the SYSPM.

8.3.2 Operations

SYSPM routes transaction to CMX_PERFMON. See the chip-specific SYSPM information for details on the performance monitors. SYSPM also provides read-only identification registers that describe the types of monitors in each sub-slot.

8.3.3 Clocking

This module has no clocking considerations.

8.3.4 Interrupts

This module has no interrupts.

8.3.5 External signals

This module has no external signals.

8.3.6 Initialization

Initialization is not required for this module.

8.4 Memory Map and register definition

This section includes the SYSPM module memory map and detailed descriptions of all registers.

8.4.1 SYSPM register descriptions

8.4.1.1 SYSPM memory map

SYSPM base address: 4001_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Configuration 0 (CFGSS0)	32	R	FF00_00FFh
4h	Configuration 1 (CFGSS1)	32	R	0000_0000h
8h	Configuration 2 (CFGSS2)	32	R	0103_0002h
Ch	Configuration 3 (CFGSS3)	32	R	0203_0002h
200h	Performance Monitor Control Register (PMCR0)	32	RW	0000_0000h
218h	Performance Monitor Event Counter (PMECTR1_HI_0)	8	R	00h
21Ch	Performance Monitor Event Counter (PMECTR1_LO_0)	32	R	0000_0000h
220h	Performance Monitor Event Counter (PMECTR2_HI_0)	8	R	00h
224h	Performance Monitor Event Counter (PMECTR2_LO_0)	32	R	0000_0000h
228h	Performance Monitor Event Counter (PMECTR3_HI_0)	8	R	00h
22Ch	Performance Monitor Event Counter (PMECTR3_LO_0)	32	R	0000_0000h
300h	Performance Monitor Control Register (PMCR1)	32	RW	0000_0000h
318h	Performance Monitor Event Counter (PMECTR1_HI_1)	8	R	00h
31Ch	Performance Monitor Event Counter (PMECTR1_LO_1)	32	R	0000_0000h
320h	Performance Monitor Event Counter (PMECTR2_HI_1)	8	R	00h
324h	Performance Monitor Event Counter (PMECTR2_LO_1)	32	R	0000_0000h
328h	Performance Monitor Event Counter (PMECTR3_HI_1)	8	R	00h
32Ch	Performance Monitor Event Counter (PMECTR3_LO_1)	32	R	0000_0000h

8.4.1.2 Configuration 0 (CFGSS0)

Offset

Register	Offset
CFGSS0	0h

Function

This register describes the type of monitor, its hardware revision level, number of implemented counters in the monitor.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSC								NCTRS							
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HRL								ID							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 MSC	Miscellaneous • If ID = 0000_0001b, MSC = number of events • If ID = 0000_0010b, — MSC = location identifier — MSC = 00b - CM0+ — MSC = 01b - CMx Code — MSC = 10b - CMx Sys — MSC = 11b- FlexSPI • If ID 0000_0011b, MSC = 11b = Configuration slot • If ID = 0000_0011b, MSC = 11b • For any other ID value, MSC = 00b
23-16 NCTRS	Number of Counters
15-8	Hardware revision level

Table continues on the next page...

Table continued from the previous page...

Field	Function
HRL	
7-0 ID	Identifier 0000_0000b - CFGSS not present 0000_0001b - CFGSS PSAM configuration 0000_0010b - CFGSS PERFMON configuration 0000_0011b - CFGSS Configuration

8.4.1.3 Configuration 1 (CFGSS1)

Offset

Register	Offset
CFGSS1	4h

Function

This register describes the type of monitor, its hardware revision level, number of implemented counters in the monitor and the location of the monitor (if it is a PERFMON).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSC								NCTRS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HRL								ID							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MSC	Miscellaneous • If ID = 0000_0001b, MSC = number of events • If ID = 0000_0010b,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— MSC = location identifier — MSC = 00b - CM0+ — MSC = 01b - CMx Code — MSC = 10b - CMx Sys — MSC = 11b- FlexSPI • If ID 0000_0011b, MSC = 11b = Configuration slot • If ID = 0000_0011b, MSC = 11b • For any other ID value, MSC = 00b
23-16 NCTRS	Number of Counters
15-8 HRL	Hardware revision level
7-0 ID	Identifier 0000_0000b - CFGSS not present 0000_0001b - CFGSS PSAM configuration 0000_0010b - CFGSS PERFMON configuration 0000_0011b - CFGSS Configuration

8.4.1.4 Configuration 2 (CFGSS2)

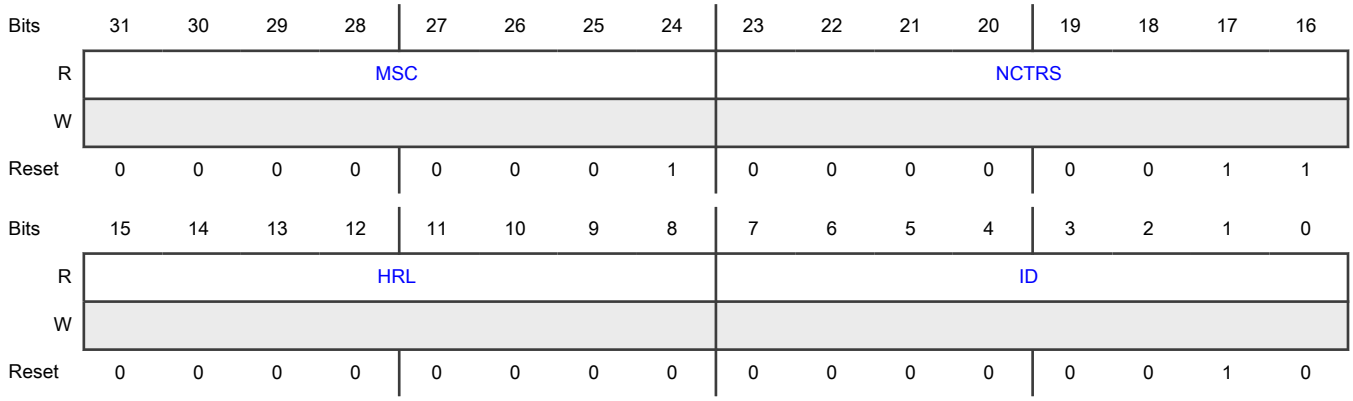
Offset

Register	Offset
CFGSS2	8h

Function

This register describes the type of monitor, its hardware revision level, number of implemented counters in the monitor.

Diagram



Fields

Field	Function
31-24 MSC	Miscellaneous • If ID = 0000_0001b, MSC = number of events • If ID = 0000_0010b, — MSC = location identifier — MSC = 00b - CM0+ — MSC = 01b - CMx Code — MSC = 10b - CMx Sys — MSC = 11b- FlexSPI • If ID 0000_0011b, MSC = 11b = Configuration slot • If ID = 0000_0011b, MSC = 11b • For any other ID value, MSC = 00b
23-16 NCTRS	Number of Counters
15-8 HRL	Hardware revision level
7-0 ID	Identifier 0000_0000b - CFGSS not present 0000_0001b - CFGSS PSAM configuration 0000_0010b - CFGSS PERFMON configuration 0000_0011b - CFGSS Configuration

8.4.1.5 Configuration 3 (CFGSS3)

Offset

Register	Offset
CFGSS3	Ch

Function

This register describes the type of monitor, its hardware revision level, number of implemented counters in the monitor.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSC								NCTRS							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HRL								ID							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-24 MSC	Miscellaneous • If ID = 0000_0001b, MSC = number of events • If ID = 0000_0010b, — MSC = location identifier — MSC = 00b - CM0+ — MSC = 01b - CMx Code — MSC = 10b - CMx Sys — MSC = 11b- FlexSPI • If ID 0000_0011b, MSC = 11b = Configuration slot • If ID = 0000_0011b, MSC = 11b • For any other ID value, MSC = 00b
23-16 NCTRS	Number of Counters
15-8	Hardware revision level

Table continues on the next page...

Table continued from the previous page...

Field	Function
HRL	
7-0 ID	Identifier 0000_0000b - CFGSS not present 0000_0001b - CFGSS PSAM configuration 0000_0010b - CFGSS PERFMON configuration 0000_0011b - CFGSS Configuration

8.4.1.6 Performance Monitor Control Register (PMCR0 - PMCR1)

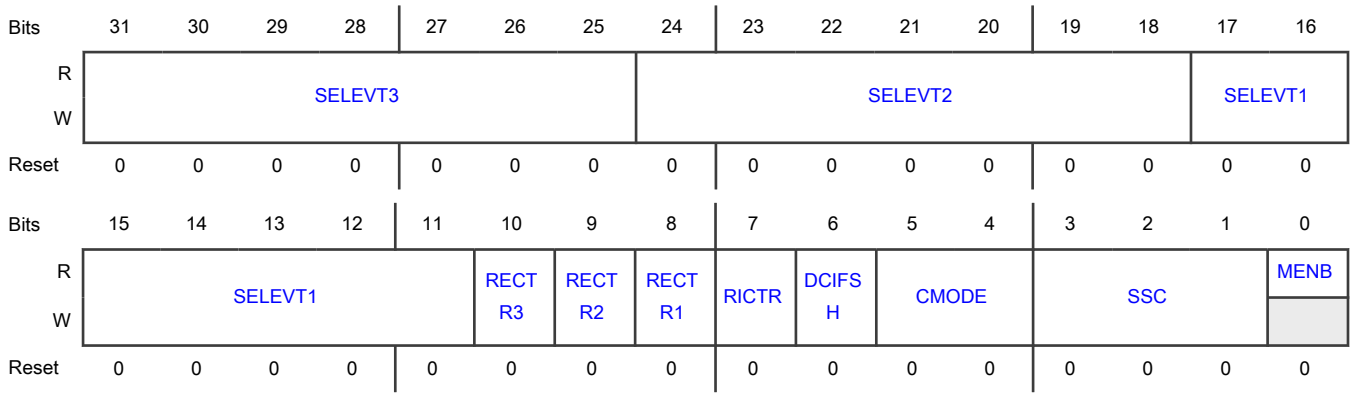
Offset

Register	Offset
PMCR0	200h
PMCR1	300h

Function

The performance monitor control register is used to select which events will be counted in PMCRn register, the count mode, start/stop control and the enables for the counters. See the chip-specific SYSPM information for detailed events.

Diagram



Fields

Field	Function
31-25 SELEVT3	Select Event 3 See Operations to select which event to be counted in PMECTR3.
24-18	Select Event 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
SELEVT2	See Operations to select which event to be counted in PMECTR2.
17-11 SELEVT1	Select Event 1 See Operations to select which event to be counted in PMECTR1.
10 RECTR3	Reset Event Counter 3 Write a one to clear event counter 3. This field must be zero to restart event counter 3. 0b - Counter runs normally 1b - Counter value resets at the end of the cycle
9 RECTR2	Reset Event Counter 2 Write a one to clear event counter 2. This field must be zero to restart event counter 2.
8 RECTR1	Reset Event Counter 1 Write a one to clear event counter 1. This field must be zero to restart event counter 1.
7 RICTR	Resets the Instruction Counter Write a one to clear the instruction counter. This field must be zero to restart the instruction counter. 0b - do not reset the instruction counter 1b - clear the instruction counter
6 DCIFSH	Disable Counters if Stopped or Halted 0b - Continue counting 1b - Stops counting when the CPU is halted
5-4 CMODE	Count Mode Determines if events are counted for privileged mode, user mode, or both privileged and user. This setting affects the operation of all three of the event counters. 00b - count in both user and privileged modes 01b - Reserved 10b - count only in user mode 11b - count only in privileged mode
3-1 SSC	Start/Stop Control This 3-bit field provides a three-phase mechanism to start/stop the counters. It includes a prioritized scheme with local start > local stop > global start > global stop > conditional TSTART > TSTOP. The global and conditional start/stop affect all configured PM/PSAM module concurrently so counters are "coherent". 000b - Idle 001b - local stop

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - local start 011b - local start 100b - Reserved 101b - Reserved 110b - Reserved 111b - Reserved
0 MENB	Module is Enabled This bit signals the performance monitor is enabled. Write a non-zero value to the SSC bit field to start the counter. 0b - Disable the performance monitor. 1b - Enable the performance monitor.

8.4.1.7 Performance Monitor Event Counter (PMECTR1_HI_0)

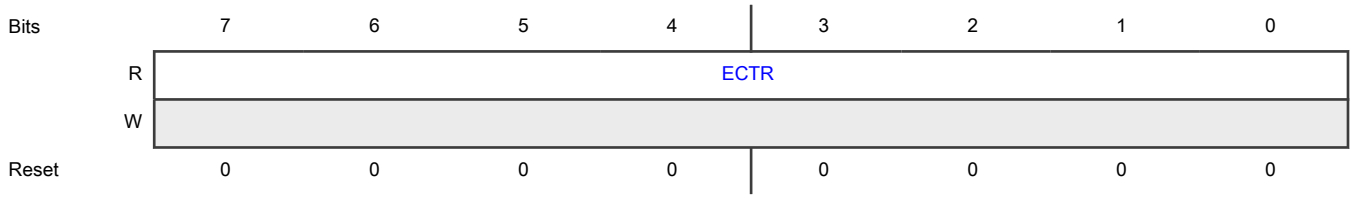
Offset

Register	Offset
PMECTR1_HI_0	218h

Function

{PMECTR1_HI, PMECTR1_LO} is the 40-bit counter that counts the event selected by the PMCR0[SELEVT1] configuration.

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter This is the upper 8-bits of event1 counter. The value in this register increments each time the event selected in PMCR0[SELEVT1] occurs.

8.4.1.8 Performance Monitor Event Counter (PMECTR1_LO_0)

Offset

Register	Offset
PMECTR1_LO_0	21Ch

Function

{PMECTR1_HI, PMECTR1_LO} is the 40-bit counter that counts the event selected by the PMCR0[SELEVT1] configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ECTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ECTR	Event Counter This is the lower 32-bits of event1 counter. The 40-bit value increments each time the event selected in PMCR0[SELEVT1] occurs.

8.4.1.9 Performance Monitor Event Counter (PMECTR2_HI_0)

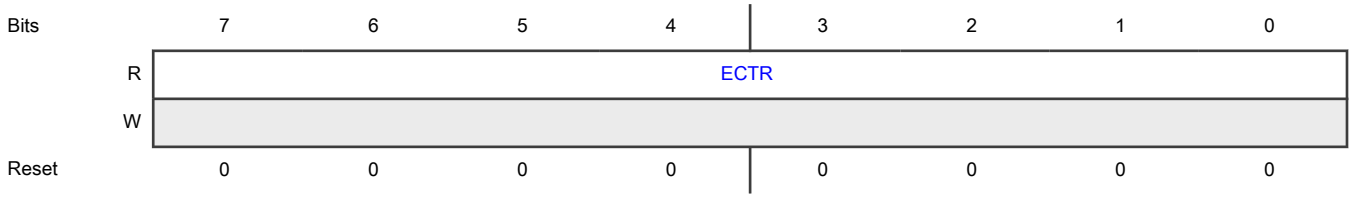
Offset

Register	Offset
PMECTR2_HI_0	220h

Function

{PMECTR2_HI, PMECTR2_LO} is the 40-bit counter that counts the event selected by the PMCR0[SELEVT2] configuration.

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter This is the upper 8-bits of event1 counter. The value in this register increments each time the event selected in PMCR0[SELEVT2] occurs.

8.4.1.10 Performance Monitor Event Counter (PMECTR2_LO_0)

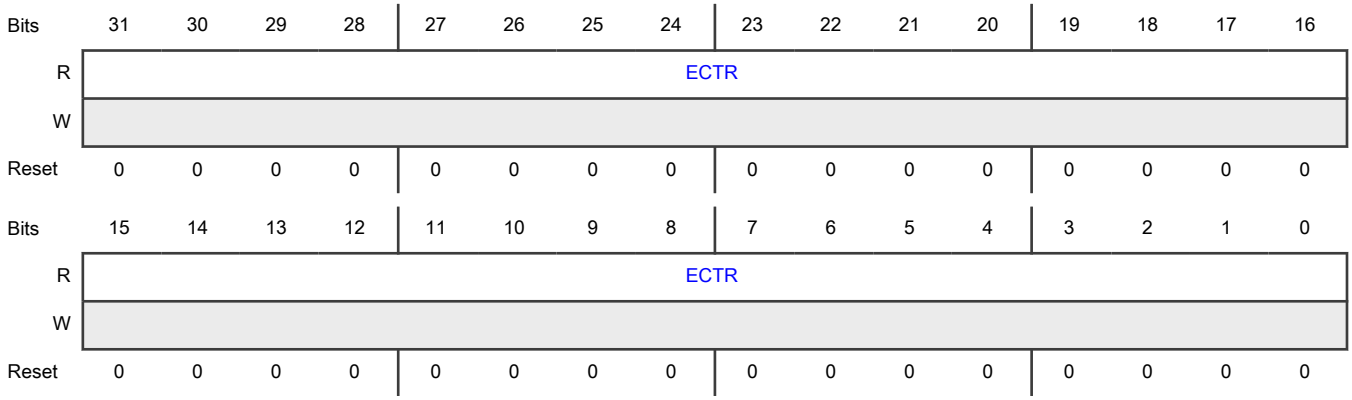
Offset

Register	Offset
PMECTR2_LO_0	224h

Function

{PMECTR2_HI, PMECTR2_LO} is the 40-bit counter that counts the event selected by the PMCR0[SELEVT2] configuration.

Diagram



Fields

Field	Function
31-0 ECTR	Event Counter This is the lower 32-bits of event1 counter. The 40-bit value increments each time the event selected in PMCR0[SELEVT2] occurs.

8.4.1.11 Performance Monitor Event Counter (PMECTR3_HI_0)

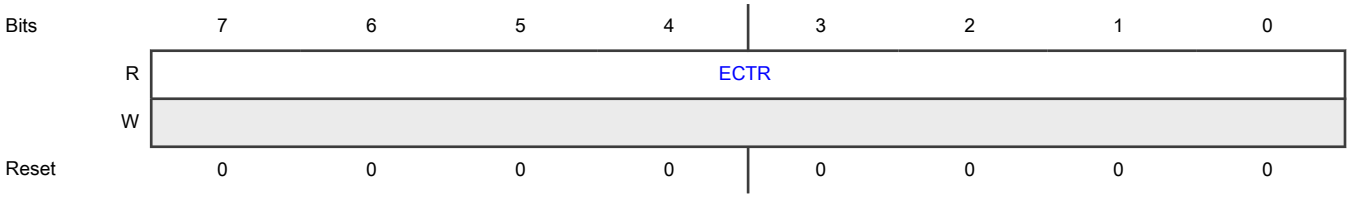
Offset

Register	Offset
PMECTR3_HI_0	228h

Function

{PMECTR3_HI, PMECTR3_LO} is the 40-bit counter that counts the event selected by the PMCR0[SELEVT3] configuration.

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter This is the upper 8-bits of event1 counter. The value in this register increments each time the event selected in PMCR0[SELEVT3] occurs.

8.4.1.12 Performance Monitor Event Counter (PMECTR3_LO_0)

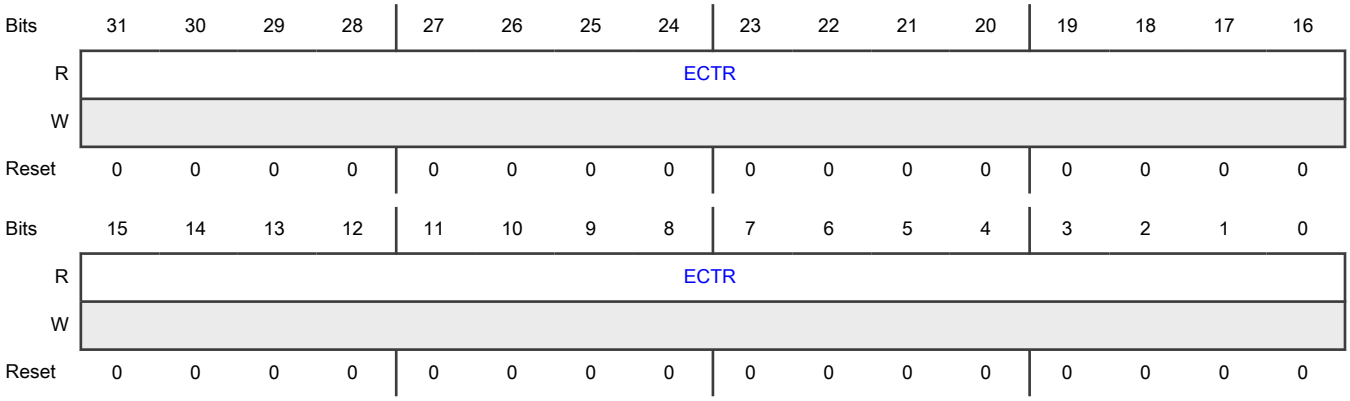
Offset

Register	Offset
PMECTR3_LO_0	22Ch

Function

{PMECTR3_HI, PMECTR3_LO} is the 40-bit counter that counts the event selected by the PMCR0[SELEVT3] configuration.

Diagram



Fields

Field	Function
31-0 ECTR	Event Counter This is the lower 32-bits of event1 counter. The 40-bit value increments each time the event selected in PMCR0[SELEVT3] occurs.

8.4.1.13 Performance Monitor Event Counter (PMECTR1_HI_1)

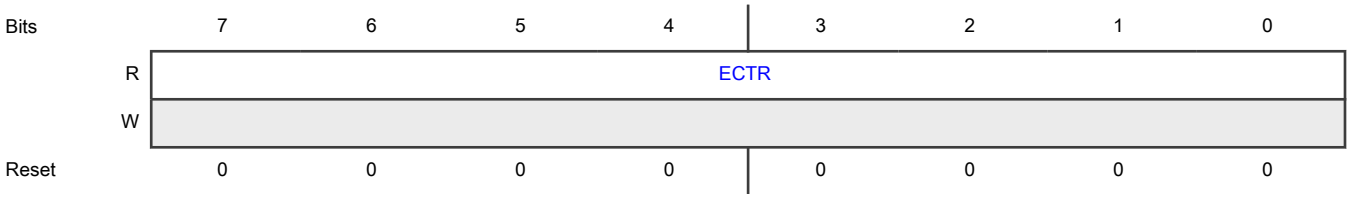
Offset

Register	Offset
PMECTR1_HI_1	318h

Function

{PMECTR1_HI, PMECTR1_LO} is the 40-bit counter that counts the event selected by the PMCR1[SELEVT1] configuration.

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter This is the upper 8-bits of event1 counter. The value in this register increments each time the event selected in PMCR1[SELEVT1] occurs.

8.4.1.14 Performance Monitor Event Counter (PMECTR1_LO_1)

Offset

Register	Offset
PMECTR1_LO_1	31Ch

Function

{PMECTR1_HI, PMECTR1_LO} is the 40-bit counter that counts the event selected by the PMCR1[SELEVT1] configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ECTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ECTR	Event Counter This is the lower 32-bits of event1 counter. The 40-bit value increments each time the event selected in PMCR1[SELEVT1] occurs.

8.4.1.15 Performance Monitor Event Counter (PMECTR2_HI_1)

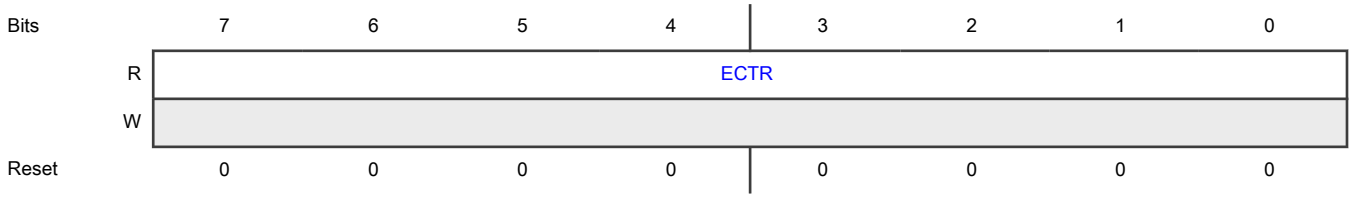
Offset

Register	Offset
PMECTR2_HI_1	320h

Function

{PMECTR2_HI, PMECTR2_LO} is the 40-bit counter that counts the event selected by the PMCR1[SELEVT2] configuration.

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter This is the upper 8-bits of event1 counter. The value in this register increments each time the event selected in PMCR1[SELEVT2] occurs.

8.4.1.16 Performance Monitor Event Counter (PMECTR2_LO_1)

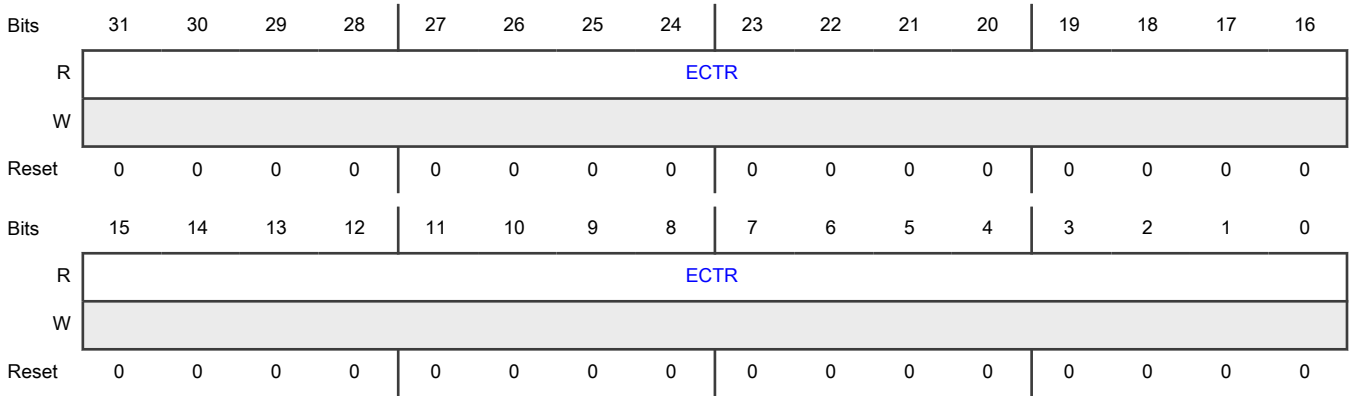
Offset

Register	Offset
PMECTR2_LO_1	324h

Function

{PMECTR2_HI, PMECTR2_LO} is the 40-bit counter that counts the event selected by the PMCR1[SELEVT2] configuration.

Diagram



Fields

Field	Function
31-0 ECTR	Event Counter This is the lower 32-bits of event1 counter. The 40-bit value increments each time the event selected in PMCR1[SELEVT2] occurs.

8.4.1.17 Performance Monitor Event Counter (PMECTR3_HI_1)

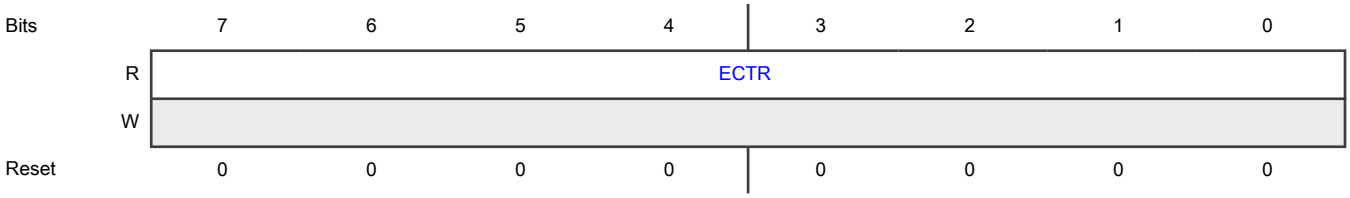
Offset

Register	Offset
PMECTR3_HI_1	328h

Function

{PMECTR3_HI, PMECTR3_LO} is the 40-bit counter that counts the event selected by the PMCR1[SELEVT3] configuration.

Diagram



Fields

Field	Function
7-0 ECTR	Event Counter This is the upper 8-bits of event1 counter. The value in this register increments each time the event selected in PMCR1[SELEVT3] occurs.

8.4.1.18 Performance Monitor Event Counter (PMECTR3_LO_1)

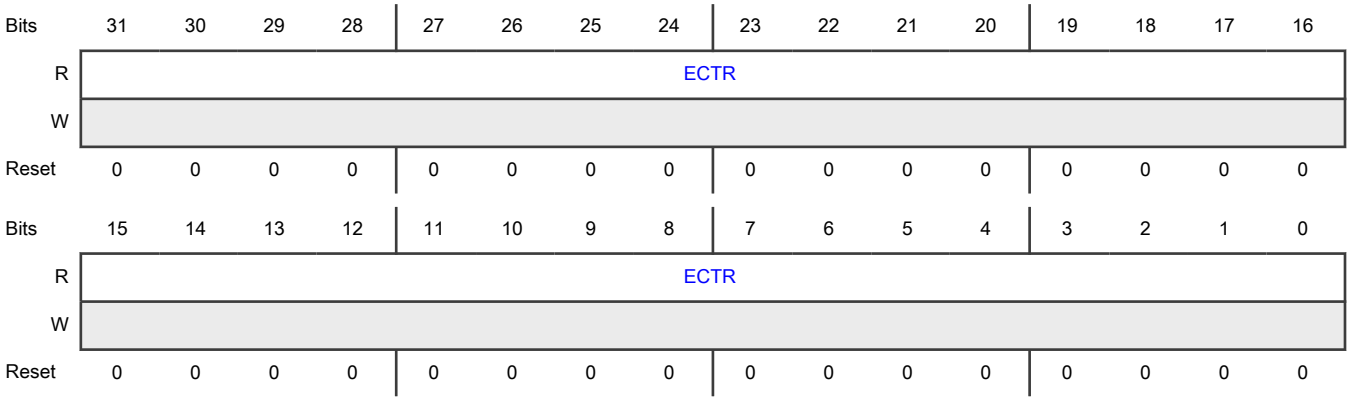
Offset

Register	Offset
PMECTR3_LO_1	32Ch

Function

{PMECTR3_HI, PMECTR3_LO} is the 40-bit counter that counts the event selected by the PMCR1[SELEVT3] configuration.

Diagram



Fields

Field	Function
31-0 ECTR	Event Counter This is the lower 32-bits of event1 counter. The 40-bit value increments each time the event selected in PMCR1[SELEVT3] occurs.

Chapter 9
Trusted Resource Domain Controller (TRDC)

9.1 Chip-specific TRDC information

Table 32. Reference links to related information

Topic	Related module	Reference
Full description	TRDC	TRDC
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

9.1.1 Module instances

This device has one instance of the TRDC module.

9.1.2 TRDC configuration

NOTE

Not all the registers and fields in this chapter are available for this device, see the following table for the available MBC, domain, slave port, and blocks.

Table 33. TRDC slave port configuration

No.	Domain[1]	Slave port No.	Slave memory	Number of block x block size / Regions	Non-secure address
MBC0	Domain0 Domain1 Domain2	SLV0	Flash – 1 MB	32 x 32 KB	0000_0000 – 000F_FFFF
		SLV1	Flash IFR0 – 32 KB	4 x 8 KB	0200_0000 – 0200_7FFF
		SLV2	Flash IFR1 – 8 KB	1 x 8 KB	0210_0000 – 0210_1FFF
		SLV3	ROM – 96 KB	12 x 8 KB	0480_0000 – 0481_7FFF
MBC1	Domain0 Domain1 Domain2	SLV0	CTCM0 – 16 KB (with ECC) CTCM1 – 16 KB (with ECC)	2 x 8 KB	0400_0000 – 0400_3FFF
		SLV1	STCM0 – 16 KB (with ECC) STCM1 – 16 KB (with ECC) STCM2 – 32 KB (with ECC)	8 x 8 KB	2000_0000 – 2000_FFFF
		SLV2	STCM3 – 32 KB STCM4 – 8 KB	5 x 8 KB	2001_0000 – 2001_9FFF
		SLV3	STCM5 – 8 KB (with ECC)	2 x 4 KB	2001_A000 – 2001_BFFF
MBC2	Domain0 Domain1	SLV0	PBRIDGE2	75[2] x 4 KB	4000_0000 – 4007_FFFF

Table continues on the next page...

Table 33. TRDC slave port configuration (continued)

No.	Domain[1]	Slave port No.	Slave memory	Number of block x block size / Regions	Non-secure address
	Domain2	SLV1	Radio Pbridge in Fast Peripheral 1	4 x 64 KB	4800_0000 – 487F_FFFF
		SLV2	NBU part in Fast Peripheral 1	16 x 4 KB	48A0_0000 – 48A0_FFFF
MRC	Domain0 Domain1 Domain2		NBU part	8 regions	4880_0000 – 489F_FFFF

1. The address ranges for each domain are identical. The different domains provide a mechanism for different masters to have different access permissions when accessing a given memory space. The MDA_Wr_m[DID] configures the domain ID for a given master and this is used to select which set of domain access registers to use.
2. See Peripheral Bridge2 (PBRIDGE2) memory map for details.

Table 34. TRDC master port configuration

MDAC#	Master
MDAC0	CM33
MDAC1	eDMA
MDAC2	Data stream buffer
MDAC3	Radio NBU

9.2 Overview

The Trusted Resource Domain Controller (TRDC) provides an integrated, scalable architectural framework for access control, system memory protection and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions and slave peripherals to processing *domains* to support enforcement of robust operational environments. First, each bus mastering resource is assigned to a domain identifier (domainID, DID). Typically, each processor is assigned to a unique domainID and all remaining non-processor bus masters assigned to a different domainID. Next, the access control policies for the individual domains are programmed into any number of registers implemented in the slave memory block and region checkers. Finally, all accesses throughout the device are monitored concurrently to determine the validity of each and every access. If a reference from a given domain has sufficient access rights, it is allowed to continue, else the access is aborted and error information captured.

The access control scheme defined by the TRDC supports a 4-level model, combining the traditional privileged (also known as supervisor) and user modes with an additional signal defining the secure, nonsecure attributes of each memory reference. The result is a 4-level hierarchical access control mechanism, where:

SecurePriv > SecureUser > NonsecurePriv > NonsecureUser

with different access control policies based on read, write and execute references. Combined with the secure/nonsecure and privileged/user attributes, a domainID is associated with every system bus transaction and forms the hardware basis for implementation of the TRDC's access control mechanisms.

9.2.1 Block diagram

As previously noted, the TRDC implementation is distributed across multiple submodules instantiated throughout the device. The TRDC submodules include:

- TRDC_MGR

— The Manager (MGR) submodule coordinates all programming model reads and writes

- TRDC_DAC
  - — The Domain Assignment Controller (DAC) handles resource assignments and generation of the domain identifiers.
- TRDC_MBC
  - — The Memory Block Checker (MBC) implements access controls for on-chip internal memories and slave peripherals based on a fixed-sized block format
- TRDC_MRC
  - — The Memory Region Checker (MRC) implements the access controls for external off-chip memories and peripherals based on the pre-programmed region descriptor registers.

NOTE

TRDC typically consists of various submodules like DAC, MGR, MBC and MRC. However, all the submodules as shown in block digram below may not be present for this configuration. For chip-specific implementation details of this module's instances, see the chip specific information.

See Figure 9 for a simplified TRDC block diagram focusing on topology and connections.

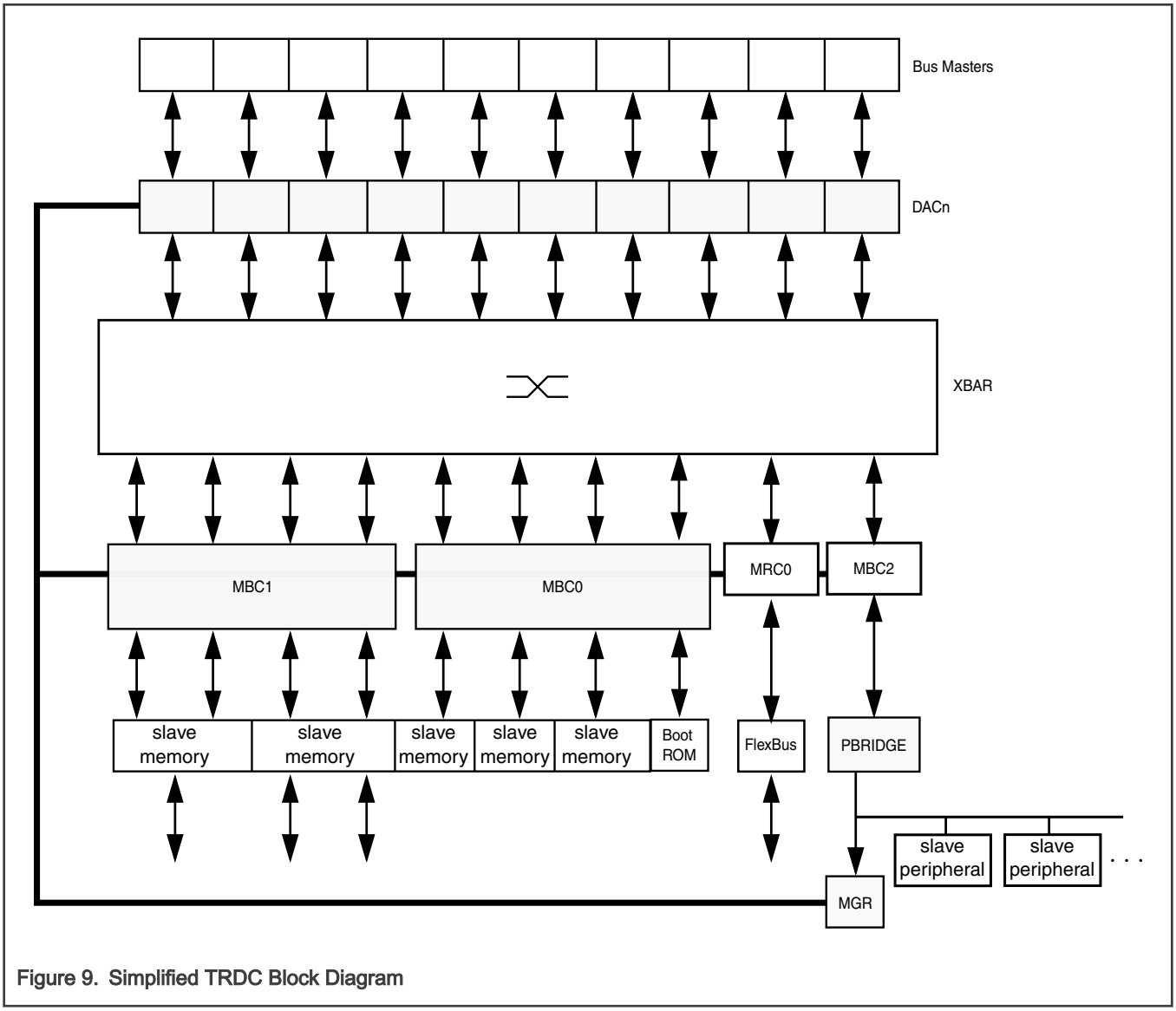


Figure 9. Simplified TRDC Block Diagram

9.2.2 Features

The key features of the TRDC include:

- Assignment of chip resources to processing "domains"
  - Processor cores, non-core bus masters, slave memories and slave peripherals
    - Processors typically assigned to unique domainIDs
    - Non-processor bus masters assigned to a shared domainID
    - Number of supported DIDs is typically (number of CPUs + 1)
  - Each processing domain is assigned a unique domain identifier (domainID, DID)
  - DomainID is an attribute associated with every system bus transaction
  - Used in conjunction with secure/nonsecure, privileged/user attributes
- Access rights to slave targets defined in memory block checkers for on-chip memories and peripherals and memory region checkers for external memories and peripherals
- Built upon a 4-level hierarchical access control model
  - SecurePriv > SecureUser > NonsecurePriv > NonsecureUser
  - Defined by multiple sets of user-programmable R/W/X (Read, Write, eXecute) flags per access state
- Programming model and hardware implementation is distributed across multiple submodules
  - Supports a broad, highly-configurable architecture definition
  - Supports efficient mechanisms for memory space context switch state changes using NSE (Nonsecure Enables) for memory blocks and regions

9.3 Functional description

This section provides more details on the operation and implementation of the TRDC submodules.

9.3.1 Modes of operation

The TRDC module does not support any special modes of operation. As a memory-mapped device located in the core platform's clock domain, it responds based strictly on the memory addresses of the connected system buses. Domain assignment and access control functions are evaluated on a reference-by-reference basis using the addresses and attributes connected to the system bus port(s).

9.3.2 Flash Logical Window

The FLW = Flash Logical Window logic provides a logical window (remapping) between a fixed physical address window and a programmable flash array window. Note that the whole flash image is also available at the normal flash array addresses.

The fixed physical address base and maximum window size is determined per project. The fixed physical address base is available in the read-only FLW Physical Base register. The functional size of the window is configured by programming the window size in the number of 32KByte blocks in the FLW Logical Block Count register. The flash array address base is set in the FLW Array Base register.

When the FLW is disabled, the physical address window is not mapped to any memory storage and all accesses to the window will fault. When the FLW is enabled, the physical address window, up to the size configured by the Logical Block Count, is mapped to the flash array window. When the FLW is enabled, access to the window beyond the size configured by the Logical Block Count will fault.

9.3.3 Manager (TRDC_MGR)

The MGR submodule is responsible for coordinating TRDC's programming model reads and writes.

The complete programming model is large. Additionally, the programming model is distributed across the various submodule instances (DAC, MBC, MRC). The MGR submodule provides the required hardware management for all programming model accesses, generating and distributing the appropriate address decodes and write data to the DAC, MBC and MRC submodule instances, collecting and combining all the register read data buses and responses.

MGR implements the Control Register, all the read-only hardware configuration registers (HWCFG[0-1], DACFGn, MBCFG, MRCFG) and the program-visible versions of the domain error capture registers. To support the domain error reporting functionality, the MGR submodule collects the individual captured error indicator output signals from all the submodule instances and remaps them so they are organized into an instance bitmap by domain as specified by the DERRLOCn register array.

9.3.4 Domain Assignment Controller (TRDC_DAC)

The DAC submodule is responsible for the generation of the domainID on every memory transaction for every bus master in the device. The resulting domainID and {nonsecure, privileged} signals are generated and then treated as address attributes and associated with each transaction as it moves through the system.

As a simple example, assume a system with two domainIDs.

- domainID 0 is for critical tasks.
- domainID 1 is for non-critical tasks.
- Two domainIDs require two MDA_Wm_n registers, MDA_W0 and MDA_W1.

During startup, software initializes the MDA_W[0-1] registers.

1. MDA_W0 = 0x8000_0F81
    - VLD = 1
    - PE = 2
    - DIDS = 0
    - DID = 1
2. MDA_W1 = 0x8000_0FC2
    - VLD = 1
    - PE = 3
    - DIDS = 0
    - DID = 2

A special mode is provided for driving the DID of non-processor masters. This is enabled with the DID bypass (DIDB) bit. The DAC has a DID input that can be driven by a master and used directly as the DID output. This allows variability in the DID as dictated by the master and is intended to be used by DMA masters that masquerade as, (that is, drive the DID of) the master that programmed it.

9.3.5 Access evaluation

Fundamental to the TRDC's operation is the actual access violation check performed by the memory block checker (MBC) and memory region (MRC).

Previous descriptions have detailed the domain assignment mechanisms and the tracking of the domain identifier as an address attribute through the system bus switching fabric(s). As transactions reach the slave memory and peripheral controllers, the address is used to select the appropriate memory region descriptor or block configuration register. Once the appropriate register has been selected, the next function is the actual access evaluation.

For this step, the 3-bit M{B,R}ACSEL field of the selected RGD or BLK_CFG register selects the appropriate GLBAC (global access control policy). The GLBAC register defines the access rights for the domain based on the 4-level model combined with the transaction's access attributes (read = 0/write = 1, secure = 0/nonsecure = 1, execute =1/non-execute=0, user = 0/privileged = 1).

The 4-level model is defined by the concatenation of two address attributes: secure/nonsecure and user/privileged attributes. These two attribute signals (nonsecure, privileged) and the resulting access levels are defined as:

```
if {nonsecure, privileged} == 0b00, then level = SecureUser
if {nonsecure, privileged} == 0b01, then level = SecurePriv(ileged)
if {nonsecure, privileged} == 0b10, then level = NonsecureUser
if {nonsecure, privileged} == 0b11, then level = NonsecurePriv(ileged)
```

The resulting hierarchical access control model specifies:

```
SecurePriv , SecureUser , NonsecurePriv , NonsecureUser
```

After the access evaluation is complete and the access error signal generated, the transaction is allowed to continue if it has sufficient access rights, else the access is aborted with the address and attribute information captured in the appropriate error registers.

9.3.6 Memory Block Checker (TRDC_MBC)

The TRDC's Memory Block Checker provides domain-based access control for all system bus references targeted to on-chip internal memories and slave peripherals. Using programmed block configuration registers which define the access rights per domain and block, the MBC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access rights are allowed to complete, while references that have insufficient rights are terminated with an access error response.

The TRDC architectural framework supports up to 8 MBC instances where each MBC can support up to 4 sub-regions (MEM[s]). Note monitoring of an AXI memory counts as 2 sub-regions since there are independent read and write channels per AXI memory. Conversely, an AHB connection counts as a single subregion.

Each sub-memory is divided up in equal sized blocks. Sub-memory 0 (MEM0) can support up to 512 blocks while sub-memory 1-3 (MEM1-MEM3) can each support up to 64 blocks. The number of blocks and size of the block is defined in MBCm_MEMs_GLBCFG[NBLKS,SIZE_LOG2] register fields, where m is the MBC instance number, and s is the sub-memory number.

Each MBCm_DOMd_BLK_CFG_W contains eight access control structures. Each structure is comprised of a 1-bit NSE (NonSecure Enable) plus a 3-bit MBACSEL (Memory Block Access Control Select). The NSE controls secure/nonsecure access to the corresponding block, while the MBACSEL field selects a MBCm_MEMN_GLBACr register which controls the read/write/execute access to the corresponding block. There are 8 programmable MBCm_MEMn_GLBACr registers per block checker, each containing contain read-write-execute control flags for each of the four operating modes:

```
SPR, SPW, SPX - Secure    Privileged {Read, Write, Execute}
SUR, SUW, SUX - Secure    User       {Read, Write, Execute}
NPR, NPW, NPX - Nonsecure Privileged {Read, Write, Execute}
NUR, NUW, NUX - Nonsecure User       {Read, Write, Execute}
```

GLBAC1-7 also have a lock bit MBCm_MEMn_GLBACr[LK]. When LK=1, the GLBACr register is read-only until the next reset. Furthermore, if MBCm_DOMd_BLK_CFG_W[MBACSEL] selects a GLBAC that is locked, the 3-bit MBCACSEL field is also locked until the next reset. GLBAC0 cannot be locked.

The MBC also has MBCm_DOMd_MEMs_BLK_NSE_W registers. These registers are a bit map of the block NSE bits. Each word supports up to 32 blocks

```
MBCm_DOMd_MEM0_BLK_NSE_W[0-15] has a maximum of 16 words to support 512 blocks
MBCm_DOMd_MEM[1-3]_BLK_NSE_W[0-1] has a maximum of 2 words to support 64 blocks.
```

The block NSE bits can be programmed by writing these registers or by writing the MBCm_MEMN_BLK_{INDEX, SET, CLR, CLR_ALL} registers.

The MBCm_MEMn_NSE_BLK_{INDEX, SET, CLR, CLR_ALL} registers provide a mechanism to efficiently and quickly manipulate the NSE bitmap with bitmasks to set or clear individual blocks within the bitmap and a single operation to perform a clear all across multiple DIDs. The BLK_INDEX register provides the control definition for the BLK_SET and BLK_CLR operations.

9.3.6.1 Memory block hit determination

MBCm_MEMs_GLBCFG[NBLK, SIZE_LOG2] is used to determine which bits of the address correspond to block number.

```
block_n_hit = (addr[MSB:LSB] == n)
where LSB=SIZE_LOG2
MSB=f{NBLKS,LSB}
```

Note that the block hit determination is based only on the address comparison of the first byte being accessed; that is, the MBC does not check the size of the access to make sure it entirely fits within the region.

9.3.6.2 Memory block access evaluation

For a block n hit, the MBC logic evaluates the access rights defined by the MBCm_DOMd_BLK_CFG_Ww registers. The domainID attribute and block number hit selects the appropriate MBCm_DOMd_BLK_CFG_Ww register to use in the access evaluation. The {R,W,X}, nonsecure and privilege attributes of the access are compared with the MBC global access control policy defined in MBCm_DOMd_BLK_CFG_W[MBACSEL, NSE] fields.

While TRDC_CR[GVLDB] = 1, and for each sub-memory being monitored, the MBC performs a reduction-AND of all the individual error terms from each access evaluation.

Unlike the MRC access evaluation and termination, the MBC access evaluation terminates the access with a bus error and reports an access error for only one condition - when the access doesn't have sufficient access rights. By nature, the block checker can only hit in ONE block. And by implementation, there are no "miss" accesses. Accesses outside of the range covered by the MBC are not sent to the MBC. There also aren't any individual block valid bits. When TRDC_CR[GVLDB] = 1, checking on all blocks is always enabled.

9.3.7 Memory Region Checker (TRDC_MRC)

The TRDC's Memory Region Controller provides domain-based, hardware access control for all system bus references targeted at non-peripheral memory spaces. Using pre-programmed *region descriptors* which define memory spaces and their associated access rights per domain identifier, the MRC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with an access error response.

The TRDC architectural framework supports up to 16 MRC instances, where each MRC instance can support [1-16] region descriptors, concurrently monitoring up to 8 slave memory ports. Note, monitoring an AXI protocol port counts as two ports since there are independent read address and write address channels. Conversely, an AHB connection counts as a single port.

A partial, simplified one-dimensional block diagram of an instance of the MRC module is shown in Figure 10. In this figure, a single slave bus port is shown. When the remaining slave ports are considered, the MRC hardware's two-dimensional connection matrix, broadcasting each system bus slave port access across the implemented memory region descriptors, is the resulting structure with the basic access evaluation macro shown as the replicated submodule block. In Figure 10, the system bus fabric slave port is shown on the left, the memory region descriptor registers are in the middle, and the peripheral bus interface on the right side. The evaluation macro contains two magnitude comparators connected to the start and end address registers from each region descriptor as well as the combinational logic blocks to determine the region hit and, based on the domainID associated with the reference, access violations.

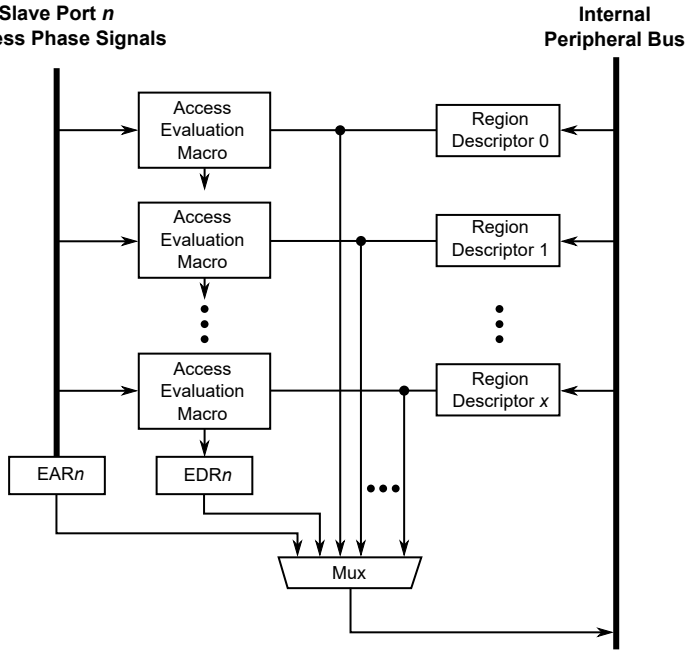


Figure 10. MRC access evaluation macro hardware structure

For each access, the MRC hardware performs a two-step evaluation process. First, the access is compared to all memory region descriptors to determine hit/miss status, and then, using the domainID associated with the reference plus the address attributes {write, nonsecure, privileged}, the access rights are examined using the method detailed in Access evaluation.

9.3.7.1 Memory region descriptor hit determination

To determine if the current reference hits in a given region, two magnitude comparators are used in conjunction with the region's start and end addresses. The boolean equation for this portion of the hit determination is defined as:

```
region_hit_n =
   ((addr >= start_addr) & (addr <= end_addr)) & valid
```

where `addr` is the current system reference address, `start_addr` and `end_addr` are the start and end addresses, and `valid` is the valid bit, all from memory region descriptor n.

Note there are *no hardware checks* to verify that `end_addr ≥ start_addr`, and it is software's responsibility to properly load appropriate values into these fields of the region descriptor.

Also, the region hit determination is based only on an address comparison of the first byte being accessed; that is, the MRC does *not* check the size of the access to make sure it entirely fits within the region.

9.3.7.2 Memory region access evaluation

For each region descriptor hit, the MRC logic evaluates the access rights defined by the MRCm_DOMd_RGDr_Ww registers. Specifically, the domainID attribute selects the appropriate MRCm_DOMd_RGDr_Ww register to use in the access evaluation. The {R,W,X}, nonsecure and privilege attributes of the access are compared with the MRC global access control policy defined in MRCm_DOMd_RGDr_W0[MRACSEL] and MRCm_DOMd_RGDr_W1[NSE] fields. See Access evaluation for additional details on this function.

While CR[GVLDR] = 1, and for each bus slave port being monitored, the MRC performs a **reduction-AND** of all the individual *(no_hit | error)* terms from each access evaluation macro. Recall as specified in Memory region descriptor hit determination, an *invalid* MRC_DOMd_RGDr_Ww (MRCm_DOMd_RGDr_W1[VLD] = 0) forces the region_hit_n evaluation to be negated.

This expression then terminates the bus cycle with an error and reports an access error for three conditions:

1. If the access does not hit in *any* region descriptor, an access error is reported.
2. If the access hits in a single region descriptor and that region signals a domain violation, then an access error is reported.
3. If the access hits in multiple (overlapping) regions and one region signals a violation, then an access error is reported.

The third condition reflects that priority is given to access denying over access allowing for overlapping regions.

Unimplemented domain IDs (DIDs) do not have any associated region descriptors and therefore have no access rights.

9.3.8 Interrupts

This module outputs an interrupt signal which can be connected to the system's interrupt controller. Please check chip specific interrupt assignment for details. Interrupt is asserted on detection of access violation by any checker, and it remains asserted until DERRLOCn registers are cleared

9.4 External signals

The TRDC module does not directly support any external interfaces.

The *internal* interfaces include a standard 32-bit slave bus for all programming model accesses, connections to the address phase signals associated with AHB and/or AXI system buses and connections to the slave peripheral buses as shown in TRDC block diagram.

9.5 Initialization

Out of reset, CR[GVLDR, GVLDB, GVLDM] bits are cleared, TRDC is disabled and in deny by default mode allowing secure privileged startup code to configure the entire programming model. In deny by default mode, only the DIDs that aren't denied by default are allowed to access memory and peripherals. The allowed DIDs are implementation defined. The initialization process typically is performed in 3 steps:

1. Read the various hardware configuration registers (HWCFG{0,1}, DACFGn, MBCm_MEMs_GLBCFG, MRC_GLBCFG) to obtain the implemented hardware capabilities for the device.
2. Using the information retrieved in Step 1 coupled with the desired domain architecture, program all the register data structures for domain assignment (MDA_Wm_n), memory block configuration (MBCm_DOMd_MEMs_BLK_CFG_Ww) and memory region descriptors (MRCm_DOMd_RGDr_Ww). There are individual valid bits included in these registers which typically would be asserted. Additionally, there are also lock mechanisms available if register settings need to be configured and marked as read-only.
3. Once the TRDC programming model is loaded, the CR is written with GVLD {R, B, M} asserted. Upon the assertion of the 3 GVLD bits, deny by default mode is exited and at that point, the TRDC is fully operational. The TRDC remains enabled until the next reset.

9.6 Application information

As described in Overview, the TRDC architecture is intended to provide a broad, highly-capable framework for access control, system memory protection and peripheral isolation. The resulting microarchitecture implementation is distributed throughout the core platform, highly configurable via hardware design parameters and software programmability.

9.6.1 Master domain assignments

The typical use case related to master domain assignments is to include one (or more) processor core(s) in a single domain, possibly coupled with other bus master devices like DMA, etc. The definition of a domain may be static, based simply on the combination of a processor and other optional bus masters. Alternatively, the processor's operation may be configured to dynamically select between a very small number of domains.

For example, "critical" tasks, whether safety-critical, performance-critical, etc. can be grouped together in one domain and all other tasks into a second domain. Non-processor bus masters typically have a single MDA_Wm_n configuration register associated with them. The domainID and {nonsecure, privileged} attributes are usually statically assigned, unless the module can "inherit" attributes from a processor programming it (certain modules like DMAs).

The MDA_Wm_n registers, along with the memory region descriptors and the peripheral domain access control registers all have lock features that allow the register resources to be converted into read-only resources to protect the written configuration.

9.6.2 Memory region descriptor management

There are important concepts to consider in managing the memory region descriptors.

Recall that the association between each MRCn instance and the slave memory regions it monitors is SoC-dependent. The device-specific configuration specifies the number of implemented memory region descriptors in a given instance and the specific port numbers associated with the slave memories being monitored.

9.6.3 Domain error capture management

Recall when a domain access violation is detected by either a memory region checker or a memory block checker, address and attribute information of the offending access is captured. The DERRLOCn read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_n and DERR_W1_n, the DERRLOCn registers provide the instance number details.

These registers are organized as a word array, *indexed by the faulting domain number*, with the contents of each register providing a bitmap signaling the instance number(s) associated with all submodules containing captured error information for that domain.

It is recommended that the exception handler begin by reading the HWCFG1 register to determine its domainID. Next, it uses the just-retrieved domainID to index into the DERRLOCn array. The resulting DERRLOCn value is then examined to determine the instance number of the reporting MBC and/or MRC submodule.

There may be multiple access violations, across multiple instances, pending for a given domain. It is suggested that exception handler use a "find first one" instruction (alternatively known as "count leading zeroes") to quickly and efficiently find the instance number containing access violation details. Once the instance number has been determined, the captured error address (DERR_W0_n) and attribute information (DERR_W1_n) can easily be retrieved from the domain error registers. The 2-bit DERR_W1_n[EST] field provides information signaling whether one or more errors have been detected by the submodule instance. A special write to DERR_W3_n is required to reset (and rearm) the EST error capture state machine.

This process is repeated until all errors associated with a given domainID have been processed.

A graphical representation of this 2-step retrieval of the domain error address and attributes is shown in Figure 11.

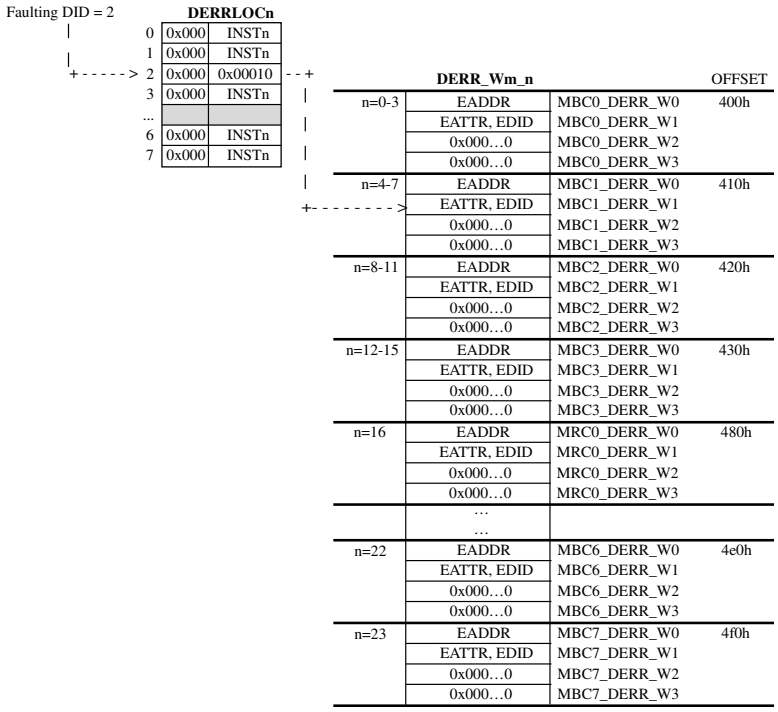


Figure 11. Two-step retrieval of domain error address and attributes

9.7 Register descriptions

If TZ-M is enabled, the TRDC programming model can only be accessed in SecurePrivileged mode. If TZ-M is disabled, it can only be accessed in Privileged mode regardless of Secure or NonSecure;q that is it can't be accessed in User mode. Unless noted otherwise, the programming model registers can be accessed via 8-, 16- or 32-bit reads and 32-bit write references. Attempted accesses in a different operating mode, using unsupported write data sizes, writes to read-only resources, or access to reserved spaces are terminated with an error unless noted otherwise. The TRDC programming model is partitioned into these groups of registers:

- Basic hardware control and configuration
- Domain errors: location and details
- Master domain assignments
- Memory block and region checkers

It should be noted that many of the programming model registers in TRDC are organized as 2-, 3- or 4- dimensional data structures. For the 2-dimensional structures, the generic arrays contain "m" words representing the "columns" and "n" instances of the structure representing the "rows". These may be described as `structure[n][m]`. They appear in the address space of the programming model memory map in the standard C language row-major layout, that is, the "m" words representing the "column" appear sequentially with the entire row replicated "n" times. The TRDC register structure uses the naming convention of *TRDC_<regname>_Wm_n*, where the column number "m" appears as a numeric suffix on the W (32-bit word) identifier, and the row identifier "n" appears as the final numerical suffix.

For the 4-dimensional structures, the generic arrays contain "m", "d", "s" or "r", and "w" indices corresponding to MBC/MRC instance, domain index, sub-memory or region index and word index. The naming convention for the MBC block configuration and MRC region descriptor word registers are: *MBCm_DOMd_MEMs_BLK_CFG_Ww* and *MRCm_DOMd_RGDr_Ww*

For the reset values in this section, a "*" indicates an initial value determined by the hardware configuration.

9.7.1 TRDC register descriptions

9.7.1.1 TRDC memory map

TRDC base address: 4002_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRDC Register (TRDC_CR)	32	RW	See section
F0h	Hardware Configuration Register 0 (TRDC_HWCFG0)	32	R	2103_0403h
F4h	TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)	32	R	See section
100h - 103h	Domain Assignment Configuration Register (DACFG0 - DACFG3)	8	R	See section
140h	Memory Block Configuration Register (MBC0_CFG0)	32	R	3004_3020h
144h	Memory Block Configuration Register (MBC0_CFG1)	32	R	300C_3001h
148h	Memory Block Configuration Register (MBC1_CFG0)	32	R	3008_3002h
14Ch	Memory Block Configuration Register (MBC1_CFG1)	32	R	3002_3005h
150h	Memory Block Configuration Register (MBC2_CFG0)	32	R	3004_304Fh
154h	Memory Block Configuration Register (MBC2_CFG1)	32	R	3000_3010h
158h	Memory Block Configuration Register (MBC3_CFG0)	32	R	0000_0000h
15Ch	Memory Block Configuration Register (MBC3_CFG1)	32	R	0000_0000h
160h - 167h	Memory Region Configuration Register (MRCFG0 - MRCFG7)	8	R	See section
1C0h	TRDC IDAU Control Register (TRDC_IDAU_CR)	32	RW	0000_0008h
1E0h	TRDC FLW Control (TRDC_FLW_CTL)	32	RW	0000_0000h
1E4h	TRDC FLW Physical Base (TRDC_FLW_PBASE)	32	R	0100_0000h
1E8h	TRDC FLW Array Base (TRDC_FLW_ABASE)	32	RW	0000_0000h
1ECh	TRDC FLW Block Count (TRDC_FLW_BCNT)	32	RW	0000_0000h
1FCh	TRDC Fault Domain ID (TRDC_FDID)	32	RW	0000_0000h
200h - 208h	TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC2)	32	R	0000_0000h
400h	MBC Domain Error Word0 Register (MBC0_DERR_W0)	32	R	0000_0000h
404h	MBC Domain Error Word1 Register (MBC0_DERR_W1)	32	R	0000_0000h
40Ch	MBC Domain Error Word3 Register (MBC0_DERR_W3)	32	RW	0000_0000h
410h	MBC Domain Error Word0 Register (MBC1_DERR_W0)	32	R	0000_0000h
414h	MBC Domain Error Word1 Register (MBC1_DERR_W1)	32	R	0000_0000h
41Ch	MBC Domain Error Word3 Register (MBC1_DERR_W3)	32	RW	0000_0000h
420h	MBC Domain Error Word0 Register (MBC2_DERR_W0)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
424h	MBC Domain Error Word1 Register (MBC2_DERR_W1)	32	R	0000_0000h
42Ch	MBC Domain Error Word3 Register (MBC2_DERR_W3)	32	RW	0000_0000h
480h	MRC Domain Error Word0 Register (MRC0_DERR_W0)	32	R	0000_0000h
484h	MRC Domain Error Word1 Register (MRC0_DERR_W1)	32	R	0000_0000h
48Ch	MRC Domain Error Word3 Register (MRC0_DERR_W3)	32	RW	0000_0000h
800h	DAC Master Domain Assignment Register (MDA_W0_0_DFMT0)	32	RW	0000_0000h
820h	DAC Master Domain Assignment Register (MDA_W0_1_DFMT1)	32	RW	2000_0000h
840h	DAC Master Domain Assignment Register (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	DAC Master Domain Assignment Register (MDA_W0_3_DFMT1)	32	RW	2000_0000h
1000h	MBC Global Configuration Register (MBC0_MEM0_GLBCFG)	32	R	000C_0020h
1004h	MBC Global Configuration Register (MBC0_MEM1_GLBCFG)	32	R	000C_0004h
1008h	MBC Global Configuration Register (MBC0_MEM2_GLBCFG)	32	R	000C_0001h
100Ch	MBC Global Configuration Register (MBC0_MEM3_GLBCFG)	32	R	000C_000Ch
1010h	MBC NonSecure Enable Block Index (MBC0_NSE_BLK_INDEX)	32	RW	0000_0000h
1014h	MBC NonSecure Enable Block Set (MBC0_NSE_BLK_SET)	32	W	0000_0000h
1018h	MBC NonSecure Enable Block Clear (MBC0_NSE_BLK_CLR)	32	W	0000_0000h
101Ch	MBC NonSecure Enable Block Clear All (MBC0_NSE_BLK_CLR_ALL)	32	RW	0000_0000h
1020h	MBC Global Access Control (MBC0_MEMN_GLBAC0)	32	RW	0000_0000h
1024h	MBC Global Access Control (MBC0_MEMN_GLBAC1)	32	RW	0000_0000h
1028h	MBC Global Access Control (MBC0_MEMN_GLBAC2)	32	RW	0000_0000h
102Ch	MBC Global Access Control (MBC0_MEMN_GLBAC3)	32	RW	0000_0000h
1030h	MBC Global Access Control (MBC0_MEMN_GLBAC4)	32	RW	0000_0000h
1034h	MBC Global Access Control (MBC0_MEMN_GLBAC5)	32	RW	0000_0000h
1038h	MBC Global Access Control (MBC0_MEMN_GLBAC6)	32	RW	0000_0000h
103Ch	MBC Global Access Control (MBC0_MEMN_GLBAC7)	32	RW	0000_0000h
1040h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1044h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1048h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
104Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1140h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1180h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
11A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
11A8h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
11C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
11D0h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
11D4h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
11F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1240h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1244h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1248h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
124Ch	MBC Memory Block Configuration Word (MBC0_DOM1_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1340h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1380h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
13A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
13A8h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
13C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM2_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13D0h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
13D4h	MBC Memory Block Configuration Word (MBC0_DOM1_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
13F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM1_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
1440h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
1444h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
1448h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
144Ch	MBC Memory Block Configuration Word (MBC0_DOM2_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
1540h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
1580h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
15A0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
15A8h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
15C8h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
15D0h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
15D4h	MBC Memory Block Configuration Word (MBC0_DOM2_MEM3_BLK_CFG_W1)	32	RW	0000_0000h
15F0h	MBC Memory Block NonSecure Enable Word (MBC0_DOM2_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
2000h	MBC Global Configuration Register (MBC1_MEM0_GLBCFG)	32	R	000C_0002h
2004h	MBC Global Configuration Register (MBC1_MEM1_GLBCFG)	32	R	000C_0008h
2008h	MBC Global Configuration Register (MBC1_MEM2_GLBCFG)	32	R	000C_0005h
200Ch	MBC Global Configuration Register (MBC1_MEM3_GLBCFG)	32	R	000C_0002h
2010h	MBC NonSecure Enable Block Index (MBC1_NSE_BLK_INDEX)	32	RW	0000_0000h
2014h	MBC NonSecure Enable Block Set (MBC1_NSE_BLK_SET)	32	W	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2018h	MBC NonSecure Enable Block Clear (MBC1_NSE_BLK_CLR)	32	W	0000_0000h
201Ch	MBC NonSecure Enable Block Clear All (MBC1_NSE_BLK_CLR_ALL)	32	RW	0000_0000h
2020h	MBC Global Access Control (MBC1_MEMN_GLBAC0)	32	RW	0000_0000h
2024h	MBC Global Access Control (MBC1_MEMN_GLBAC1)	32	RW	0000_0000h
2028h	MBC Global Access Control (MBC1_MEMN_GLBAC2)	32	RW	0000_0000h
202Ch	MBC Global Access Control (MBC1_MEMN_GLBAC3)	32	RW	0000_0000h
2030h	MBC Global Access Control (MBC1_MEMN_GLBAC4)	32	RW	0000_0000h
2034h	MBC Global Access Control (MBC1_MEMN_GLBAC5)	32	RW	0000_0000h
2038h	MBC Global Access Control (MBC1_MEMN_GLBAC6)	32	RW	0000_0000h
203Ch	MBC Global Access Control (MBC1_MEMN_GLBAC7)	32	RW	0000_0000h
2040h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
2140h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
2180h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
21A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
21A8h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
21C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
21D0h	MBC Memory Block Configuration Word (MBC1_DOM0_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
21F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM0_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
2240h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
2340h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
2380h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
23A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23A8h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
23C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
23D0h	MBC Memory Block Configuration Word (MBC1_DOM1_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
23F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM1_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
2440h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
2540h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
2580h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
25A0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h
25A8h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
25C8h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
25D0h	MBC Memory Block Configuration Word (MBC1_DOM2_MEM3_BLK_CFG_W0)	32	RW	0000_0000h
25F0h	MBC Memory Block NonSecure Enable Word (MBC1_DOM2_MEM3_BLK_NSE_W0)	32	RW	0000_0000h
3000h	MBC Global Configuration Register (MBC2_MEM0_GLBCFG)	32	R	000C_004Fh
3004h	MBC Global Configuration Register (MBC2_MEM1_GLBCFG)	32	R	000C_0004h
3008h	MBC Global Configuration Register (MBC2_MEM2_GLBCFG)	32	R	000C_0010h
300Ch	MBC Global Configuration Register (MBC2_MEM3_GLBCFG)	32	R	000C_0000h
3010h	MBC NonSecure Enable Block Index (MBC2_NSE_BLK_INDEX)	32	RW	0000_0000h
3014h	MBC NonSecure Enable Block Set (MBC2_NSE_BLK_SET)	32	W	0000_0000h
3018h	MBC NonSecure Enable Block Clear (MBC2_NSE_BLK_CLR)	32	W	0000_0000h
301Ch	MBC NonSecure Enable Block Clear All (MBC2_NSE_BLK_CLR_ALL)	32	RW	0000_0000h
3020h	MBC Global Access Control (MBC2_MEMN_GLBAC0)	32	RW	0000_0000h
3024h	MBC Global Access Control (MBC2_MEMN_GLBAC1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3028h	MBC Global Access Control (MBC2_MEMN_GLBAC2)	32	RW	0000_0000h
302Ch	MBC Global Access Control (MBC2_MEMN_GLBAC3)	32	RW	0000_0000h
3030h	MBC Global Access Control (MBC2_MEMN_GLBAC4)	32	RW	0000_0000h
3034h	MBC Global Access Control (MBC2_MEMN_GLBAC5)	32	RW	0000_0000h
3038h	MBC Global Access Control (MBC2_MEMN_GLBAC6)	32	RW	0000_0000h
303Ch	MBC Global Access Control (MBC2_MEMN_GLBAC7)	32	RW	0000_0000h
3040h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
3044h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
3048h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
304Ch	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
3050h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
3054h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
3058h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
305Ch	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
3060h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
3064h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
3140h	MBC Memory Block NonSecure Enable Word (MBC2_DOM0_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
3144h	MBC Memory Block NonSecure Enable Word (MBC2_DOM0_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
3148h	MBC Memory Block NonSecure Enable Word (MBC2_DOM0_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
3180h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
31A0h	MBC Memory Block NonSecure Enable Word (MBC2_DOM0_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
31A8h	MBC Memory Block Configuration Word (MBC2_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
31ACh	MBC Memory Block Configuration Word (MBC2_DOM0_MEM2_BLK_CFG_W1)	32	RW	0000_0000h
31C8h	MBC Memory Block NonSecure Enable Word (MBC2_DOM0_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
3240h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
3244h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
3248h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
324Ch	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
3250h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
3254h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
3258h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
325Ch	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
3260h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
3264h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
3340h	MBC Memory Block NonSecure Enable Word (MBC2_DOM1_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
3344h	MBC Memory Block NonSecure Enable Word (MBC2_DOM1_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
3348h	MBC Memory Block NonSecure Enable Word (MBC2_DOM1_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
3380h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
33A0h	MBC Memory Block NonSecure Enable Word (MBC2_DOM1_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
33A8h	MBC Memory Block Configuration Word (MBC2_DOM1_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
33ACh	MBC Memory Block Configuration Word (MBC2_DOM1_MEM2_BLK_CFG_W1)	32	RW	0000_0000h
33C8h	MBC Memory Block NonSecure Enable Word (MBC2_DOM1_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
3440h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
3444h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
3448h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W2)	32	RW	0000_0000h
344Ch	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
3450h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
3454h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
3458h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
345Ch	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
3460h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W8)	32	RW	0000_0000h
3464h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM0_BLK_CFG_W9)	32	RW	0000_0000h
3540h	MBC Memory Block NonSecure Enable Word (MBC2_DOM2_MEM0_BLK_NSE_W0)	32	RW	0000_0000h
3544h	MBC Memory Block NonSecure Enable Word (MBC2_DOM2_MEM0_BLK_NSE_W1)	32	RW	0000_0000h
3548h	MBC Memory Block NonSecure Enable Word (MBC2_DOM2_MEM0_BLK_NSE_W2)	32	RW	0000_0000h
3580h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
35A0h	MBC Memory Block NonSecure Enable Word (MBC2_DOM2_MEM1_BLK_NSE_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
35A8h	MBC Memory Block Configuration Word (MBC2_DOM2_MEM2_BLK_CFG_W0)	32	RW	0000_0000h
35ACh	MBC Memory Block Configuration Word (MBC2_DOM2_MEM2_BLK_CFG_W1)	32	RW	0000_0000h
35C8h	MBC Memory Block NonSecure Enable Word (MBC2_DOM2_MEM2_BLK_NSE_W0)	32	RW	0000_0000h
4000h	MRC Global Configuration Register (MRC0_GLBCFG)	32	R	0000_0008h
4010h	MRC NonSecure Enable Region Indirect (MRC0_NSE_RGN_INDIRECT)	32	RW	0000_0000h
4014h	MRC NonSecure Enable Region Set (MRC0_NSE_RGN_SET)	32	RW	0000_0000h
4018h	MRC NonSecure Enable Region Clear (MRC0_NSE_RGN_CLR)	32	RW	0000_0000h
401Ch	MRC NonSecure Enable Region Clear All (MRC0_NSE_RGN_CLR_ALL)	32	RW	0000_0000h
4020h	MRC Global Access Control (MRC0_GLBAC0)	32	RW	0000_0000h
4024h	MRC Global Access Control (MRC0_GLBAC1)	32	RW	0000_0000h
4028h	MRC Global Access Control (MRC0_GLBAC2)	32	RW	0000_0000h
402Ch	MRC Global Access Control (MRC0_GLBAC3)	32	RW	0000_0000h
4030h	MRC Global Access Control (MRC0_GLBAC4)	32	RW	0000_0000h
4034h	MRC Global Access Control (MRC0_GLBAC5)	32	RW	0000_0000h
4038h	MRC Global Access Control (MRC0_GLBAC6)	32	RW	0000_0000h
403Ch	MRC Global Access Control (MRC0_GLBAC7)	32	RW	0000_0000h
4040h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD0_W0)	32	RW	0000_0000h
4044h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD0_W1)	32	RW	0000_0000h
4048h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD1_W0)	32	RW	0000_0000h
404Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD1_W1)	32	RW	0000_0000h
4050h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD2_W0)	32	RW	0000_0000h
4054h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD2_W1)	32	RW	0000_0000h
4058h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD3_W0)	32	RW	0000_0000h
405Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD3_W1)	32	RW	0000_0000h
4060h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD4_W0)	32	RW	0000_0000h
4064h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD4_W1)	32	RW	0000_0000h
4068h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD5_W0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
406Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD5_W1)	32	RW	0000_0000h
4070h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD6_W0)	32	RW	0000_0000h
4074h	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD6_W1)	32	RW	0000_0000h
4078h	MRC Region Descriptor Word 0 (MRC0_DOM0_RGD7_W0)	32	RW	0000_0000h
407Ch	MRC Region Descriptor Word 1 (MRC0_DOM0_RGD7_W1)	32	RW	0000_0000h
40C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM0_RGD_NSE)	32	RW	0000_0000h
4140h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD0_W0)	32	RW	0000_0000h
4144h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD0_W1)	32	RW	0000_0000h
4148h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD1_W0)	32	RW	0000_0000h
414Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD1_W1)	32	RW	0000_0000h
4150h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD2_W0)	32	RW	0000_0000h
4154h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD2_W1)	32	RW	0000_0000h
4158h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD3_W0)	32	RW	0000_0000h
415Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD3_W1)	32	RW	0000_0000h
4160h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD4_W0)	32	RW	0000_0000h
4164h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD4_W1)	32	RW	0000_0000h
4168h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD5_W0)	32	RW	0000_0000h
416Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD5_W1)	32	RW	0000_0000h
4170h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD6_W0)	32	RW	0000_0000h
4174h	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD6_W1)	32	RW	0000_0000h
4178h	MRC Region Descriptor Word 0 (MRC0_DOM1_RGD7_W0)	32	RW	0000_0000h
417Ch	MRC Region Descriptor Word 1 (MRC0_DOM1_RGD7_W1)	32	RW	0000_0000h
41C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM1_RGD_NSE)	32	RW	0000_0000h
4240h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD0_W0)	32	RW	0000_0000h
4244h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD0_W1)	32	RW	0000_0000h
4248h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD1_W0)	32	RW	0000_0000h
424Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD1_W1)	32	RW	0000_0000h
4250h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD2_W0)	32	RW	0000_0000h
4254h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD2_W1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4258h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD3_W0)	32	RW	0000_0000h
425Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD3_W1)	32	RW	0000_0000h
4260h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD4_W0)	32	RW	0000_0000h
4264h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD4_W1)	32	RW	0000_0000h
4268h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD5_W0)	32	RW	0000_0000h
426Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD5_W1)	32	RW	0000_0000h
4270h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD6_W0)	32	RW	0000_0000h
4274h	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD6_W1)	32	RW	0000_0000h
4278h	MRC Region Descriptor Word 0 (MRC0_DOM2_RGD7_W0)	32	RW	0000_0000h
427Ch	MRC Region Descriptor Word 1 (MRC0_DOM2_RGD7_W1)	32	RW	0000_0000h
42C0h	MRC Region Descriptor NonSecure Enable (MRC0_DOM2_RGD_NSE)	32	RW	0000_0000h

9.7.1.2 TRDC Register (TRDC_CR)

Offset

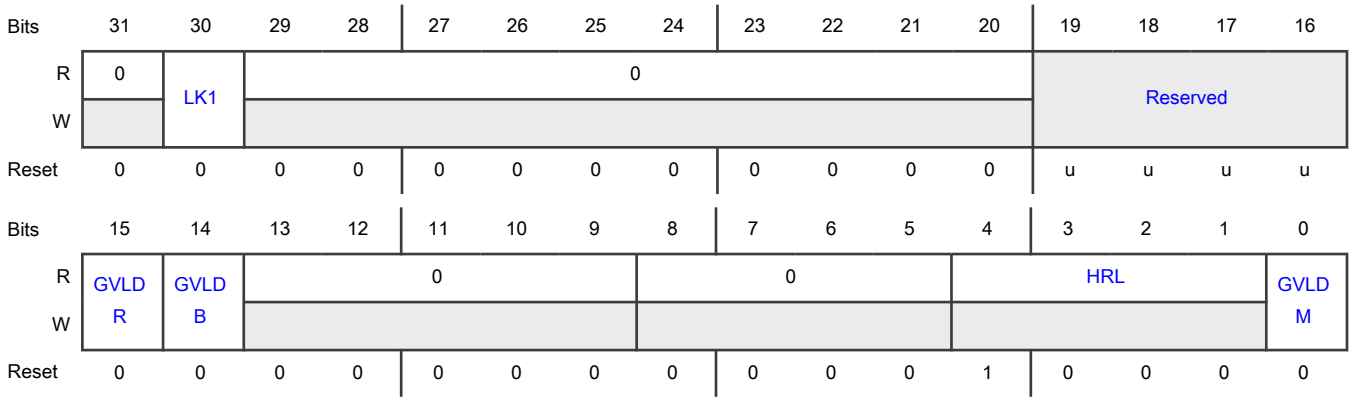
Register	Offset
TRDC_CR	0h

Function

This register provides status about the TRDC and global enable bits for the entire module's operation. A fully operational TRDC requires all global bits GVLD {R,B,M} to be asserted. Undefined behavior results if they are not all asserted. If there are no region checkers (MRCs) present in the configuration, GVLDR need not be asserted for a fully operational TRDC.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 —	Reserved
30 LK1	Lock Status This read-only field is the lock status of the TRDC_CR. This field is set when all GVLD bits are set; GVLDR=GVLDB=GVLDM=1b1. Once set, this bit remains asserted until the next reset. 0b - The CR can be written by any secure privileged write. 1b - The CR is locked (read-only) until the next reset.
29-20 —	Reserved
19-16 —	Reserved
15 GVLDR	Global Valid for Memory Region Checkers TRDC global MRC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MRCs are disabled. 1b - TRDC MRCs are enabled.
14 GVLDB	Global Valid for Memory Block Checkers TRDC global MBC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC MBCs are disabled. 1b - TRDC MBCs are enabled.
13-9 —	Reserved
8-5 —	Reserved
4-1 HRL	Hardware Revision Level This read-only field specifies the TRDC's hardware and definition revision level. It can be read by software to determine the functional definition of the module.
0 GVLDM	Global Valid for Domain Assignment Controllers TRDC global DAC enable/disable. Once set, this bit remains set until the next reset. 0b - TRDC DACs are disabled. 1b - TRDC DACs are enabled.

9.7.1.3 Hardware Configuration Register 0 (TRDC_HWCFG0)

Offset

Register	Offset
TRDC_HWCFG0	F0h

Function

This read-only register contains information on the TRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory block checkers (MBCs) and memory region checkers (MRCs). The register value at reset is device-specific. Attempted writes are error terminated.

Access: f TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NMRC				0				NMBC			
W																
Reset	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								0				NDID			
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-28 MID	Module ID This field defines major version ID of this module.
27-24 NMRC	Number of MRCs This field defines the number of Memory Region Checkeres on the device [1-8].
23-19 —	Reserved
18-16 NMBC	Number of MBCs This field defines the number of Memory Block Checkers on the device [1-4].
15-8 NMSTR	Number of bus masters This read-only field defines the number of bus masters.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 —	Reserved
3-0 NDID	Number of domains This read-only field defines the number of domains on the device [1-8].

9.7.1.4 TRDC Hardware Configuration Register 1 (TRDC_HWCFG1)

Offset

Register	Offset
TRDC_HWCFG1	F4h

Function

This register contains information on the TRDC's hardware configuration. It provides a mechanism for software to determine its domain number by simply reading the register. See Domain error capture management for more details on typical usage. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													DID		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	u	u	u

Fields

Field	Function
31-3 —	Reserved
2-0 DID	Domain identifier number This field provides the domain number [0-7] of the requesting bus master.

9.7.1.5 Domain Assignment Configuration Register (DACFG0 - DACFG3)

Offset

Register	Offset
DACFG0	100h
DACFG1	101h
DACFG2	102h
DACFG3	103h

Function

This register defines the number of implemented domain assignment registers for bus master m, where m+1 can specify from 1 to 64 bus masters. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error terminated.

Register read will not return transfer error when DACFG for a master doesn't exist but it is in the same 32-bit group as DACFG for an existing master. For example, when there are only 2 DACFG0-1 registers for 2 master, and masters for DACFG2-3 don't exist, then access to DACFG2-3 won't return transfer error.

Typically, processor bus masters have one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
DACFG0	01h
DACFG1–DACFG3	81h

Fields

Field	Function
7 NCM	Non-CPU Master This read-only field signals that bus master m is a non-CPU master. It specifies that the format of the associated MDA_Wr_m register defines a non-processor domain assignment. This field is zero for a non-existent bus master.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Bus master is a processor. 1b - Bus master is a non-processor.
6-4 —	Reserved
3-0 NMDAR	Number of master domain assignment registers for bus master m This read-only field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where zero indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDAm.

9.7.1.6 Memory Block Configuration Register (MBC0_CFG0)

Offset

Register	Offset
MBC0_CFG0	140h

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV1_BLKSZL2							SLV1_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV0_BLKSZL2							SLV0_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30-26 SLV1_BLKSZL2	Block size log2 in slave 1.
25-16 SLV1_NMBLK	Number of blocks in slave 1.
15 —	Reserved
14-10 SLV0_BLKSZL2	Block size log2 in slave 0.
9-0 SLV0_NMBLK	Number of blocks in slave 0.

9.7.1.7 Memory Block Configuration Register (MBC0_CFG1)

Offset

Register	Offset
MBC0_CFG1	144h

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV3_BLKSZL2							SLV3_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV2_BLKSZL2							SLV2_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 —	Reserved
30-26 SLV3_BLKSZL2	Block size log2 in slave 3.
25-16 SLV3_NMBLK	Number of blocks in slave 3.
15 —	Reserved
14-10 SLV2_BLKSZL2	Block size log2 in slave 2.
9-0 SLV2_NMBLK	Number of blocks in slave 2.

9.7.1.8 Memory Block Configuration Register (MBC1_CFG0)

Offset

Register	Offset
MBC1_CFG0	148h

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV1_BLKSZL2							SLV1_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV0_BLKSZL2							SLV0_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31 —	Reserved
30-26 SLV1_BLKSZL2	Block size log2 in slave 1.
25-16 SLV1_NMBLK	Number of blocks in slave 1.
15 —	Reserved
14-10 SLV0_BLKSZL2	Block size log2 in slave 0.
9-0 SLV0_NMBLK	Number of blocks in slave 0.

9.7.1.9 Memory Block Configuration Register (MBC1_CFG1)

Offset

Register	Offset
MBC1_CFG1	14Ch

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV3_BLKSZL2							SLV3_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV2_BLKSZL2							SLV2_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31 —	Reserved
30-26 SLV3_BLKSZL2	Block size log2 in slave 3.
25-16 SLV3_NMBLK	Number of blocks in slave 3.
15 —	Reserved
14-10 SLV2_BLKSZL2	Block size log2 in slave 2.
9-0 SLV2_NMBLK	Number of blocks in slave 2.

9.7.1.10 Memory Block Configuration Register (MBC2_CFG0)

Offset

Register	Offset
MBC2_CFG0	150h

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV1_BLKSZL2							SLV1_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV0_BLKSZL2							SLV0_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	1	0	0	1	1	1	1

Fields

Field	Function
31 —	Reserved
30-26 SLV1_BLKSZL2	Block size log2 in slave 1.
25-16 SLV1_NMBLK	Number of blocks in slave 1.
15 —	Reserved
14-10 SLV0_BLKSZL2	Block size log2 in slave 0.
9-0 SLV0_NMBLK	Number of blocks in slave 0.

9.7.1.11 Memory Block Configuration Register (MBC2_CFG1)

Offset

Register	Offset
MBC2_CFG1	154h

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV3_BLKSZL2							SLV3_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV2_BLKSZL2							SLV2_NMBLK							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-26 SLV3_BLKSZL2	Block size log2 in slave 3.
25-16 SLV3_NMBLK	Number of blocks in slave 3.
15 —	Reserved
14-10 SLV2_BLKSZL2	Block size log2 in slave 2.
9-0 SLV2_NMBLK	Number of blocks in slave 2.

9.7.1.12 Memory Block Configuration Register (MBC3_CFG0)

Offset

Register	Offset
MBC3_CFG0	158h

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV1_BLKSZL2					SLV1_NMBLK									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV0_BLKSZL2					SLV0_NMBLK									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-26 SLV1_BLKSZL2	Block size log2 in slave 1.
25-16 SLV1_NMBLK	Number of blocks in slave 1.
15 —	Reserved
14-10 SLV0_BLKSZL2	Block size log2 in slave 0.
9-0 SLV0_NMBLK	Number of blocks in slave 0.

9.7.1.13 Memory Block Configuration Register (MBC3_CFG1)

Offset

Register	Offset
MBC3_CFG1	15Ch

Function

This register defines the hardware configuration of two of the sub-blocks within a memory block checker.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv read-only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SLV3_BLKSZL2					SLV3_NMBLK									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SLV2_BLKSZL2					SLV2_NMBLK									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-26 SLV3_BLKSZL2	Block size log2 in slave 3.
25-16 SLV3_NMBLK	Number of blocks in slave 3.
15 —	Reserved
14-10 SLV2_BLKSZL2	Block size log2 in slave 2.
9-0 SLV2_NMBLK	Number of blocks in slave 2.

9.7.1.14 Memory Region Configuration Register (MRCFG0 - MRCFG7)

Offset

For r = 0 to 7:

Register	Offset
MRCFGr	160h + (r × 1h)

Function

This read-only register defines the number of implemented memory region descriptors for each MRCr, where r+1 can specify up to 16 instances. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. A zero value indicates a non-existent memory region checker instance. Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	7	6	5	4	3	2	1	0
R	0				NMRGD			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
MRCFG0	08h
MRCFG1–MRCFG7	00h

Fields

Field	Function
7-5 —	Reserved
4-0 NMRGD	Number of memory region descriptors for memory region checker n Number of memory region descriptors for MRCr. This field specifies the number of memory region descriptors associated with a given memory region checker instance. The value is limited to the range [0-16], where zero indicates a non-existent MRC instance and non-zero values indicate the number of implemented memory region descriptors [0-16] associated with the TRDC_MRCm submodule.

9.7.1.15 TRDC IDAU Control Register (TRDC_IDAU_CR)

Offset

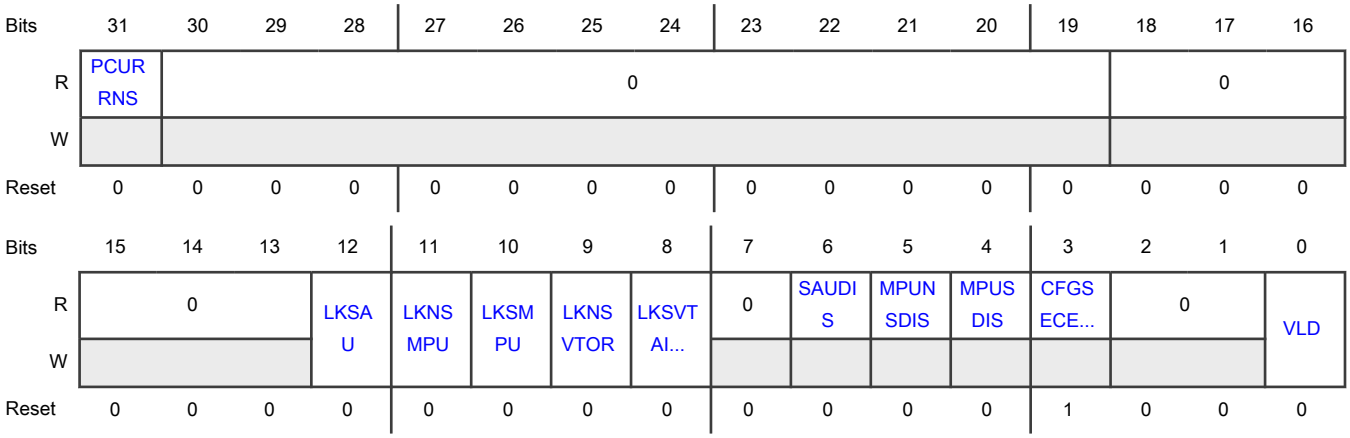
Register	Offset
TRDC_IDAU_CR	1C0h

Function

This register defines the configuration for the Implementation-Defined Attribution Unit which is tightly-coupled to the TZ-M extensions in the processor core. Many of the fields in this register explicitly control processor TZ-M functions.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31 PCURRNS	Processor current security Current security state of the processor. 0b - Processor is in Secure state 1b - Processor is in Nonsecure state
30-19 —	Reserved
18-16 —	Reserved
15-13 —	Reserved
12 LKSAU	Lock SAU Asserting this bit prevents changes to the processor's Secure SAU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the SAU_CTRL, SAU_RNR, SAU_RBAR and SAU_RLAR registers from software or from a debug agent connected to the processor
11 LKNSMPU	Lock Nonsecure MPU Asserting this bit prevents changes to the processor's Nonsecure MPU memory regions already programmed. All processor writes to the registers are ignored. This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Disable writes to the MPU_CTRL_NS, MPU_RNR_NS, MPU_RBAR_NS, MPU_RLAR_NS, MPU_RBAR_A_NSn and MPU_RLAR_A_NSn from software or from a debug agent connected to the processor
10 LKSMPU	Lock Secure MPU Asserting this signal prevents changes to the processor's programmed Secure MPU memory regions and all processor writes to the registers are ignored.This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock these registers 1b - Disable writes to the MPU_CTRL, MPU_RNR, MPU_RBAR, MPU_RLAR, MPU_RBAR_An and MPU_RLAR_An from software or from a debug agent connected to the processor in Secure state
9 LKNSVTOR	Lock Nonsecure Vector Table Offset Register Asserting this signal prevents changes to the processor's Nonsecure vector table base address.This bit is sticky, once set, it can only be cleared with a reset. 0b - Unlock this register 1b - Disable writes to the VTOR_NS register
8 LKSVTAIRCR	Lock Secure VTOR, Application interrupt and Reset Control Registers This bit is sticky, once set, it can only be cleared with a reset. Asserting this signal prevents processor changes to: • The Secure vector table base address. • Handling of Secure interrupt priority. • BusFault, HardFault, and NMI security target settings in the processor. 0b - Unlock these registers 1b - Disable writes to the VTOR_S, AIRCR[PRIS], and AIRCR[BFHFNMINS] registers
7 —	Reserved
6 SAUDIS	Security Attribution Unit Disable 0b - SAU is enabled 1b - SAU is disabled
5 MPUNSDIS	NonSecure Memory Protection Unit Disabled 0b - Nonsecure MPU is enabled 1b - Nonsecure MPU is disabled
4 MPUSDIS	Secure Memory Protection Unit Disabled 0b - Secure MPU is enabled 1b - Secure MPU is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 CFGSECEXT	Configure Security Extension 0b - ARMv8M Security Extension is disabled 1b - ARMv8-M Security Extension is enabled
2-1 —	Reserved
0 VLD	Valid When VLD =0, all address attribute from IDAU is nonSecure. When VLD=1, values in other fields of this register are valid.

9.7.1.16 TRDC FLW Control (TRDC_FLW_CTL)

Offset

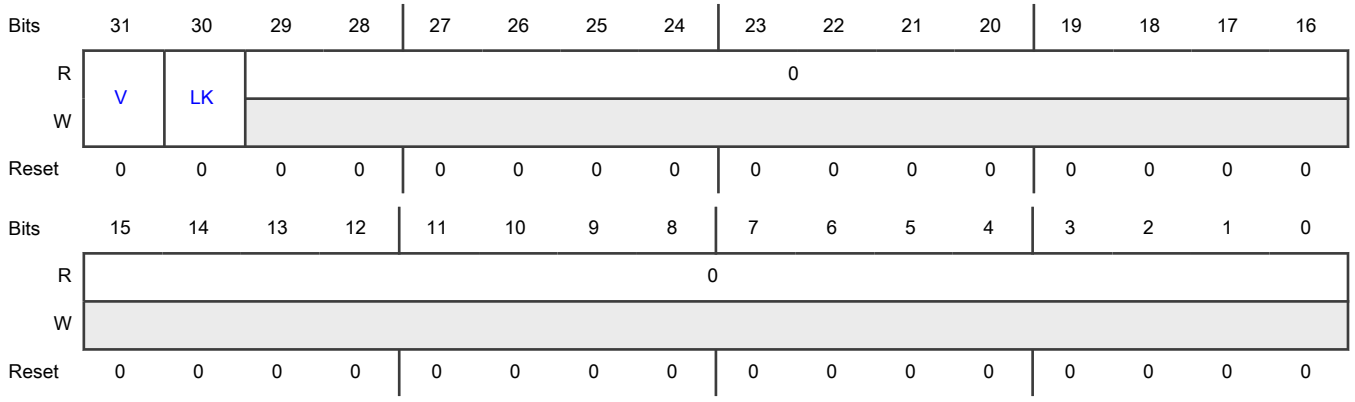
Register	Offset
TRDC_FLW_CTL	1E0h

Function

This register provides control of the FLW = Flash Logical Window operation.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31	Valid bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
V	0b - FLW function is disabled. 1b - FLW function is enabled.
30 LK	Lock bit 0b - FLW registers may be modified. 1b - FLW registers are locked until the next reset.
29-0 —	Reserved

9.7.1.17 TRDC FLW Physical Base (TRDC_FLW_PBASE)

Offset

Register	Offset
TRDC_FLW_PBASE	1E4h

Function

This read-only register gives the physical base address of the Flash Logical Window.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PBASE															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PBASE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PBASE	Physical base address Physical address of the base of the FLW (mod 32KBytes).

9.7.1.18 TRDC FLW Array Base (TRDC_FLW_ABASE)

Offset

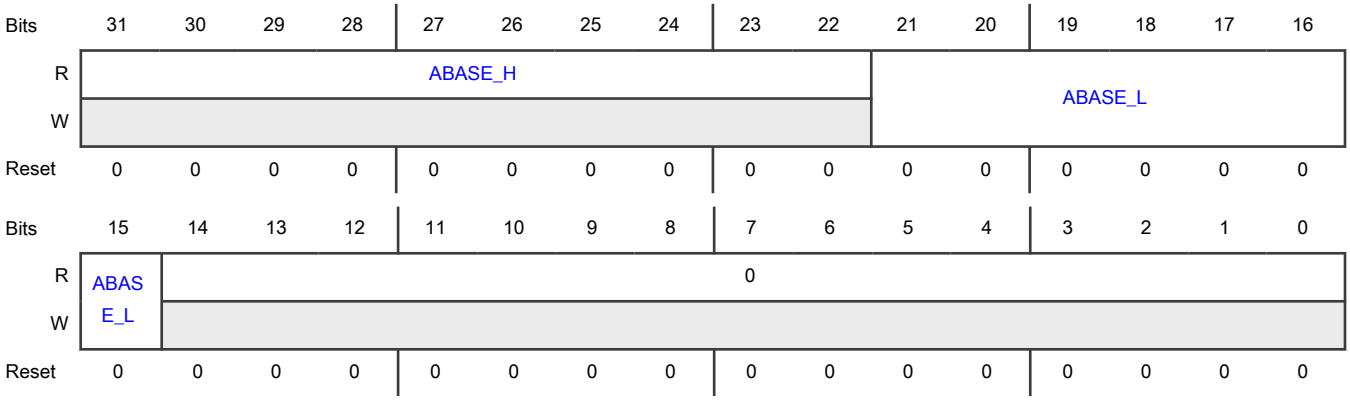
Register	Offset
TRDC_FLW_ABASE	1E8h

Function

This register gives the flash array base address of the Flash Logical Window.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-22 ABASE_H	Array base address high Flash array address upper bits of the base of the FLW (mod 32KBytes).
21-15 ABASE_L	Array base address low Flash array address lower bits of the base of the FLW (mod 32KBytes).
14-0 —	Reserved

9.7.1.19 TRDC FLW Block Count (TRDC_FLW_BCNT)

Offset

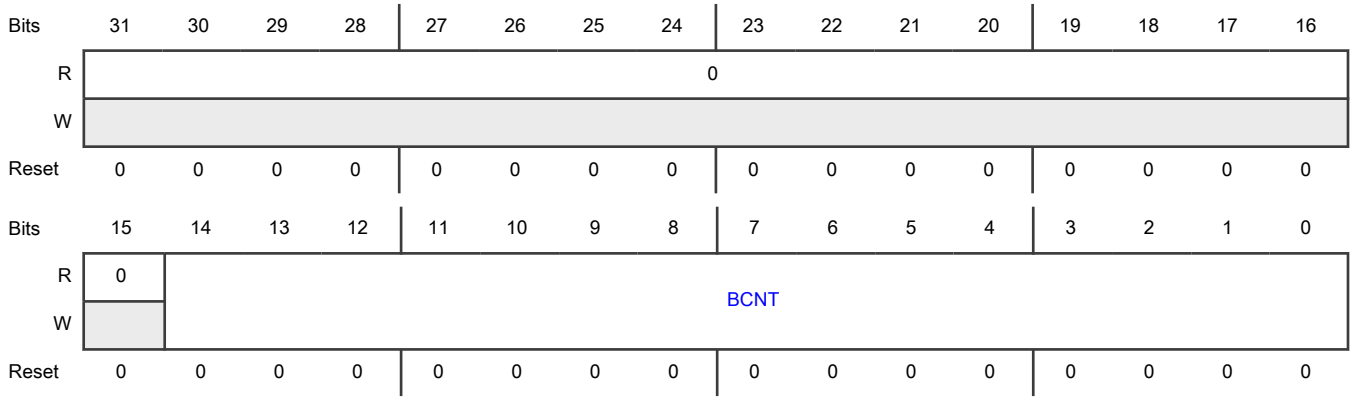
Register	Offset
TRDC_FLW_BCNT	1ECh

Function

This register gives the size of the Flash Logic Window in 32KByte blocks.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write. This register is not writable if TRDC_FLW_CTL[LK]=1.

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 BCNT	Block Count Size of FLW in number of 32KByte blocks.

9.7.1.20 TRDC Fault Domain ID (TRDC_FDID)

Offset

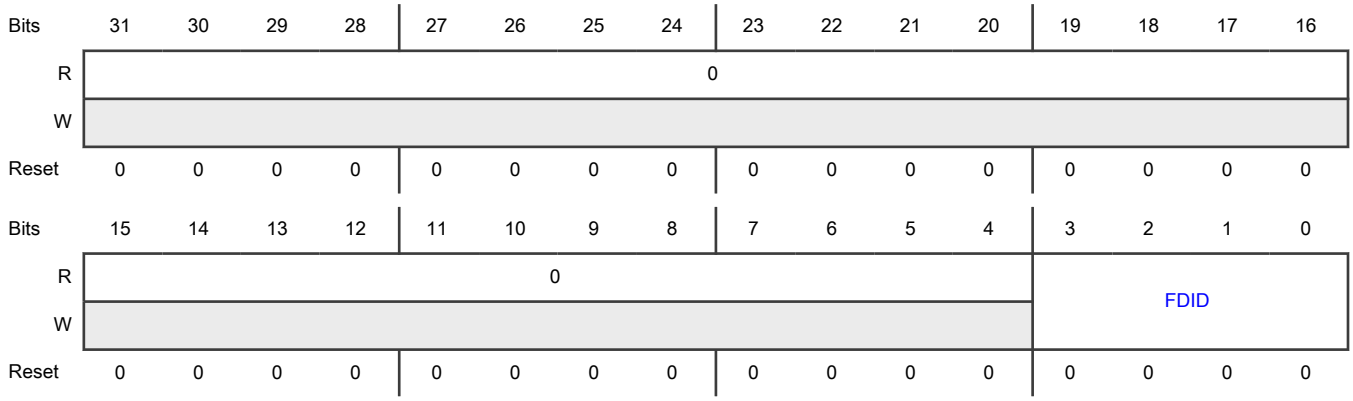
Register	Offset
TRDC_FDID	1FCh

Function

In the event of an access error, this register is used to specify the domainID of the faulting reference before indexing into the Domain Error registers.

If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FDID	Domain ID of Faulted Access This field indicates the domainID of the fault used to index the array of domain error information. The user queries the DERRLOCx registers to find the DomainID of the faulting access and must write this register with the domain ID before reading the Domain Error registers.

9.7.1.21 TRDC Domain Error Location Register (TRDC_DERRLOC0 - TRDC_DERRLOC2)

Offset

Register	Offset
TRDC_DERRLOC0	200h
TRDC_DERRLOC1	204h
TRDC_DERRLOC2	208h

Function

This array of read-only registers provide the instance number of the submodule where (an) access violation(s) occurred. These registers are organized as a word array, indexed by the faulting domain number, d. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, and DERR_W3_*i*register arrays. See Domain error capture management for more details.

When an access violation is detected by either a Memory Region Checker (MRC) or a Memory Block Checker (MBC), address and attribute information of the offending access is captured. Using the faulting domainID number as the index, d, this array of read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error terminated.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								MRCINST							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	mbc3_err_slv				mbc2_err_slv				mbc1_err_slv				mbc0_err_slv			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 MRCINST	MRC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MRC. The least-significant bit of this field (bit 16) corresponds to MRC instance 0. The most-significant bit of this field (bit 23) corresponds to MRC instance 7 (MRC instance = i-16 where i is the register bit number), and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MRCs. For each bit in this field: • 0 - The memory region checker has not detected an access violation or is not physically present. • 1 - The memory region checker has detected one or more access violations for this domain.
15-12 mbc3_err_slv	MBC3 ERROR SLAVE • 0001b - Error in Slave memory 0 • 0010b - Error in Slave memory 1 • 0100b - Error in Slave memory 2 • 1000b - Error in Slave memory 3
11-8 mbc2_err_slv	MBC2 ERROR SLAVE • 0001b - Error in Slave memory 0 • 0010b - Error in Slave memory 1 • 0100b - Error in Slave memory 2 • 1000b - Error in Slave memory 3
7-4	MBC1 ERROR SLAVE

Table continues on the next page...

Table continued from the previous page...

Field	Function
mbc1_err_slv	• 0001b - Error in Slave memory 0 • 0010b - Error in Slave memory 1 • 0100b - Error in Slave memory 2 • 1000b - Error in Slave memory 3
3-0 mbc0_err_slv	MBC0 ERROR SLAVE • 0001b - Error in Slave memory 0 • 0010b - Error in Slave memory 1 • 0100b - Error in Slave memory 2 • 1000b - Error in Slave memory 3

9.7.1.22 MBC Domain Error Word0 Register (MBC0_DERR_W0 - MBC2_DERR_W0)

Offset

Register	Offset
MBC0_DERR_W0	400h
MBC1_DERR_W0	410h
MBC2_DERR_W0	420h

Function

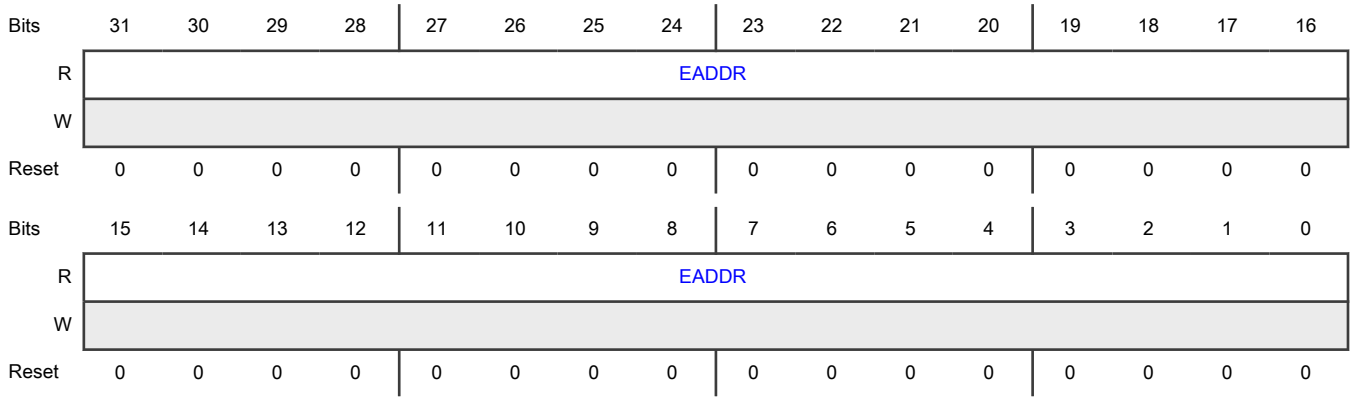
This read-only register array provides the address of an access violation detected by memory block checker (MBC). These registers are organized as a word array, which is indexed by the MBC instance number. That is, the index, i, of this array is the instance number of MBC with the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory mapped error capture detail registers are organized as 24 sequential 16 byte entries.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

Attempted writes are error terminated as are attempted reads of an MBC instance that is not physically present.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-0 EADDR	Error address This is the unaliased virtual address of the access that generated the access violation.

9.7.1.23 MBC Domain Error Word1 Register (MBC0_DERR_W1 - MBC2_DERR_W1)

Offset

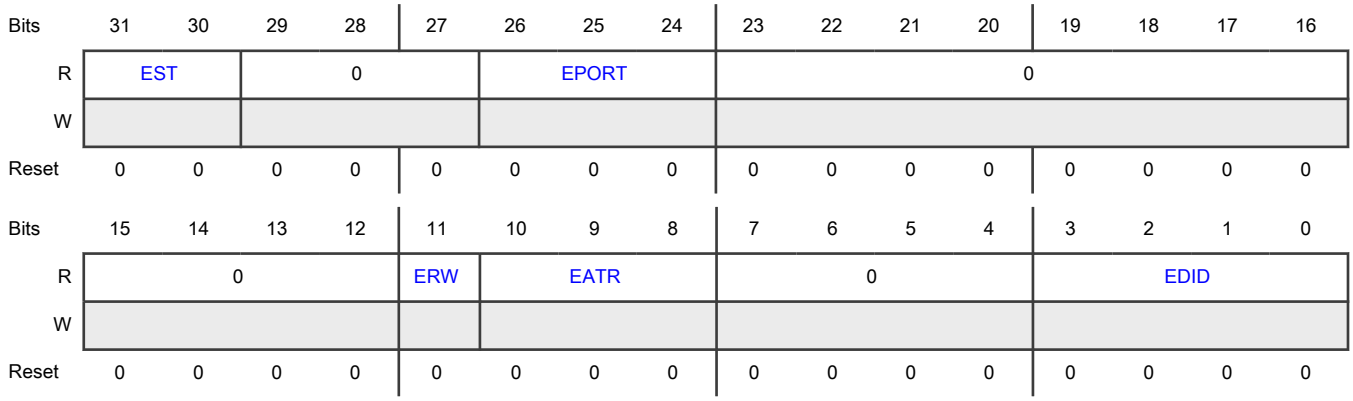
Register	Offset
MBC0_DERR_W1	404h
MBC1_DERR_W1	414h
MBC2_DERR_W1	424h

Function

This read-only register array provides the attributes of an access violation detected by Memory Block Checker (MBC). These registers are organized as a word array, which is indexed by the violating submodule instance number. Refer to register Domain Error Location(DERRLOCd), Domain Error Word0 Register(DERR_W0_i), and Domain error capture management for more information.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram



Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the block checker or region checker. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MBC that detected the access violation. The MBC port number connection is device-specific. See the chip configuration details for more information. This is the the MBC slave that has the violation. 000b - mbcxslv0 001b - mbcxslv1 010b - mbcxslv2 011b - mbcxslv3
23-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 ERW	Error read/write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error attributes This field captures attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

9.7.1.24 MBC Domain Error Word3 Register (MBC0_DERR_W3 - MBC2_DERR_W3)

Offset

Register	Offset
MBC0_DERR_W3	40Ch
MBC1_DERR_W3	41Ch
MBC2_DERR_W3	42Ch

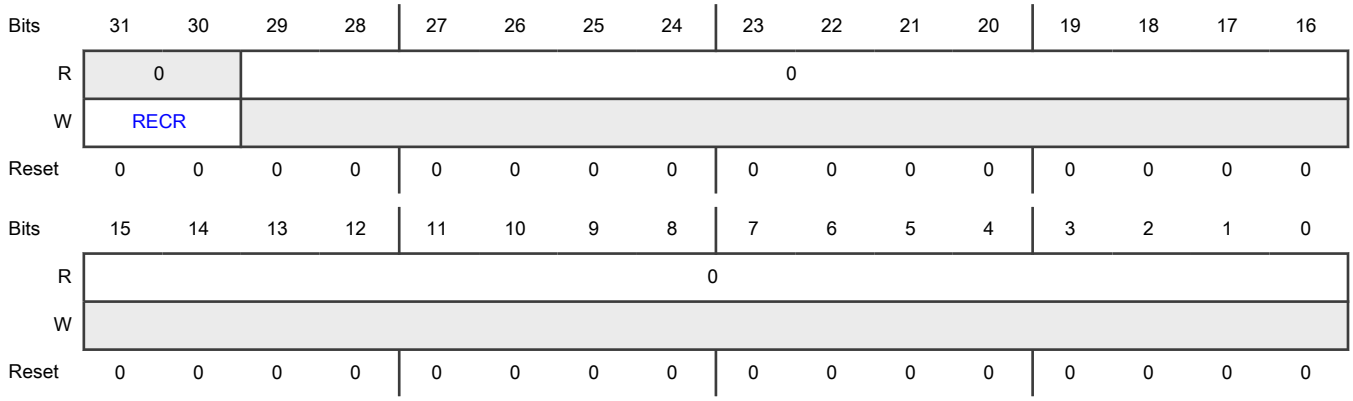
Function

This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a Memory Region Checker (MRC) or Memory Block Checker (MBC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

Diagram



Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

9.7.1.25 MRC Domain Error Word0 Register (MRC0_DERR_W0)

Offset

Register	Offset
MRC0_DERR_W0	480h

Function

This read-only register array provides the address of an access violation detected by memory region checker (MRC). These registers are organized as a word array, which is indexed by the MRC instance number. That is, the index, i, of this array is the instance number of MRC with the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory mapped error capture detail registers are organized as 24 sequential 16 byte entries.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

Attempted writes are error terminated as are attempted reads of an MRC instance that is not physically present.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EADDR	Error address This is the unaliased virtual address of the access that generated the access violation.

9.7.1.26 MRC Domain Error Word1 Register (MRC0_DERR_W1)

Offset

Register	Offset
MRC0_DERR_W1	484h

Function

This read-only register array provides the attributes of an access violation detected by memory region checker (MRC). These registers are organized as a word array, which is indexed by the violating submodule instance number. Refer to register Domain Error Location(DERRLOCd), Domain Error Word0 Register(DERR_W0_i), and Domain error capture management for more information.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EST		0		EPORT				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ERW	EATR			0				EDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the block checker or region checker. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MRC that detected the access violation. The MRC port number connection is device-specific. See the chip configuration details for more information.
23-12 —	Reserved
11 ERW	Error read/write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error attributes This field captures certain attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

9.7.1.27 MRC Domain Error Word3 Register (MRC0_DERR_W3)

Offset

Register	Offset
MRC0_DERR_W3	48Ch

Function

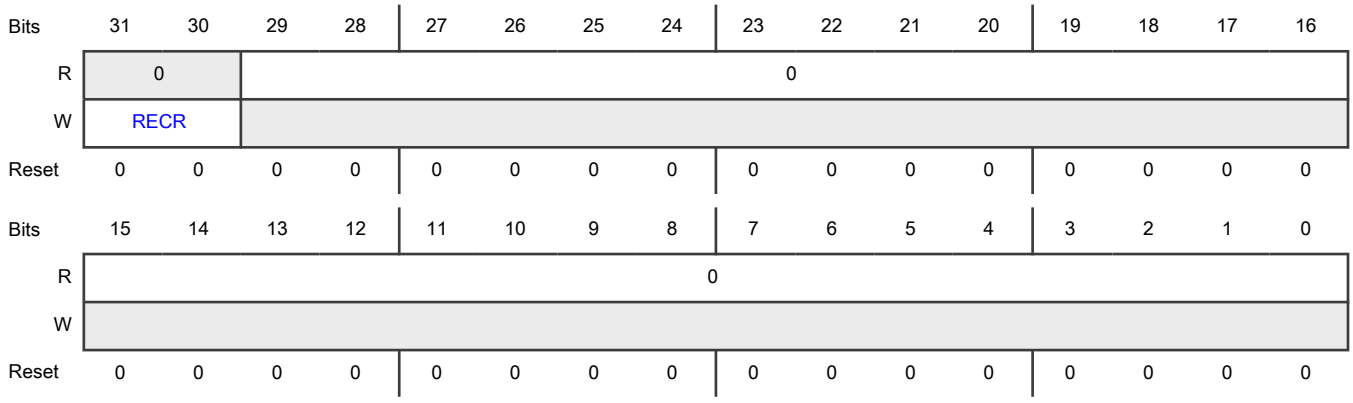
This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a memory region checker (MRC) or Memory Block Checker (MBC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/Write

Diagram



Fields

Field	Function
31-30	Rearm Error Capture Registers

Table continues on the next page...

Table continued from the previous page...

Field	Function
RECR	This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

9.7.1.28 DAC Master Domain Assignment Register (MDA_W0_0_DFMT0)

Offset

Register	Offset
MDA_W0_0_DFMT0	800h

Function

The MDA_Wr_m registers provide a 2-dimensional data structure for assigning bus masters to domains. The number of implemented registers is defined by DACFGm[NMDAR]. This per-master domain assignment is then repeated for each bus master (MDAm). Thus, m specifies the master number and r refers to the specific MDA register for a given bus master.

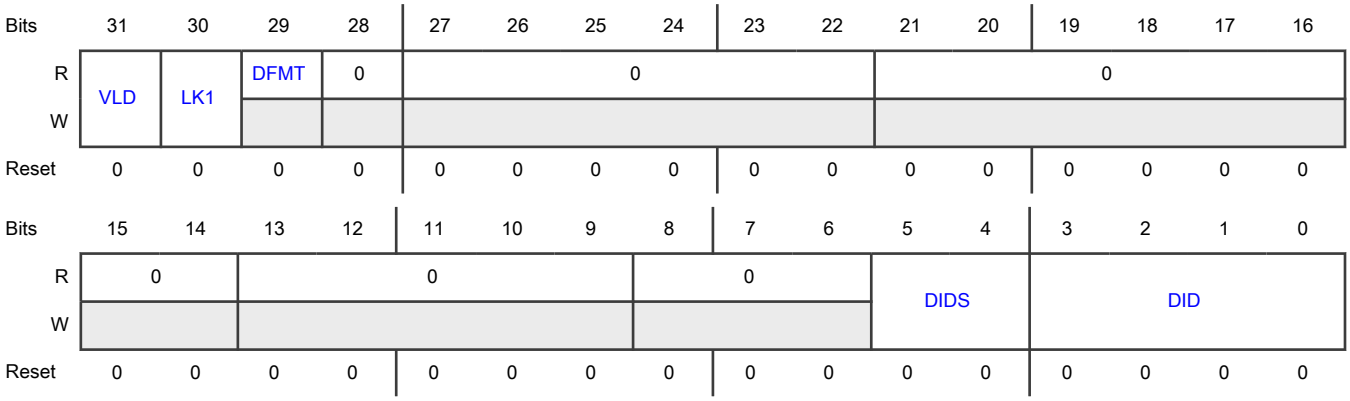
Each Wr within the MDAm structure is a word-sized definition; there are two formats supported, one for processor cores and another for non-processors. Processor masters typically support one or more Wr domain definitions, while non-processor masters support a single Wr.

The DAC submodule is responsible for the generation of domain identifiers for every transaction from every bus master. If there is a single Wr for a given master, then the specified domain identifier is used directly. If there are multiple Wr values for a given master, then the DAC evaluates the conditional terms to determine a “hit”. For all Wr hits, their corresponding domain identifiers are simply logically summed together (boolean OR). Use cases are typically expected to *hit in a single Wr* for a processor master. Special care is needed if *none* of the conditional terms hit in any Wr evaluation; for this case, the generated DID = 0 and software needs to be aware of any potential access rights granted for this DID.

Each MDA_Wr_m register has one of two programming models depending on the state of the domain format field, DFMT. The model described in this section is for DFMT = 0. This definition allows three different specifications of the DID for processors. The DFMT = 1 model is described in Master Domain Assignment (MDA_Wr_m_DFMT1) register.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROZ 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-14 —	Reserved
13-9 —	Reserved
8-6 —	Reserved
5-4 DIDS	DID Select This field selects the source of the domain identifier. 00b - Use MDAm[3:0] as the domain identifier. 01b - Use the input DID as the domain identifier. 10b - Use MDAm[3:2] concatenated with the low-order 2 bits of the input DID (DID_in[1:0]) as the domain identifier. 11b - Reserved for future use.
3-0 DID	Domain identifier This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDS=00b. DID[3:2] is used when DIDS=10b

9.7.1.29 DAC Master Domain Assignment Register (MDA_W0_1_DFMT1 - MDA_W0_3_DFMT1)

Offset

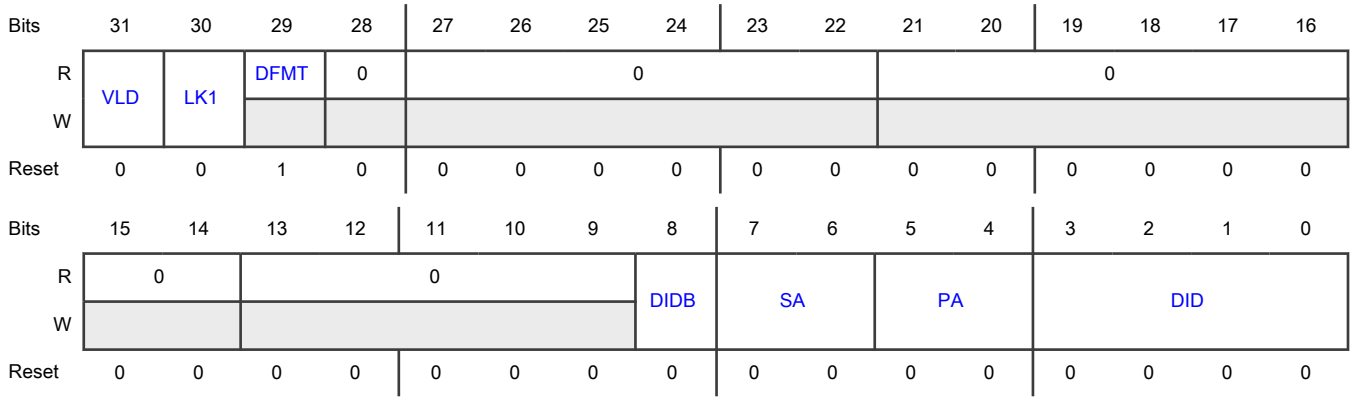
Register	Offset
MDA_W0_1_DFMT1	820h
MDA_W0_2_DFMT1	840h
MDA_W0_3_DFMT1	860h

Function

This register is identical to Master Domain Assignment (MDA_Wr_m_DFMT0) register except that the domain format field, DFMT, is 1 and the PID field is not used. This format supports two different specifications of the DID for non-core bus masters.

Access: If TZ-M is enabled, then SecurePrivileged, else NonsecurePriv Read/write

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLDM] = 1. If CR[GVLDM] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the TRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLDM] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment format. **NOTE** This bitfield access is ROO 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved
27-22 —	Reserved
21-16	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>15-14
—</td><td>Reserved</td></tr>
<tr><td>13-9
—</td><td>Reserved</td></tr>
<tr><td>8
DIDB</td><td>DID Bypass
If asserted, this bit enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters, for example, a DMA to masquerade as a processor.
Once set, this field is “sticky” and remains set until the next reset.
<table><tr><th>DAC</th><th>DID Input</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>DMA Domain ID</td></tr><tr><td>2</td><td>USB Domain ID</td></tr></table>
0b - Use MDAn[3:0] as the domain identifier.
1b - Use the DID input as the domain identifier.</td></tr>
<tr><td>7-6
SA</td><td>Secure attribute
This field defines the secure/nonsecure attribute for non-processor cores.
NOTE
The bus master’s input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. If TZM_ENB = 0, the master attribute is forced to nonsecure. A nonsecure write cannot program this field to 2'b00, which is a security level higher than the mode of the process that is writing it.
Reset value of SA:
• if TZM_ENB=0, SA reset value = 2'b01
• if TZM_ENB=1, SA reset value = 2'b00
00b - Force the bus attribute for this master to secure.
01b - Force the bus attribute for this master to nonsecure.
10b - Use the bus master’s secure/nonsecure attribute directly.
11b - Use the bus master’s secure/nonsecure attribute directly.</td></tr>
<tr><td>5-4
PA</td><td>Privileged attribute
This field defines the privileged/user attribute for non-processor cores.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The bus master's input privileged/user attribute is used if PA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to user. 01b - Force the bus attribute for this master to privileged. 10b - Use the bus master's privileged/user attribute directly. 11b - Use the bus master's privileged/user attribute directly.
3-0 DID	Domain identifier This 4-bit field is the domain ID attribute that is sent on the accesses from the bus master connected to the DAC when DIDB=0.

9.7.1.30 MBC Global Configuration Register (MBC0_MEM0_GLBCFG - MBC2_MEM3_GLBCFG)

Offset

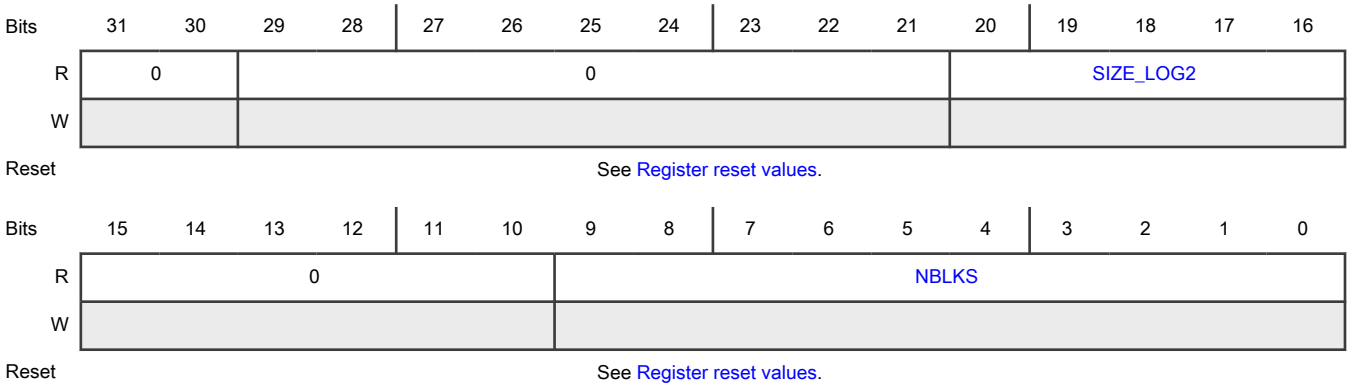
For m = 0 to 2; r = 0 to 3:

Register	Offset
MBCm_MEMr_GLBCFG	1000h + (m × 1000h) + (r × 4h)

Function

These MBC global configuration read-only registers contain information on the MBC's hardware configuration. Specifically, it defines the number of memory blocks and the size of each block in each MBC mem (r).

Diagram



Register reset values

Register	Reset value
MBC0_MEM0_GLBCFG	000C_0020h
MBC0_MEM1_GLBCFG	000C_0004h
MBC0_MEM2_GLBCFG	000C_0001h
MBC0_MEM3_GLBCFG	000C_000Ch
MBC1_MEM0_GLBCFG	000C_0002h
MBC1_MEM1_GLBCFG	000C_0008h
MBC1_MEM2_GLBCFG	000C_0005h
MBC1_MEM3_GLBCFG	000C_0002h
MBC2_MEM0_GLBCFG	000C_004Fh
MBC2_MEM1_GLBCFG	000C_0004h
MBC2_MEM2_GLBCFG	000C_0010h
MBC2_MEM3_GLBCFG	000C_0000h

Fields

Field	Function
31-30 —	Reserved
29-21 —	Reserved
20-16 SIZE_LOG2	Log2 size per block For example SIZE_LOG2=0x0C is 2^12=4 KB blocks.
15-10 —	Reserved
9-0 NBLKS	Number of blocks in this memory

9.7.1.31 MBC NonSecure Enable Block Index (MBC0_NSE_BLK_INDEX - MBC2_NSE_BLK_INDEX)

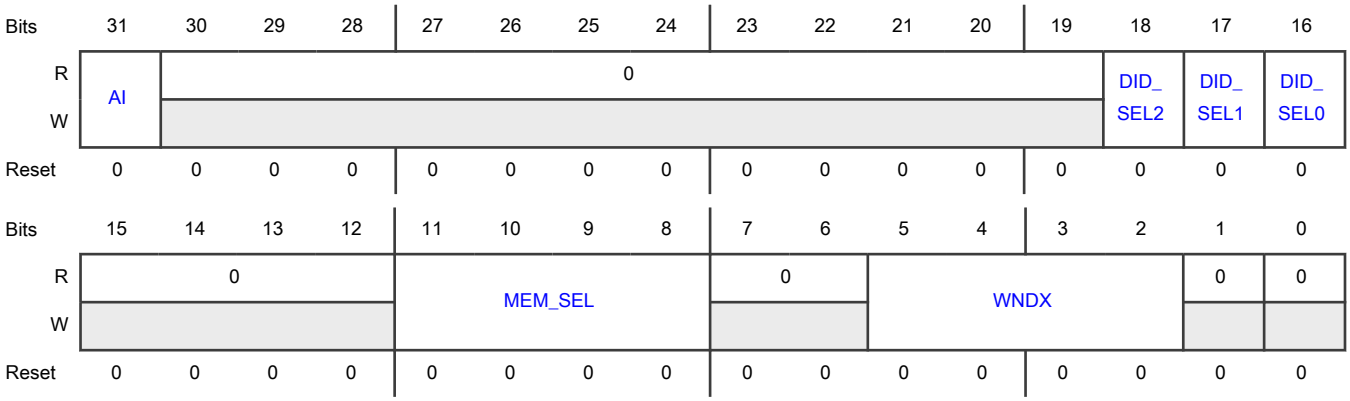
Offset

Register	Offset
MBC0_NSE_BLK_INDEX	1010h
MBC1_NSE_BLK_INDEX	2010h
MBC2_NSE_BLK_INDEX	3010h

Function

This R/W register defines the selected memories that are affected by the writes to the NSE_BLK_SET and NSE_BLK_CLR registers.

Diagram



Fields

Field	Function
31 AI	Auto Increment 0b - No effect. 1b - Add 1 to the WNDX field after the register write.
30-19 —	Reserved
18-16 DID_SELn	DID Select Destination domain bitmap select. 0b - No effect. 1b - Selects NSE bits for this domain.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-8 MEM_SEL	Memory Select Destination memory bitmap select. • Bit [11] - MBC MEM 3 • Bit [10] - MBC MEM 2 • Bit [9] - MBC MEM 1 • Bit [8] - MBC MEM 0
7-6 —	Reserved
5-2 WNDX	Word index into the block NSE bitmap. It selects the BLK_NSE_Wn register, where WNDX determines the value of n.
1 —	Reserved
0 —	Reserved

9.7.1.32 MBC NonSecure Enable Block Set (MBC0_NSE_BLK_SET - MBC2_NSE_BLK_SET)

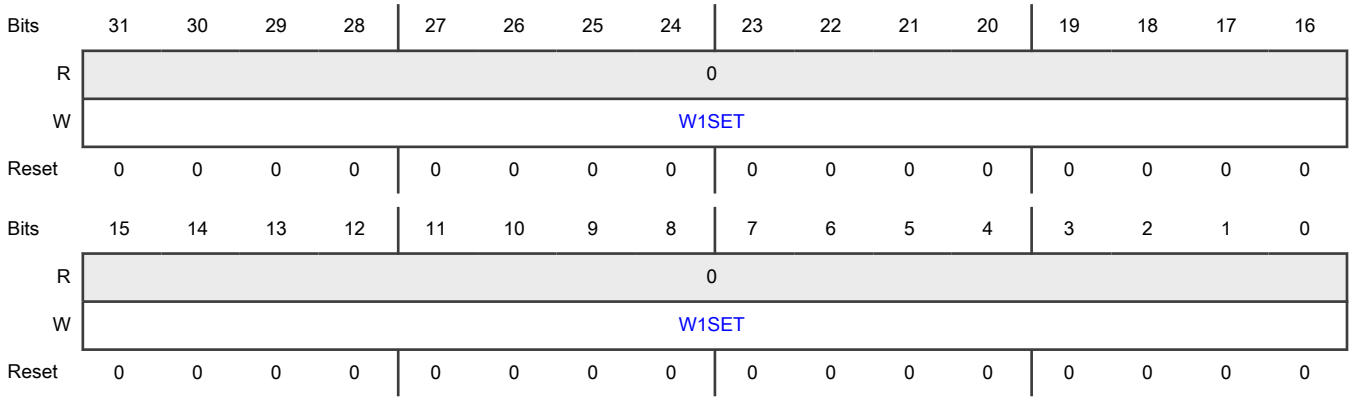
Offset

Register	Offset
MBC0_NSE_BLK_SET	1014h
MBC1_NSE_BLK_SET	2014h
MBC2_NSE_BLK_SET	3014h

Function

A write to this register sets the appropriate NSE Bits for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. If NSE_BLK_INDEX[AI] = 1, then the NSE_BLK_INDEX[WDNX] field is incremented by 1 (modulo-16) after the write is completed. This register reads as zero.

Diagram



Fields

Field	Function
31-0 W1SET	Write-1 Set If set to 1, sets appropriate NSE bit for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. Write with value 0 has no effect.

9.7.1.33 MBC NonSecure Enable Block Clear (MBC0_NSE_BLK_CLR - MBC2_NSE_BLK_CLR)

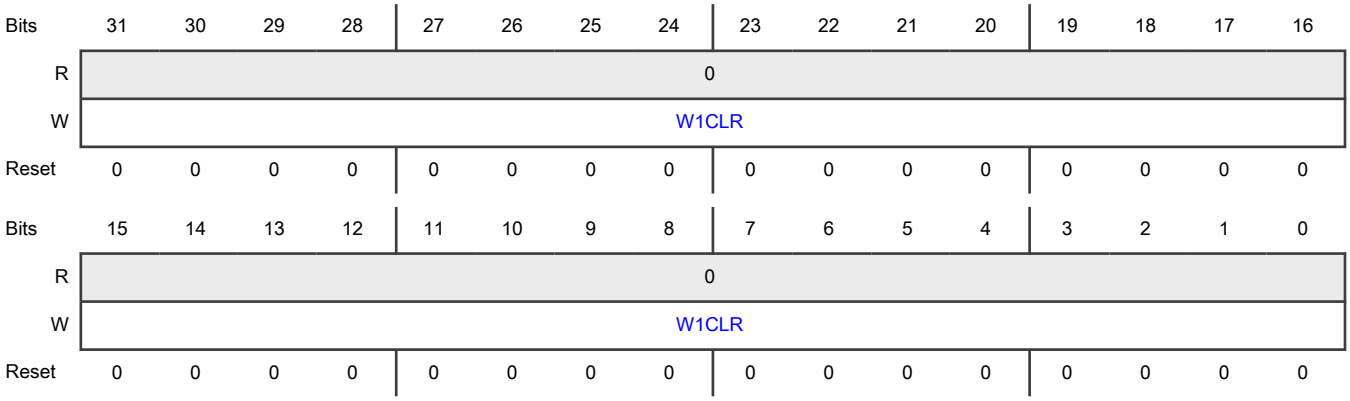
Offset

Register	Offset
MBC0_NSE_BLK_CLR	1018h
MBC1_NSE_BLK_CLR	2018h
MBC2_NSE_BLK_CLR	3018h

Function

A write to this location clears the appropriate NSE bits as defined by the W1CLR[n]=1 for the selected domains and memories Defined at the word location NSE_BLK_INDEX[WDNX]. If NSE_BLK_INDEX[AI] = 1, then the NSC_BLK_INDEX[WDNX] field is incremented by 1 (modulo-16) after the write is completed. This register reads as zero.

Diagram



Fields

Field	Function
31-0 W1CLR	Write-1 Clear If set to 1, Clear the appropriate NSE bit for the selected domains and memories defined at the word location NSE_BLK_INDEX[WDNX]. Write with value 0 has no effect.

9.7.1.34 MBC NonSecure Enable Block Clear All (MBC0_NSE_BLK_CLR_ALL - MBC2_NSE_BLK_CLR_ALL)

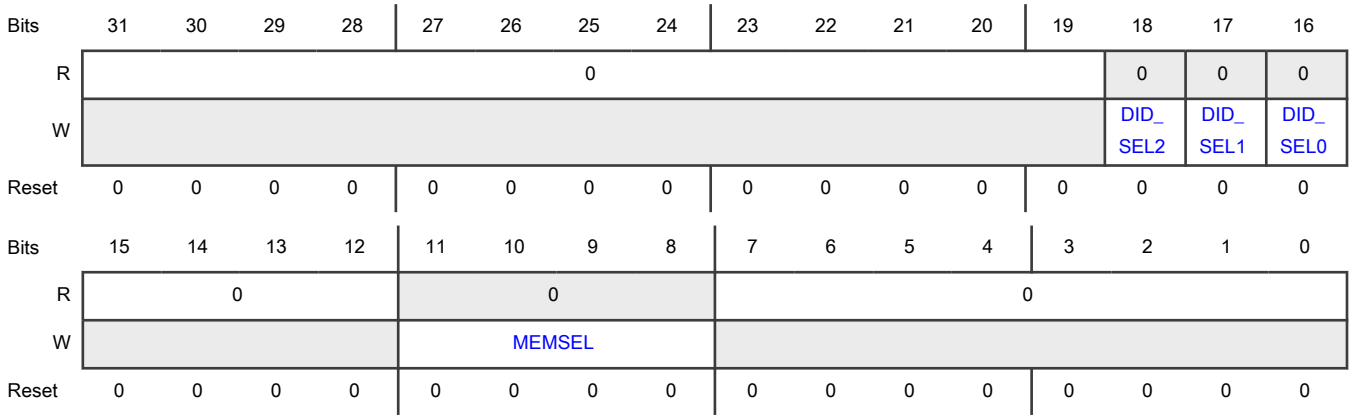
Offset

Register	Offset
MBC0_NSE_BLK_CLR_ALL	101Ch
MBC1_NSE_BLK_CLR_ALL	201Ch
MBC2_NSE_BLK_CLR_ALL	301Ch

Function

A write to this location clears all the block NSE bits for the selected domains and memories defined in the write data. This register reads as zero.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-16 DID_SELn	DID Select Destination domain bitmap select. 0b - No effect. 1b - Clear all NSE bits for this domain.
15-12 —	Reserved
11-8 MEMSEL	Memory Select Destination memory bitmap select. • Bit [11] - clear all MEM 3 NSE bits • Bit [10] - clear all MEM 2 NSE bits • Bit [9] - clear all MEM 1 NSE bits • Bit [8] - clear all MEM 0 NSE bits
7-0 —	Reserved

9.7.1.35 MBC Global Access Control (MBC0_MEMN_GLBAC0)

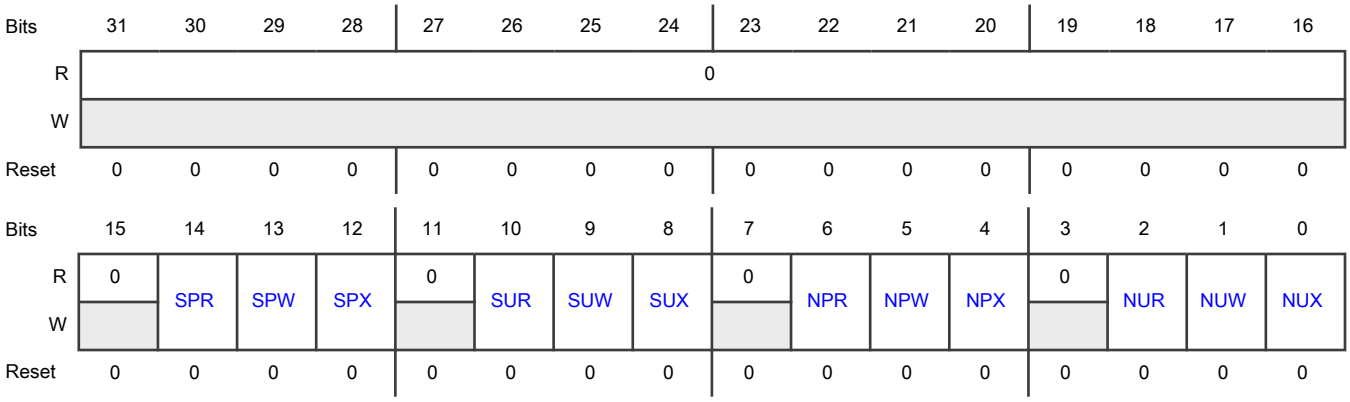
Offset

Register	Offset
MBC0_MEMN_GLBAC0	1020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these contol access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1	NonsecureUser Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUW	NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.36 MBC Global Access Control (MBC0_MEMN_GLBAC1 - MBC0_MEMN_GLBAC7)

Offset

Register	Offset
MBC0_MEMN_GLBAC1	1024h
MBC0_MEMN_GLBAC2	1028h
MBC0_MEMN_GLBAC3	102Ch
MBC0_MEMN_GLBAC4	1030h
MBC0_MEMN_GLBAC5	1034h
MBC0_MEMN_GLBAC6	1038h
MBC0_MEMN_GLBAC7	103Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these contol access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.37 MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0 - MBC2_DOM2_MEM2_BLK_CFG_W1)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_CFG_W0	1040h
MBC0_DOM0_MEM0_BLK_CFG_W1	1044h
MBC0_DOM0_MEM0_BLK_CFG_W2	1048h
MBC0_DOM0_MEM0_BLK_CFG_W3	104Ch
MBC0_DOM0_MEM1_BLK_CFG_W0	1180h
MBC0_DOM0_MEM2_BLK_CFG_W0	11A8h
MBC0_DOM0_MEM3_BLK_CFG_W0	11D0h
MBC0_DOM0_MEM3_BLK_CFG_W1	11D4h
MBC0_DOM1_MEM0_BLK_CFG_W0	1240h
MBC0_DOM1_MEM0_BLK_CFG_W1	1244h
MBC0_DOM1_MEM0_BLK_CFG_W2	1248h
MBC0_DOM1_MEM0_BLK_CFG_W3	124Ch
MBC0_DOM1_MEM1_BLK_CFG_W0	1380h
MBC0_DOM1_MEM2_BLK_CFG_W0	13A8h
MBC0_DOM1_MEM3_BLK_CFG_W0	13D0h
MBC0_DOM1_MEM3_BLK_CFG_W1	13D4h
MBC0_DOM2_MEM0_BLK_CFG_W0	1440h
MBC0_DOM2_MEM0_BLK_CFG_W1	1444h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC0_DOM2_MEM0_BLK_CFG_W2	1448h
MBC0_DOM2_MEM0_BLK_CFG_W3	144Ch
MBC0_DOM2_MEM1_BLK_CFG_W0	1580h
MBC0_DOM2_MEM2_BLK_CFG_W0	15A8h
MBC0_DOM2_MEM3_BLK_CFG_W0	15D0h
MBC0_DOM2_MEM3_BLK_CFG_W1	15D4h
MBC1_DOM0_MEM0_BLK_CFG_W0	2040h
MBC1_DOM0_MEM1_BLK_CFG_W0	2180h
MBC1_DOM0_MEM2_BLK_CFG_W0	21A8h
MBC1_DOM0_MEM3_BLK_CFG_W0	21D0h
MBC1_DOM1_MEM0_BLK_CFG_W0	2240h
MBC1_DOM1_MEM1_BLK_CFG_W0	2380h
MBC1_DOM1_MEM2_BLK_CFG_W0	23A8h
MBC1_DOM1_MEM3_BLK_CFG_W0	23D0h
MBC1_DOM2_MEM0_BLK_CFG_W0	2440h
MBC1_DOM2_MEM1_BLK_CFG_W0	2580h
MBC1_DOM2_MEM2_BLK_CFG_W0	25A8h
MBC1_DOM2_MEM3_BLK_CFG_W0	25D0h
MBC2_DOM0_MEM0_BLK_CFG_W0	3040h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC2_DOM0_MEM0_BLK_CFG_W1	3044h
MBC2_DOM0_MEM0_BLK_CFG_W2	3048h
MBC2_DOM0_MEM0_BLK_CFG_W3	304Ch
MBC2_DOM0_MEM0_BLK_CFG_W4	3050h
MBC2_DOM0_MEM0_BLK_CFG_W5	3054h
MBC2_DOM0_MEM0_BLK_CFG_W6	3058h
MBC2_DOM0_MEM0_BLK_CFG_W7	305Ch
MBC2_DOM0_MEM0_BLK_CFG_W8	3060h
MBC2_DOM0_MEM0_BLK_CFG_W9	3064h
MBC2_DOM0_MEM1_BLK_CFG_W0	3180h
MBC2_DOM0_MEM2_BLK_CFG_W0	31A8h
MBC2_DOM0_MEM2_BLK_CFG_W1	31ACh
MBC2_DOM1_MEM0_BLK_CFG_W0	3240h
MBC2_DOM1_MEM0_BLK_CFG_W1	3244h
MBC2_DOM1_MEM0_BLK_CFG_W2	3248h
MBC2_DOM1_MEM0_BLK_CFG_W3	324Ch
MBC2_DOM1_MEM0_BLK_CFG_W4	3250h
MBC2_DOM1_MEM0_BLK_CFG_W5	3254h
MBC2_DOM1_MEM0_BLK_CFG_W6	3258h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC2_DOM1_MEM0_BLK_CFG_W7	325Ch
MBC2_DOM1_MEM0_BLK_CFG_W8	3260h
MBC2_DOM1_MEM0_BLK_CFG_W9	3264h
MBC2_DOM1_MEM1_BLK_CFG_W0	3380h
MBC2_DOM1_MEM2_BLK_CFG_W0	33A8h
MBC2_DOM1_MEM2_BLK_CFG_W1	33ACh
MBC2_DOM2_MEM0_BLK_CFG_W0	3440h
MBC2_DOM2_MEM0_BLK_CFG_W1	3444h
MBC2_DOM2_MEM0_BLK_CFG_W2	3448h
MBC2_DOM2_MEM0_BLK_CFG_W3	344Ch
MBC2_DOM2_MEM0_BLK_CFG_W4	3450h
MBC2_DOM2_MEM0_BLK_CFG_W5	3454h
MBC2_DOM2_MEM0_BLK_CFG_W6	3458h
MBC2_DOM2_MEM0_BLK_CFG_W7	345Ch
MBC2_DOM2_MEM0_BLK_CFG_W8	3460h
MBC2_DOM2_MEM0_BLK_CFG_W9	3464h
MBC2_DOM2_MEM1_BLK_CFG_W0	3580h
MBC2_DOM2_MEM2_BLK_CFG_W0	35A8h
MBC2_DOM2_MEM2_BLK_CFG_W1	35ACh

Function

MBC[m]_DOM[d]_MEM[s]_BLK_CFG_W[w], where,

- m - mbc index
- d - domain index
- s - memory slave index
- w - word index

These registers are read/write for both the alternate view of the NSE bit and the 3-bit MBACSEL field. This is an array of 4 bit fields defining the block configuration for the given submemory. Each 4-bit field includes an alternative view of the associated NSE bit plus a 3-bit access control select that selects the global access control value that applies to the referenced memory block.

For a given memory block, B, if the value programmed in the MBACSELn field selects a locked MBC_MEMN_GLBACr register, the MBACSEL field is locked until the next reset. Depending on BLK_CFG_W, the NSEn/MBASELn fields correspond to different blocks. The following table describes the relationship between W (word index) and B (block number).

MEM0 supports up to 512 blocks ; W0 - W63

MEM1-3 supports up to 64 block ; W0 - W7

NOTE

This register is shown with fields such that it covers entire 32 bit register. However, actual fields may be less if number of blocks are such that fields don't align to 32 bits. These absent fields may be shown but user can refer to table below to deduce actual fields and treat absent fields as reserved.

Table 35. Block Config Word to Block Number Relationship

Block Config Word	NSE7/ MBACSEL7	NSE6/ MBACSEL6	NSE5/ MBACSEL5	NSE4/ MBACSEL4	NSE3/ MBACSEL3	NSE2/ MBACSEL2	NSE1/ MBACSEL1	NSE0/ MBACSEL0
W0	block 7	block 6	block 5	block 4	block 3	block 2	block 1	block 0
W1	block 15	block 14	block 13	block 12	block 11	block 10	block 9	block 8
W2	block 23	block 22	block 21	block 20	block 19	block 18	block 17	block 16
W3	block 31	block 30	block 29	block 28	block 27	block 26	block 25	block 24
W4	block 39	block 38	block 37	block 36	block 35	block 34	block 33	block 32
W5	block 47	block 46	block 45	block 44	block 43	block 42	block 41	block 40
W6	block 55	block 54	block 53	block 52	block 51	block 50	block 49	block 48
W7	block 63	block 62	block 61	block 60	block 59	block 58	block 57	block 56
For MEM0, block 64 - block 511								
W64	block 71	block 70	block 69	block 68	block 67	block 66	block 65	block 64
.								
.								
.								

Table continues on the next page...

Table 35. Block Config Word to Block Number Relationship (continued)

Block Config Word	NSE7/ MBACSEL7	NSE6/ MBACSEL6	NSE5/ MBACSEL5	NSE4/ MBACSEL4	NSE3/ MBACSEL3	NSE2/ MBACSEL2	NSE1/ MBACSEL1	NSE0/ MBACSEL0
W63	block 511	block 510	block 509	block 508	block 507	block 506	block 505	block 504

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NSE7	MBACSEL7			NSE6	MBACSEL6			NSE5	MBACSEL5			NSE4	MBACSEL4		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NSE3	MBACSEL3			NSE2	MBACSEL2			NSE1	MBACSEL1			NSE0	MBACSEL0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NSE7	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
30-28 MBACSEL7	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111b - select MBC_MEMN_GLBAC7 access control policy for block B
27 NSE6	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
26-24 MBACSEL6	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
23 NSE5	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
22-20 MBACSEL5	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
19 NSE4	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
18-16 MBACSEL4	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
15 NSE3	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
14-12 MBACSEL3	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
11 NSE2	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
10-8 MBACSEL2	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
7 NSE1	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
6-4	Memory Block Access Control Select for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBACSEL1	This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B
3 NSE0	NonSecure Enable for block B 0b - Secure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in this register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).
2-0 MBACSEL0	Memory Block Access Control Select for block B This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL7 = r and MBC_MEMN_GLBACr[LK] = 1. 000b - select MBC_MEMN_GLBAC0 access control policy for block B 001b - select MBC_MEMN_GLBAC1 access control policy for block B 010b - select MBC_MEMN_GLBAC2 access control policy for block B 011b - select MBC_MEMN_GLBAC3 access control policy for block B 100b - select MBC_MEMN_GLBAC4 access control policy for block B 101b - select MBC_MEMN_GLBAC5 access control policy for block B 110b - select MBC_MEMN_GLBAC6 access control policy for block B 111b - select MBC_MEMN_GLBAC7 access control policy for block B

9.7.1.38 MBC Memory Block NonSecure Enable Word (MBC0_DOM0_MEM0_BLK_NSE_W0 - MBC2_DOM2_MEM2_BLK_NSE_W0)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_NSE_W0	1140h
MBC0_DOM0_MEM1_BLK_NSE_W0	11A0h
MBC0_DOM0_MEM2_BLK_NSE_W0	11C8h
MBC0_DOM0_MEM3_BLK_NSE_W0	11F0h
MBC0_DOM1_MEM0_BLK_NSE_W0	1340h
MBC0_DOM1_MEM1_BLK_NSE_W0	13A0h
MBC0_DOM1_MEM2_BLK_NSE_W0	13C8h
MBC0_DOM1_MEM3_BLK_NSE_W0	13F0h
MBC0_DOM2_MEM0_BLK_NSE_W0	1540h
MBC0_DOM2_MEM1_BLK_NSE_W0	15A0h
MBC0_DOM2_MEM2_BLK_NSE_W0	15C8h
MBC0_DOM2_MEM3_BLK_NSE_W0	15F0h
MBC1_DOM0_MEM0_BLK_NSE_W0	2140h
MBC1_DOM0_MEM1_BLK_NSE_W0	21A0h
MBC1_DOM0_MEM2_BLK_NSE_W0	21C8h
MBC1_DOM0_MEM3_BLK_NSE_W0	21F0h
MBC1_DOM1_MEM0_BLK_NSE_W0	2340h
MBC1_DOM1_MEM1_BLK_NSE_W0	23A0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC1_DOM1_MEM2_BLK_NSE_W0	23C8h
MBC1_DOM1_MEM3_BLK_NSE_W0	23F0h
MBC1_DOM2_MEM0_BLK_NSE_W0	2540h
MBC1_DOM2_MEM1_BLK_NSE_W0	25A0h
MBC1_DOM2_MEM2_BLK_NSE_W0	25C8h
MBC1_DOM2_MEM3_BLK_NSE_W0	25F0h
MBC2_DOM0_MEM0_BLK_NSE_W0	3140h
MBC2_DOM0_MEM0_BLK_NSE_W1	3144h
MBC2_DOM0_MEM0_BLK_NSE_W2	3148h
MBC2_DOM0_MEM1_BLK_NSE_W0	31A0h
MBC2_DOM0_MEM2_BLK_NSE_W0	31C8h
MBC2_DOM1_MEM0_BLK_NSE_W0	3340h
MBC2_DOM1_MEM0_BLK_NSE_W1	3344h
MBC2_DOM1_MEM0_BLK_NSE_W2	3348h
MBC2_DOM1_MEM1_BLK_NSE_W0	33A0h
MBC2_DOM1_MEM2_BLK_NSE_W0	33C8h
MBC2_DOM2_MEM0_BLK_NSE_W0	3540h
MBC2_DOM2_MEM0_BLK_NSE_W1	3544h
MBC2_DOM2_MEM0_BLK_NSE_W2	3548h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MBC2_DOM2_MEM1_BLK_NSE_W0	35A0h
MBC2_DOM2_MEM2_BLK_NSE_W0	35C8h

Function

MBC[m]_DOM[d]_MEM[s]_BLK_NSE_W[w], where,

- m - mbc index
- d - domain index
- s - memory slave index
- w - word index

This is the bitmap of the individual NSE bits for the memory block.

NOTE

This register is shown with fields such that it covers entire 32 bit register. However, actual fields may be less if number of blocks are such that fields don't align to 32 bits. These absent fields may be shown but user can refer to table below to deduce actual fields and treat absent fields as reserved.

The following table describes the relationship between W (word index) and B (block number).

Table 36. Block Config Word to Block Number Relationship

Block NSE Word	BIT31	BIT30-BIT1	BIT0
W0	block 31	block 30 - block 1	block 0
W1	block 63	block 62 - block 33	block 32
For MEM0, block 64 - block 511			
W2	block 95	block 94 - block 65	block 64
. . .			
W3	block 511	block 510 - block 481	block 480

The NSE bitmap can be accessed directly by register reads and writes to these locations, or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and BLK_CFG[NSEn] registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24	BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BITb	Bit b NonSecure Enable [b = 0 - 31] 0b - Secure accesses to block B are based on corresponding MBACSEL field in register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]), nonsecure accesses to block B are not allowed. 1b - Secure accesses to block B are are not allowed, nonsecure accesses to block B are based on corresponding MBACSEL field in register (MBCm_DOMd_MEMs_BLK_CFG_Ww[MBACSEL]).

9.7.1.39 MBC Global Access Control (MBC1_MEMN_GLBAC0)

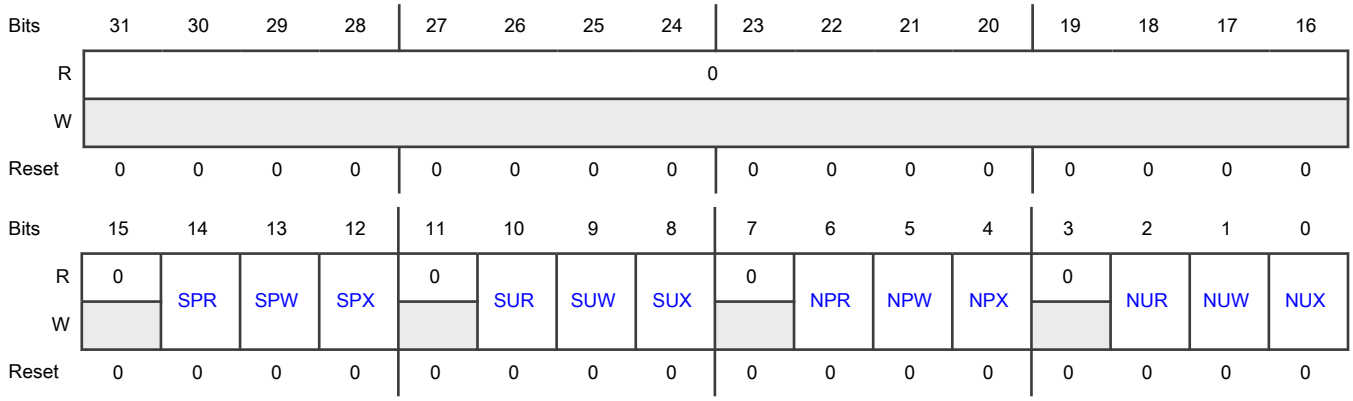
Offset

Register	Offset
MBC1_MEMN_GLBAC0	2020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these contol access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9	SecureUser Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUW	SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.40 MBC Global Access Control (MBC1_MEMN_GLBAC1 - MBC1_MEMN_GLBAC7)

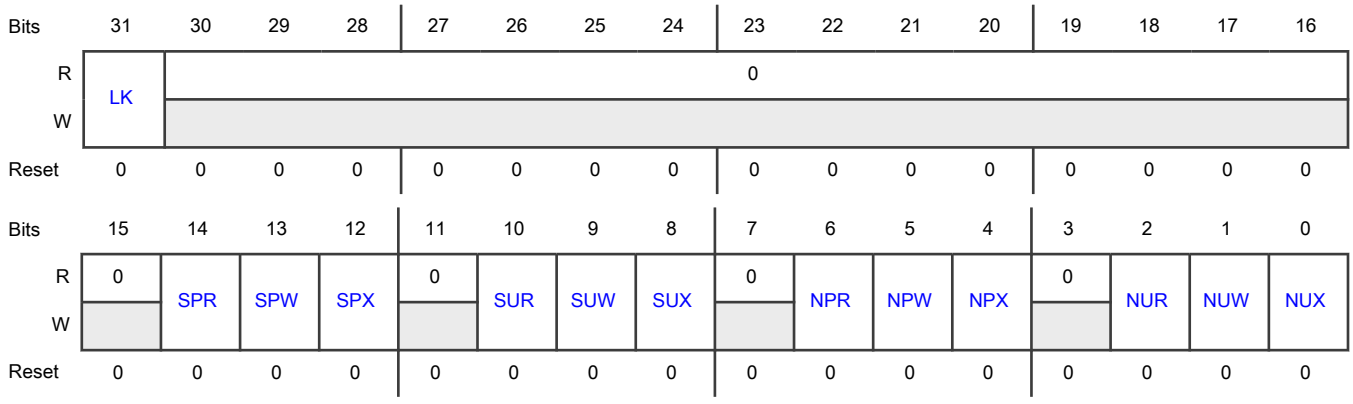
Offset

Register	Offset
MBC1_MEMN_GLBAC1	2024h
MBC1_MEMN_GLBAC2	2028h
MBC1_MEMN_GLBAC3	202Ch
MBC1_MEMN_GLBAC4	2030h
MBC1_MEMN_GLBAC5	2034h
MBC1_MEMN_GLBAC6	2038h
MBC1_MEMN_GLBAC7	203Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these contol access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.41 MBC Global Access Control (MBC2_MEMN_GLBAC0)

Offset

Register	Offset
MBC2_MEMN_GLBAC0	3020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these contol access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.42 MBC Global Access Control (MBC2_MEMN_GLBAC1 - MBC2_MEMN_GLBAC7)

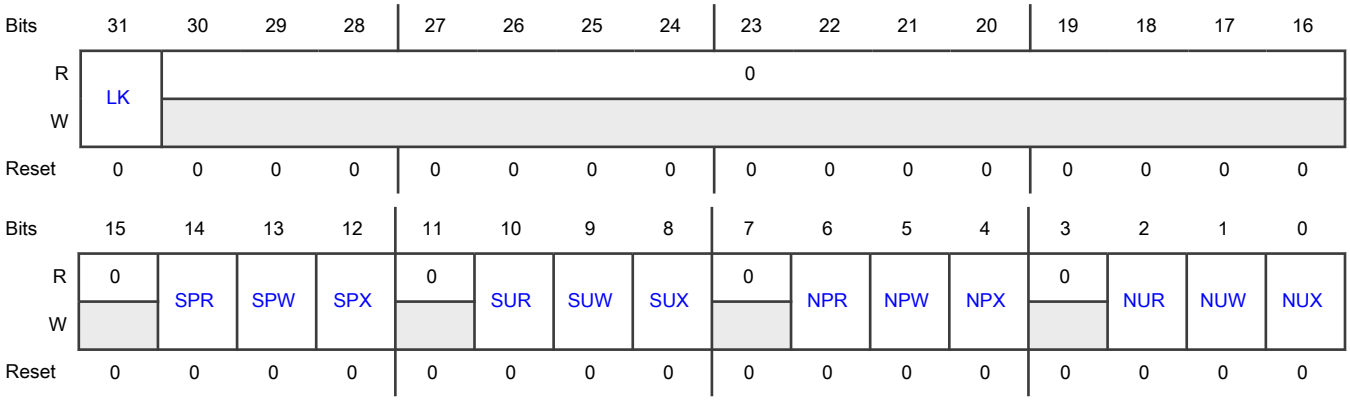
Offset

Register	Offset
MBC2_MEMN_GLBAC1	3024h
MBC2_MEMN_GLBAC2	3028h
MBC2_MEMN_GLBAC3	302Ch
MBC2_MEMN_GLBAC4	3030h
MBC2_MEMN_GLBAC5	3034h
MBC2_MEMN_GLBAC6	3038h
MBC2_MEMN_GLBAC7	303Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these contol access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.43 MRC Global Configuration Register (MRC0_GLBCFG)

Offset

Register	Offset
MRC0_GLBCFG	4000h

Function
This read-only register describes the number of region descriptors for this memory.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												NRGNS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 NRGNS	Number of regions [1-16]

9.7.1.44 MRC NonSecure Enable Region Indirect (MRC0_NSE_RGN_INDIRECT)

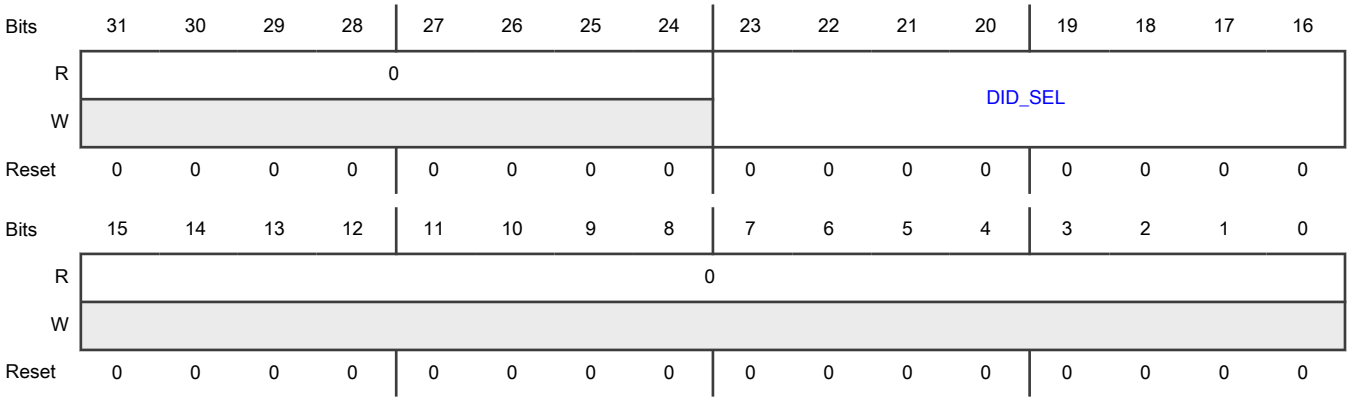
Offset

Register	Offset
MRC0_NSE_RGN_INDIRECT	4010h

Function

This R/W register defines the selected domains that are affected by writes to the NSE_RGN_SET and NSE_RGN_CLR registers.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 DID_SEL	DID Select Destination domain bitmap select. • Bit [23] = 1, update domain 7 • Bit [22] = 1, update domain 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Bit [21] = 1, update domain 5 • Bit [20] = 1, update domain 4 • Bit [19] = 1, update domain 3 • Bit [18] = 1, update domain 2 • Bit [17] = 1, update domain 1 • Bit [16] = 1, update domain 0
15-0 —	Reserved

9.7.1.45 MRC NonSecure Enable Region Set (MRC0_NSE_RGN_SET)

Offset

Register	Offset
MRC0_NSE_RGN_SET	4014h

Function

A write to this location sets the appropriate NSE bits in region descriptors for the selected domain in MRC NonSecure Enable Region Indirect register. This register reads as 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	W1SET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 W1SET	Write-1 Set • Bit [15] = Set the NSE bits for RGD15 • Bit [14] = Set the NSE bits for RGD14 • Bit [13] = Set the NSE bits for RGD13 • Bit [12] = Set the NSE bits for RGD12 • Bit [11] = Set the NSE bits for RGD11 • Bit [10] = Set the NSE bits for RGD10 • Bit [9] = Set the NSE bits for RGD9 • Bit [8] = Set the NSE bits for RGD8 • Bit [7] = Set the NSE bits for RGD7 • Bit [6] = Set the NSE bits for RGD6 • Bit [5] = Set the NSE bits for RGD5 • Bit [4] = Set the NSE bits for RGD4 • Bit [3] = Set the NSE bits for RGD3 • Bit [2] = Set the NSE bits for RGD2 • Bit [1] = Set the NSE bits for RGD1 • Bit [0] = Set the NSE bits for RGD0

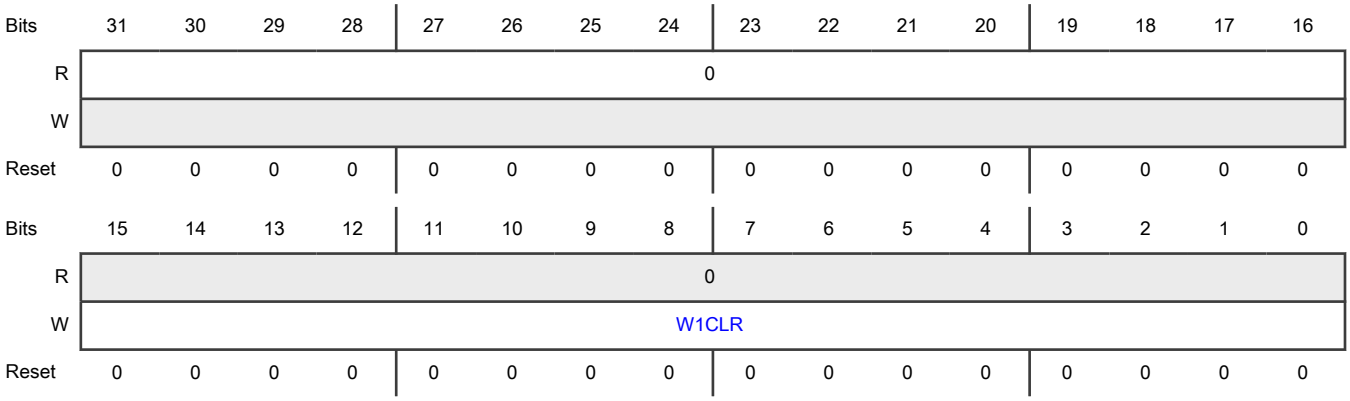
9.7.1.46 MRC NonSecure Enable Region Clear (MRC0_NSE_RGN_CLR)

Offset

Register	Offset
MRC0_NSE_RGN_CLR	4018h

Function
A write to this location clears the appropriate NSE bits in region descriptors for the selected domain in MRC NonSecure Enable Region Indirect register. This register reads as 0.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 W1CLR	Write-1 Clear • Bit [15] = Clear the NSE bits for RGD15 • Bit [14] = Clear the NSE bits for RGD14 • Bit [13] = Clear the NSE bits for RGD13 • Bit [12] = Clear the NSE bits for RGD12 • Bit [11] = Clear the NSE bits for RGD11 • Bit [10] = Clear the NSE bits for RGD10 • Bit [9] = Clear the NSE bits for RGD9 • Bit [8] = Clear the NSE bits for RGD8 • Bit [7] = Clear the NSE bits for RGD7 • Bit [6] = Clear the NSE bits for RGD6 • Bit [5] = Clear the NSE bits for RGD5 • Bit [4] = Clear the NSE bits for RGD4 • Bit [3] = Clear the NSE bits for RGD3 • Bit [2] = Clear the NSE bits for RGD2 • Bit [1] = Clear the NSE bits for RGD1 • Bit [0] = Clear the NSE bits for RGD0

9.7.1.47 MRC NonSecure Enable Region Clear All (MRC0_NSE_RGN_CLR_ALL)

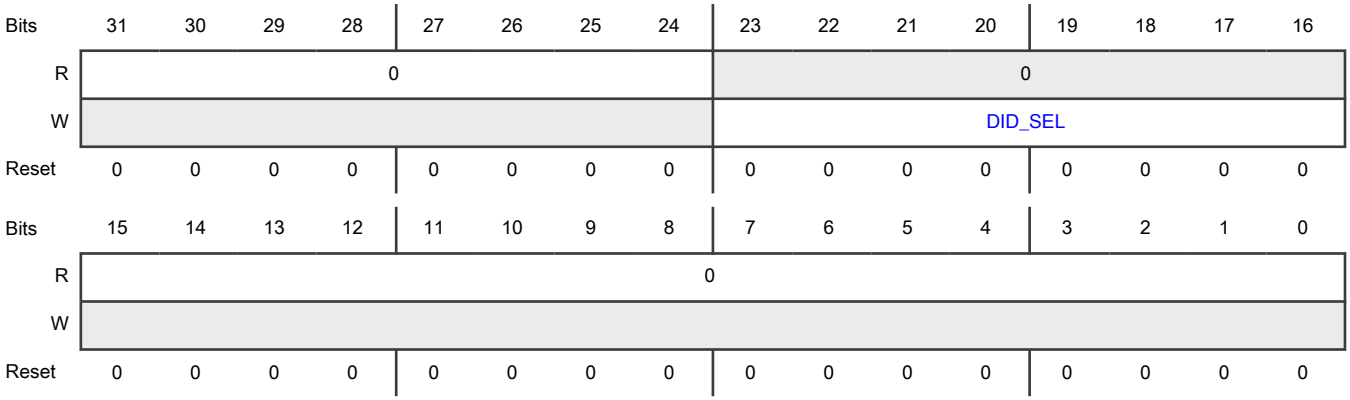
Offset

Register	Offset
MRC0_NSE_RGN_CLR_ALL	401Ch

Function

A write to this register clears all the region NSE bits for the selected domains defined in the write data.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 DID_SEL	DID Select Destination domain bitmap select. • Bit [23] = 1 clear NSE bits for domain 7 • Bit [22] = 1 clear NSE bits for domain 6 • Bit [21] = 1 clear NSE bits for domain 5 • Bit [20] = 1 clear NSE bits for domain 4 • Bit [19] = 1 clear NSE bits for domain 3 • Bit [18] = 1 clear NSE bits for domain 2 • Bit [17] = 1 clear NSE bits for domain 1 • Bit [16] = 1 clear NSE bits for domain 0
15-0 —	Reserved

9.7.1.48 MRC Global Access Control (MRC0_GLBAC0)

Offset

Register	Offset
MRC0_GLBAC0	4020h

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these contol access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10 SUR	SecureUser Read SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.49 MRC Global Access Control (MRC0_GLBAC1 - MRC0_GLBAC7)

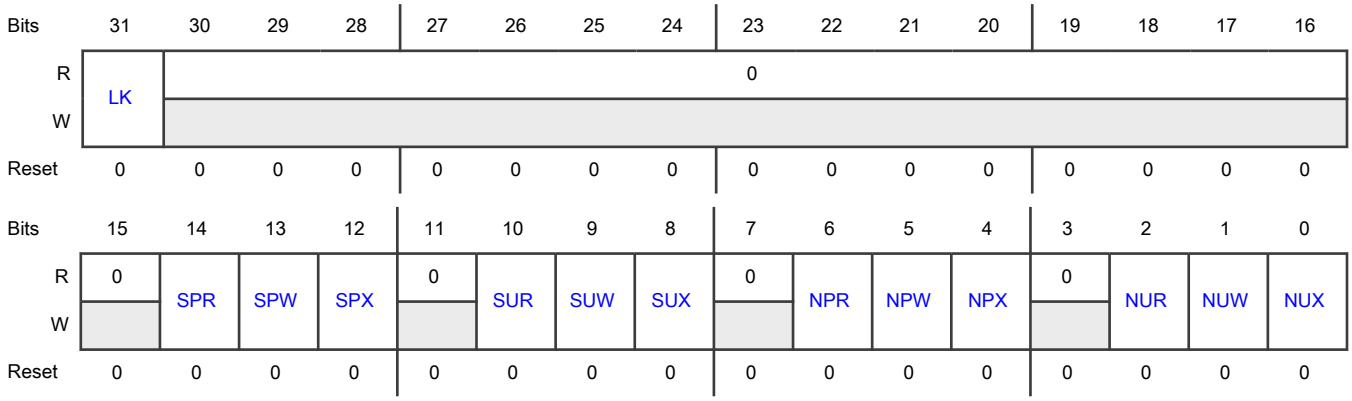
Offset

Register	Offset
MRC0_GLBAC1	4024h
MRC0_GLBAC2	4028h
MRC0_GLBAC3	402Ch
MRC0_GLBAC4	4030h
MRC0_GLBAC5	4034h
MRC0_GLBAC6	4038h
MRC0_GLBAC7	403Ch

Function

These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, NonsecureUser. The lock bit provides the ability to lock these contol access flags, to lock the MRC_BLK_CFG[MRACSEL] field, and to lock the MRC_RGD[STRT_ADDR, END_ADDR, VLD] fields when a locked MRC_GLBAC register is selected.

Diagram



Fields

Field	Function
31 LK	LOCK This 1-bit field provides a mechanism to limit writes to the this register to protect its contents. Once set, this bit remains asserted until the next reset. 0b - This register is not locked and can be altered. 1b - This register is locked (read-only) and cannot be altered.
30-15 —	Reserved
14 SPR	SecurePriv Read SecurePriv read access control flag 0b - Read access is not allowed in Secure Privilege mode. 1b - Read access is allowed in Secure Privilege mode.
13 SPW	SecurePriv Write SecurePriv write access control flag 0b - Write access is not allowed in Secure Privilege mode. 1b - Write access is allowed in Secure Privilege mode.
12 SPX	SecurePriv Execute SecurePriv execute access control flag 0b - Execute access is not allowed in Secure Privilege mode. 1b - Execute access is allowed in Secure Privilege mode.
11 —	Reserved
10	SecureUser Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SUR	SecureUser read access control flag 0b - Read access is not allowed in Secure User mode. 1b - Read access is allowed in Secure User mode.
9 SUW	SecureUser Write SecureUser write access control flag 0b - Write access is not allowed in Secure User mode. 1b - Write access is allowed in Secure User mode.
8 SUX	SecureUser Execute SecureUser execute access control flag 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6 NPR	NonsecurePriv Read NonsecurePriv read access control flag 0b - Read access is not allowed in Nonsecure Privilege mode. 1b - Read access is allowed in Nonsecure Privilege mode.
5 NPW	NonsecurePriv Write NonsecurePriv write access control flag 0b - Write access is not allowed in Nonsecure Privilege mode. 1b - Write access is allowed in Nonsecure Privilege mode.
4 NPX	NonsecurePriv Execute NonsecurePriv execute access control flag 0b - Execute access is not allowed in Nonsecure Privilege mode. 1b - Execute access is allowed in Nonsecure Privilege mode.
3 —	Reserved
2 NUR	NonsecureUser Read NonsecureUser read access control flag 0b - Read access is not allowed in Nonsecure User mode. 1b - Read access is allowed in Nonsecure User mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 NUW	NonsecureUser Write NonsecureUser write access control flag 0b - Write access is not allowed in Nonsecure User mode. 1b - Write access is allowed in Nonsecure User mode.
0 NUX	NonsecureUser Execute NonsecureUser execute access control flag 0b - Execute access is not allowed in Nonsecure User mode. 1b - Execute access is allowed in Nonsecure User mode.

9.7.1.50 MRC Region Descriptor Word 0 (MRC0_DOM0_RGD0_W0 - MRC0_DOM2_RGD7_W0)

Offset

For d = 0 to 2; r = 0 to 7:

Register	Offset
MRC0_DOMd_RGDr_W0	4040h + (d × 100h) + (r × 8h)

Function

MRC[m]_DOM[d]_RGD[r]_W[w], where,

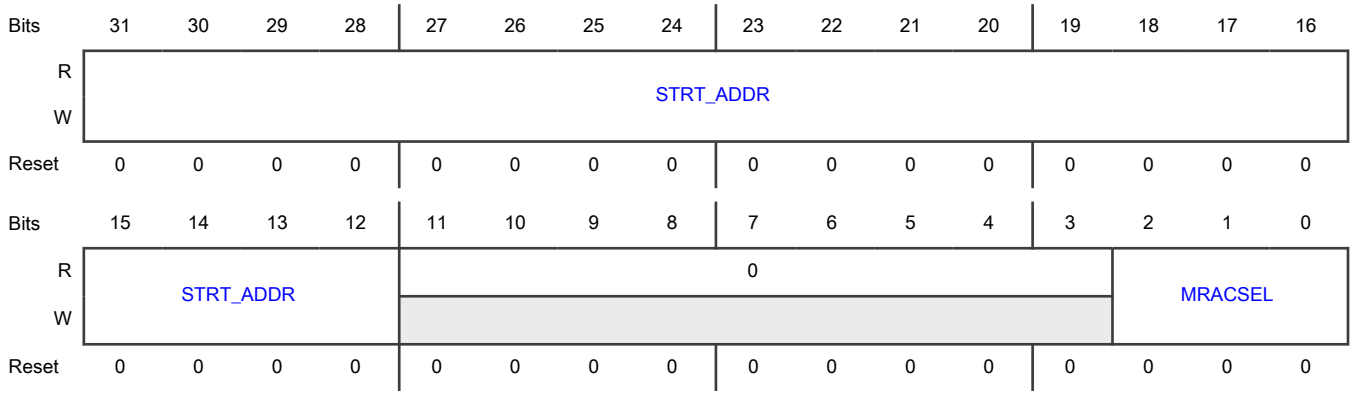
- m - mrc index
- d - domain index
- r - region descriptor index
- w - word index

The MRC[m]_DOM[d]_RGD[r]_W[w] registers provide a 3-dimensional data structure for defining access control policies per domain for each supported memory region.

Memory map/register definition:

There two word-size registers (W[w]) defined in each memory region. There are up to 8 memory region checkers (MRCs), m, each supporting up to 16 memory region descriptors (RGD), r. The number of MRCs is defined by HWCFG0[NMRC].

Diagram



Fields

Field	Function
31-12 STRT_ADDR	Start Address 0-mod-4K physical start address [31:12] of region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.
11-3 —	Reserved
2-0 MRACSEL	Memory Region Access Control Select This field selects the global access control associated with region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1. 000b - Select MRC_GLBAC0 access control policy 001b - Select MRC_GLBAC1 access control policy 010b - Select MRC_GLBAC2 access control policy 011b - Select MRC_GLBAC3 access control policy 100b - Select MRC_GLBAC4 access control policy 101b - Select MRC_GLBAC5 access control policy 110b - Select MRC_GLBAC6 access control policy 111b - Select MRC_GLBAC7 access control policy

9.7.1.51 MRC Region Descriptor Word 1 (MRC0_DOM0_RGD0_W1 - MRC0_DOM2_RGD7_W1)

Offset

For d = 0 to 2; r = 0 to 7:

Register	Offset
MRC0_DOMd_RGDr_W1	4044h + (d × 100h) + (r × 8h)

Function

MRC[m]_DOM[d]_RGD[r]_W[w], where,

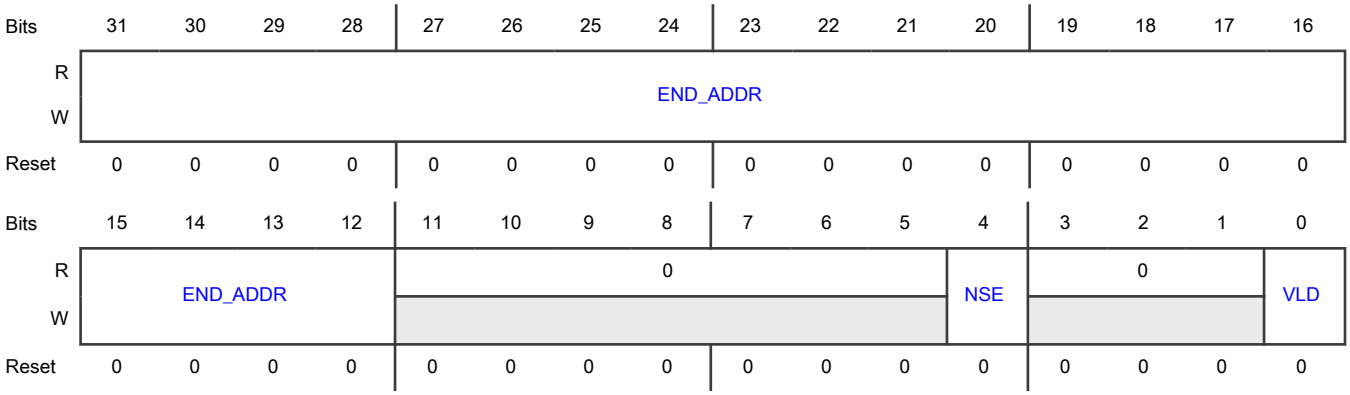
- m - mrc index
- d - domain index
- r - region descriptor index
- w - word index

The MRC[m]_[d]_rgd[r]_w[w] registers provide a 3-dimensional data structure for defining access control policies per domain for each supported memory region.

Memory map/register definition:

There two word-size registers (W[w]) defined in each memory region. There are up to 8 memory region checkers (MRCs), m, each supporting up to 16 region descriptors (RGD), r. The number of MRCs is defined by HWCFG0[NMRC].

Diagram



Fields

Field	Function
31-12 END_ADDR	End Address (4K-1)-mod-4K physical end address [31:12] of region r. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.
11-5 —	Reserved
4	NonSecure Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
NSE	Alternate view of NSE flag for Region r. 0b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]). 1b - Secure accesses to region r are are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in this register (MRCm_DOMd_RGDr_Ww[MRACSEL]).
3-1 —	Reserved
0 VLD	Valid If set, this region is valid and accesses are checked for this region. **NOTE** This field is locked (read-only) if MRACSEL = r and MRC_GLBACr[LK] = 1.

9.7.1.52 MRC Region Descriptor NonSecure Enable (MRC0_DOM0_RGD_NSE - MRC0_DOM2_RGD_NSE)

Offset

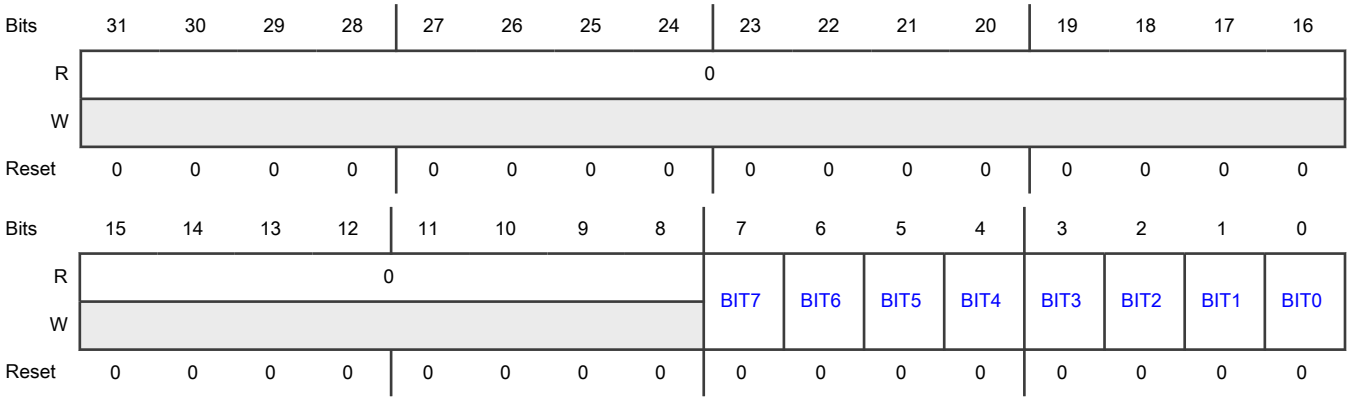
Register	Offset
MRC0_DOM0_RGD_NSE	40C0h
MRC0_DOM1_RGD_NSE	41C0h
MRC0_DOM2_RGD_NSE	42C0h

Function

This register is the bitmap of the individual NSE bits for the region descriptors. Bit n corresponds to region n.

These bits can also be written or "indirectly" through writes to NSE_SET, NSE_CLR, NSE_CLR_ALL, and RGD_W0[NSE] registers.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 BITn	Bit n NonSecure Enable [n = 0 - 15] 0b - Secure accesses to region r are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in register (MRCm_DOMd_RGDr_Ww[MRACSEL]). 1b - Secure accesses to region r are are not allowed, nonsecure accesses to region r are based on corresponding MRACSEL field in register (MRCm_DOMd_RGDr_Ww[MRACSEL]).

Chapter 10
Enhanced Direct Memory Access (eDMA)

10.1 Chip-specific eDMA information

Table 37. Reference links to related information

Topic	Related module	Reference
Full description	eDMA	eDMA
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

10.1.1 Module instances

This device has one instance of the eDMA module.

10.1.2 Direct Memory Access Multiplexer (DMAMUX)

10.1.2.1 DMAMUX0 request sources

The peripheral request line for each channel of the Enhanced Direct Memory Access Controller 0 (eDMA0) is driven from a Direct Memory Access Multiplexer (DMAMUX). This is a flexible configuration that allows the user to select the appropriate peripheral to connect to each channel of the DMA Controller. The DMAMUX for this device allows up to 64 DMA request signals to be mapped to each channel. Because of the mux, there is not a correlation between any of the DMA request sources and a specific DMA channel.

The DMAMUX is an integrated component of the DMA Controller. This allows each DMA channel and DMA Multiplexer to configured as secure or non-secure on initial configuration of the device and not be changed without the correct permissions. For more information, please refer to the DMA Controller (DMA) chapter of the reference manual.

Table 38. DMAMUX0 request assignments

DMAMUX0 number	Alias	Source description
0	—	Disabled
System modules		
1	WUU0	Wake up event
Security modules		
2	EdgeLock enclave	Data request
Timer modules		
3	LPTMR0	Counter match event
4	LPTMR1	Counter match event
5	TPM0	Channel 0 request
6	TPM0	Channel 1 request
7	TPM0	Channel 2 request

Table continues on the next page...

Table 38. DMAMUX0 request assignments (continued)

DMAMUX0 number	Alias	Source description
8	TPM0	Channel 3 request
9	TPM0	Channel 4 request
10	TPM0	Channel 5 request
11	TPM0	Counter overflow request
12	TPM1	Channel 0 request
13	TPM1	Channel 1 request
14	TPM1	Channel 2 request
15	TPM1	Channel 3 request
16	TPM1	Channel 4 request
17	TPM1	Channel 5 request
18	TPM1	Counter overflow request
RF modules		
19	RF	Radio Bric Input data request
20	RF	Radio Bric Output data request
Communication modules		
21	LPI2C0	Master / Slave receive request
22	LPI2C0	Master / Slave transmit request
23	LPI2C1	Master / Slave receive request
24	LPI2C1	Master / Slave transmit request
25	I3C0	Master / Slave receive request
26	I3C0	Master / Slave transmit request
27	LPSPI0	Master/Slave receive request
28	LPSPI0	Master/Slave transmit request
29	LPSPI1	Master/Slave receive request
30	LPSPI1	Master/Slave transmit request
31	LPUART0	Receive request
32	LPUART0	Transmit request
33	LPUART1	Receive request
34	LPUART1	Transmit request
35	FLEXIO0	Shift register 0 request
36	FLEXIO0	Shift register 1 request
37	FLEXIO0	Shift register 2 request

Table continues on the next page...

Table 38. DMAMUX0 request assignments (continued)

DMAMUX0 number	Alias	Source description
38	FLEXIO0	Shift register 3 request
39	FLEXIO0	Shift register 4 request
40	FLEXIO0	Shift register 5 request
41	FLEXIO0	Shift register 6 request
42	FLEXIO0	Shift register 7 request
43	CAN0[1]	DMA request
HMI modules		
44	GPIOA	Pin event request 0
45	GPIOA	Pin event request 1
46	GPIOB	Pin event request 0
47	GPIOB	Pin event request 1
48	GPIOC	Pin event request 0
49	GPIOC	Pin event request 1
50	GPIOD	Pin event request 0
51	GPIOD	Pin event request 1
Analog modules		
52	ADC-GP0	FIFO A request
53	ADC-GP0	FIFO B request
54	CMP-GP0	DMA request
55	CMP-GP1	DMA request
56 - 63	—	—

1. Refer to Ordering information for parts have this module.

10.2 Overview

The enhanced direct memory access (eDMA) controller is capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 16 channels

10.2.1 Block diagram

Figure 12 illustrates the components of the eDMA system, including the eDMA module (engine).

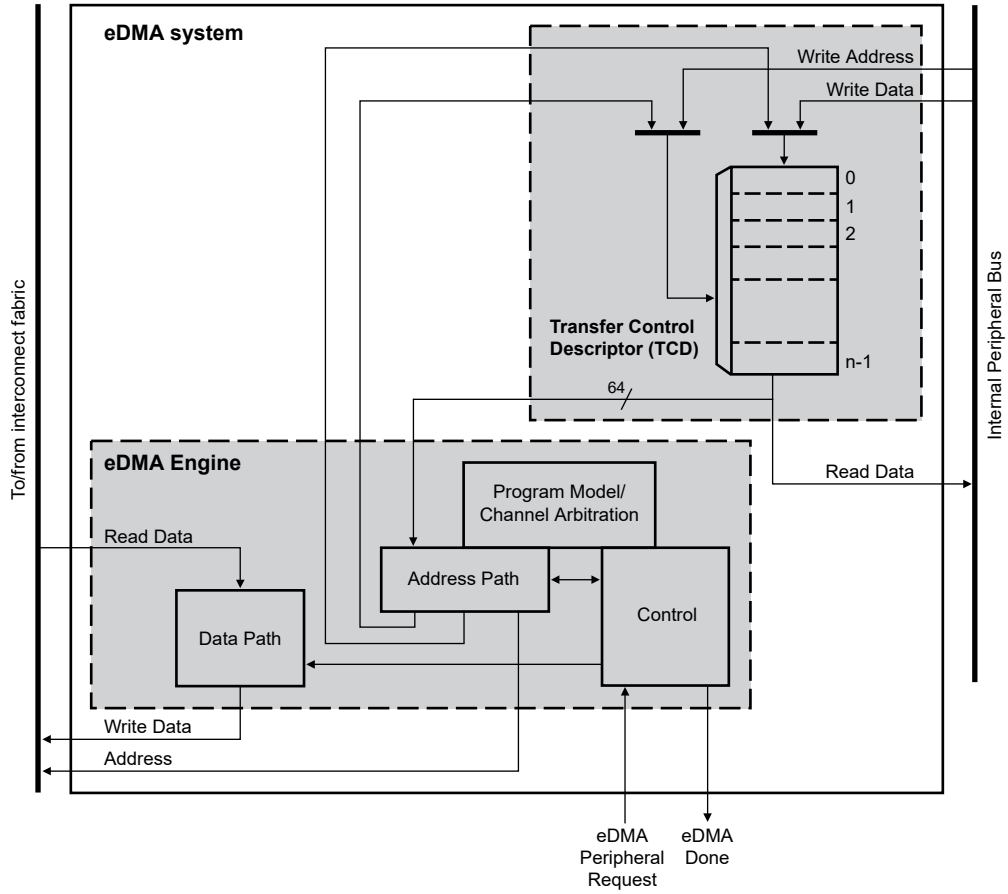


Figure 12. Block diagram

10.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 39. eDMA engine submodules

Submodule	Function
Address path	This block: • Implements a primary channel and secondary (preempt) channel • Manages all master bus-address calculations All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the primary channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by CH*n*_PRI[ECP]) where a large data transfer can be preempted to minimize the time another channel is blocked from execution.

Table continues on the next page...

Table 39. eDMA engine submodules (continued)

Submodule	Function
	When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of a new TCD*n* from memory as part of a scatter/gather operation. See Dynamic scatter/gather for more details.
Data path	This block implements the bus master read/write data path. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the first stage of the bus pipeline (address phase), and the data path module implements the second stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has been moved from the source to the destination. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, the eDMA performs two reads, then one 32-bit write.

The transfer control descriptor local memory is further partitioned into:

Table 40. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, and manages accesses from the eDMA engine as well as references from the internal peripheral bus. In simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

10.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and is not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for complex address calculations
- 16-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage for all transfers

  - Connections to the crossbar switch for bus mastering the data movement
- TCD organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel that are logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

10.3 Functional description

The operation of eDMA is described in the following subsections.

10.3.1 Modes of operation

The eDMA operates in the following modes:

Table 41. Modes of operation

Mode	Description
Normal	In Normal mode, eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the TCD. The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	eDMA operation is configurable in Debug mode via the control register: • If CSR[EDBG] is cleared to 0, eDMA continues to operate. • If CSR[EDBG] is set to 1, eDMA stops transferring data. If Debug mode is entered when a channel is active, eDMA continues operation until the channel retires.

10.3.2 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

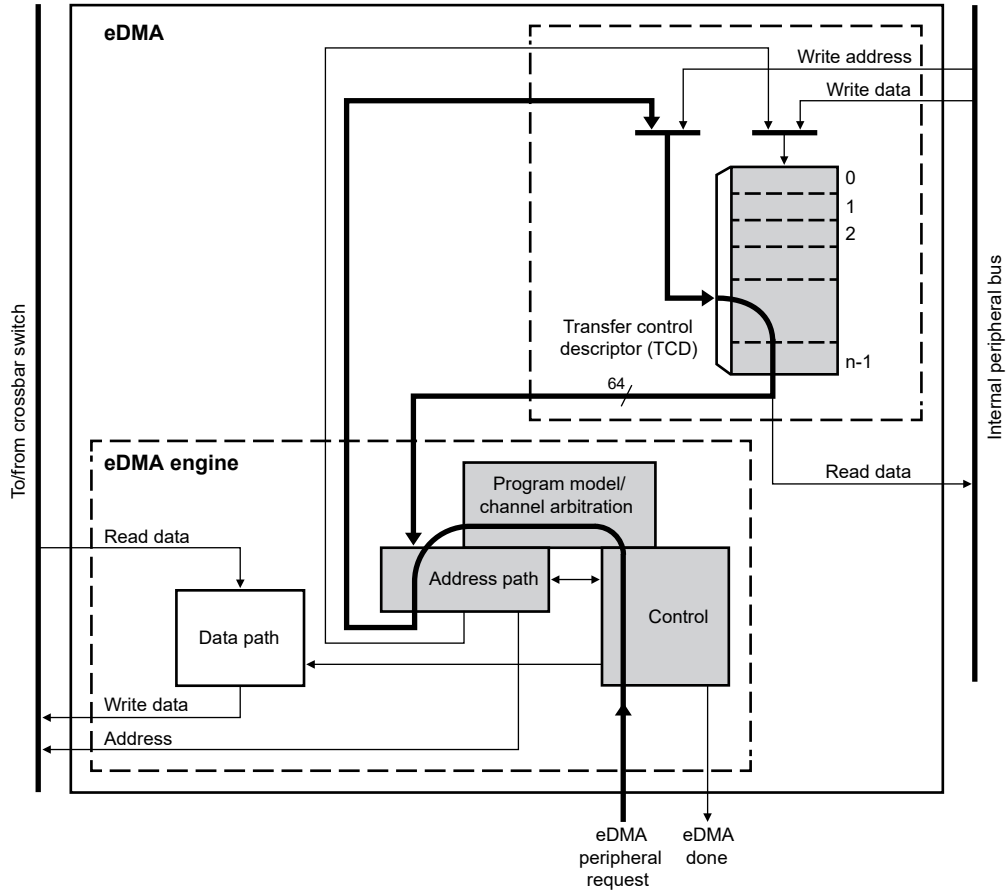


Figure 13. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] field follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration.

In the next cycle, the channel arbitration begins using fixed-priority plus the optional round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor is read from the local memory and then loaded into the eDMA engine address path's primary or secondary channel execution registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path registers.

The following diagram illustrates the second part of the basic data flow:

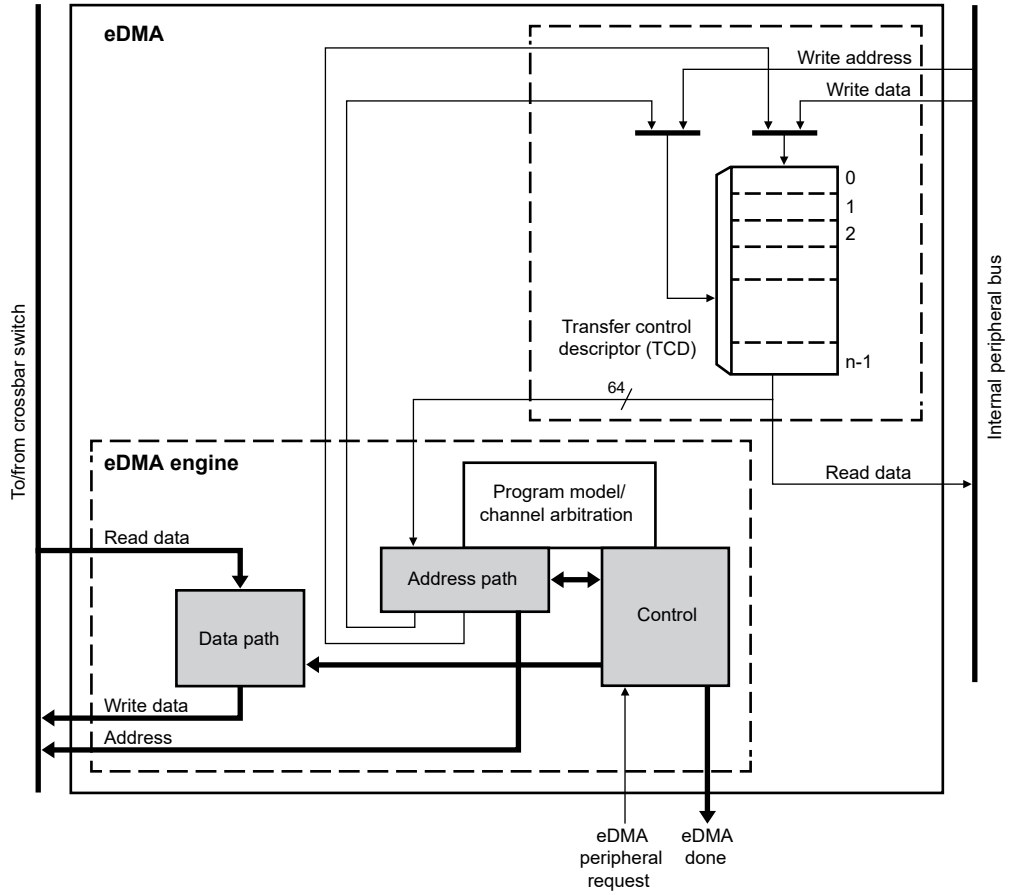


Figure 14. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) go through the required sequence of source reads and destination writes to perform the actual data movement. The source reads are initiated, and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the byte count, NBYTES, has been transferred.

After NBYTES of data has been moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD (for example, SADDR, DADDR, CITER). If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER field. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

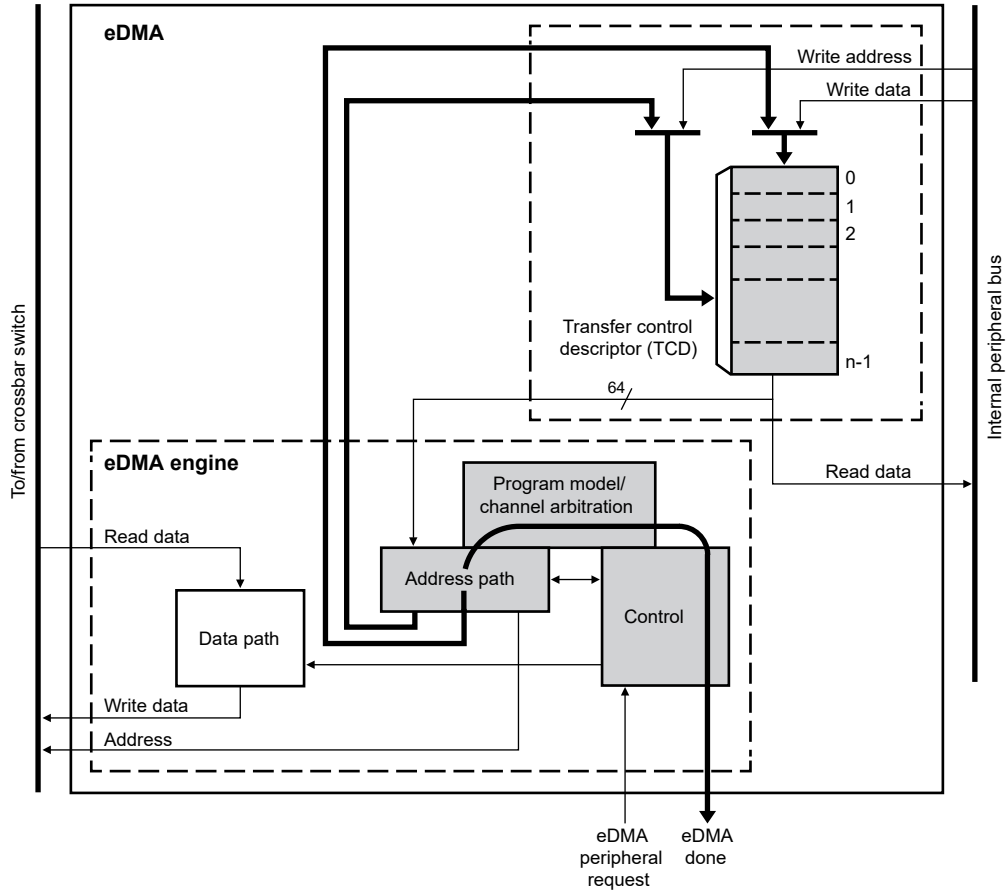


Figure 15. eDMA operation, part 3

10.3.3 Fault reporting and handling

Channel errors are reported in the Error Status register (CHn_ES) and can be caused by any of the following:

- A configuration error, which is an illegal setting in the transfer control descriptor
- An active channel canceled via a "cancel transfer with error" hardware or software request
- An error termination to a bus master read or write cycle

A configuration error is reported when an inconsistent state is represented by one of these factors:

- Starting source or destination address
- Source or destination offsets
- Minor loop byte count
- Transfer size

Each of these possible causes is detailed below:

- The addresses and offsets must be aligned on zero-modulo-transfer-sized boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size.

NOTE

To aid in debugging, set the Halt After Error field in the DMA's Control Status register, CSR[HAE]. Upon any error condition, the DMA is halted after the error is recorded. The DMA remains halted and does not process any channel service requests. After the error is fixed, the DMA may be enabled again by clearing the Halt field, CSR[HALT].

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (TCDn_DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[ELINK] field does not equal the TCDn_BITER[ELINK] field.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, are reported as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion if properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel field in the eDMA error register is set to 1. At the same time, the details of the error condition are loaded into the Error Status register (CHn_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag, and the possible assertion of an interrupt request are not affected when an error is detected.

After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

The error status fields are read-only. These error indicators are sticky and cannot be cleared. They show the last recorded error until the DMA is reset. CHn_ES[ERR] is used to determine if a new error condition exists. This field is the logical OR of each channel's error interrupt field (ERR).

After the software has resolved all errors and cleared all of the error interrupt fields, the MP_ES[VLD] is cleared to 0 but the cause of the last error is still indicated.

10.3.4 Channel preemption

The eDMA uses a priority vector value to determine the highest priority channel requesting service.

The priority vector is a combination of:

1. the channel's group priority, CHn_GRPRI
2. the channel's priority level, CHn_PRI[APL]
3. the channel number

Priority vector = ((CHn_GRPRI << 8) + (CHn_PRI[APL] << 5) + CHn_*)

A channel requesting service with the highest priority vector value will receive the next execution slot.

An execution slot is available:

1. immediately if the eDMA is idle
2. when an active channel retires
3. when valid preemption conditions exist

NOTE

Preemption is strictly priority based. Preemption is not bound by a specific group number as defined by CHn_GRPRI.

Channel preemption is enabled on a per-channel basis by setting the CHn_PRI[ECP] field. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher-priority channel. After the preempting channel has completed all of its minor loop data transfers, the preempted channel is restored and resumes execution.

After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended, and the higher-priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted.

A channel's ability to preempt another channel can be disabled by setting CHn_PRI[DPA] to 1. When a channel's preempt ability is disabled, that channel cannot suspend a lower-priority channel's data transfer, regardless of the lower-priority channel's ECP setting. This allows for a pool of low-priority, large-data-moving channels to be defined.

You can configure these low-priority channels to not preempt each other, thus preventing a low-priority channel from consuming the preempt slot normally available to a true high-priority channel. When you enable round-robin channel arbitration mode (CSR[ERCA] is set to 1), any channel with a priority level equal to 0 (CHn_PRI[APL] = 0) has preemption disabled and cannot preempt another channel.

10.3.5 Clocking

This module has no clocking considerations.

10.3.6 Interrupts

This module has no interrupts.

10.4 Memory map/register definition

The eDMA programming model is partitioned into three parts:

1. The first part defines a number of registers providing overall control functions and is known as the management page.
2. The second part corresponds to the channel (CH) control, status, and configuration.
3. The third part corresponds to the local TCD memory.

TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the data movement operation. Each TCDn definition is presented as 11 registers of 16 or 32 bits. See "TCD memory map" for details.

TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

TCD structure

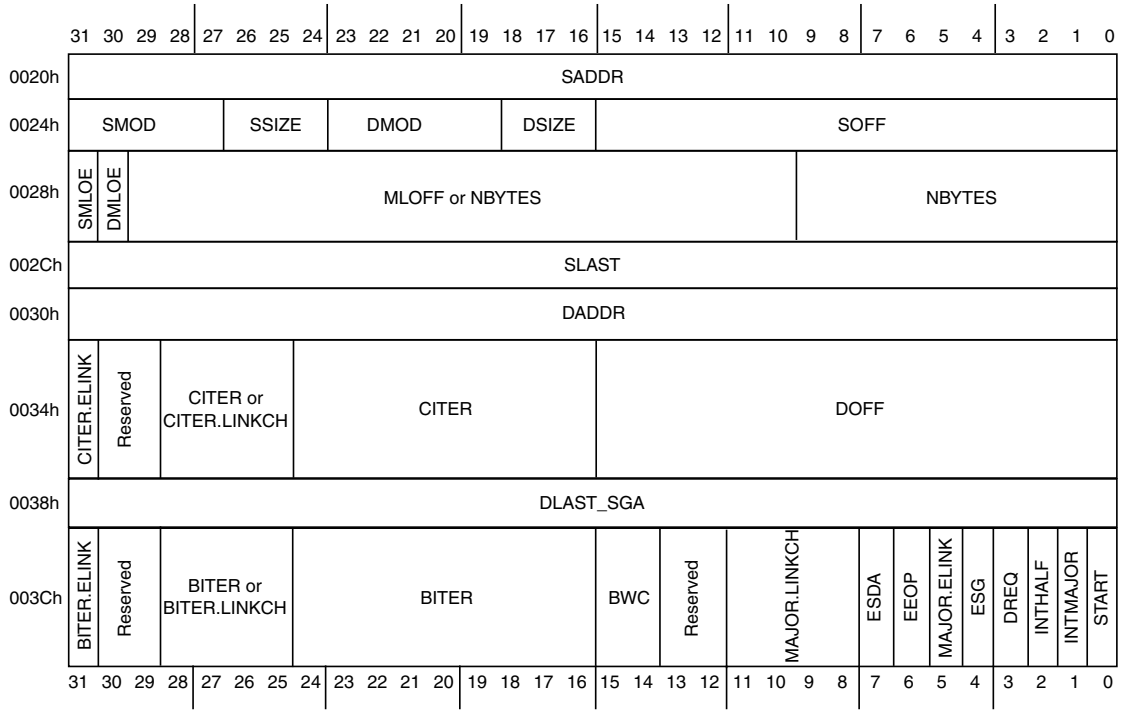


Figure 16. TCD structure

Accesses to reserved memory and fields

- Reading reserved fields in a register returns the value of zero.
- Writes to reserved fields in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

10.4.1 DMA MP register descriptions

10.4.1.1 MP memory map

DMA3.MP base address: 4000_2000h

NOTE

For registers in the following table with *Protection*, see the REG_PROT details for more information.

Offset	Register	Width (In bits)	Access	Reset value	Protection
0h	Management Page Control (MP_CSR)	32	RW	0031_0000h	Yes
4h	Management Page Error Status (MP_ES)	32	R	0000_0000h	No
8h	Management Page Interrupt Request Status (MP_INT)	32	R	0000_0000h	No

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value	Protection
Ch	Management Page Hardware Request Status (MP_HRS)	32	R	0000_0000h	No
100h - 13Ch	Channel Arbitration Group (CH0_GRPRI - CH15_GRPRI)	32	RW	0000_0000h	Yes

10.4.1.2 Management Page Control (MP_CSR)

Offset

Register	Offset
MP_CSR	0h

Function

The Management Page Control register defines the basic operating configuration of the DMA.

Arbitration uses a two-tier priority system; group and channel priority. The eDMA assigns each channel to a priority group. Group arbitration is fixed-priority and cannot be changed. Channel arbitration uses fixed priority and may be configured to use a selective round-robin scheme for specified channels within each priority group. For fixed-priority arbitration, eDMA selects for execution the highest priority channel requesting service in the highest priority arbitration group.

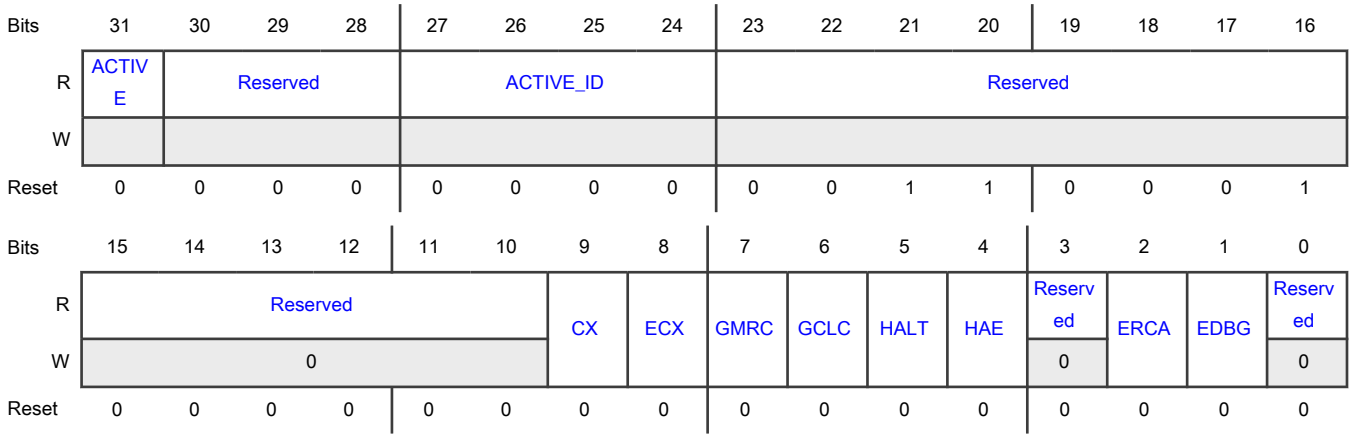
The channel priority registers assign the relative priorities within each arbitration group; see CHn_PRI. All channels with a non-zero CHn_PRI value use fixed-priority arbitration.

When you enable round-robin arbitration, all channels with channel priority set to zero do not have a priority and, of those channels requesting service, are cycled through (from high to low channel number) without regard to priority relative to each other within the same priority group. Any channel with a non-zero CHn_PRI value automatically has a higher priority over the round-robin channels. A channel's priority group is assigned in Channel Arbitration Group (CH0_GRPRI - CH15_GRPRI).

NOTE

For correct operation, changes to the CSR[ERCA, GCLC, GMRC] fields must be performed when the DMA channels are inactive; that is, when the CSR[ACTIVE] field is 0.

Diagram



Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel
30-28 —	Reserved
27-24 ACTIVE_ID	Active Channel ID This field identifies the channel number that is executing when the ACTIVE bit is 1.
23-16 —	Reserved
15-10 —	Reserved
9 CX	Cancel Transfer When set to 1, this field cancels the remaining data transfer, stops the executing channel, and forces the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. CX clears itself to 0 after the cancel has been honored. This cancel retires the channel normally as if the minor loop had been completed. 0b - Normal operation 1b - Cancel the remaining data transfer
8 ECX	Cancel Transfer With Error Cancellation of the remaining data transfer is similar to that of the CX field. Execution of the channel is stopped and the minor loop is forced to finish. The cancellation takes effect after the last write of the current read/write sequence. The ECX field clears itself to 0 after the cancel is honored. In addition to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	cancelling the transfer, ECX treats the cancel as an error condition, thus updating Management Page Error Status (MP_ES) and generating an optional error interrupt. 0b - Normal operation 1b - Cancel the remaining data transfer
7 GMRC	Global Master ID Replication Control **NOTE** If master ID replication is disabled, the nonsecure, user protection level for DMA transfers is used. 0b - Master ID replication disabled for all channels 1b - Master ID replication available and controlled by each channel's CHn_SBR[EMI] setting
6 GCLC	Global Channel Linking Control 0b - Channel linking disabled for all channels 1b - Channel linking available and controlled by each channel's link settings
5 HALT	Halt DMA Operations This field stalls the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this field is cleared to 0. 0b - Normal operation 1b - Stall the start of any new channels
4 HAE	Halt After Error When this field is set to 1, any error causes the HALT field to be set to 1. Then all service requests are ignored until the HALT field is cleared to 0. 0b - Normal operation 1b - Any error causes the HALT field to be set to 1
3 —	Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0b - Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection 1b - Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection
1 EDBG	Enable Debug When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. DMA resumes channel execution when the system exits debug mode or clears the EDBG field to 0. 0b - Debug mode disabled. When in debug mode, the DMA continues to operate

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Debug mode is enabled. When in debug mode, the DMA stalls the start of a new channel
0 —	Reserved

10.4.1.3 Management Page Error Status (MP_ES)

Offset

Register	Offset
MP_ES	4h

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle
- An uncorrectable error that occurred when the device was accessing the TCD SRAM
- A "cancel transfer with error" request was made via the corresponding cancel transfer field or input signal

Upon any error condition, the software must initialize the TCD of the channel that contains the error, as it is in an incomplete state after an error. See Fault reporting and handling for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	Reserved			ERRCHN				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							ECX	SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Valid Logical OR of all ERR status fields.

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	0b - No ERR fields are set to 1 1b - At least one ERR field is set to 1, indicating a valid error exists that software has not cleared
30-28 —	Reserved
27-24 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error or last recorded error-canceled transfer.
23-9 —	Reserved
8 ECX	Transfer Canceled The ECX operation is a management page function. When employed, the targeted channel's CHn_ES register reports an unspecified error; that is, only the ERR field is set to 1. The management page has full view of the error condition. 0b - No canceled transfers 1b - Last recorded entry was a canceled transfer by the error cancel transfer input
7 SAE	Source Address Error When this field is 1, it indicates that TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error When this field is 1, it indicates that TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error When this field is 1, it indicates that TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error When this field is 1, it indicates that TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3	NBYTES/CITER Configuration Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE	This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - The last recorded error was NBYTES equal to zero or a CITER not equal to BITER error. Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was a bus error on a destination write

10.4.1.4 Management Page Interrupt Request Status (MP_INT)

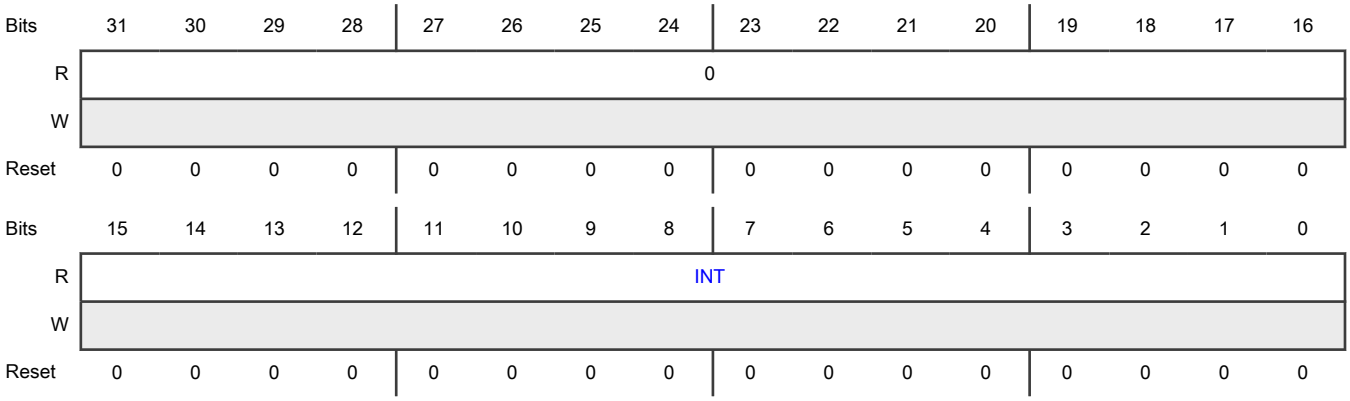
Offset

Register	Offset
MP_INT	8h

Function

This register shows the current state of the interrupt service requests for all eDMA channels.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 INT	Interrupt Request Status The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate field setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion or an error condition. The eDMA routes channel interrupt requests to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate field in the channel's interrupt request register, CHn_INT, thus negating the interrupt request. 0b - Interrupt request for corresponding channel not present 1b - Interrupt request for corresponding channel present

10.4.1.5 Management Page Hardware Request Status (MP_HRS)

Offset

Register	Offset
MP_HRS	Ch

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by eDMA's arbitration logic.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HRS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HRS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 HRS	Hardware Request Status The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel. 0b - Hardware service request for corresponding channel is not present 1b - Hardware service request for corresponding channel is present

10.4.1.6 Channel Arbitration Group (CH0_GRPRI - CH15_GRPRI)

Offset

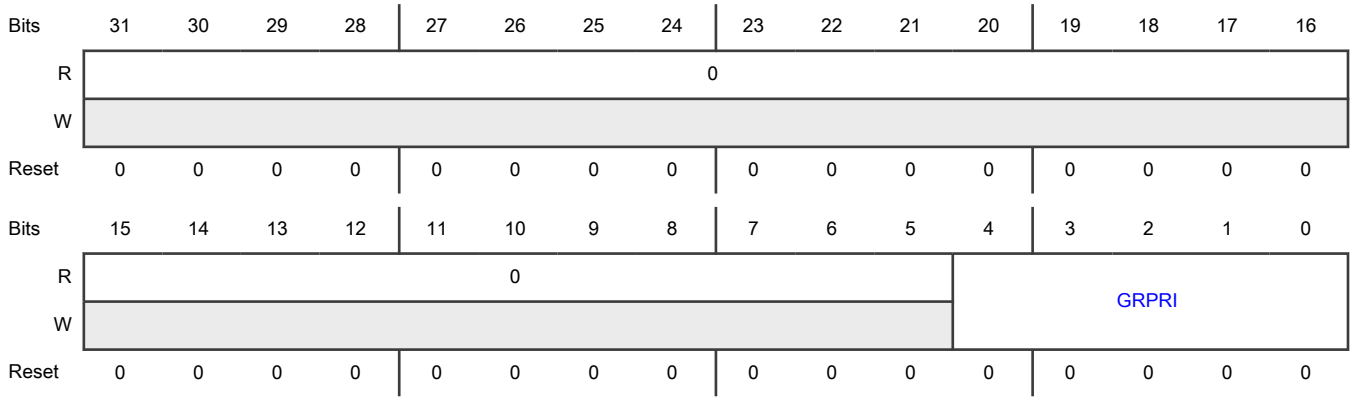
For n = 0 to 15:

Register	Offset
CHn_GRPRI	100h + (n × 4h)

Function

The contents of this register define the arbitration group associated with each channel. Using a fixed-priority group arbitration scheme, eDMA evaluates the arbitration group priorities by numeric value from highest group number to lowest; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. The range of the group priority values is limited to the values of 0 through 31. Within each arbitration group, the channel priority assignment CH*n*_PRI determines the highest-priority channel.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 GRPRI	Arbitration Group For Channel n Fixed-priority arbitration group number.

10.4.2 DMA TCD register descriptions

10.4.2.1 TCD memory map

DMA3.TCD base address: 4000_3000h

NOTE

For registers in the following table with *Protection*, see the REG_PROT details for more information.

Offset	Register	Width (In bits)	Access	Reset value	Protection
0h - F000h	Channel Control and Status (CH0_CSR - CH15_CSR)	32	RW	0000_0000h	Yes
4h - F004h	Channel Error Status (CH0_ES - CH15_ES)	32	RW	0000_0000h	No
8h - F008h	Channel Interrupt Status (CH0_INT - CH15_INT)	32	RW	0000_0000h	No
Ch - F00Ch	Channel System Bus (CH0_SBR - CH15_SBR)	32	RW	0000_0012h	Yes
10h - F010h	Channel Priority (CH0_PRI - CH15_PRI)	32	RW	0000_0000h	Yes
14h - F014h	Channel Multiplexor Configuration (CH0_MUX - CH15_MUX)	32	RW	0000_0000h	No

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value	Protection
20h - F020h	TCD Source Address (TCD0_SADDR - TCD15_SADDR)	32	RW	See section	Yes
24h - F024h	TCD Signed Source Address Offset (TCD0_SOFF - TCD15_SOFF)	16	RW	See section	Yes
26h - F026h	TCD Transfer Attributes (TCD0_ATTR - TCD15_ATTR)	16	RW	See section	Yes
28h - F028h	TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD15_NBYTES_MLOFFNO)	32	RW	See section	Yes
28h - F028h	TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD15_NBYTES_MLOFFYES)	32	RW	See section	No
2Ch - F02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD15_SLAST_SDA)	32	RW	See section	Yes
30h - F030h	TCD Destination Address (TCD0_DADDR - TCD15_DADDR)	32	RW	See section	Yes
34h - F034h	TCD Signed Destination Address Offset (TCD0_DOFF - TCD15_DOFF)	16	RW	See section	Yes
36h - F036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD15_CITER_ELINKNO)	16	RW	See section	Yes
36h - F036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD15_CITER_ELINKYES)	16	RW	See section	No
38h - F038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD15_DLAST_SGA)	32	RW	See section	Yes
3Ch - F03Ch	TCD Control and Status (TCD0_CSR - TCD15_CSR)	16	RW	See section	Yes
3Eh - F03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD15_BITER_ELINKNO)	16	RW	See section	Yes
3Eh - F03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD15_BITER_ELINKYES)	16	RW	See section	No

10.4.2.2 Channel Control and Status (CH0_CSR - CH15_CSR)

Offset

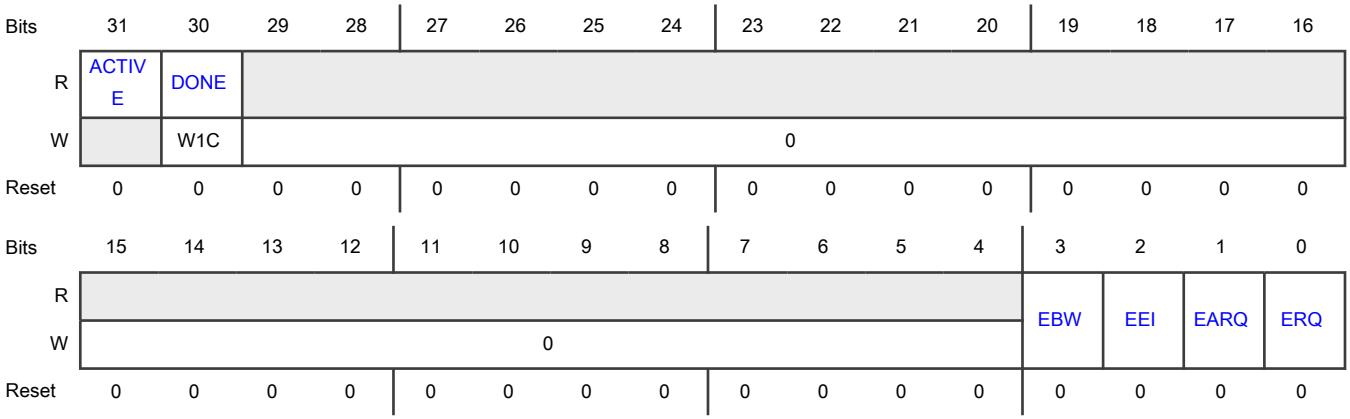
For n = 0 to 15:

Register	Offset
CHn_CSR	0h + (n × 1000h)

Function

This register contains several fields related to hardware and interrupt requests, configuration, and status for the given channel.

Diagram



Fields

Field	Function
31 ACTIVE	Channel Active The ACTIVE field indicates the channel was selected by arbitration and is executing the prescribed transfers. The eDMA sets it to 1 when channel service begins, and clears it to 0 as the minor loop completes or when any error condition is detected. Except for dynamic scatter/gather or dynamic channel linking, you must not modify the transfer control descriptor when a channel is active.
30 DONE	Channel Done The DONE field indicates the eDMA has completed the major loop. The eDMA engine sets this field as the CITER count reaches zero. If enabled, the eDMA generates an interrupt request corresponding to this completed channel. The software clears it, or the hardware clears it when the channel is activated. **NOTE** This field must be cleared to 0 before writing the MAJORELINK or ESG fields.
29-4 —	Reserved
3 EBW	Enable Buffered Writes When buffered writes are enabled, all writes except for the last write sequence of the minor loop are signaled by the eDMA as bufferable. 0b - Buffered writes on system bus disabled. Buffered writes on system bus disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Buffered writes on system bus enabled. Bufferable write signal asserted on all system bus writes except during last write sequence
2 EEI	Enable Error Interrupt The EEI field enables the error interrupt signal for the channel. The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller. 0b - Error signal for corresponding channel does not generate error interrupt 1b - Assertion of error signal for corresponding channel generates error interrupt request
1 EARQ	Enable Asynchronous DMA Request The enable asynchronous DMA request field (EARQ) does not affect DMA operations. When set to 1, this field allows the hardware service request enable field (ERQ) to propagate out of the DMA to the power controller. When cleared to 0, this field masks the hardware service request enable field to the power controller. 0b - Disable asynchronous DMA request for the channel 1b - Enable asynchronous DMA request for the channel
0 ERQ	Enable DMA Request Disable a channel's hardware service request at the source before clearing the channel's ERQ field. The DMA hardware request input signal and the enable request field (ERQ) must be asserted before a channel's hardware service request is accepted. The state of the eDMA enable request field does not affect a channel service request made explicitly through software or channel linking. The state of the ERQ field does not affect the channel's START field. 0b - DMA hardware request signal for corresponding channel disabled 1b - DMA hardware request signal for corresponding channel enabled

10.4.2.3 Channel Error Status (CH0_ES - CH15_ES)

Offset

For n = 0 to 15:

Register	Offset
CHn_ES	4h + (n × 1000h)

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle

The ERR field signals the presence of an error for the channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate field in this register. The outputs of this register are enabled by the contents of the CHn_CSR[EEI] field,

then logically summed across all channels to form an error interrupt request, which may be routed to the interrupt controller. In addition, this enabled error status is logically OR'd onto the channel done interrupt, CHn_INT[INT], thus forming a done or error interrupt on a per channel basis.

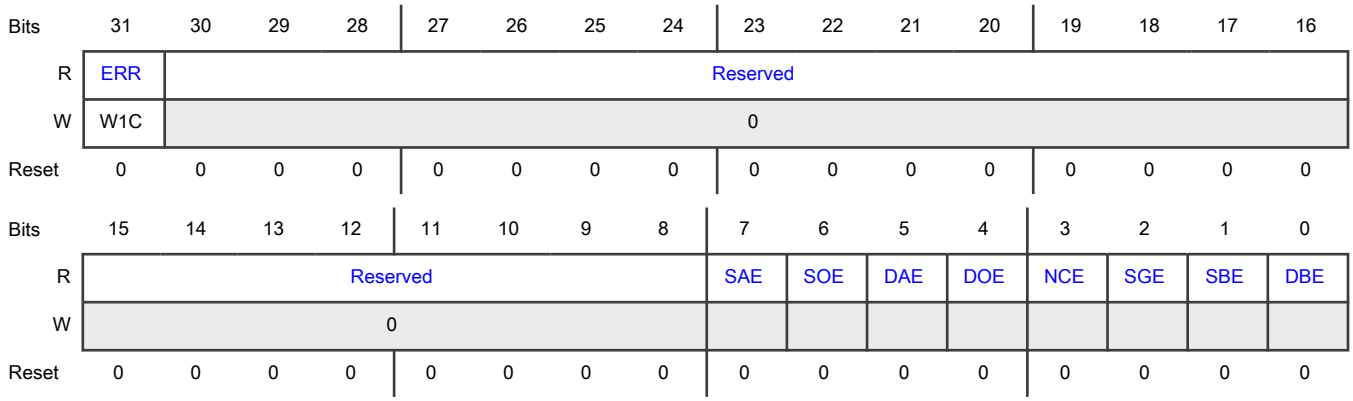
During the execution of the interrupt service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when eDMA detects an error. The contents of this ERR register field can also be polled because a non-zero value indicates the presence of a channel error, regardless of the state of the EEI mask.

The state of any given channel's error indicators is affected by writes to this register. Writing a 1 to the ERR field clears the channel's error status, and writing a 0 has no effect.

An unspecified error, where only the ERR field is set to 1, indicates that either a transfer was cancelled with an error. The Management Page Error Status register has full view of the error condition.

See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 ERR	Error In Channel 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
30-8 —	Reserved
7 SAE	Source Address Error TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG]is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was bus error on source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was bus error on destination write

10.4.2.4 Channel Interrupt Status (CH0_INT - CH15_INT)

Offset

For n = 0 to 15:

Register	Offset
CHn_INT	8h + (n × 1000h)

Function

The INT field signals the presence of an interrupt request for the channel. Depending on the appropriate bit setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion or an error condition.

The outputs of this register are directly routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. On writes to INT, a 1 clears the channel's interrupt request. A zero has no effect on the channel's current interrupt status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															INT
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 INT	Interrupt Request 0b - Interrupt request for corresponding channel cleared 1b - Interrupt request for corresponding channel active

10.4.2.5 Channel System Bus (CH0_SBR - CH15_SBR)

Offset

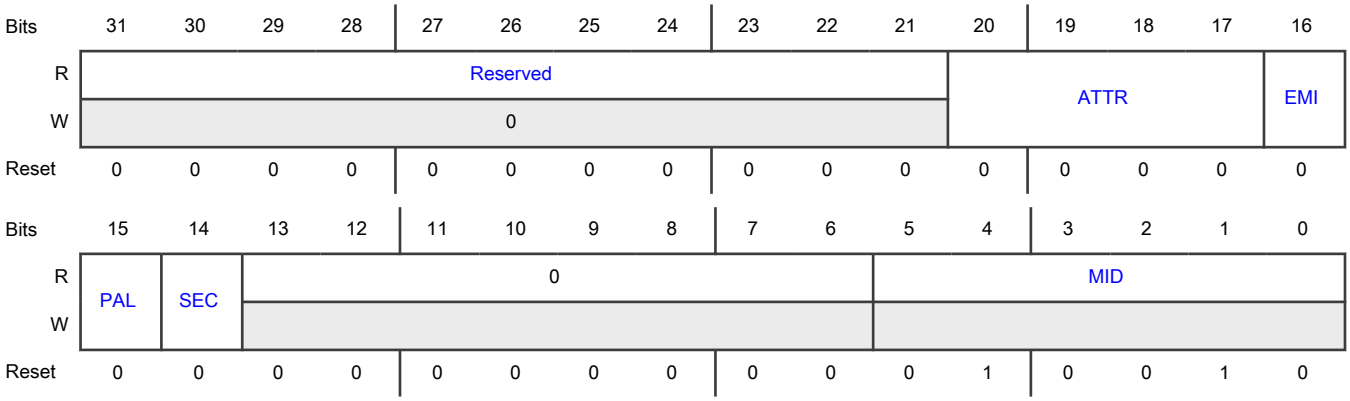
For n = 0 to 15:

Register	Offset
CHn_SBR	Ch + (n × 1000h)

Function

The Channel System Bus register places identification and attribute information on the system bus interface for the eDMA.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-17 ATTR	Attribute Output DMA's system bus attribute output value.
16 EMI	Enable Master ID Replication The eDMA master ID replication field allows the eDMA to use the same protection level and system bus ID of the master programming the eDMA's TCD. When enabled, the eDMA uses the master ID and protection level stored in the CHn_SBR registers, instead of the eDMA's default values. When a master (for example a core) programs a TCD, its master ID is captured when the TCD*n*_CSR control attributes are written. A scatter/gather operation does not affect the CHn_SBR registers. You can write the EMI only if MP_CSR[GMRC] = 1, which means Global Master ID Replication Control is enabled; otherwise, the EMI is forced to zero. **NOTE** If master ID replication is disabled, the nonsecure, user protection level for DMA transfers is used. 0b - Master ID replication is disabled 1b - Master ID replication is enabled
15	Privileged Access Level

Table continues on the next page...

Table continued from the previous page...

Field	Function
PAL	This field controls DMA's protection level on the system bus when the channel is active. **NOTE** The value written into this register cannot exceed the security and privilege level of the core or other master writing the channel's system bus register; CH*n*_SBR. The order of precedence is SecurePriv>SecureUser>NonsecurePriv>NonsecureUser 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14 SEC	Security Level DMA's security level on the system bus when the channel is active. **NOTE** The value written into this register cannot exceed the security and privilege level of the core or other master writing the channel's system bus register; CH*n*_SBR. The order of precedence is SecurePriv>SecureUser>NonsecurePriv>NonsecureUser 0b - Nonsecure protection level for DMA transfers 1b - Secure protection level for DMA transfers
13-6 —	Reserved
5-0 MID	Master ID This field controls the DMA's master ID on the system bus when the channel is active. **NOTE** The ID captured in this register reflects the master ID of the core or other master writing the channel's security attributes, TCD*n*_SBR[SEC].

10.4.2.6 Channel Priority (CH0_PRI - CH15_PRI)

Offset

For n = 0 to 15:

Register	Offset
CHn_PRI	10h + (n × 1000h)

Function

The contents of these registers define unique priorities associated with each channel within the same channel group. Channel grouping is programmed via Channel Arbitration Group (CH0_GRPRI - CH15_GRPRI).

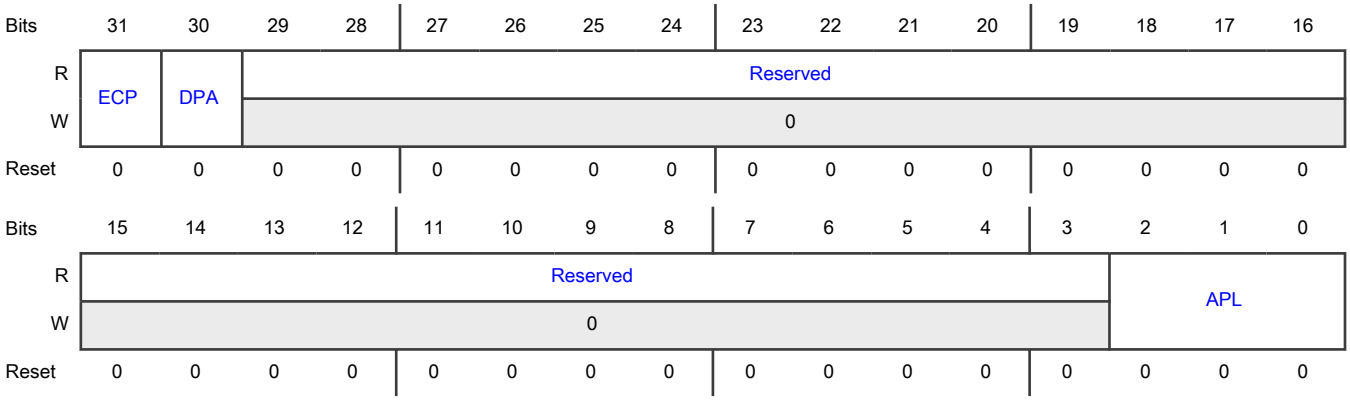
The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected based on channel number with the higher channel number having higher priority.

If more than one channel in a group has an arbitration priority level value of zero, then the arbitration mode field MP_CSR[ERCA] is used to determine the arbitration scheme for all channels with APL=0 within a group.

When you enable round-robin channel arbitration (MP_CSR[ERCA] = 1), all channels with APL=0 within a group will use a round-robin arbitration scheme, which rotates among these channels requesting service without regard to priority. Round-robin provides a fairness mechanism within an arbitration group.

When you enable fixed-priority channel arbitration (MP_CSR[ERCA] = 0), eDMA selects channels with APL=0 based on channel number, with the higher channel number having higher priority.

Diagram



Fields

Field	Function
31 ECP	Enable Channel Preemption 0b - Channel cannot be suspended by a higher-priority channel's service request 1b - Channel can be temporarily suspended by a higher-priority channel's service request
30 DPA	Disable Preempt Ability 0b - Channel can suspend a lower-priority channel 1b - Channel cannot suspend any other channel, regardless of channel priority
29-3 —	Reserved
2-0 APL	Arbitration Priority Level Channel priority level for arbitration within the assigned arbitration group.

10.4.2.7 Channel Multiplexor Configuration (CH0_MUX - CH15_MUX)

Offset

For n = 0 to 15:

Register	Offset
CHn_MUX	14h + (n × 1000h)

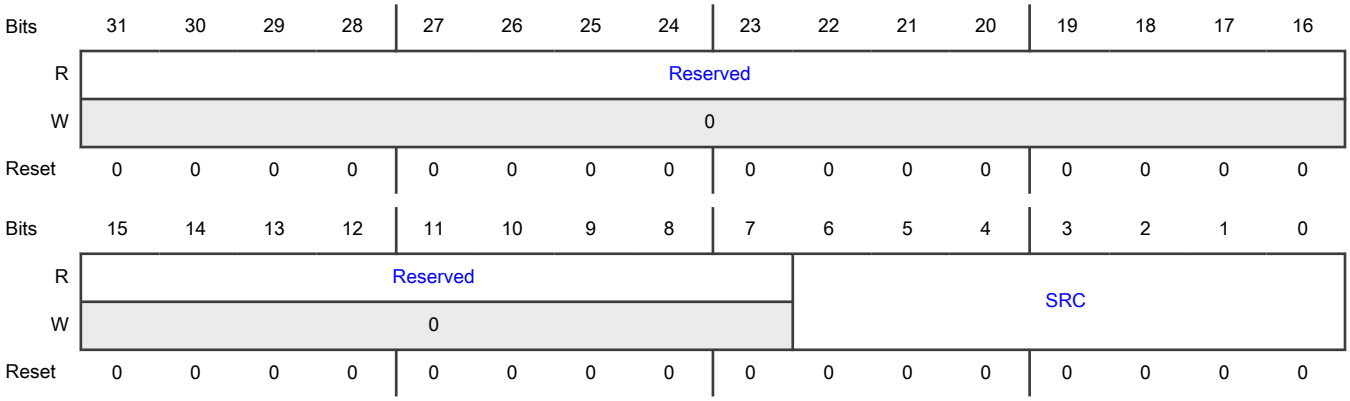
Function

Each of the DMA channels can be independently associated with various peripherals in the system. The Channel Multiplexor Configuration register selects the peripheral assigned to each channel. Service requests from the peripheral should be disabled when configuring a channel to a peripheral source.

Each channel must have a unique value when selecting a peripheral slot in the channel mux configuration. The only value that may overlap is source 0. If there is an attempt to write a mux configuration value that is already consumed by any channel, a mux configuration of 0 (SRC = 0) will be written.

All channels will default to source 0. When a particular peripheral is needed, the channel's mux configuration is set to that source number. When the peripheral is no longer needed, the mux configuration for that channel should be written to 0, thus releasing the resource.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 SRC	Service Request Source Hardware service request source for the channel. **NOTE** With the exception of 0, attempts to write a value already in use will be forced to 0.

10.4.2.8 TCD Source Address (TCD0_SADDR - TCD15_SADDR)

Offset

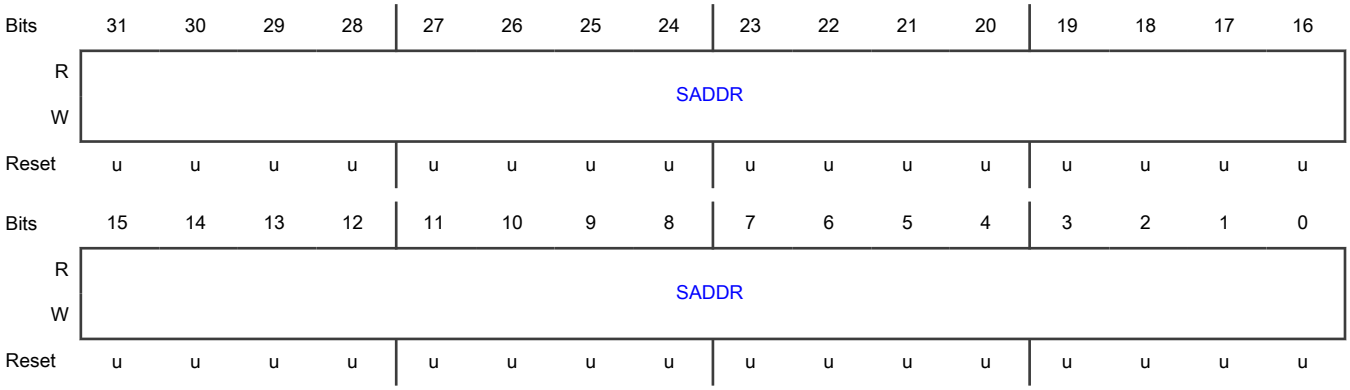
For n = 0 to 15:

Register	Offset
TCDn_SADDR	20h + (n × 1000h)

Function

This register contains the address for the read transactions.

Diagram



Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

10.4.2.9 TCD Signed Source Address Offset (TCD0_SOFF - TCD15_SOFF)

Offset

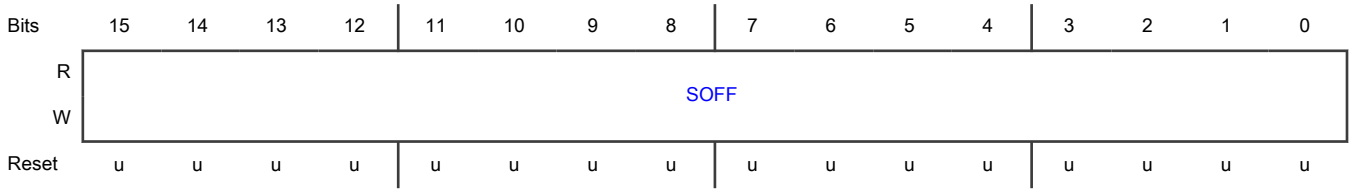
For n = 0 to 15:

Register	Offset
TCDn_SOFF	24h + (n × 1000h)

Function

This register contains the sign-extended value added to Source Address register after each read transaction.

Diagram



Fields

Field	Function
15-0 SOFF	Source Address Signed Offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

10.4.2.10 TCD Transfer Attributes (TCD0_ATTR - TCD15_ATTR)

Offset

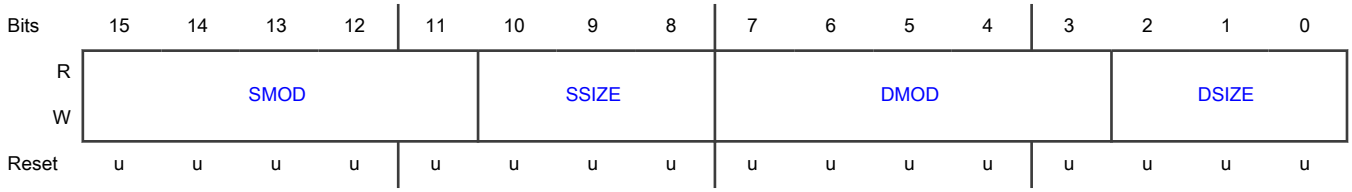
For n = 0 to 15:

Register	Offset
TCDn_ATTR	26h + (n × 1000h)

Function

This register contains size and option modulo addressing information for source and destination addresses.

Diagram



Fields

Field	Function
15-11 SMOD	Source Address Modulo This field defines a specific address range, which is the value after the SADDR + SOFF calculation is performed on the original register value. Setting this field makes it easy to implement a circular data queue. For data queues requiring power-of-2-sized bytes, the queue must start at a 0-modulo-size address and the SMOD field must be set to the appropriate value for the queue, freezing the required number of upper address bits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The value programmed into this field specifies the number of lower address bits that are allowed to change. For a circular queue application, you typically set TCDn_SOFF[SOFF] to the transfer size to implement post-increment addressing, with the SMOD function constraining the addresses to a 0-modulo-size range. 0_0000b - Source address modulo feature disabled 0_0001b - Source address modulo feature enabled for any non-zero value [1-31]
10-8 SSIZE	Source Data Transfer Size 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - Reserved 111b - Reserved
7-3 DMOD	Destination Address Modulo See the SMOD definition.
2-0 DSIZE	Destination Data Transfer Size See the SSIZE definition.

10.4.2.11 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD15_NBYTES_MLOFFNO)

Offset

For n = 0 to 15:

Register	Offset
TCDn_NBYTES_MLOFFNO	28h + (n × 1000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR), or destination address (TCDn_DADDR), upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. When minor loop offsets are enabled, the minor loop offset value (TCDn_NBYTES_MLOFFYES[MLOFF]) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both, prior to the addresses being written back to the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (SMLOE or DMLOE is 1), TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is redefined. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is used to specify multiple fields:

- A source enable bit (SMLOE) to specify the minor loop offset must be applied to the source address (TCDn_SADDR) upon minor loop completion
- A destination enable bit (DMLOE) to specify the minor loop offset must be applied to the destination address (TCDn_DADDR) upon minor loop completion
- The sign extended minor loop offset value (MLOFF)

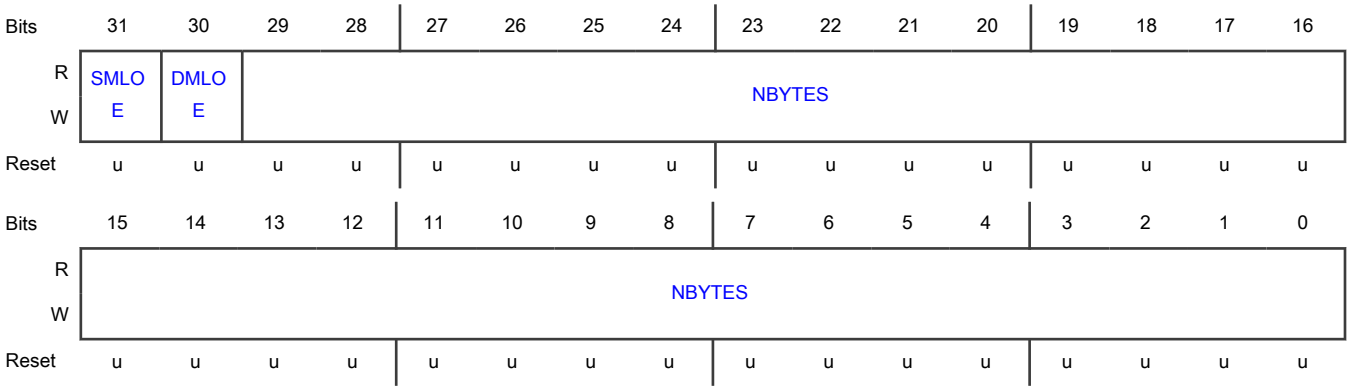
The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is defined as follows:

- If SMLOE = 0 and DMLOE = 0, then see the TCDn_NBYTES_MLOFFNO register description.
- If either SMLOE or DMLOE is 1, then see the TCDn_NBYTES_MLOFFYES register description.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-0	Number of Bytes To Transfer Per Service Request

Table continues on the next page...

Table continued from the previous page...

Field	Function
NBYTES	Number of bytes to be transferred for each service request of the channel. When a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the byte transfer count has been reached. This process is normally an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the byte count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major loop iteration count (CITER) is decremented by one and written back to the TCD memory. If the major iteration count is complete, additional processing is performed.

10.4.2.12 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD15_NBYTES_MLOFFYES)

Offset

For n = 0 to 15:

Register	Offset
TCDn_NBYTES_MLOFFYES	28h + (n × 1000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offset is an address offset value added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion occurs when the channel has finished the service request and has transferred NBYTES. Minor loop offsets are enabled by setting either the source enable bit (SMLOE) or the destination enable bit (DMLOE).

The source enable bit (SMLOE) specifies the minor loop offset value (MLOFF) that is to be applied to the source address (TCDn_SADDR) upon minor loop completion. The destination enable bit (DMLOE) specifies the minor loop offset (MLOFF) that is to be applied to the destination address (TCDn_DADDR) upon minor loop completion.

If the major loop is complete, the minor loop offsets are ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

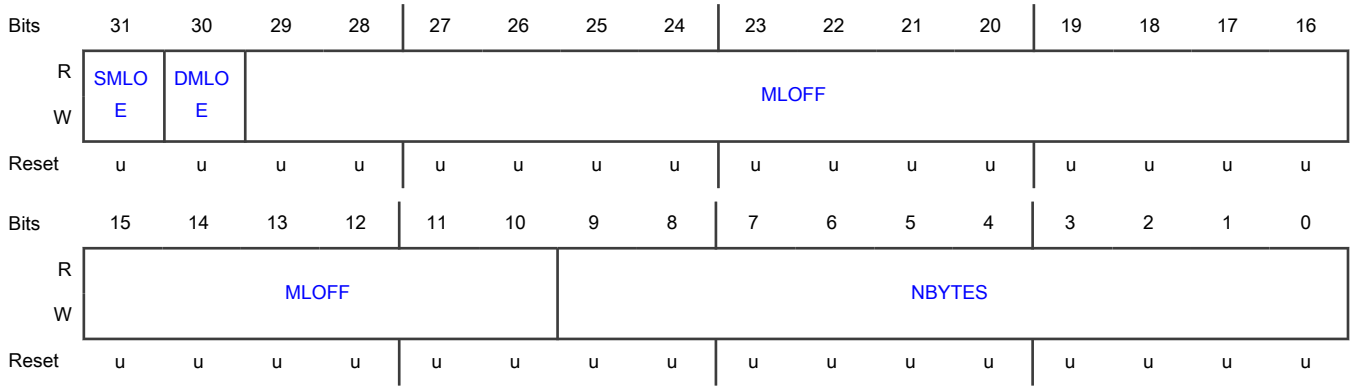
When you enable the minor loop offset overlay (either SMLOE or DMLOE is 1), eDMA redefines TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES specifies the sign-extended minor loop offset value (MLOFF). The same offset value (MLOFF) applies to both source and destination minor loop offsets. When the minor loop offset is enabled, you must align it to the transfer size of the source or destination it is associated with. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFNO) defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFYES is defined as follows:

- If either minor loop offset is enabled (SMLOE or DMLOE = 1), then see the TCDn_NBYTES_MLOFFYES register description.
- If SMLOE and DMLOE are both 0, then see the TCDn_NBYTES_MLOFFNO register description.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-10 MLOFF	Minor Loop Offset If SMLOE or DMLOE is 1, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Number of Bytes To Transfer Per Service Request The number of bytes to be transferred in each service request of the channel. As a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the minor byte transfer count has been reached. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is complete, additional processing is performed.

10.4.2.13 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD15_SLAST_SDA)

Offset

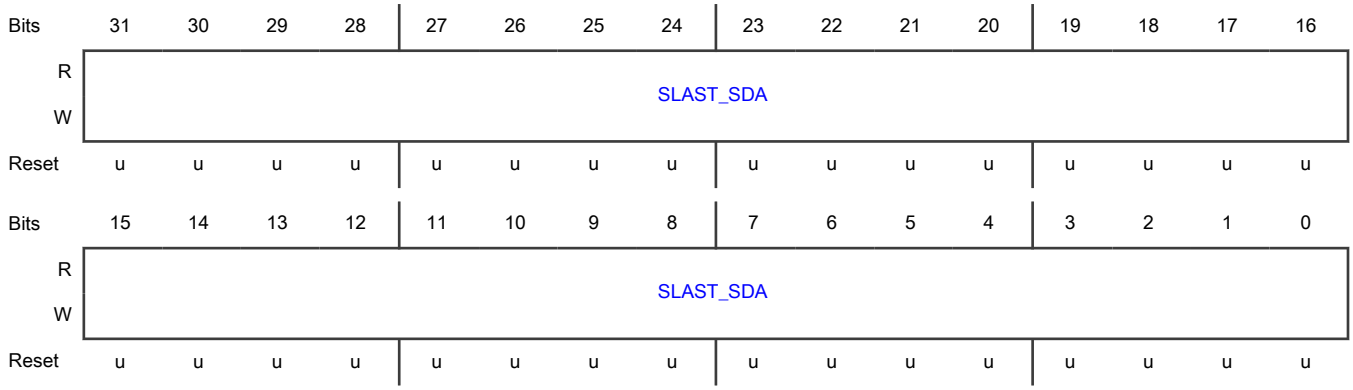
For n = 0 to 15:

Register	Offset
TCDn_SLAST_SDA	2Ch + (n × 1000h)

Function

This register contains the value added to the source address when the major loop is complete. When the store destination address option is enabled, this field provides a pointer to memory for storing the final destination address.

Diagram



Fields

Field	Function
31-0 SLAST_SDA	Last Source Address Adjustment / Store DADDR Address Source last address adjustment or the system memory address for destination address (DADDR) storage. If (TCDn_CSR[ESDA] = 0), then: • Adjustment value is added to the source address at the completion of the major iteration count. This value can be used to restore the source address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final source address adjustment. Otherwise: • This address points to the 32-bit-aligned memory location where the destination address (DADDR) is to be stored in system memory. By saving the final destination address in system memory via the ESDA feature, you are able to compute the size of a variable destination data buffer by simply subtracting the beginning DADDR from the final, saved DADDR. This feature is used together with the scatter/gather operation to prevent the loss of the final DADDR, which is overwritten during the scatter/gather operation. The "Store Destination Address" (SDA) value must be a 32-bit-aligned location because the eDMA forces the lower two address bits of the SLAST_SDA field to zero when ESDA is enabled. The module performs this write operation when the major loop is done; that is, when the major iteration count (CITER) decrements to zero.

10.4.2.14 TCD Destination Address (TCD0_DADDR - TCD15_DADDR)

Offset

For n = 0 to 15:

Register	Offset
TCDn_DADDR	30h + (n × 1000h)

Function

This register contains the address for the write transactions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

10.4.2.15 TCD Signed Destination Address Offset (TCD0_DOFF - TCD15_DOFF)

Offset

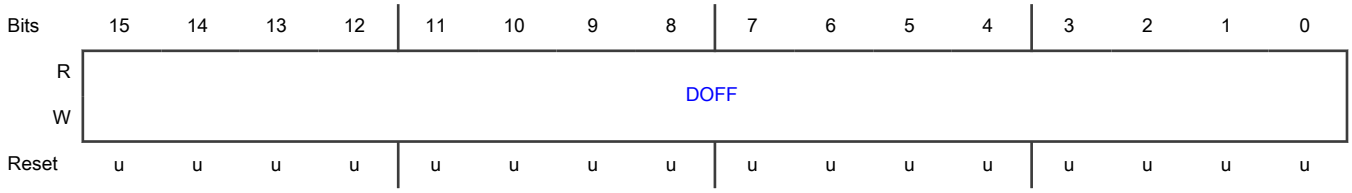
For n = 0 to 15:

Register	Offset
TCDn_DOFF	34h + (n × 1000h)

Function

This register contains the sign-extended value added to Destination Address register after each write transaction.

Diagram



Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset that is applied to the current destination address to form the next-state value as each destination write is completed.

10.4.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD15_CITER_ELINKNO)

Offset

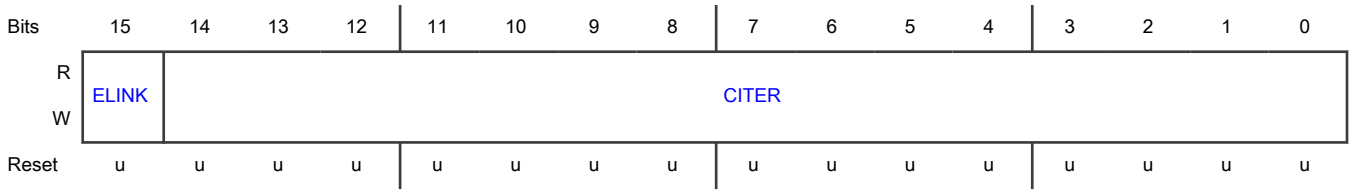
For n = 0 to 15:

Register	Offset
TCDn_CITER_ELINKNO	36h + (n × 1000h)

Function

If TCDn_CITER[ELINK] is 0, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel to 1. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

10.4.2.17 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD15_CITER_ELINKYES)

Offset

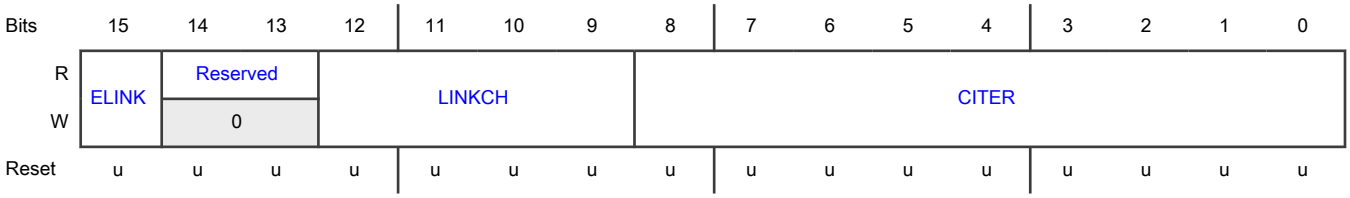
For n = 0 to 15:

Register	Offset
TCDn_CITER_ELINKYES	36h + (n × 1000h)

Function

If TCDn_CITER[ELINK] is 1, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. When enabled, an internal mechanism sets the TCDn_CSR[START] field of the specified channel (LINKCH) upon minor loop completion. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-13 —	Reserved
12-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted the eDMA engine initiates a channel service request to the channel defined by this field by writing that channel's TCDn_CSR[START] field to 1.
8-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

10.4.2.18 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD15_DLAST_SGA)

Offset

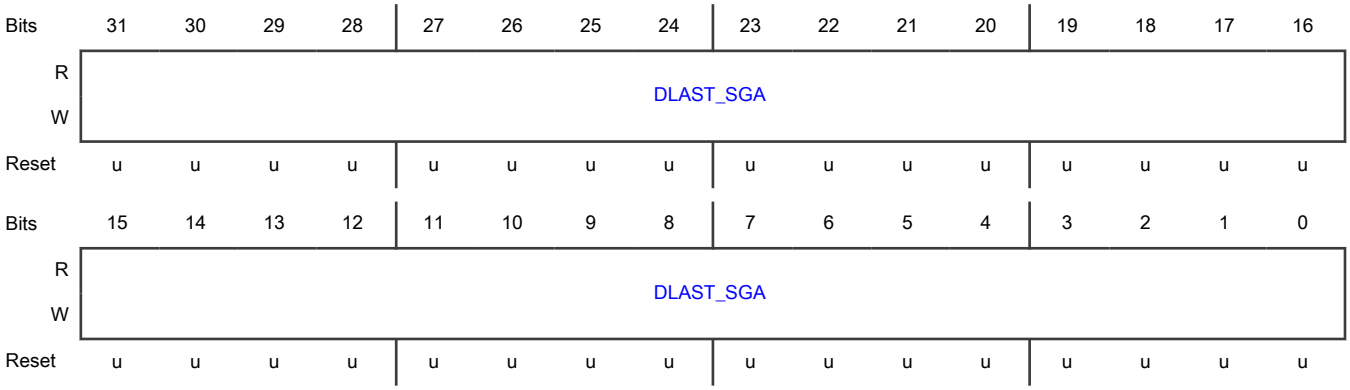
For n = 0 to 15:

Register	Offset
TCDn_DLAST_SGA	38h + (n × 1000h)

Function

This register contains the value added to the destination address when the major loop is complete. When the Scatter/Gather option is enabled, this field provides a pointer to memory for fetching a transfer control descriptor to reprogram the channel.

Diagram



Fields

Field	Function
31-0 DLAST_SGA	Last Destination Address Adjustment / Scatter Gather Address Adjustment of the last destination address or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value is added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo 32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo 32-byte, or else a configuration error is reported.

10.4.2.19 TCD Control and Status (TCD0_CSR - TCD15_CSR)

Offset

For n = 0 to 15:

Register	Offset
TCDn_CSR	3Ch + (n × 1000h)

Function

This register is used to enable optional features.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BWC				MAJORLINKCH				ESDA	EEOP	MAJO REL...	ESG	DREQ	INTHA LF	INTMA JOR	STAR T
W			0													
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	0

Fields

Field	Function
15-14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces eDMA to stall after the completion of each read/write access, to control the bus request bandwidth seen by the system bus interconnect. **NOTE** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00b - No eDMA engine stalls 01b - Reserved 10b - eDMA engine stalls for 4 cycles after each R/W 11b - eDMA engine stalls for 8 cycles after each R/W
13-12 —	Reserved
11-8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field to 1.
7 ESDA	Enable Store Destination Address As the channel completes the major loop by either the current iteration counter (CITER) decrementing to 0, or by receiving an enabled end-of-packet signal, this field enables writing the destination address (DADDR) to the address stored in the SLAST_SDA field. The value written to system memory is the last DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation. When the SDA bit is 1, SLAST_SDA contains the write pointer instead of the final source address offset. Because this is a pointer and not a final offset, a last source address offset of zero is applied to SADDR instead of the SLAST_SGA value. 0b - Ability to store destination address to system memory disabled 1b - Ability to store destination address to system memory enabled
6 EEOP	Enable End-Of-Packet Processing When enabled by the EEOP field, an end-of-packet hardware input signal directs eDMA to discontinue executing the active channel, and to treat the shutdown as the major-loop-completed event. If the EEOP field is 1, the end-of-packet signal from supported peripherals is accepted. If the EEOP field is 0, the end-of-packet input is ignored. With an end-of-packet retirement, the current TCD destination address (or ESDA-saved destination address), minus the software-saved initial address (DADDR), reflects the total amount of data transferred. 0b - End-of-packet operation disabled 1b - End-of-packet hardware input signal enabled
5 MAJORELINK	Enable Link When Major Loop Complete As the channel completes the major loop, this flag enables linking to another channel defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. **NOTE** To support the dynamic linking coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses TCDn_DLAST_SGA as a memory pointer to a 0-modulo 32-bit address containing a 32-byte data structure, which is loaded as the transfer control descriptor into local memory. **NOTE** To support the dynamic scatter/gather coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Current channel's TCD is normal format 1b - Current channel's TCD specifies scatter/gather format.
3 DREQ	Disable Request If this flag is 1, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches 0. 0b - No operation. Channel's ERQ field not affected 1b - Clear the ERQ field to 0 upon major loop completion, thus disabling hardware service requests. Channel's ERQ field cleared to 0 when major loop complete
2 INTHALF	Enable Interrupt If Major Counter Half-complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER = (BITER/2)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes, or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE** If BITER = 1, do not use INTHALF; use INTMAJOR instead. 0b - Halfway point interrupt disabled 1b - Halfway point interrupt enabled
1 INTMAJOR	Enable Interrupt If Major count complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count (CITER) reaches 0. 0b - End-of-major loop interrupt disabled 1b - End-of-major loop interrupt enabled
0 START	Channel Start If this flag is 1, the channel is requesting service. The eDMA hardware automatically clears this flag to 0 after the channel begins execution. 0b - Channel not explicitly started 1b - Channel explicitly started via a software-initiated service request

10.4.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD15_BITER_ELINKNO)

Offset

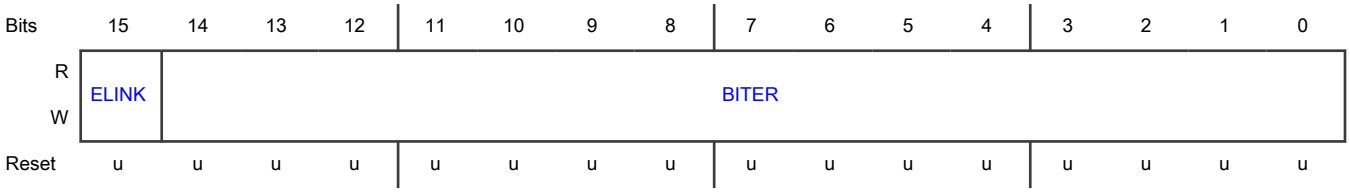
For n = 0 to 15:

Register	Offset
TCDn_BITER_ELINKNO	3Eh + (n × 1000h)

Function

If the TCDn_BITER[ELINK] field is 0, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enables Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

10.4.2.21 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD15_BITER_ELINKYES)

Offset

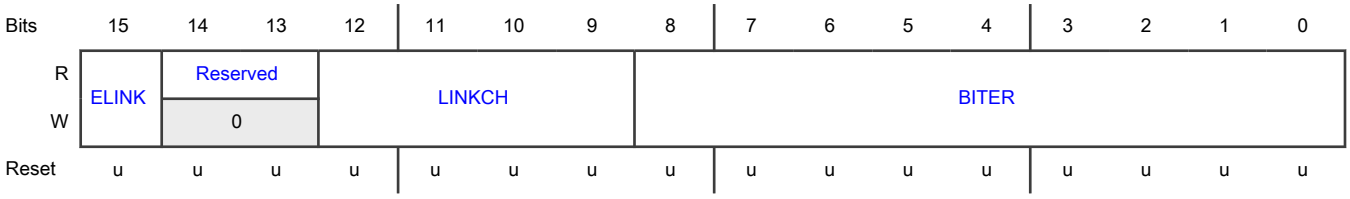
For n = 0 to 15:

Register	Offset
TCDn_BITER_ELINKYES	3Eh + (n × 1000h)

Function

If the TCDn_BITER[ELINK] field is set, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-13 —	Reserved
12-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field.
8-0	Starting Major Iteration Count

Table continues on the next page...

Table continued from the previous page...

Field	Function
BITER	As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

10.5 External signals

This module has no external signals.

10.6 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

10.6.1 eDMA initialization

To initialize the eDMA:

1. Write to the MP_CSR if a configuration other than the default is wanted.
2. Write the channel priority levels to the CHn_PRI registers and group priority levels to the CHn_GRPRI registers if a configuration other than the default is wanted.
3. Enable error interrupts in the CHn_CSR[EEI] registers if they are wanted.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the CHn_CSR[ERQ] registers.
6. Request channel service via either:
   - Software: setting TCDn_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in Table 42, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, defined by TCDn_SADDR, to the destination, defined by TCD_DADDR, continue until the number of bytes specified by TCDn_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCDn_SADDR, TCDn_DADDR, and TCDn_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, then eDMA executes further post-processing, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 42. TCD control and status (TCDn_CSR) fields

TCDn_CSR field name	Description
START	Control field to start the channel explicitly when using a software-initiated DMA service (automatically cleared by hardware)
EEOP	Control field to enable end-of-packet processing

Table continues on the next page...

Table 42. TCD control and status (TCDn_CSR) fields (continued)

TCDn_CSR field name	Description
ESDA	Control field to enable storing of the destination address to system memory after the major loop completes
DREQ	Control field to disable hardware-initiated DMA service requests after major loop completion
BWC	Control field for throttling the bandwidth control of a channel
ESG	Control field to enable the scatter-gather feature
INTHALF	Control field to enable interrupt when major loop is half-complete
INTMAJOR	Control field to enable interrupt when major loop completes

Table 43. Channel control and status (CHn_CSR) fields

CHn_CSR field name	Description
ACTIVE	Status field indicating the channel is currently in execution
DONE	Status field indicating major loop completion (cleared by software when a channel begins execution)
EEI	Control field to enable error interrupts
EARQ	Control field to enable external, asynchronous wakeup event in conjunction with the ERQ field
ERQ	Control field to enable hardware service requests

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

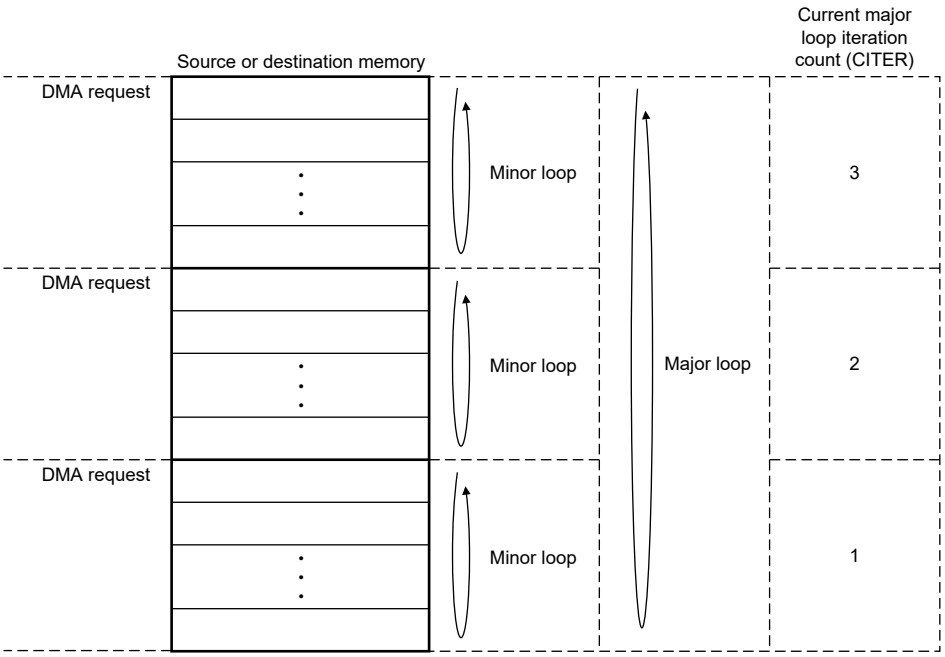


Figure 17. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings are related.

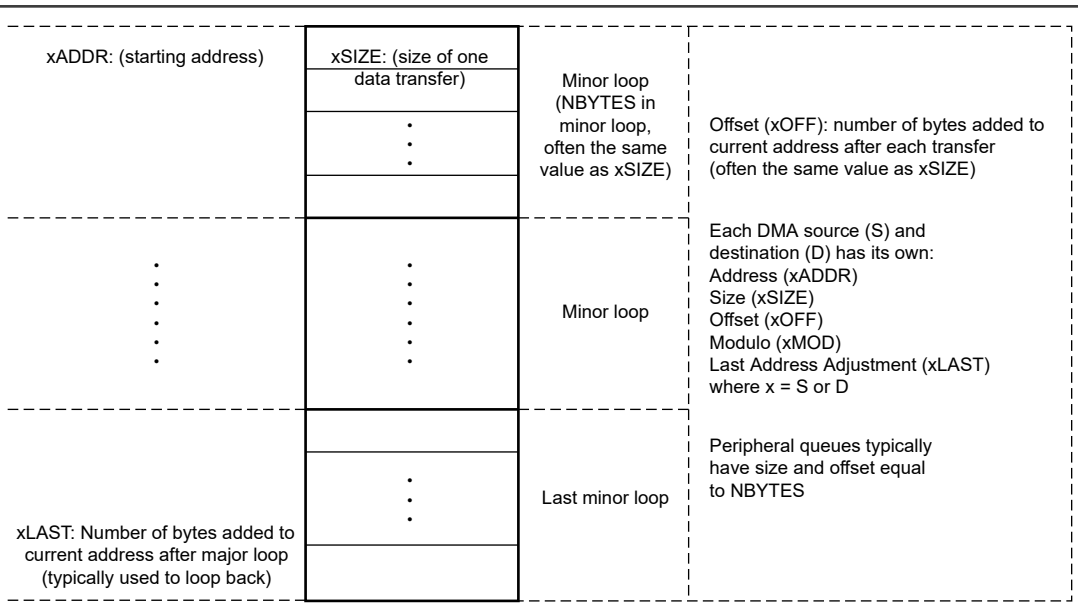


Figure 18. Memory array terms

10.6.2 eDMA arbitration

The eDMA uses a layered arbitration scheme composed of multiple priority levels. The eDMA uses a fixed-priority arbitration scheme with optional round-robin arbitration under specific conditions. The priorities are evaluated in the following order:

Table 44. eDMA arbitration priorities

Priority	Scheme	Description
1 (Highest)	Arbitration group priority	Each channel is assigned an arbitration group via the CHn_GRPRI registers. Priority is given to the highest value (31 being the highest possible value) down to the lowest value (zero, the default).
2	Channel priority	Each channel is assigned a channel priority level via the CHn_PRI registers. The channel priority is a relative priority level within an arbitration group. Priority is given to the highest value (seven being the highest possible value) down to the lowest value (zero, the default). Channel priorities within each arbitration group need not be unique. If multiple channels have the same channel priority level, the channel number will be used to determine priority as defined in row three.
3	Channel number	When two or more channels have the same arbitration group priority and channel priority, the channel number (CHn_NUM) is used to determine the highest priority. Priority is giver to the highest channel number. Lowest priority is channel 0. The channel numbers are static and cannot be changed in the programmer's model.
4 (Lowest)	Round-robin	When round-robin is enabled, any channel configured for round-robin operation has lowest priority within an arbitration group. Round-robin is enabled by setting the MP_CSR[ERCA] field to 1. After being enabled, channels with a channel priority of zero (CHn_PRI=0) will use round-robin arbitration. Round-robin arbitration will rotate the channel selection among the channels requesting service with CHn_PRI=0 within the arbitration group. Any non-zero channel within the arbitration group will continue to use fixed-priority arbitration, and if requesting service will be selected over any round-robin channels.

For fixed arbitration, the overall priority can be considered a number composed of three concatenated priority levels: CHn_GRPRI:CHn_PRI:CH_NUM. The largest number has the highest priority and the lowest number has the lowest priority.

For round-robin arbitration, the priority number is CHn_GRPRI:0:X. The module rotates through the CHn_PRI=0 channels requesting service without regard to priority among these channels. Any channel within the arbitration group for which CHn_PRI is greater than 0 will be serviced before the round-robin channels.

10.6.3 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data.

The channel number causing the error is recorded in the Error Status register (CHn_ES). If the error source is not removed before the next activation of the problematic channel, the error is detected and recorded again. Setting the halt after error field, CSR[HAE], will halt the DMA and prevent recurrence of the error.

10.6.4 Arbitration mode considerations

This section discusses arbitration considerations for eDMA.

10.6.4.1 Fixed group arbitration, fixed channel arbitration

In this mode, eDMA selects for execution the channel service request from the highest-priority channel in the highest-priority group. If eDMA is programmed so that the channels within a high-priority group have a high number of requests or large data transfers, that group may consume all the bandwidth of the eDMA controller. That is, no lower-priority groups are serviced if there is always at least one DMA request pending on a channel in the highest-priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly.

10.6.4.2 Fixed group arbitration, round-robin channel arbitration

The highest-priority group with a request is serviced. Lower-priority groups are serviced if no pending requests exist in the higher-priority groups.

Within each group, channels are serviced starting with the highest non-zero channel priority. For all channels with a channel priority programmed to 0, selection begins with the highest channel number requesting service and then rotates through to the lowest channel number requesting service. The round-robin channel arbitration can provide a fairness mechanism to lower-priority channels.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, fixed channel arbitration, but all the channels in the highest-priority group will be serviced. Service latency is short on the highest-priority group, but could potentially be very much longer as the group priority decreases.

10.6.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

10.6.5.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the CH*n*_CSR[DONE] field is set to 1 and an interrupt is generated if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte-wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source, and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
```

```
TCDn_DLAST_SGA= -16
TCDn_CSR[INTMAJ] = 1
TCDn_CSR[START] = 1 (should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] field requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes:
   - CH*n*_CSR[DONE] = 0
   - TCD*n*_CSR[START] = 0
   - CH*n*_CSR[ACTIVE] = 1
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
   f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
   g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
   h. Write 32 bits to location 0x200C → last iteration of the minor loop → major loop complete.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

10.6.5.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop, transferring 16 bytes per iteration. After the channel's hardware requests are enabled via the CHn_CSR[ERQ] register field, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware (eDMA peripheral) requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:

a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
h. Write 32 bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0.
8. The channel retires, which concludes one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware (eDMA peripheral) requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: Channel TCD data from local memory to internal register file.
13. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
    b. Write 32 bits to location 0x2010 → first iteration of the minor loop.
    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
    d. Write 32 bits to location 0x2014 → second iteration of the minor loop.
    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
    f. Write 32 bits to location 0x2018 → third iteration of the minor loop.
    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
    h. Write 32 bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
16. The channel retires, which concludes with the major loop complete. The eDMA goes idle or services the next channel.

10.6.5.3 Using the modulo feature

The modulo feature of the eDMA allows implementation of a circular data queue in which the size of the queue is a power of 2. xMOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature. Modulo addressing applies to cases where the minor loop offset is enabled; that is, the upper address bits remain the same after the minor loop offset is added to the source or destination address.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value but the 28 upper address bits (0x1234567*x*) retain their original value. In this example, the source address is set to 0x12345670, the offset is set to four bytes, and the MOD field is set to four, which allows for a 2^4 byte (16 byte) queue size.

Table 45. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

10.6.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

10.6.6.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software-initiated service requests.

1. The first method is to read the TCDn_CITER field and test for a change.
2. The second method, extracted from the sequence shown below, is to test the TCDn_CSR[START] field and the CHn_CSR[ACTIVE] field. The minor-loop-complete condition is indicated by both fields reading 0 after TCDn_CSR[START] is set to 1. Polling the CHn_CSR[ACTIVE] field only may be inconclusive because the active status may be missed if the channel execution is short in duration.

The CHn_CSR and TCDn_CSR status fields execute the following sequence for a software-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	1	0	0	Initiate channel service request via software.
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

The best method to test for minor-loop completion when using hardware-initiated (that is, peripheral-initiated) service requests is to read the TCDn_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status fields execute the following sequence for a hardware-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	0	0	0	Initiate channel service request via hardware (peripheral request asserted).
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

For both activation types, the major-loop-complete status is explicitly indicated via the CHn_CSR[DONE] field.

The TCDn_CSR[START] field is cleared to 0 automatically when the channel begins execution, regardless of how the channel activates.

10.6.6.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCDn_SADDR, TCDn_DADDR, and TCDn_NBYTES values if they are read when a channel executes. The true values of SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file, and not the values in the TCD local memory for that channel. The addresses, SADDR and DADDR, and NBYTES (which decrements to zero as the transfer progresses), can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

10.6.6.3 Checking channel preemption status

A preemptive situation is one in which a preempt-enabled channel is executing and a higher-priority request becomes active. When round-robin channel arbitration mode is enabled, all channels with their channel priority set to 0 lose their preempt ability. Channel priorities of 0 are treated as equal, that is, they are constantly rotating, when round-robin arbitration mode is enabled.

The CHn_CSR[ACTIVE] field for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended when the preempting channel executes one major loop iteration. If two CHn_CSR[ACTIVE] fields are set simultaneously in the global TCD map, a higher-priority channel is actively preempting a lower-priority channel.

10.6.7 Channel linking

Channel linking (or chaining) is a mechanism in which one channel sets the TCDn_CSR[START] field of another channel (or itself), thus initiating a service request for that channel. When properly enabled, the eDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCDn_CITER[ELINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, using an initial field setting of:

```
TCDn_CITER[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJORELINK] = 1
TCDn_CSR[MAJORLINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] field

2. Minor loop done → set TCD12_CSR[START] field
3. Minor loop done → set TCD12_CSR[START] field
4. Minor loop done, major loop done→ set TCD7_CSR[START] field

When minor loop linking is enabled (TCDn_CITER[ELINK] = 1), the TCDn_CITER[CITER] field uses a nine-bit vector to form the current iteration count. When minor loop linking is disabled (TCDn_CITER[ELINK] = 0), the TCDn_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCDn_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

NOTE

The TCDn_CITER[ELINK] field and the TCDn_BITER[ELINK] field must be equal — if they are not, a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop halfway done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, that is, use another channel's TCD, at the end of a loop.

Table 46. Channel linking parameters

Wanted link behavior	TCD control field name	Description
Link at end of minor loop	TCDn_CITER[ELINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	TCDn_CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of major loop	TCDn_CSR[MAJORELINK]	Enable channel-to-channel linking on major loop completion
	TCDn_CSR[MAJORLINKCH]	Link channel number when linking at end of major loop

10.6.8 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

10.6.8.1 Dynamically changing the channel priority

To change group or channel priority levels:

1. Halt the DMA by writing 1 to the CSR[HALT] field.
2. Change the group or channel priorities as wanted.
3. Enable normal DMA operations by writing 0 to the CSR[HALT] field.

10.6.8.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] field during channel execution (see the diagram in TCD structure). This field is read from the TCD local memory at the end of channel execution, thus allowing you to enable the feature during channel execution.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic channel link by enabling the TCDn_CSR[MAJORELINK] field at the same time the eDMA engine is retiring the channel. TCDn_CSR[MAJORELINK] would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

We recommend that you use the following coherency model when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] field.

2. Read back the TCDn_CSR[MAJORELINK] field.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCDn_CSR[MAJORELINK] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field is set to 1, indicating the major loop is complete.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing to the TCDn_CSR[MAJORELINK] field. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution.

10.6.8.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in the eDMA programmer's model, thus replacing the current descriptor.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] field at the same time the eDMA engine is retiring the channel. The TCDn_CSR[ESG] field would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods are recommended for executing a dynamic scatter/gather request. Whenever the TCDn_CSR is written, the TCD local memory controller forces the TCDn_CSR[ESG] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field has been set to 1, indicating the major loop is complete. If attempting to set the ESG, ensure the DONE field is cleared to 0.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing the TCDn_CSR[MAJORELINK] or TCDn_CSR[ESG] fields. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution and is set to 1 upon major loop completion.

10.6.8.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCDn_CSR[MAJORELINK] field is 0, the TCDn_CSR[MAJORLINKCH] field is not used by the eDMA. In this case, the TCDn_CSR[MAJORLINKCH] bits may be used for other purposes. This method uses the TCDn_CSR[MAJORLINKCH] field as a TCDn_CSR identification (ID).

When the descriptors are built, write a unique TCDn_CSR ID in the TCDn_CSR[MAJORLINKCH] field for each TCDn_CSR associated with a channel using dynamic scatter/gather.

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the TCDn_CSR[DREQ] field to 1 will prevent future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the 16-bit TCDn_CSR control/status field.
5. Test the TCDn_CSR[ESG] request status and TCDn_CSR[MAJORLINKCH] value:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.

- If ESG = 0 and the MAJORLINKCH (ID) did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
- If ESG = 0 and the MAJORLINKCH (ID) changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

10.6.8.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCDn_DLAST_SGA field as a TCD identification (ID).

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the DREQ field to 1 will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the TCDn_CSR[ESG] field.
5. Test the TCDn_CSR[ESG] request status:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.
   - If ESG = 0, read the 32-bit TCDn_DLAST_SGA field.
   - If ESG = 0 and the TCDn_DLAST_SGA did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
   - If ESG = 0 and the TCDn_DLAST_SGA changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

10.6.9 Suspend/resume a DMA channel with active hardware service requests

The DMA allows you to move data from memory or peripheral registers to another location in memory or to peripheral registers without CPU interaction. After the DMA and peripherals are configured and active, it is rare but supported to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, you must follow a specific procedure. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (DSPI), Sigma Delta Analog to Digital Convertor (SDADC), or other module.

10.6.9.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status (MP_HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ field to 0 on the appropriate DMA channel.

For example, assume the DSPI is set as a master for transmitting data via a DMA service request when the TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If you need to suspend the DMA/DSPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the DSPI by verifying that MP_HRS[HRS] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ field to 0. If a service request is present, wait until the request has been processed and the HRS field reads 0.

10.6.9.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ field to 1.
2. Enable the DMA service request at the peripheral.

Chapter 11
Miscellaneous System Control Module (MSCM)

11.1 Chip-specific MSCM information

Table 47. Reference links to related information

Topic	Related module	Reference
Full description	MSCM	MSCM
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

11.1.1 Module instances

This device has one instance of the MSCM module.

11.1.2 OCMDR1[OCMSZ]

OCMDR1[OCMSZ] = 1000b, indicates the device boot ROM is 96 KB.

11.1.3 SID[FAMID]

For MCX W71, SID[FAMID] = 0000b.

11.2 Overview

The Miscellaneous System Control Module (MSCM) contains CPU configuration registers and on-chip memory controller registers.

11.2.1 Block Diagram

The MSCM block diagram is shown below in Figure 19:

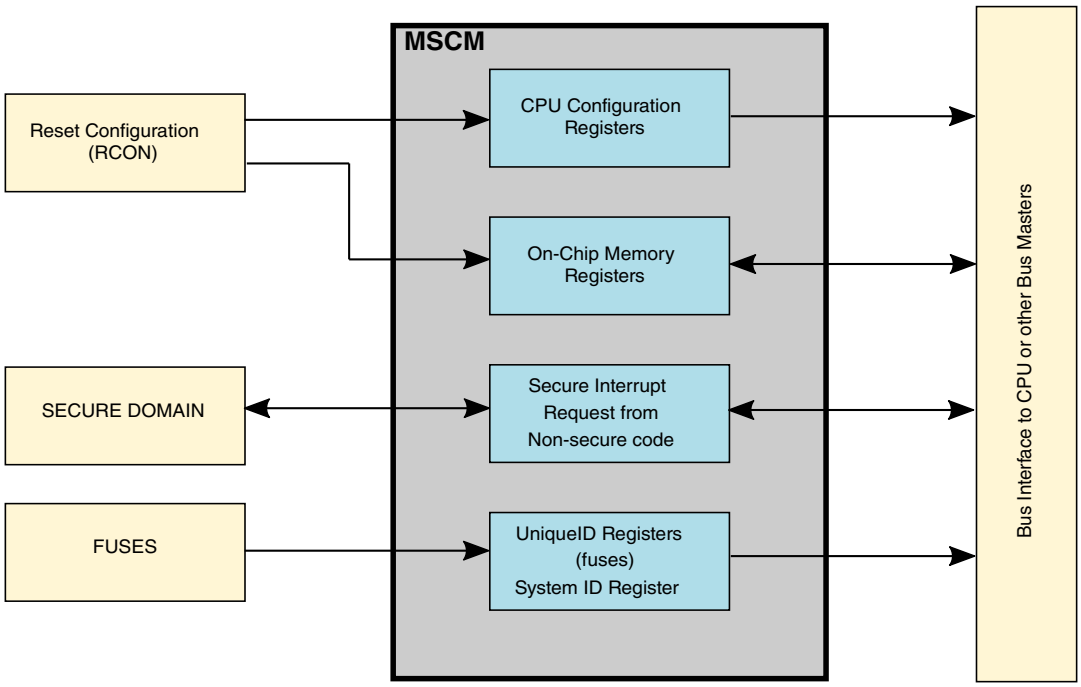


Figure 19. MSCM Block Diagram

11.2.2 Features

The MSCM has the following registers:

- Generic processor "x" configuration registers - These registers are only accessible by the main processor core
- Configuration registers containing information for processor 0 (main processor core) - These registers are accessible to any bus master
- On-Chip Memory Descriptor Registers - Provides static information about the attached memories. These registers are only accessible to the processor core or debugger
- Secure interrupt register - Provides a mechanism for non-secure code to generate a low latency pulsed exception into the secure privileged domain and can only be written in nonsecure mode
- Unique ID registers - Loaded directly from UUID fuse words
- System ID register - Loaded from IFR

11.3 Functional Description

11.3.1 Chip Configuration and Boot

The device's logical definition is controlled via chip-specific configuration bits, supported memory sizes and packing options. Collectively, these configuration bits define a reset configuration value (RCON).

Once the core has fetched the reset vector(s), core and system configuration information is read from a globally-accessible slave peripheral that properly converts the information into more appropriate values. More specifically, the core accesses configuration information from a common set of peripheral addresses and the chip configuration logic properly evaluates based on the requesting processor and returns the appropriate value for the given processor, including core identification.

As an example, there is a single 32-bit read-only location for the core identification. A 32-bit read from this location returns a 4-character ASCII string: 0x4D333301 .

The programming model associated with the core configuration information is included as part of the Miscellaneous System Control Module (MSCM). It specifically includes multiple views of the processor configuration; one view that is available generically to the core, and other views that are available to any bus masters in the system.

11.4 MSCM Memory Map/Register Definition

11.4.1 CPU Configuration Memory Map and Registers

The CPU configuration portion of the MSCM module provides a set of memory-mapped read-only addresses defining the processor set-up. This portion of the MSCM programming model can only be accessed with 32-bit read references; any other size is terminated with an error. If the processor is logically not included in the chip configuration, then reads of its configuration registers return zeroes.

The CPU Configuration registers are organized based on the logical processor number (not any type of physical port number) and partitioned into the following equal sections:

Table 48. CPU Configuration Register Sections

Offset addresses	Function
0x000 - 0x01F	Defines the generic processor "x" configuration. This region is only accessible to the processor core; reads by non-core bus masters are treated as read-as-zero (RAZ) accesses.
0x020 - 0x03F	Defines the configuration information for processor 0 (CP0). This region is accessible to any bus master.

11.4.2 MSCM register descriptions

11.4.2.1 MSCM memory map

MSCM base address: 4001_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Processor X Type Register (CPxTYPE)	32	R	See section
4h	Processor X Number Register (CPxNUM)	32	R	See section
8h	Processor X Master Register (CPxMASTER)	32	R	See section
Ch	Processor X Count Register (CPxCOUNT)	32	R	0000_0000h
10h	Processor X Configuration Register 0 (CPxCFG0)	32	R	See section
14h	Processor X Configuration Register 1 (CPxCFG1)	32	R	See section
18h	Processor X Configuration Register 2 (CPxCFG2)	32	R	See section
1Ch	Processor X Configuration Register 3 (CPxCFG3)	32	R	See section
20h	Processor 0 Type Register (CP0TYPE)	32	R	4D33_3301h
24h	Processor 0 Number Register (CP0NUM)	32	R	0000_0000h
28h	Processor 0 Master Register (CP0MASTER)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2Ch	Processor 0 Count Register (CP0COUNT)	32	R	0000_0000h
30h	Processor 0 Configuration Register 0 (CP0CFG0)	32	R	0408_0000h
34h	Processor 0 Configuration Register 1 (CP0CFG1)	32	R	0000_0000h
38h	Processor 0 Configuration Register 2 (CP0CFG2)	32	R	0001_0001h
3Ch	Processor 0 Configuration Register 3 (CP0CFG3)	32	R	0000_0231h
400h	On-Chip Memory Descriptor Register (OCMDR0)	32	R	EB08_9000h
404h	On-Chip Memory Descriptor Register (OCMDR1)	32	R	D804_7000h
408h	On-Chip Memory Descriptor Register (OCMDR2)	32	R	E504_1000h
40Ch	On-Chip Memory Descriptor Register (OCMDR3)	32	R	E704_1000h
410h	On-Chip Memory Descriptor Register (OCMDR4)	32	R	D704_1000h
414h	On-Chip Memory Descriptor Register (OCMDR5)	32	R	E404_1000h
800h	Secure Interrupt Request (SECURE_IRQ)	32	RW	0000_0000h
810h	Unique ID 0 (UID0)	32	R	See section
814h	Unique ID 1 (UID1)	32	R	See section
818h	Unique ID 2 (UID2)	32	R	See section
81Ch	Unique ID 3 (UID3)	32	R	See section
820h	System ID (SID)	32	R	See section

11.4.2.2 Processor X Type Register (CPxTYPE)

Offset

Register	Offset
CPxTYPE	0h

Function

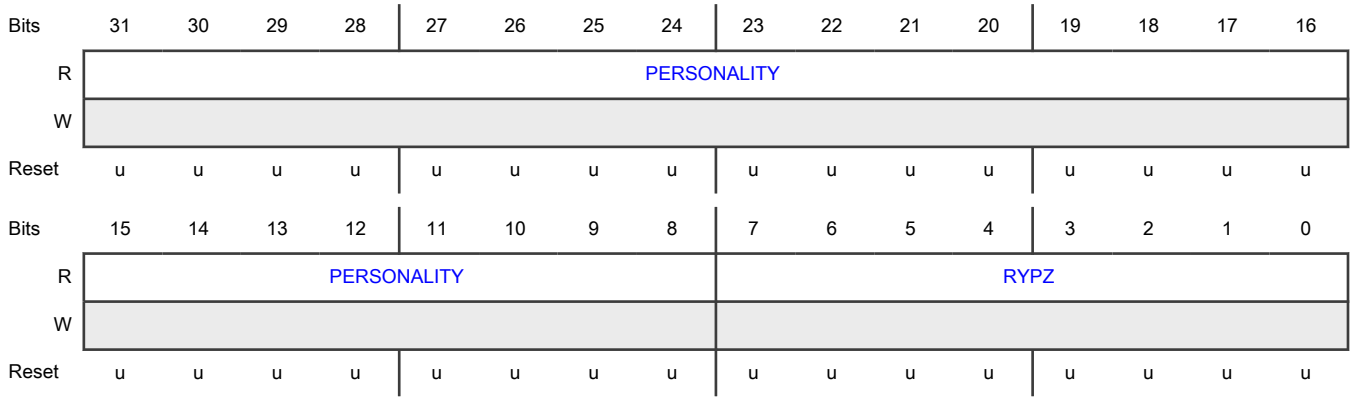
The register provides a CPU-specific response indicating the personality of the core making the access. The 32-bit response includes 3 ASCII characters that define the CPU type, along with a byte that defines the logical revision number. The logical revision number follows Arm's rYpZ nomenclature.

NOTE

CPxTYPE value(s) for this device:

- If CPU0 is making the access, then the value read is 0x4D333301
- If the read access is not from a CPU, then the value read is 0x00000000

Diagram



Fields

Field	Function
31-8 PERSONALITY	Processor x Personality This read-only field defines the processor personality for CPUx
7-0 RYPZ	Processor x Revision This read-only field defines the processor revision for CPUx: 0x00 corresponds to the r0p0 core release. 0x01 corresponds to the r0p1 core release. ...

11.4.2.3 Processor X Number Register (CPxNUM)

Offset

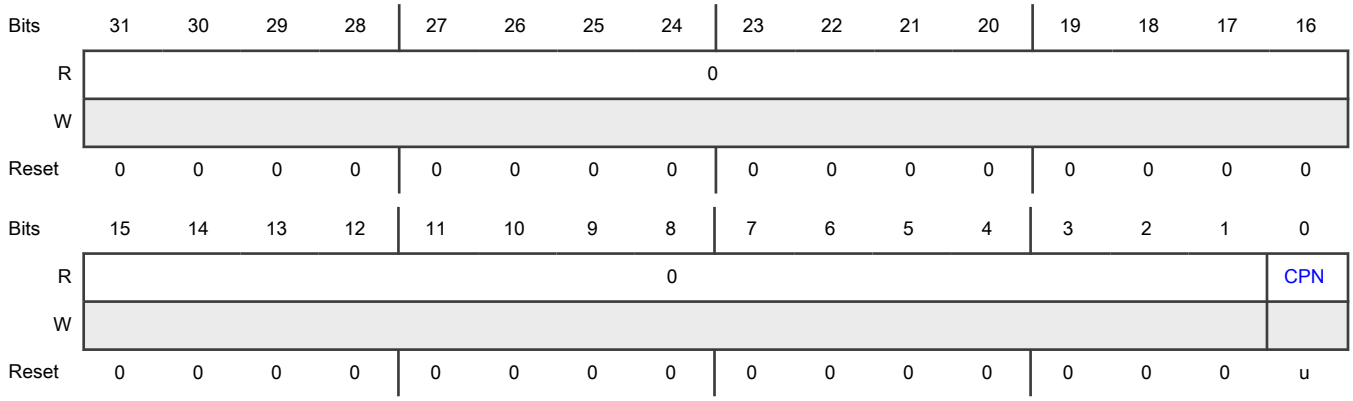
Register	Offset
CPxNUM	4h

Function

The register provides a CPU-specific response indicating the logical processor number of the core making the access. The logical processor number is always 0.

If the read access is not from a CPU, then the value read is 0x00000000.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CPN	Processor x Number This zero-filled word defines the logical processor number for CPUx If single core configuration, then CPN = 0

11.4.2.4 Processor X Master Register (CPxMASTER)

Offset

Register	Offset
CPxMASTER	8h

Function

The register provides a CPU-specific response indicating the physical bus master number of the core that is making the access. The 32-bit response defines the physical master number for processor x.

- CPxMASTER = 0x00000000

NOTE

A read from a CPU returns the appropriate processor information. Reads from a non-CPU bus master returns all zeroes. Write accesses are terminated with an error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										PPMN					
W																
Reset	0	0	0	0	0	0	0	0	0	0	u	u	u	u	u	u

Fields

Field	Function
31-6 —	Reserved
5-0 PPMN	Processor x Physical Master Number This read-only field defines the physical bus master number for CPUx. PPMN = 0x00

11.4.2.5 Processor X Count Register (CPxCOUNT)

Offset

Register	Offset
CPxCOUNT	Ch

Function

The register indicates the total number of processor cores in the chip configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														PCNT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1-0 PCNT	Processor Count This read-only field defines the processor count for the chip configuration: PCNT = 00 (Single Core)

11.4.2.6 Processor X Configuration Register 0 (CPxCFG0)

Offset

Register	Offset
CPxCFG0	10h

Function

The CPxCFG0 register provides a CPU-specific response detailing configuration information, in this case, information on the Level 1 caches (if present).

NOTE

Reset values for the Processor X Configuration Register 0:

- For CPU0, the CPxCFG0 value read is 0x04080000
- If the read access is not from a CPU, then the value read is 0x00000000

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+ICSZ)}$, where ICSZ is non-zero; a ICSZ = 0 indicates the memory is not present. • if no Instruction Cache, then ICSZ = 0x00 • if a 1 Kbyte Instruction Cache, then ICSZ = 0x01 • if a 2 Kbyte Instruction Cache, then ICSZ = 0x02 • if a 4 Kbyte Instruction Cache, then ICSZ = 0x03 • if an 8 Kbyte Instruction Cache, then ICSZ = 0x04 • if a 16 Kbyte Instruction Cache, then ICSZ = 0x05 • if a 32 Kbyte Instruction Cache, then ICSZ = 0x06 • if a 64 Kbyte Instruction Cache, then ICSZ = 0x07 • if a 128 Kbyte Instruction Cache, then ICSZ = 0x08
23-16 ICWY	Level 1 Instruction Cache Ways This read-only field provides the number of cache ways for the Instruction Cache. ICWY=0x00 indicates not present.
15-8 DCSZ	Level 1 Data Cache Size This read-only field provides an encoded value of the Data Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+DCSZ)}$, where DCSZ is non-zero; a DCSZ = 0 indicates the memory is not present. • if no Data Cache, then DCSZ = 0x00 • if a 1 Kbyte Data Cache, then DCSZ = 0x01 • if a 2 Kbyte Data Cache, then DCSZ = 0x02 • if a 4 Kbyte Data Cache, then DCSZ = 0x03 • if an 8 Kbyte Data Cache, then DCSZ = 0x04 • if a 16 Kbyte Data Cache, then DCSZ = 0x05 • if a 32 Kbyte Data Cache, then DCSZ = 0x06 • if a 64 Kbyte Data Cache, then DCSZ = 0x07 • if a 128 Kbyte Data Cache, then DCSZ = 0x08
7-0 DCWY	Level 1 Data Cache Ways This read-only field provides the number of cache ways for the Data Cache. DCWY=0x00 indicates not present.

11.4.2.7 Processor X Configuration Register 1 (CPxCFG1)

Offset

Register	Offset
CPxCFG1	14h

Function

The CPxCFG1 register provides a CPU-specific response detailing configuration information, in this case, information on a Level 2 cache (if present).

NOTE

Reset values for the Processor X Configuration Register 1:

- For CPU0, the CPxCFG1 value read is 0x00000000
- If the read access is not from a CPU, then the value read is 0x00000000

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	L2SZ								L2WY							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 L2SZ	Level 2 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+L2SZ)}$, where L2SZ is non-zero; a L2SZ = 0 indicates the memory is not present. • if no Level 2 Cache, then L2SZ = 0x00 • if a 1 Kbyte Level 2 Cache, then L2SZ = 0x01 • if a 2 Kbyte Level 2 Cache, then L2SZ = 0x02 • if a 4 Kbyte Level 2 Cache, then L2SZ = 0x03 • if an 8 Kbyte Level 2 Cache, then L2SZ = 0x04 • if a 16 Kbyte Level 2 Cache, then L2SZ = 0x05

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• if a 32 Kbyte Level 2 Cache, then L2SZ = 0x06 • if a 64 Kbyte Level 2 Cache, then L2SZ = 0x07 • if a 128 Kbyte Level 2 Cache, then L2SZ = 0x08
23-16 L2WY	Level 2 Instruction Cache Ways This read-only field provides the number of cache ways for the Instruction Cache. L2WY=0x00 indicates not present.
15-0 —	Reserved

11.4.2.8 Processor X Configuration Register 2 (CPxCFG2)

Offset

Register	Offset
CPxCFG2	18h

Function

The CPxCFG2 register provides a CPU-specific response detailing configuration information, in this case, information on tightly-coupled local memories (if present).

NOTE

Reset values for the Processor X Configuration Register 2:

- For CPU0, the CPxCFG2 value read is 0x00010001
- If the read access is not from a CPU, then the value read is 0x00000000

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMLSZ								Reserved							
W																
Reset	u	u	u	u	u	u	u	u	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMUSZ								Reserved							
W																
Reset	u	u	u	u	u	u	u	u	0	0	0	0	0	0	0	1

Fields

Field	Function
31-24 TMLSZ	Tightly-coupled Memory Lower Size This field provides an encoded value of the tightly-coupled local memory lower size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+TMLSZ)}$, where TMLSZ is non-zero; a TMLSZ = 0 indicates the memory is not present. • if no TCML, then TMLSZ = 0x00 • if a 1 Kbyte TCML, then TMLSZ = 0x01 • if a 2 Kbyte TCML, then TMLSZ = 0x02 • if a 4 Kbyte TCML, then TMLSZ = 0x03 • if an 8 Kbyte TCML, then TMLSZ = 0x04 • if a 16 Kbyte TCML, then TMLSZ = 0x05 • if a 32 Kbyte TCML, then TMLSZ = 0x06 • if a 64 Kbyte TCML, then TMLSZ = 0x07 • if a 128 Kbyte TCML, then TMLSZ = 0x08 • if a 256 Kbyte TCML, then TMLSZ = 0x09
23-16 —	Reserved
15-8 TMUSZ	Tightly-coupled Memory Upper Size This field provides an encoded value of the tightly-coupled local memory upper size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+TMUSZ)}$, where TMUSZ is non-zero; a TMUSZ = 0 indicates the memory is not present. • if no TCMU, then TMUSZ = 0x00 • if a 1 Kbyte TCMU, then TMUSZ = 0x01 • if a 2 Kbyte TCMU, then TMUSZ = 0x02 • if a 4 Kbyte TCMU, then TMUSZ = 0x03 • if an 8 Kbyte TCMU, then TMUSZ = 0x04 • if a 16 Kbyte TCMU, then TMUSZ = 0x05 • if a 32 Kbyte TCMU, then TMUSZ = 0x06 • if a 64 Kbyte TCMU, then TMUSZ = 0x07 • if a 128 Kbyte TCMU, then TMUSZ = 0x08 • if a 256 Kbyte TCMU, then TMUSZ = 0x09
7-0 —	Reserved

11.4.2.9 Processor X Configuration Register 3 (CPxCFG3)

Offset

Register	Offset
CPxCFG3	1Ch

Function

The CPxCFG3 register provides a CPU-specific response detailing configuration information, in this case, information on processor options.

Details regarding Arm core features, such as Trust Zone, Jazelle, SIMD/NEON, can be found in Arm documentation.

NOTE

Reset values for the Processor X Configuration Register 3:

- For CPU0, the CPxCFG3 value read is 0x00000231
- If the read access is not from a CPU, then the value read is 0x00000000

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						SBP		0	BB	CMP	TZ	MMU	JAZ	SIMD	FPU
W																
Reset	0	0	0	0	0	0	u	u	0	u	u	u	u	u	u	u

Fields

Field	Function
31-10 —	Reserved
9-8 SBP	System Bus Ports This field defines the number of physical connections to the system bus fabric for this processor.
7 —	Reserved
6 BB	Bit Banding This field defines if the processor supports "bit banding".

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Bit Banding is not supported. 1b - Bit Banding is supported.
5 CMP	Core Memory Protection unit This field indicates if the core memory protection hardware is included in the processor. 0b - Core Memory Protection is not included. 1b - Core Memory Protection is included.
4 TZ	Trust Zone This field indicates if the Trust Zone capabilities are supported in the processor. 0b - Trust Zone support is not included. 1b - Trust Zone support is included.
3 MMU	Memory Management Unit Memory Management Unit. This field indicates if the virtual memory management capabilities are supported in the processor. 0b - MMU support is not included. 1b - MMU support is included.
2 JAZ	Jazelle support This field indicates if Jazelle hardware is supported in the processor. 0b - Jazelle support is not included. 1b - Jazelle support is included.
1 SIMD	SIMD/NEON instruction support This field indicates if the instruction set extensions supporting SIMD and/or NEON capabilities are supported in the processor. 0b - SIMD/NEON support is not included. 1b - SIMD/NEON support is included.
0 FPU	Floating Point Unit This field indicates if hardware support for floating point capabilities are supported in the processor. 0b - FPU support is not included. 1b - FPU support is included.

11.4.2.10 Processor 0 Type Register (CP0TYPE)

Offset

Register	Offset
CP0TYPE	20h

Function

The register provides the personality of Processor 0. The 32-bit response includes 3 ASCII characters defining the CPU type, along with a byte defining the logical revision number. The logical revision number follows Arm's rYpZ nomenclature.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERSONALITY															
W																
Reset	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERSONALITY								RYPZ							
W																
Reset	0	0	1	1	0	0	1	1	0	0	0	0	0	0	0	1

Fields

Field	Function
31-8 PERSONALITY	Processor 0 Personality This read-only field defines the processor personality for CP0
7-0 RYPZ	Processor 0 Revision This read-only field defines the processor revision for CPU0: 0x00 corresponds to the r0p0 core release. 0x01 corresponds to the r0p1 core release. ...

11.4.2.11 Processor 0 Number Register (CP0NUM)

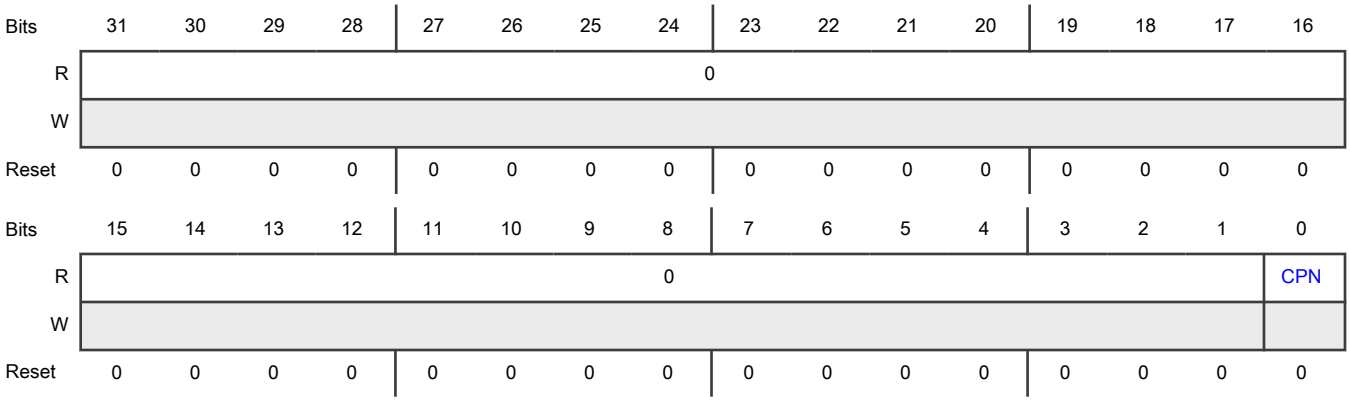
Offset

Register	Offset
CP0NUM	24h

Function

The register provides the logical processor number of Processor 0. The logical processor number is always 0.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CPN	Processor 0 Number This zero-filled word defines the logical processor number for CPU0 If single core configuration, then CPN = 0

11.4.2.12 Processor 0 Master Register (CP0MASTER)

Offset

Register	Offset
CP0MASTER	28h

Function

The register provides the physical bus master number of Processor 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										PPMN					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5-0 PPMN	Processor 0 Physical Master Number This read-only field defines the physical bus master number for CPU0

11.4.2.13 Processor 0 Count Register (CP0COUNT)

Offset

Register	Offset
CP0COUNT	2Ch

Function

The register indicates the total number of processor cores in the chip configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														PCNT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1-0 PCNT	Processor Count This read-only field defines the processor count for the chip configuration: PCNT = 00 (Single Core)

11.4.2.14 Processor 0 Configuration Register 0 (CP0CFG0)

Offset

Register	Offset
CP0CFG0	30h

Function

The CP0CFG0 register provides information on the CPU0 Level 1 caches (if present).

NOTE

Reset values for the Processor 0 Configuration Register 0:

- CP0CFG0 = 0x04080000

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 ICSZ	Level 1 Instruction Cache Size

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+ICSZ)}$, where ICSZ is non-zero; a ICSZ = 0 indicates the memory is not present. • if no Instruction Cache, then ICSZ = 0x00 • if a 1 Kbyte Instruction Cache, then ICSZ = 0x01 • if a 2 Kbyte Instruction Cache, then ICSZ = 0x02 • if a 4 Kbyte Instruction Cache, then ICSZ = 0x03 • if an 8 Kbyte Instruction Cache, then ICSZ = 0x04 • if a 16 Kbyte Instruction Cache, then ICSZ = 0x05 • if a 32 Kbyte Instruction Cache, then ICSZ = 0x06 • if a 64 Kbyte Instruction Cache, then ICSZ = 0x07 • if a 128 Kbyte Instruction Cache, then ICSZ = 0x08
23-16 ICWY	Level 1 Instruction Cache Ways This read-only field provides the number of cache ways for the Instruction Cache. ICWY=0x00 indicates not present.
15-8 DCSZ	Level 1 Data Cache Size This read-only field provides an encoded value of the Data Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+DCSZ)}$, where DCSZ is non-zero; a DCSZ = 0 indicates the memory is not present. • if no Data Cache, then DCSZ = 0x00 • if a 1 Kbyte Data Cache, then DCSZ = 0x01 • if a 2 Kbyte Data Cache, then DCSZ = 0x02 • if a 4 Kbyte Data Cache, then DCSZ = 0x03 • if an 8 Kbyte Data Cache, then DCSZ = 0x04 • if a 16 Kbyte Data Cache, then DCSZ = 0x05 • if a 32 Kbyte Data Cache, then DCSZ = 0x06 • if a 64 Kbyte Data Cache, then DCSZ = 0x07 • if a 128 Kbyte Data Cache, then DCSZ = 0x08
7-0 DCWY	Level 1 Data Cache Ways This read-only field provides the number of cache ways for the Data Cache. DCWY=0x00 indicates not present.

11.4.2.15 Processor 0 Configuration Register 1 (CP0CFG1)

Offset

Register	Offset
CP0CFG1	34h

Function

The CP0CFG1 register provides information on CPU0 Level 2 cache (if present).

NOTE

Reset values for the Processor 0 Configuration Register 1:

- CP0CFG1 = 0x00000000

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	L2SZ								L2WY							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 L2SZ	Level 2 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+L2SZ)}$, where L2SZ is non-zero; a L2SZ = 0 indicates the memory is not present. • if no Level 2 Cache, then L2SZ = 0x00 • if a 1 Kbyte Level 2 Cache, then L2SZ = 0x01 • if a 2 Kbyte Level 2 Cache, then L2SZ = 0x02 • if a 4 Kbyte Level 2 Cache, then L2SZ = 0x03 • if an 8 Kbyte Level 2 Cache, then L2SZ = 0x04 • if a 16 Kbyte Level 2 Cache, then L2SZ = 0x05 • if a 32 Kbyte Level 2 Cache, then L2SZ = 0x06

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• if a 64 Kbyte Level 2 Cache, then L2SZ = 0x07 • if a 128 Kbyte Level 2 Cache, then L2SZ = 0x08
23-16 L2WY	Level 2 Instruction Cache Ways This read-only field provides the number of cache ways for the Instruction Cache. L2WY=0x00 indicates not present.
15-0 —	Reserved

11.4.2.16 Processor 0 Configuration Register 2 (CP0CFG2)

Offset

Register	Offset
CP0CFG2	38h

Function

The CP0CFG2 register provides information on CPU0 tightly-coupled local memories (if present).

NOTE

Reset values for the Processor 0 Configuration Register 2:

- CP0CFG2 = 0x00010001

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMLSZ								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMUSZ								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-24 TMLSZ	Tightly-coupled Memory Lower Size This field provides an encoded value of the tightly-coupled local memory lower size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+TMLSZ)}$, where TMLSZ is non-zero; a TMLSZ = 0 indicates the memory is not present. • if no TCML, then TMLSZ = 0x00 • if a 1 Kbyte TCML, then TMLSZ = 0x01 • if a 2 Kbyte TCML, then TMLSZ = 0x02 • if a 4 Kbyte TCML, then TMLSZ = 0x03 • if an 8 Kbyte TCML, then TMLSZ = 0x04 • if a 16 Kbyte TCML, then TMLSZ = 0x05 • if a 32 Kbyte TCML, then TMLSZ = 0x06 • if a 64 Kbyte TCML, then TMLSZ = 0x07 • if a 128 Kbyte TCML, then TMLSZ = 0x08 • if a 256 Kbyte TCML, then TMLSZ = 0x09
23-16 —	Reserved
15-8 TMUSZ	Tightly-coupled Memory Upper Size This field provides an encoded value of the tightly-coupled local memory upper size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+TMUSZ)}$, where TMUSZ is non-zero; a TMUSZ = 0 indicates the memory is not present. • if no TCMU, then TMUSZ = 0x00 • if a 1 Kbyte TCMU, then TMUSZ = 0x01 • if a 2 Kbyte TCMU, then TMUSZ = 0x02 • if a 4 Kbyte TCMU, then TMUSZ = 0x03 • if an 8 Kbyte TCMU, then TMUSZ = 0x04 • if a 16 Kbyte TCMU, then TMUSZ = 0x05 • if a 32 Kbyte TCMU, then TMUSZ = 0x06 • if a 64 Kbyte TCMU, then TMUSZ = 0x07 • if a 128 Kbyte TCMU, then TMUSZ = 0x08 • if a 256 Kbyte TCMU, then TMUSZ = 0x09
7-0 —	Reserved

11.4.2.17 Processor 0 Configuration Register 3 (CP0CFG3)

Offset

Register	Offset
CP0CFG3	3Ch

Function

The CP0CFG3 register provides information on Processor 0 options.

Details regarding Arm core features, such as Trust Zone, Jazelle, SIMD/NEON, can be found in Arm documentation.

NOTE

Reset values for the Processor 0 Configuration Register 3:

- CP0CFG3 = 0x00000231

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						SBP		0	BB	CMP	TZ	MMU	JAZ	SIMD	FPU
W																
Reset	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1

Fields

Field	Function
31-10 —	Reserved
9-8 SBP	System Bus Ports This field defines the number of physical connections to the system bus fabric for this processor.
7 —	Reserved
6 BB	Bit Banding This field defines if the processor supports "bit banding". 0b - Bit Banding is not supported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Bit Banding is supported.
5 CMP	Core Memory Protection unit This field indicates if the core memory protection hardware is included in the processor. 0b - Core Memory Protection is not included. 1b - Core Memory Protection is included.
4 TZ	Trust Zone This field indicates if the Trust Zone capabilities are supported in the processor. 0b - Trust Zone support is not included. 1b - Trust Zone support is included.
3 MMU	Memory Management Unit Memory Management Unit. This field indicates if the virtual memory management capabilities are supported in the processor. 0b - MMU support is not included. 1b - MMU support is included.
2 JAZ	Jazelle support This field indicates if Jazelle hardware is supported in the processor. 0b - Jazelle support is not included. 1b - Jazelle support is included.
1 SIMD	SIMD/NEON instruction support This field indicates if the instruction set extensions supporting SIMD and/or NEON capabilities are supported in the processor. 0b - SIMD/NEON support is not included. 1b - SIMD/NEON support is included.
0 FPU	Floating Point Unit This field indicates if hardware support for floating point capabilities are supported in the processor. 0b - FPU support is not included. 1b - FPU support is included.

11.4.2.18 On-Chip Memory Descriptor Register (OCMDR0)

Offset

Register	Offset
OCMDR0	400h

Function

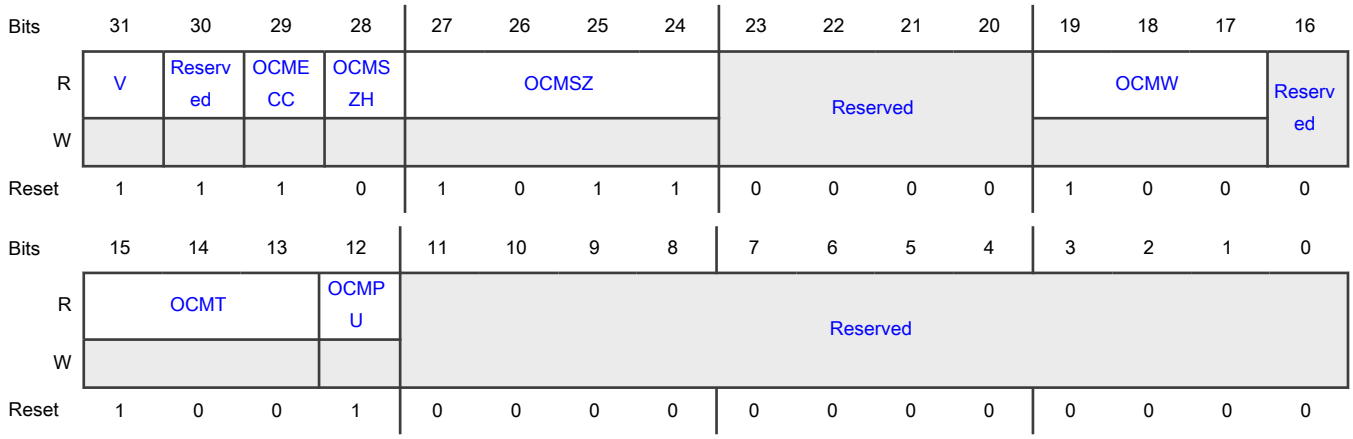
This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- The access permissions for this module are set in the TRDC.
- 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Writes from other bus masters are ignored.
- Any access with a size other than 32 bits are terminated with an error.

The following table describes the OCMDRn reset values and the associated memory type:

OCMDRn	Reset Value	On-Chip Memory Type
OCMDR0	0xEB089000	Program Flash
OCMDR1	0xD8047000	ROM
OCMDR2	0xE5041000	System RAM
OCMDR3	0xE7041000	System RAM
OCMDR4	0xD7041000	System RAM
OCMDR5	0xE4041000	System RAM

Diagram



Fields

Field	Function
31 V	V OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OCMEMn is not present. 1b - OCMEMn is present.
30 —	FMT This Reserved field always has the value of 1.
29 OCMECC	OCMECC OCMEM has support for ECC. 0b - OCMEMn does not have ECC support. 1b - OCMEMn has ECC support.
28 OCMSZH	OCMSZH OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0b - OCMEMn is a power-of-2 capacity. 1b - OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.
27-24 OCMSZ	OCMSZ OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. 0000b - no OCMEMn 0001b - 1KB OCMEMn 0010b - 2KB OCMEMn 0011b - 4KB OCMEMn 0100b - 8KB OCMEMn 0101b - 16KB OCMEMn 0110b - 32KB OCMEMn 0111b - 64KB OCMEMn 1000b - 128KB OCMEMn 1001b - 256KB OCMEMn 1010b - 512KB OCMEMn 1011b - 1MB OCMEMn 1100b - 2MB OCMEMn 1101b - 4MB OCMEMn 1110b - 8MB OCMEMn 1111b - 16MB OCMEMn
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 OCMW	OCMW OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000b-001b - Reserved 010b - OCMEMn 32-bits wide 011b - OCMEMn 64-bits wide 100b - OCMEMn 128-bits wide 101b - OCMEMn 256-bits wide 110b-111b - Reserved
16 —	Reserved
15-13 OCMT	OCMT OCMEM Type. This field defines the type of the on-chip memory: 000b - OCMEMn is a System RAM. 001b - Reserved 010b - Reserved 011b - OCMEMn is a ROM. 100b - OCMEMn is a Program Flash. 101b - Reserved 110b - Reserved 111b - Reserved
12 OCMPU	OCMPU OCMEM Memory Protection Unit. This field is reserved for this device.
11-0 —	Reserved

11.4.2.19 On-Chip Memory Descriptor Register (OCMDR1)

Offset

Register	Offset
OCMDR1	404h

Function

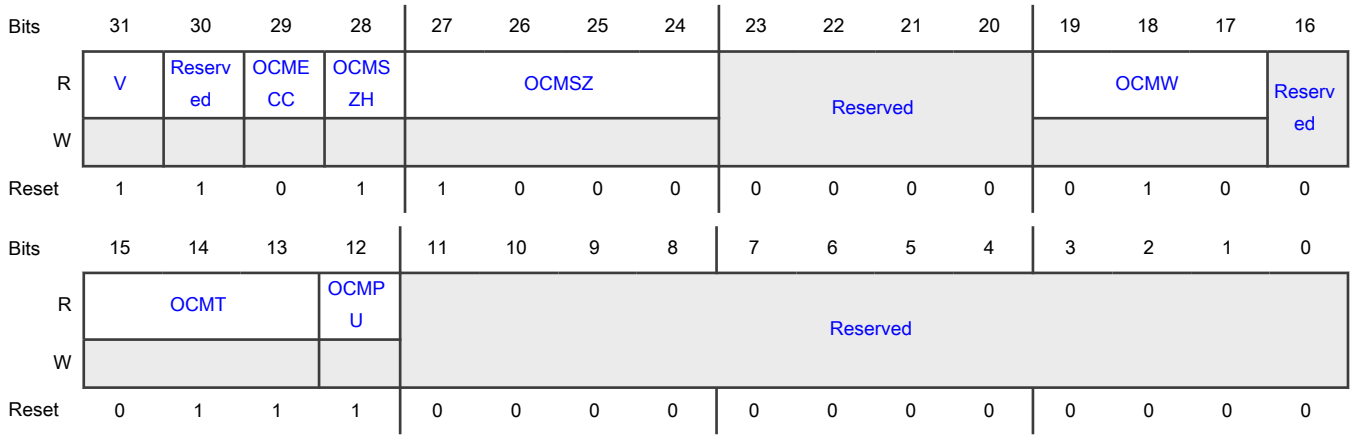
This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- The access permissions for this module are set in the TRDC.
- 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Writes from other bus masters are ignored.
- Any access with a size other than 32 bits are terminated with an error.

The following table describes the OCMDRn reset values and the associated memory type:

OCMDRn	Reset Value	On-Chip Memory Type
OCMDR0	0xEB089000	Program Flash
OCMDR1	0xD8047000	ROM
OCMDR2	0xE5041000	System RAM
OCMDR3	0xE7041000	System RAM
OCMDR4	0xD7041000	System RAM
OCMDR5	0xE4041000	System RAM

Diagram



Fields

Field	Function
31 V	V OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OCMEMn is not present. 1b - OCMEMn is present.
30 —	FMT This Reserved field always has the value of 1.
29 OCMECC	OCMECC OCMEM has support for ECC. 0b - OCMEMn does not have ECC support. 1b - OCMEMn has ECC support.
28 OCMSZH	OCMSZH OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0b - OCMEMn is a power-of-2 capacity. 1b - OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.
27-24 OCMSZ	OCMSZ OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. 0000b - no OCMEMn 0001b - 1KB OCMEMn 0010b - 2KB OCMEMn 0011b - 4KB OCMEMn 0100b - 8KB OCMEMn 0101b - 16KB OCMEMn 0110b - 32KB OCMEMn 0111b - 64KB OCMEMn 1000b - 128KB OCMEMn 1001b - 256KB OCMEMn 1010b - 512KB OCMEMn 1011b - 1MB OCMEMn 1100b - 2MB OCMEMn 1101b - 4MB OCMEMn 1110b - 8MB OCMEMn 1111b - 16MB OCMEMn
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 OCMW	OCMW OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000b-001b - Reserved 010b - OCMEMn 32-bits wide 011b - OCMEMn 64-bits wide 100b - OCMEMn 128-bits wide 101b - OCMEMn 256-bits wide 110b-111b - Reserved
16 —	Reserved
15-13 OCMT	OCMT OCMEM Type. This field defines the type of the on-chip memory: 000b - OCMEMn is a System RAM. 001b - Reserved 010b - Reserved 011b - OCMEMn is a ROM. 100b - OCMEMn is a Program Flash. 101b - Reserved 110b - Reserved 111b - Reserved
12 OCMPU	OCMPU OCMEM Memory Protection Unit. This field is reserved for this device.
11-0 —	Reserved

11.4.2.20 On-Chip Memory Descriptor Register (OCMDR2)

Offset

Register	Offset
OCMDR2	408h

Function

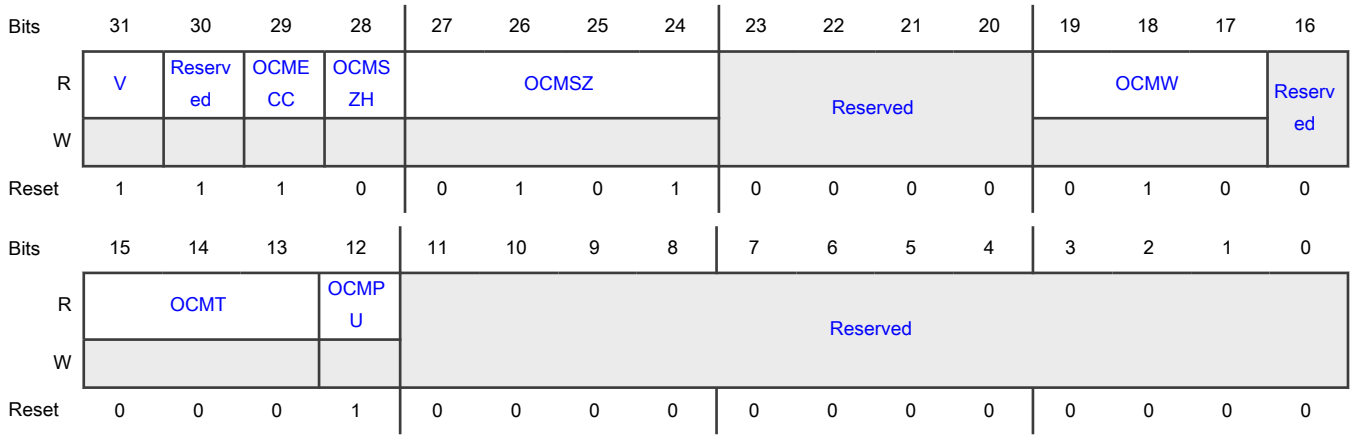
This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- The access permissions for this module are set in the TRDC.
- 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Writes from other bus masters are ignored.
- Any access with a size other than 32 bits are terminated with an error.

The following table describes the OCMDRn reset values and the associated memory type:

OCMDRn	Reset Value	On-Chip Memory Type
OCMDR0	0xEB089000	Program Flash
OCMDR1	0xD8047000	ROM
OCMDR2	0xE5041000	System RAM
OCMDR3	0xE7041000	System RAM
OCMDR4	0xD7041000	System RAM
OCMDR5	0xE4041000	System RAM

Diagram



Fields

Field	Function
31 V	V OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OCMEMn is not present. 1b - OCMEMn is present.
30 —	FMT This Reserved field always has the value of 1.
29 OCMECC	OCMECC OCMEM has support for ECC. 0b - OCMEMn does not have ECC support. 1b - OCMEMn has ECC support.
28 OCMSZH	OCMSZH OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0b - OCMEMn is a power-of-2 capacity. 1b - OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.
27-24 OCMSZ	OCMSZ OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. 0000b - no OCMEMn 0001b - 1KB OCMEMn 0010b - 2KB OCMEMn 0011b - 4KB OCMEMn 0100b - 8KB OCMEMn 0101b - 16KB OCMEMn 0110b - 32KB OCMEMn 0111b - 64KB OCMEMn 1000b - 128KB OCMEMn 1001b - 256KB OCMEMn 1010b - 512KB OCMEMn 1011b - 1MB OCMEMn 1100b - 2MB OCMEMn 1101b - 4MB OCMEMn 1110b - 8MB OCMEMn 1111b - 16MB OCMEMn
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 OCMW	OCMW OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000b-001b - Reserved 010b - OCMEMn 32-bits wide 011b - OCMEMn 64-bits wide 100b - OCMEMn 128-bits wide 101b - OCMEMn 256-bits wide 110b-111b - Reserved
16 —	Reserved
15-13 OCMT	OCMT OCMEM Type. This field defines the type of the on-chip memory: 000b - OCMEMn is a System RAM. 001b - Reserved 010b - Reserved 011b - OCMEMn is a ROM. 100b - OCMEMn is a Program Flash. 101b - Reserved 110b - Reserved 111b - Reserved
12 OCMPU	OCMPU OCMEM Memory Protection Unit. This field is reserved for this device.
11-0 —	Reserved

11.4.2.21 On-Chip Memory Descriptor Register (OCMDR3)

Offset

Register	Offset
OCMDR3	40Ch

Function

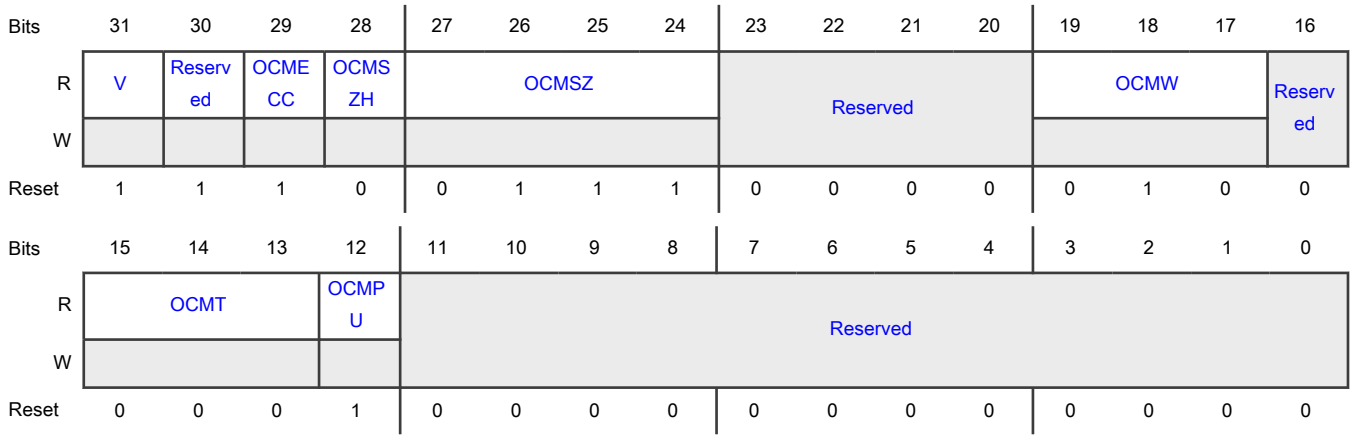
This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- The access permissions for this module are set in the TRDC.
- 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Writes from other bus masters are ignored.
- Any access with a size other than 32 bits are terminated with an error.

The following table describes the OCMDRn reset values and the associated memory type:

OCMDRn	Reset Value	On-Chip Memory Type
OCMDR0	0xEB089000	Program Flash
OCMDR1	0xD8047000	ROM
OCMDR2	0xE5041000	System RAM
OCMDR3	0xE7041000	System RAM
OCMDR4	0xD7041000	System RAM
OCMDR5	0xE4041000	System RAM

Diagram



Fields

Field	Function
31 V	V OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OCMEMn is not present. 1b - OCMEMn is present.
30 —	FMT This Reserved field always has the value of 1.
29 OCMECC	OCMECC OCMEM has support for ECC. 0b - OCMEMn does not have ECC support. 1b - OCMEMn has ECC support.
28 OCMSZH	OCMSZH OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0b - OCMEMn is a power-of-2 capacity. 1b - OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.
27-24 OCMSZ	OCMSZ OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. 0000b - no OCMEMn 0001b - 1KB OCMEMn 0010b - 2KB OCMEMn 0011b - 4KB OCMEMn 0100b - 8KB OCMEMn 0101b - 16KB OCMEMn 0110b - 32KB OCMEMn 0111b - 64KB OCMEMn 1000b - 128KB OCMEMn 1001b - 256KB OCMEMn 1010b - 512KB OCMEMn 1011b - 1MB OCMEMn 1100b - 2MB OCMEMn 1101b - 4MB OCMEMn 1110b - 8MB OCMEMn 1111b - 16MB OCMEMn
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 OCMW	OCMW OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000b-001b - Reserved 010b - OCMEMn 32-bits wide 011b - OCMEMn 64-bits wide 100b - OCMEMn 128-bits wide 101b - OCMEMn 256-bits wide 110b-111b - Reserved
16 —	Reserved
15-13 OCMT	OCMT OCMEM Type. This field defines the type of the on-chip memory: 000b - OCMEMn is a System RAM. 001b - Reserved 010b - Reserved 011b - OCMEMn is a ROM. 100b - OCMEMn is a Program Flash. 101b - Reserved 110b - Reserved 111b - Reserved
12 OCMPU	OCMPU OCMEM Memory Protection Unit. This field is reserved for this device.
11-0 —	Reserved

11.4.2.22 On-Chip Memory Descriptor Register (OCMDR4)

Offset

Register	Offset
OCMDR4	410h

Function

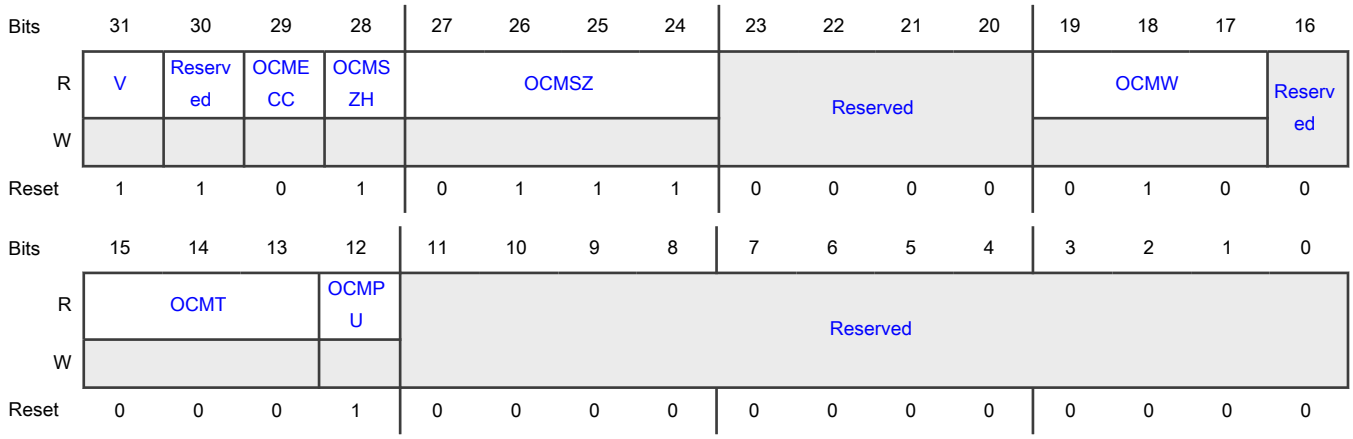
This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- The access permissions for this module are set in the TRDC.
- 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Writes from other bus masters are ignored.
- Any access with a size other than 32 bits are terminated with an error.

The following table describes the OCMDRn reset values and the associated memory type:

OCMDRn	Reset Value	On-Chip Memory Type
OCMDR0	0xEB089000	Program Flash
OCMDR1	0xD8047000	ROM
OCMDR2	0xE5041000	System RAM
OCMDR3	0xE7041000	System RAM
OCMDR4	0xD7041000	System RAM
OCMDR5	0xE4041000	System RAM

Diagram



Fields

Field	Function
31 V	V OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OCMEMn is not present. 1b - OCMEMn is present.
30 —	FMT This Reserved field always has the value of 1.
29 OCMECC	OCMECC OCMEM has support for ECC. 0b - OCMEMn does not have ECC support. 1b - OCMEMn has ECC support.
28 OCMSZH	OCMSZH OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0b - OCMEMn is a power-of-2 capacity. 1b - OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.
27-24 OCMSZ	OCMSZ OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. 0000b - no OCMEMn 0001b - 1KB OCMEMn 0010b - 2KB OCMEMn 0011b - 4KB OCMEMn 0100b - 8KB OCMEMn 0101b - 16KB OCMEMn 0110b - 32KB OCMEMn 0111b - 64KB OCMEMn 1000b - 128KB OCMEMn 1001b - 256KB OCMEMn 1010b - 512KB OCMEMn 1011b - 1MB OCMEMn 1100b - 2MB OCMEMn 1101b - 4MB OCMEMn 1110b - 8MB OCMEMn 1111b - 16MB OCMEMn
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 OCMW	OCMW OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000b-001b - Reserved 010b - OCMEMn 32-bits wide 011b - OCMEMn 64-bits wide 100b - OCMEMn 128-bits wide 101b - OCMEMn 256-bits wide 110b-111b - Reserved
16 —	Reserved
15-13 OCMT	OCMT OCMEM Type. This field defines the type of the on-chip memory: 000b - OCMEMn is a System RAM. 001b - Reserved 010b - Reserved 011b - OCMEMn is a ROM. 100b - OCMEMn is a Program Flash. 101b - Reserved 110b - Reserved 111b - Reserved
12 OCMPU	OCMPU OCMEM Memory Protection Unit. This field is reserved for this device.
11-0 —	Reserved

11.4.2.23 On-Chip Memory Descriptor Register (OCMDR5)

Offset

Register	Offset
OCMDR5	414h

Function

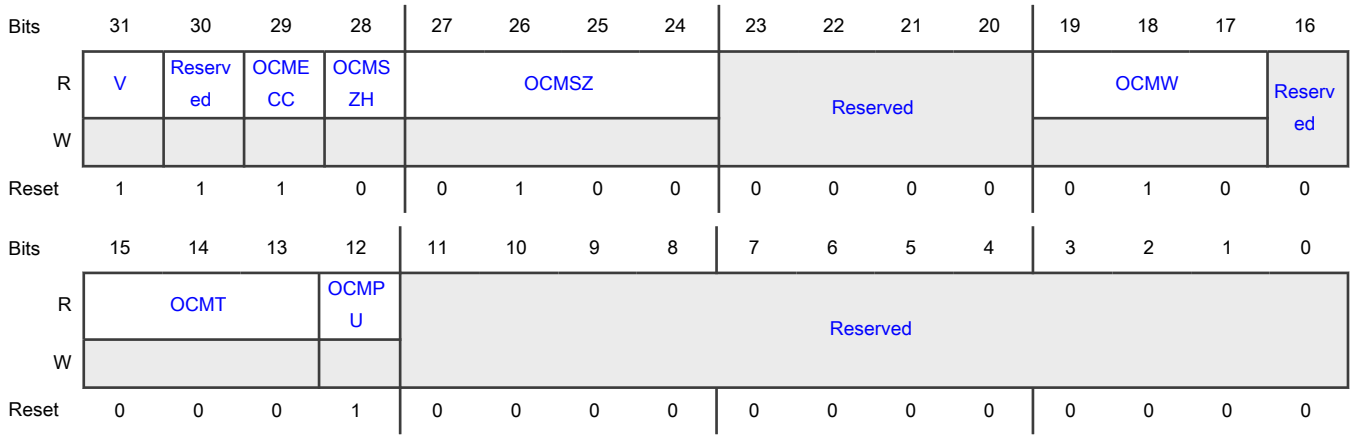
This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- The access permissions for this module are set in the TRDC.
- 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Writes from other bus masters are ignored.
- Any access with a size other than 32 bits are terminated with an error.

The following table describes the OCMDRn reset values and the associated memory type:

OCMDRn	Reset Value	On-Chip Memory Type
OCMDR0	0xEB089000	Program Flash
OCMDR1	0xD8047000	ROM
OCMDR2	0xE5041000	System RAM
OCMDR3	0xE7041000	System RAM
OCMDR4	0xD7041000	System RAM
OCMDR5	0xE4041000	System RAM

Diagram



Fields

Field	Function
31 V	V OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - OCMEMn is not present. 1b - OCMEMn is present.
30 —	FMT This Reserved field always has the value of 1.
29 OCMECC	OCMECC OCMEM has support for ECC. 0b - OCMEMn does not have ECC support. 1b - OCMEMn has ECC support.
28 OCMSZH	OCMSZH OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0b - OCMEMn is a power-of-2 capacity. 1b - OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.
27-24 OCMSZ	OCMSZ OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. 0000b - no OCMEMn 0001b - 1KB OCMEMn 0010b - 2KB OCMEMn 0011b - 4KB OCMEMn 0100b - 8KB OCMEMn 0101b - 16KB OCMEMn 0110b - 32KB OCMEMn 0111b - 64KB OCMEMn 1000b - 128KB OCMEMn 1001b - 256KB OCMEMn 1010b - 512KB OCMEMn 1011b - 1MB OCMEMn 1100b - 2MB OCMEMn 1101b - 4MB OCMEMn 1110b - 8MB OCMEMn 1111b - 16MB OCMEMn
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-17 OCMW	OCMW OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000b-001b - Reserved 010b - OCMEMn 32-bits wide 011b - OCMEMn 64-bits wide 100b - OCMEMn 128-bits wide 101b - OCMEMn 256-bits wide 110b-111b - Reserved
16 —	Reserved
15-13 OCMT	OCMT OCMEM Type. This field defines the type of the on-chip memory: 000b - OCMEMn is a System RAM. 001b - Reserved 010b - Reserved 011b - OCMEMn is a ROM. 100b - OCMEMn is a Program Flash. 101b - Reserved 110b - Reserved 111b - Reserved
12 OCMPU	OCMPU OCMEM Memory Protection Unit. This field is reserved for this device.
11-0 —	Reserved

11.4.2.24 Secure Interrupt Request (SECURE_IRQ)

Offset

Register	Offset
SECURE_IRQ	800h

Function

The SECURE_IRQ register provides a mechanism for non-secure code to generate a low latency pulsed exception into the secure privileged domain. This register is accessible in a non-secure peripheral address slot that non-secure code can write a specific value that forces an interrrupt that executes as a secure exception and enters the secure domain.

This register can only be written in nonsecure mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEC_IRQ_ARG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEC_IRQ_ARG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SEC_IRQ_ARG	Secure Interrupt Argument 32-bit secure interrupt argument. Any 32-bit write triggers a pulsed secure interrupt request.

11.4.2.25 Unique ID 0 (UID0)

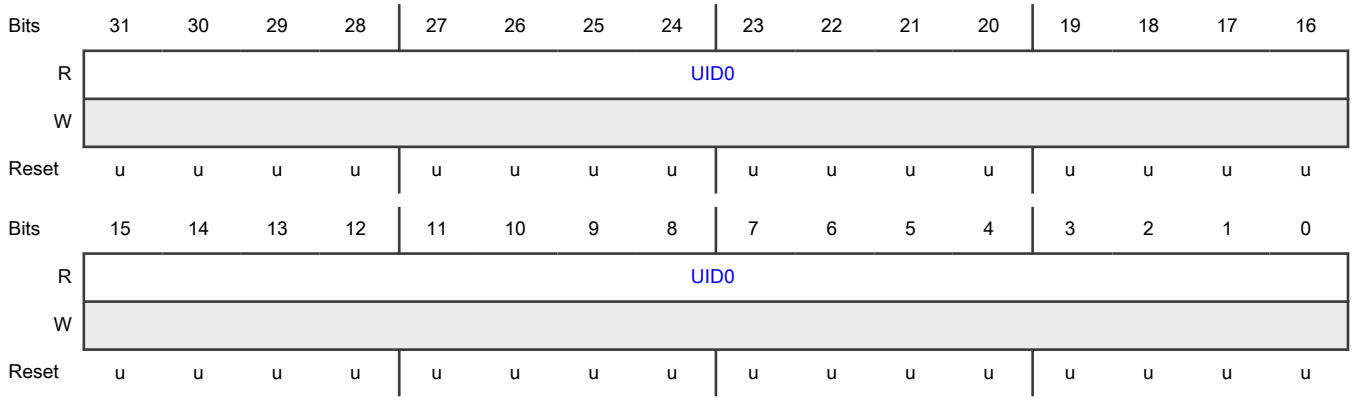
Offset

Register	Offset
UID0	810h

Function

UID[31:0] of the 128-bit UID. The UID is loaded directly from UUID fuse words.

Diagram



Fields

Field	Function
31-0 UID0	Unique ID 0 UID[31:0] of the 128-bit UID.

11.4.2.26 Unique ID 1 (UID1)

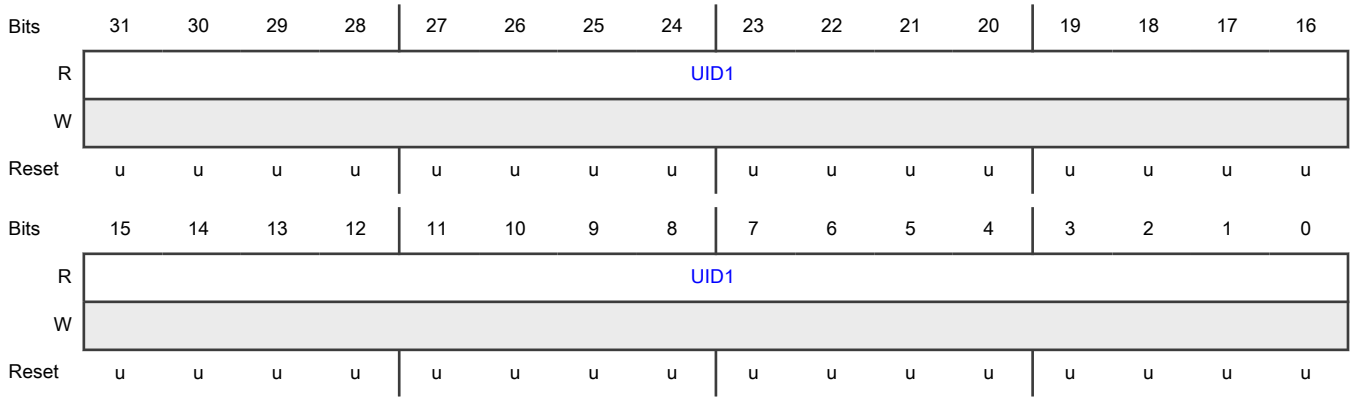
Offset

Register	Offset
UID1	814h

Function

UID[63:32] of the 128-bit UID. The UID is loaded directly from UUID fuse words.

Diagram



Fields

Field	Function
31-0 UID1	Unique ID 1 UID[63:32] of the 128-bit UID.

11.4.2.27 Unique ID 2 (UID2)

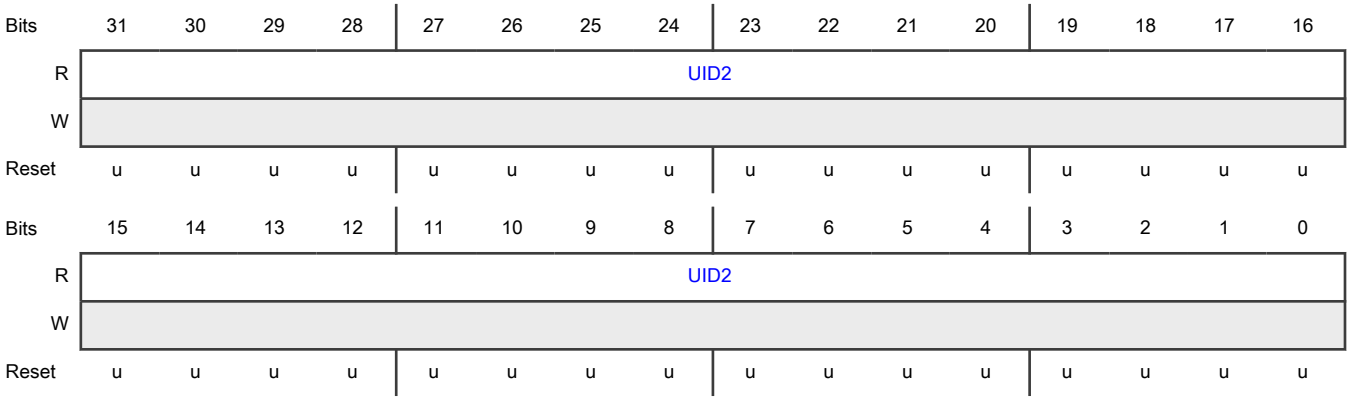
Offset

Register	Offset
UID2	818h

Function

UID[95:64] of the 128-bit UID. The UID is loaded directly from UUID fuse words.

Diagram



Fields

Field	Function
31-0 UID2	Unique ID 2 UID[95:64] of the 128-bit UID.

11.4.2.28 Unique ID 3 (UID3)

Offset

Register	Offset
UID3	81Ch

Function

UID[127:96] of the 128-bit UID. The UID is loaded directly from UUID fuse words.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	UID3															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UID3															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 UID3	Unique ID 3 UID[127:96] of the 128-bit UID.

11.4.2.29 System ID (SID)

Offset

Register	Offset
SID	820h

Function

Loaded from IFR

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FAMID				RADIOF				BLEF				FLSZ			
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RAMSZ			SEC	CAN	I3C	VREF	FLXIO	CMP	PINID			SIREV		QI	
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-28 FAMID	Family ID See the chip-specific information for details.
27-24 RADIOF	Radio Feature 0000b - 802.15.4 0001b - Bluetooth LE 0010b - Bluetooth LE + 15.4
23-20 BLEF	Bluetooth LE Feature 0000b - No Bluetooth LE present 1111b - Bluetooth LE Upgrade
19-16 FLSZ	Flash Size 1101b - 512 KB 1111b - 1 MB
15-13 RAMSZ	RAM Size 000b - 96 KB 111b - 128 KB
12 SEC	Secure Enclave Presence 0b - No Secure Enclave 1b - Secure Enclave present
11 CAN	CAN Presence 0b - No CAN 1b - CAN present
10 I3C	I3C Presence 0b - No I3C 1b - I3C present
9 VREF	VREF Presence 0b - No VREF 1b - VREF present
8 FLXIO	FlexIO Presence 0b - No FlexIO 1b - FlexIO present
7	CMP Presence

Table continues on the next page...

Table continued from the previous page...

Field	Function
CMP	0b - No CMP 1b - CMP present
6-4 PINID	Pin Identification 010b - 40HVQFN 011b - 48HVQFN 100b - 56HVQFN
3-2 SIREV	Silicon Revision 00b - Reserved 01b - 2nd Major Spin 10b - 1st Major Spin 11b - Initial mask set
1-0 QI	Qual Info 00b - Reserved 01b - Industrial 10b - Reserved 11b - Auto

Chapter 12
Secure Miscellaneous System Control Module (SMSCM)

12.1 Chip-specific SMSCM information

Table 49. Reference links to related information

Topic	Related module	Reference
Full description	SMSCM	SMSCM
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

12.1.1 Module instances

This device has one instance of the SMSCM module.

12.1.2 On-chip memory control

The SMSCM can control the following on-chip memory (OCRAM) with registers, See TCM arrays for more details about the TCM SRAM assignment.

Table 50. On-chip memory (OCRAM) control

OCRAM	SRAM instance	ECC
OCRAM0	Program Flash	—
OCRAM2	CTCM0-CTCM1	Yes
OCRAM3	STCM0-STCM2	Yes
OCRAM5	STCM5	Yes
Others	Reserved	—

12.2 Overview

The Secure Miscellaneous System Control Module (SMSCM) contains secure CPU configuration registers and secure on-chip memory controller registers.

The SMSCM has two 32-bit registers to allow software to read the values of the LIFECYCLE and LIFECYCLE_B fuses. The SMSCM inputs these fuses, which can change during POR or warm reset. The fuse are stable as reset is negated. After reset is deasserted, if the LIFECYCLE and LIFECYCLE_B values are not complements, a security violation is asserted. The security violation is sent to the security subsystem.

12.3 SMSCM Memory Map/Register Definition

12.3.1 Replicated Control Registers

For certain critical control and configuration functions, a special three D flip-flop (DFF3) register implementation is used.

This design uses a pole, anti-pole implementation where the 101b state represents the negated register state and 010b is the asserted state. In the following register descriptions, the replicated function is shown as a single read data bit plus a 3-bit write data field. The register read data bit is aligned as the "middle bit" within the 3-bit field and the 3-bit write data field is defined as "W2S, W5C". This nomenclature defines a 3-bit write data field of value 2 (010b) is needed to set the asserted state of the three

DFFs, while a 3-bit write data field of value 5 (101b) is required to clear the combined DFF3 state. All other write data values do not affect the register state. For reads, the 3-bit field returns {0,register_state,0}b, that is, the register state in the middle bit and zeroes for both adjacent bits.

To change the control state, the user must write BOTH pole and anti-pole registers. The order of the register writes, pole then anti-pole versus anti-pole then pole does not matter. The programming model is updated after the first register is written, however the control state in the hardware doesn't change until the second register is written.

The control state is asserted when the pole register is written with 010b and the anti-pole register is written with 101b. The control state is deasserted when the pole register is written with 101b and the antipole register is written with 010b. If both pole and anti-pole registers are written with the same legal value (010b or 101b), a security violation is asserted. The security violation remains asserted until system reset asserts.

12.3.2 SMSCM register descriptions

12.3.2.1 SMSCM memory map

Writes to the SMSCM programming model must be 32-bits. Non 32-bit writes are terminated with a bus error.

Reads of 8-bit, 16-bit and 32-bit size are allowed.

Writes to read-only registers and reads of write-only registers are terminated with a bus error.

The access permissions for this module are set in the TRDC.

SMSCM base address: 4001_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Debug Enable (DBGEN)	32	RW	0000_0000h
4h	Debug Enable Complement (DBGEN_B)	32	RW	0022_2222h
8h	Debug Enable Lock (DBGEN_LOCK)	32	RW	0000_0000h
20h	Debug Authentication Beacon (DBG_AUTH_BEACON)	32	RW	See section
30h	Lifecycle Fuse Word (LIFECYCLE)	32	R	See section
34h	Lifecycle Fuse Word Complement (LIFECYCLE_B)	32	R	See section
40h	ROM Lockout Register (ROM_LOCKOUT)	32	RW	0000_0000h
100h	Security Counter Register (SCTR)	32	RW	0000_0000h
104h	Security Counter Plus 1 Register (SCTRP1)	32	W	0000_0000h
10Ch	Security Counter Minus 1 Register (SCTRM1)	32	W	0000_0000h
114h	Security Counter Plus X Register (SCTRPX)	32	W	0000_0000h
11Ch	Security Counter Minus X Register (SCTRMX)	32	W	0000_0000h
400h	On-Chip Memory Descriptor Register (OCMDR0)	32	RW	0000_0000h
408h	On-Chip Memory Descriptor Register (OCMDR2)	32	RW	0000_0003h
40Ch	On-Chip Memory Descriptor Register (OCMDR3)	32	RW	0000_0003h
414h	On-Chip Memory Descriptor Register (OCMDR5)	32	RW	0000_0003h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
480h	On-Chip Memory ECC Control Register (OCMECR)	32	RW	0000_0000h
488h	On-Chip Memory ECC Interrupt Register (OCMEIR)	32	RW	0000_0000h
490h	On-Chip Memory Fault Address Register (OCMFAR)	32	R	0000_0000h
494h	On-Chip Memory Fault Attribute Register (OCMFTR)	32	R	0000_0000h
498h	On-Chip Memory ECC Fault Data High Register (OCMFDRH)	32	R	0000_0000h
49Ch	On-Chip Memory ECC Fault Data Low Register (OCMFDRL)	32	R	0000_0000h
C00h	Core Platform Control Register (CPCR)	32	RW	0000_0001h

12.3.2.2 Debug Enable (DBGEN)

Offset

Register	Offset
DBGEN	0h

Function

This register implements debug access control in six 3-bit pole, anti-pole states (DFF3) while the DBGEN_B register implements independent complement versions of the same debug access control states. This register only resets on POR. For warm resets, the value remains in the register and updated when the fuses are valid.

The DBGEN/SPIDEN/NIDEN/SPNIDEN register fields can be updated when DBGEN_LOCK[LOCK] = 000b. The ALTDBGEN register field can be updated when DBGEN_LOCK[ALT_DBGEN_LOCK] = 000b. The ALTEN register field can be updated when DBGEN_LOCK[ALT_EN_LOCK] = 000b. The register is fields are unlocked at reset.

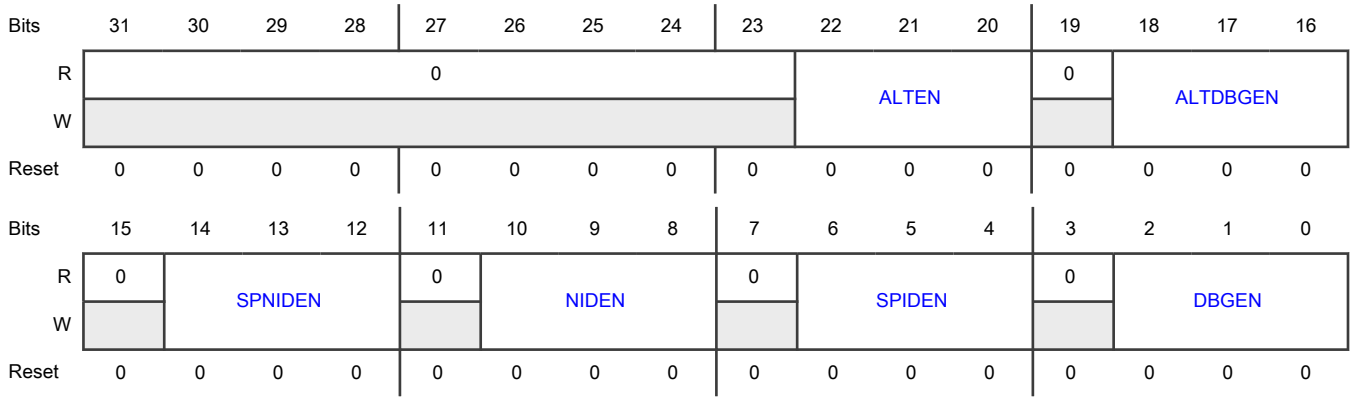
Writes to locked register fields are simply ignored and are not error terminated. Therefore, software must read the DBGEN register after each write to ensure the write completed.

As power-on reset is negated and the fuse initialization is complete, the DBGEN and DBGEN_B are loaded with values determined by the encoded converged lifecycle fuses.

The pole/anti-pole updating sequence works independently for ALTEN, ALTDBGEN, SPNIDEN, NIDEN, SPIDEN, DBGEN bitfields.

For reads, each of the DFF3 3-bit fields return {0,register_state,0}b.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 ALTEN	Alternate Enable (DFF3 bitfield) This DFF3 control provides an alternate enable. This field can only be written when DBGEN_LOCK[ALT_EN_LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Alternate Disabled. 010b - Alternate Enabled. When writing 010b - W2S - Enable Alternate. 101b - W5C - Disable Alternate.
19 —	Reserved
18-16 ALTDBGEN	Alternate Invasive Debug Enable (DFF3 bitfield) This DFF3 control provides the ability to enable invasive debug in an Alternate CPU. This field can only be written when DBGEN_LOCK[ALT_DBGEN_LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Alternate Invasive Debug Disabled. 010b - Alternate Invasive Debug Enabled. When writing 010b - W2S - Enable Alternate Invasive Debug. 101b - W5C - Disable Alternate Invasive Debug.
15 —	Reserved
14-12 SPNIDEN	Secure Non-Invasive Debug Enable (DFF3 bitfield) This DFF3 control provides the ability to enable secure non-invasive debug. This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Secure Non-Invasive Debug Disabled. 010b - Secure Non-Invasive Debug Enabled. When writing 010b - W2S - Enable Secure Non-Invasive Debug. 101b - W5C - Disable Secure Non-Invasive Debug.
11 —	Reserved
10-8 NIDEN	Non-Invasive Debug Enable (DFF3 bitfield) This DFF3 control provides the ability to enable non-invasive debug. This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Non-Invasive Debug Disabled. 010b - Non-Invasive Debug Enabled. When writing 010b - W2S - Enable Non-Invasive Debug. 101b - W5C - Disable Non-Invasive Debug.
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6-4 SPIDEN	Secure Invasive Debug Enable (DFF3 bitfield) This DFF3 control provides the ability to enable secure invasive debug. This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Secure Invasive Debug Disabled. 010b - Secure Invasive Debug Enabled. When writing 010b - W2S - Enable Secure Invasive Debug. 101b - W5C - Disable Secure Invasive Debug.
3 —	Reserved
2-0 DBGEN	Invasive Debug Enable (DFF3 bitfield) This DFF3 control provides the ability to enable invasive debug. This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Invasive Debug Disabled. 010b - Invasive Debug Enabled. When writing 010b - W2S - Enable Invasive Debug. 101b - W5C - Disable Invasive Debug.

12.3.2.3 Debug Enable Complement (DBGEN_B)

Offset

Register	Offset
DBGEN_B	4h

Function

This register implements debug access control in six 3-bit pole, anti-pole states (DFF3) while the DBGEN_B register implements independent complement versions of the same debug access control states. This register only resets on POR. For warm resets, the value remains in the register and updated when the fuses are valid.

The DBGEN_B/SPIDEN_B/NIDEN_B/SPNIDEN_B register fields can be updated when DBGEN_LOCK[LOCK] = 000b. The ALTDBGEN_B register field can be updated when DBGEN_LOCK[ALT_DBGEN_LOCK] = 000b. The ALTEN_B register field can be updated when DBGEN_LOCK[ALT_EN_LOCK] = 000b. The register is fields are unlocked at reset.

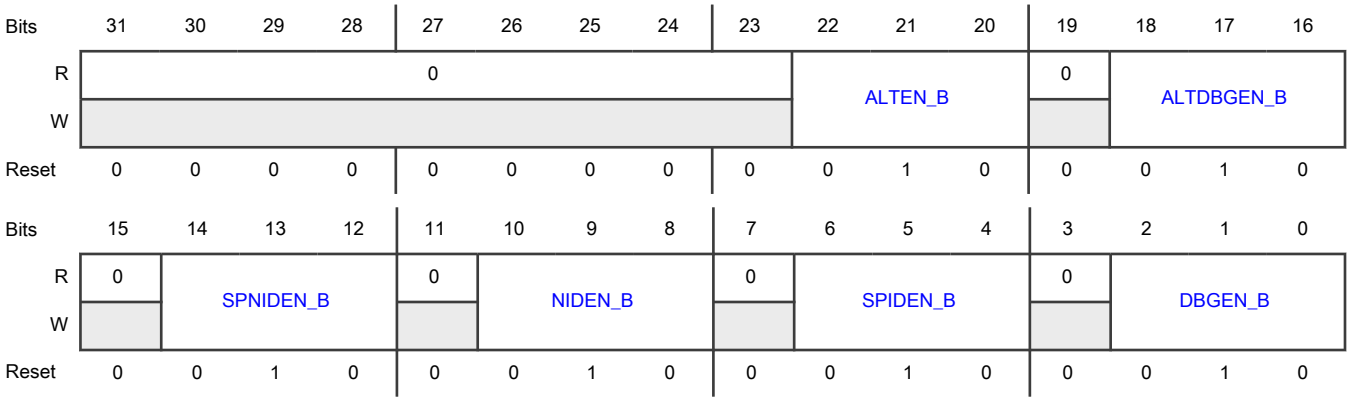
Writes to locked register fields are simply ignored and are not error terminated. Therefore, software must read the DBGEN_B register after each write to ensure the write completed.

As power-on reset is negated and the fuse initialization is complete, the DBGEN and DBGEN_B are loaded with values determined by the encoded converged lifecycle fuses.

The pole/anti-pole updating sequence works independently for ALTDBGEN, SPNIDEN, NIDEN, SPIDEN, DBGEN bitfields.

For reads, each of the DFF3 3-bit fields return {0,register_state,0}b.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 ALTEN_B	Alternate Enable Complement (DFF3 bitfield) This DFF3 control provides the complement version of ALT_EN. This field can only be written when DBGEN_LOCK[ALT_EN_LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Alternrate Enabled. 010b - Alternate Disabled. When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - W2S - Disable Alternate. 101b - W5C - Enable Alternate.
19 —	Reserved
18-16 ALTDBGEN_B	Alternate Invasive Debug Enable Complement (DFF3 bitfield) This DFF3 control provides the complement version of ALT_DBGEN. This field can only be written when DBGEN_LOCK[ALT_DBGEN_LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Alternate Invasive Debug Enabled. 010b - Alternate Invasive Debug Disabled. When writing 010b - W2S - Alternate Disable Invasive Debug. 101b - W5C - Alternate Enable Invasive Debug.
15 —	Reserved
14-12 SPNIDEN_B	Secure Non-Invasive Debug Enable Complement (DFF3 bitfield) This DFF3 control provides the complement version of SPNIDEN. **NOTE** This field behaves differently for register reads and writes. When reading 000b - Secure Non-Invasive Debug Enabled. 010b - Secure Non-Invasive Debug Disabled. When writing 010b - W2S - Disable Secure Non-Invasive Debug. 101b - W5C - Enable Secure Non-Invasive Debug.
11 —	Reserved
10-8 NIDEN_B	Non-Invasive Debug Enable Complement (DFF3 bitfield) This DFF3 control provides the complement version of NIDEN.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Non-Invasive Debug Enabled. 010b - Non-Invasive Debug Disabled. When writing 010b - W2S - Disable Non-Invasive Debug. 101b - W5C - Enable Non-Invasive Debug.
7 —	Reserved
6-4 SPIDEN_B	Secure Invasive Debug Enable - Complement (DFF3 bitfield) This DFF3 control provides the complement version of SPIDEN. This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Secure Invasive Debug Enabled. 010b - Secure Invasive Debug Disabled. When writing 010b - W2S - Disable Secure Invasive Debug. 101b - W5C - Enable Secure Invasive Debug.
3 —	Reserved
2-0 DBGEN_B	Invasive Debug Enable Complement (DFF3 bitfield) This DFF3 control provides the complement version of DBGEN. This field can only be written when DBGEN_LOCK[LOCK] = 000b **NOTE** This field behaves differently for register reads and writes. When reading 000b - Invasive Debug Enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - Invasive Debug Disabled. When writing 010b - W2S - Disable Invasive Debug. 101b - W5C - Enable Invasive Debug.

12.3.2.4 Debug Enable Lock (DBGEN_LOCK)

Offset

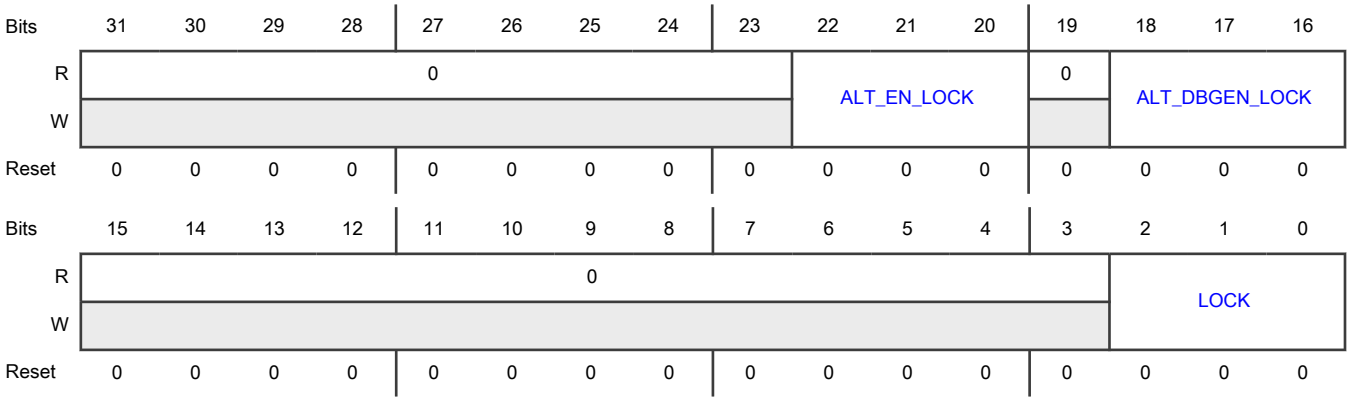
Register	Offset
DBGEN_LOCK	8h

Function

DBG_EN_LOCK fuse is used by ROM to control whether this lock is set prior to executing customer code.

Writes to locked register fields are simply ignored and are not error terminated. Therefore, software must read the DBGEN_LOCK register after each write to ensure the write completed.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 ALT_EN_LOCK	Alternate Lock (DFF3 bitfield) This DFF3 control provides the ability to prevent write accesses to the DBGEN[ALT_EN], DBGEN[ALT_EN_B] and DBGEN_LOCK[ALT_EN_LOCK] registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 000b - ALTEN, ALTEN_B, ALT_EN_LOCK unlocked. 010b - ALTEN, ALTEN_B, ALT_EN_LOCK locked. ALTEN, ALTEN_B, ALT_EN_LOCK locked and cannot be written until the next system reset. They remain readable. When writing 010b,000b,001b,011b,100b,111b - Lock DBGEN[ALTEN], DBGEN_B[ALTEN_B, and DBGEN_LOCK[ALT_EN_LOCK]. Writing any value other than 101b locks the DBGEN[ALTEN], DBGEN_B[ALTEN_B], and DBGEN_LOCK[ALT_EN_LOCK] fields. 101b - f When ALT_EN_LOCK is locked, ALT_EN_LOCK cannot be unlocked with a write of 101b to this field. When ALT_EN_LOCK is unlocked, a write of 101b to this field, ALT_EN_LOCK remains unlocked and ALTEN/ALTEN_B remains writeable.
19 —	Reserved
18-16 ALT_DBGEN_LOCK	Alternate Lock (DFF3 bitfield) This DFF3 control provides the ability to prevent write accesses to the DBGEN[ALT_DBGEN], DBGEN[ALT_DBGEN_B] and DBGEN_LOCK[ALT_DBGEN_LOCK] registers. **NOTE** This field behaves differently for register reads and writes. When reading 000b - ALT_DBGEN, ALT_DBGEN_B, ALT_DBGEN_LOCK unlocked. 010b - ALT_DBGEN, ALT_DBGEN_B, ALT_DBGEN_LOCK locked. ALT_DBGEN, ALT_DBGEN_B, ALT_DBGEN_LOCK locked and cannot be written until the next system reset. They remain readable. When writing 010b,000b,001b,011b,100b,111b - Lock DBGEN[ALTDBGEN], DBGEN_B[ALTDBGEN_B, and DBGEN_LOCK[ALT_DBGEN_LOCK]. Writing any value other than 101b locks the DBGEN[ALTDBGEN], DBGEN_B[ALTDBGEN_B], and DBGEN_LOCK[ALT_DBGEN_LOCK] fields. 101b - When ALT_DBGEN_LOCK is locked, ALT_DBGEN_LOCK cannot be unlocked with a write of 101b to this field. When ALT_DBGEN_LOCK is unlocked, a write of 101b to this field, ALT_DBGEN_LOCK remains unlocked and DBGEN/DBGEN_B remains writeable.
15-3 —	Reserved
2-0	Lock (DFF3 bitfield)

Table continues on the next page...

Table continued from the previous page...

Field	Function
LOCK	This DFF3 control provides the ability to prevent write accesses to the DBGEN[DBGEN, SPIDEN, NIDEN, SPNIDEN], DBGEN_B[DBGEN_B, SPIDEN_B, NIDEN_B, SPNIDEN_B] and DBGEN_LOCK[LOCK] register fields. **NOTE** This field behaves differently for register reads and writes. When reading 000b - DBGEN[SPNIDEN,NIDEN,SPIDEN,DBGEN], DBGEN_B[SPNIDEN_B,NIDEN_B,SPIDEN_B,DBGEN_B], and DBGEN_LOCK[LOCK] unlocked. 010b - DBGEN[SPNIDEN,NIDEN,SPIDEN,DBGEN], DBGEN_B[SPNIDEN_B,NIDEN_B,SPIDEN_B,DBGEN_B], and DBGEN_LOCK[LOCK] locked. DBGEN[SPNIDEN,NIDEN,SPIDEN,DBGEN], DBGEN_B[SPNIDEN_B,NIDEN_B,SPIDEN_B,DBGEN_B], and DBGEN_LOCK[LOCK] are locked and cannot be written until the next system reset. They remain readable. When writing 010b,000b,001b,011b,100b,111b - Lock DBGEN[SPNIDEN,NIDEN,SPIDEN,DBGEN], DBGEN_B[SPNIDEN_B,NIDEN_B,SPIDEN_B,DBGEN_B], and DBGEN_LOCK[LOCK]. Writing any value other than 101b locks DBGEN[SPNIDEN,NIDEN,SPIDEN,DBGEN], DBGEN_B[SPNIDEN_B,NIDEN_B,SPIDEN_B,DBGEN_B], and DBGEN_LOCK[LOCK]. 101b - When DBGEN_LOCK[LOCK] is locked, DBGEN_LOCK[LOCK] cannot be unlocked with a write of 101b to this field. When DBGEN_LOCK[LOCK] is unlocked, a write of 101b to this field, DBGEN_LOCK[LOCK] remains unlocked and the DBGEN[DBGEN, SPIDEN, NIDEN, SPNIDEN],DBGEN_B[DBGEN_B, SPIDEN_B, NIDEN_B, SPNIDEN_B] fields remain writeable.

12.3.2.5 Debug Authentication Beacon (DBG_AUTH_BEACON)

Offset

Register	Offset
DBG_AUTH_BEACON	20h

Function

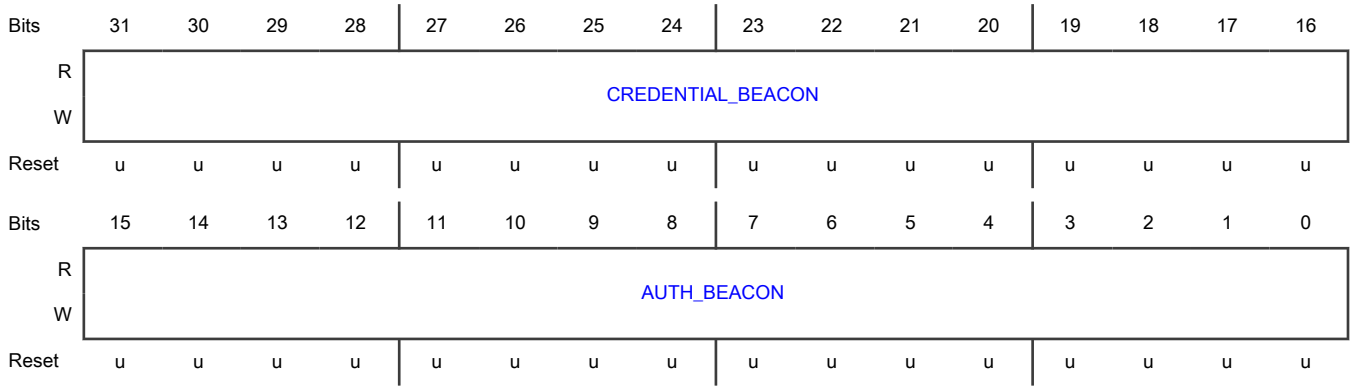
This register only resets on POR, and is not locked by DBGEN_LOCK.

It does not control any logic, written by debug authentication and used only by customer code.

NOTE

The reset value of this register is user defined.

Diagram



Fields

Field	Function
31-16 CREDENTIAL_BEACON	Credential Beacon
15-0 AUTH_BEACON	Authentication Beacon

12.3.2.6 Lifecycle Fuse Word (LIFECYCLE)

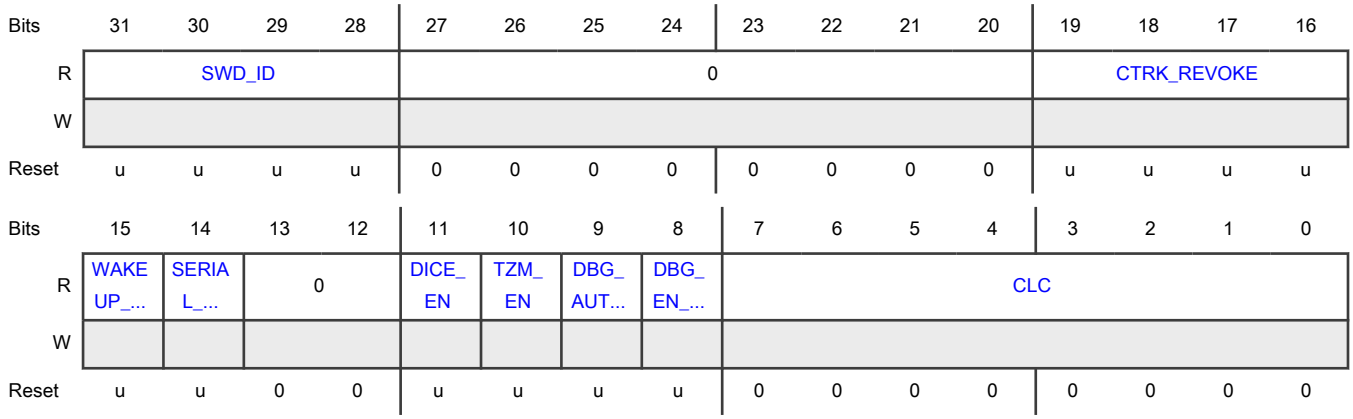
Offset

Register	Offset
LIFECYCLE	30h

Function

The LIFECYCLE register is cleared with POR and loaded directly from the fuses when the fuses are ready.

Diagram



Fields

Field	Function
31-28 SWD_ID	Serial Wire Debug Instance ID
27-20 —	Reserved
19-16 CTRK_REVOKE	Revocation indicator from OEM Firmware Authentication Public Key
15 WAKEUP_DIS	Wakeup Disabled 0b - Boot-ROM LP wakup is enabled. 1b - Boot-ROM LP wakup is disabled.
14 SERIAL_DIS	Serial Download Disabled 0b - Serial download path is enabled. 1b - Serial download path is disabled.
13-12 —	Reserved
11 DICE_EN	DICE Enable 0b - DICE is disabled by default. 1b - DICE is enabled.
10 TZM_EN	Trust Zone Mode Enable 0b - TZ-M is disabled by default, can be enabled by software. 1b - TZ-M is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 DBG_AUTH_DIS	Debug Authentication Disabled 0b - Debug Authentication enabled. 1b - Debug Authentication disabled.
8 DBG_EN_LOCK	Debug Enable Lock 0b - The debug access control registers remain open when jumping to customer code. 1b - The debug access control registers are write-locked before jumping to customer code.
7-0 CLC	Converged Lifecycle 0000_0000b - BLANK 0000_0001b - NXP Fab 0000_0011b - NXP Provisioned 0000_0111b - OEM Open 0000_1111b - OEM Secure World Closed 0001_1111b - OEM Closed 0011_1111b - OEM Return 0111_1111b - NXP Return 1001_1111b - OEM Locked 11xx_xxxxb - BRICK

12.3.2.7 Lifecycle Fuse Word Complement (LIFECYCLE_B)

Offset

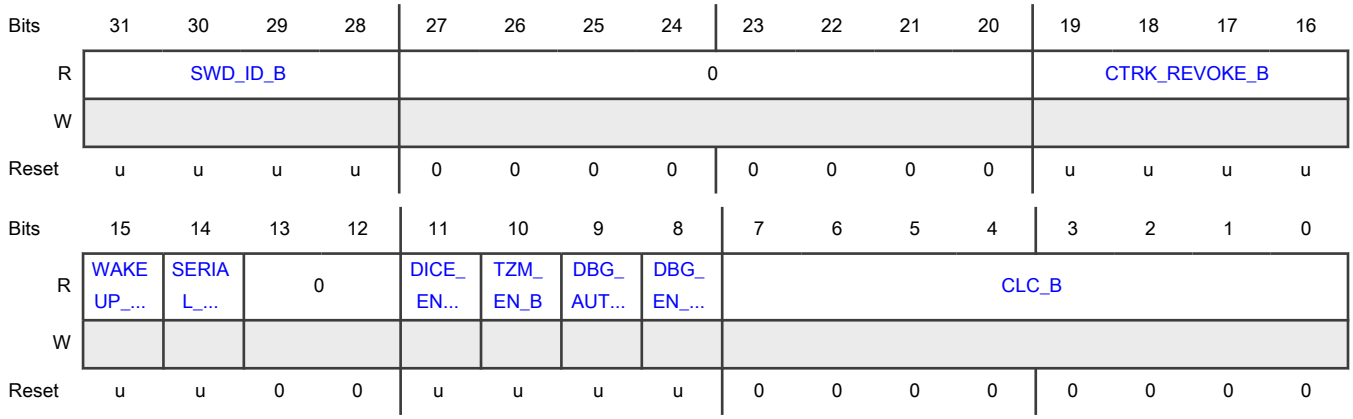
Register	Offset
LIFECYCLE_B	34h

Function

The LIFECYCLE_B register is cleared with POR and loaded directly from the fuses when the fuses are ready.

This register is the complement of the LIFECYCLE register.

Diagram



Fields

Field	Function
31-28 SWD_ID_B	Serial Wire Debug Instance ID Complement
27-20 —	Reserved
19-16 CTRK_REVOKE_B	Revocation indicator from OEM Firmware Authentication Public Key Complement
15 WAKEUP_DIS_B	Wakeup Disabled Complement 0b - Boot-ROM LP wakup is disabled. 1b - Boot-ROM LP wakup is enabled.
14 SERIAL_DIS_B	Serial Download Disabled Complement 0b - Serial download path is disabled. 1b - Serial download path is enabled.
13-12 —	Reserved
11 DICE_EN_B	DICE Enable Complement 0b - DICE is enabled. 1b - DICE is disabled by default.
10 TZM_EN_B	Trust Zone Mode Enable Complement 0b - TZ-M is enabled. 1b - TZ-M is disabled by default, can be enabled by software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 DBG_AUTH_DIS_B	Debug Authentication Disabled Complement 0b - Debug Authentication disabled. 1b - Debug Authentication enabled.
8 DBG_EN_LOCK_B	Debug Enable Lock Complement 0b - The debug access control registers are write-locked before jumping to customer code. 1b - The debug access control registers remain open when jumping to customer code.
7-0 CLC_B	Converged Lifecycle Complement 00xx_xxxxb - BRICK 0110_0000b - OEM Locked 1000_0000b - NXP Return 1100_0000b - OEM Return 1110_0000b - OEM Closed 1111_0000b - OEM Secure World Closed 1111_1000b - OEM Open 1111_1100b - NXP Provisioned 1111_1110b - NXP Fab 1111_1111b - BLANK

12.3.2.8 ROM Lockout Register (ROM_LOCKOUT)

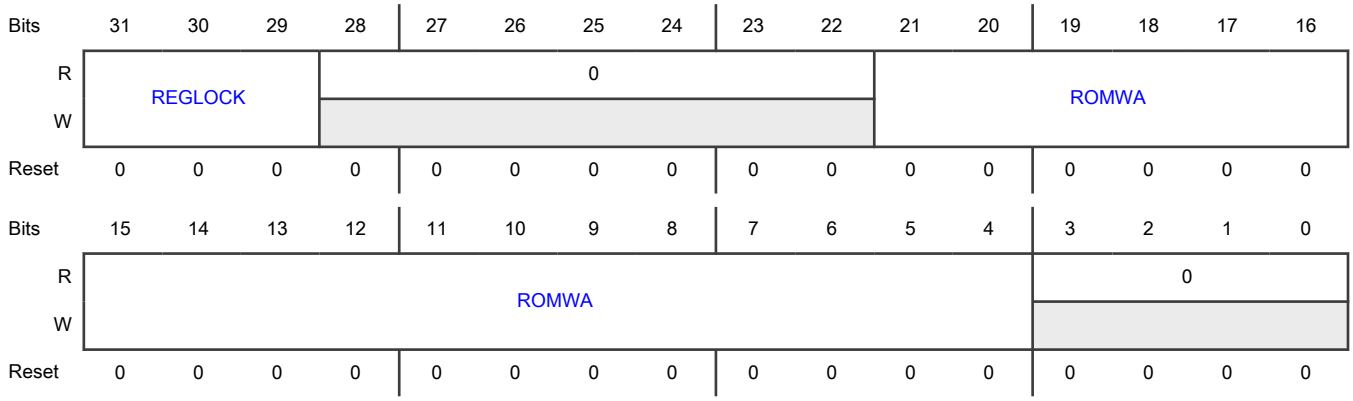
Offset

Register	Offset
ROM_LOCKOUT	40h

Function

This register provides the ability to set a watermark address in the ROM. Once enabled, accesses below the watermark address are not allowed and result in a bus error.

Diagram



Fields

Field	Function
31-29 REGLOCK	ROM_LOCKOUT Register Lock (DFF3 bitfield) This DFF3 control provides the ability to prevent write accesses to the ROM_LOCKOUT register and enable the ROM watermark feature. **NOTE** This field behaves differently for register reads and writes. When reading 000b - ROM_LOCKOUT unlocked. ROM_LOCKOUT is writable, and ROM array is fully accessible. 010b - ROM_LOCKOUT locked. ROM_LOCKOUT register cannot be written until next reset. Accesses to addresses below the ROM_LOCKOUT[ROMWA] result in an error response. When writing 010b,000b,001b,011b,100b,111b - Lock ROM_LOCKOUT register. Writing any value other than 101b locks the ROM_LOCKOUT register. 101b - Writing this value has no effect.
28-22 —	Reserved
21-4 ROMWA	ROM Watermark Address The first accessible address (0-mod-16) after the lockout region. When ROM_LOCKOUT[REGLOCK] = 010b, reads to addresses below this watermark value result in an error response.
3-0 —	Reserved

12.3.2.9 Security Counter Register (SCTR)

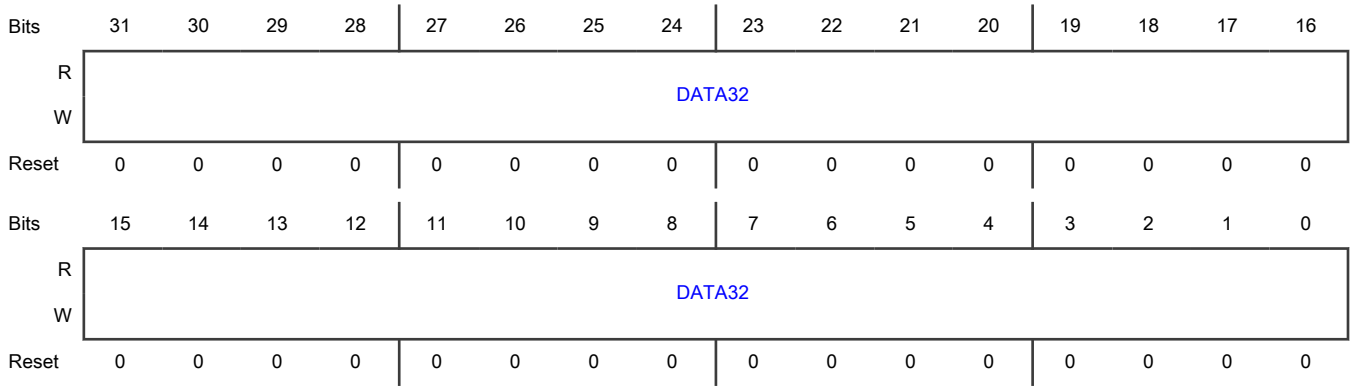
Offset

Register	Offset
SCTR	100h

Function

This register provides a security counter function, which is intended to validate the control flow integrity of CPU code execution. This register is the actual counter register; there are four additional virtual registers that support a set of arithmetic operations including {+1, -1, +x, -x} where x is any 32-bit operand.

Diagram



Fields

Field	Function
31-0 DATA32	Data, 32 bits This is the 32-bit value contained as the security counter.

12.3.2.10 Security Counter Plus 1 Register (SCTRP1)

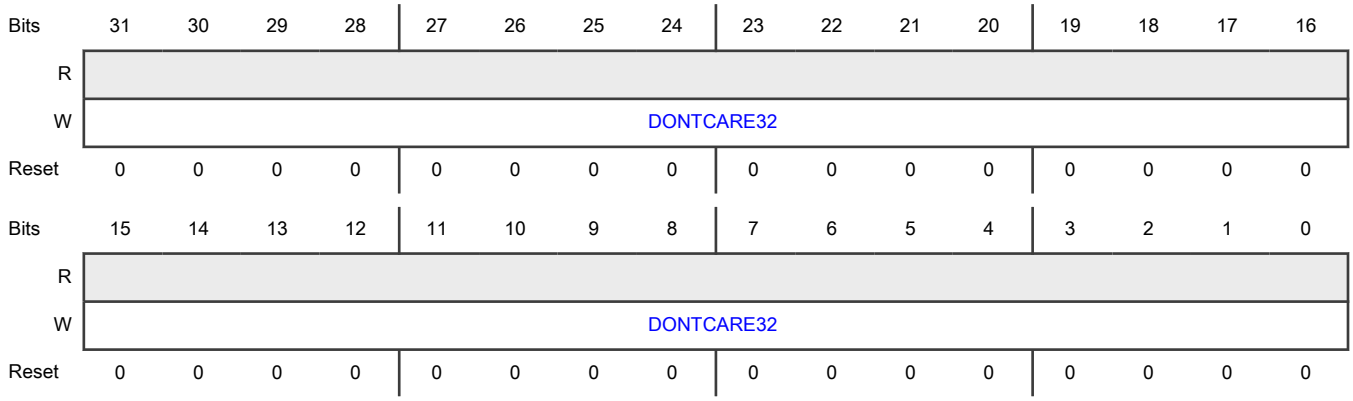
Offset

Register	Offset
SCTRP1	104h

Function

This virtual write-only register provides a security counter arithmetic function - register writes to this address increment the security counter by 1 (+1). The security counter provides a mechanism to validate the control flow integrity of CPU code execution. This is one of four virtual registers that support a set of arithmetic operations including {+1, -1, +x, -x} where x is any 32-bit operand.

Diagram



Fields

Field	Function
31-0 DONTCARE32	Don't Care Data, 32 bits The entire contents of the write data word are ignored as register writes to this location increment the security counter by 1, that, is, next-state SCTR = current-state SCTR + 1.

12.3.2.11 Security Counter Minus 1 Register (SCTRM1)

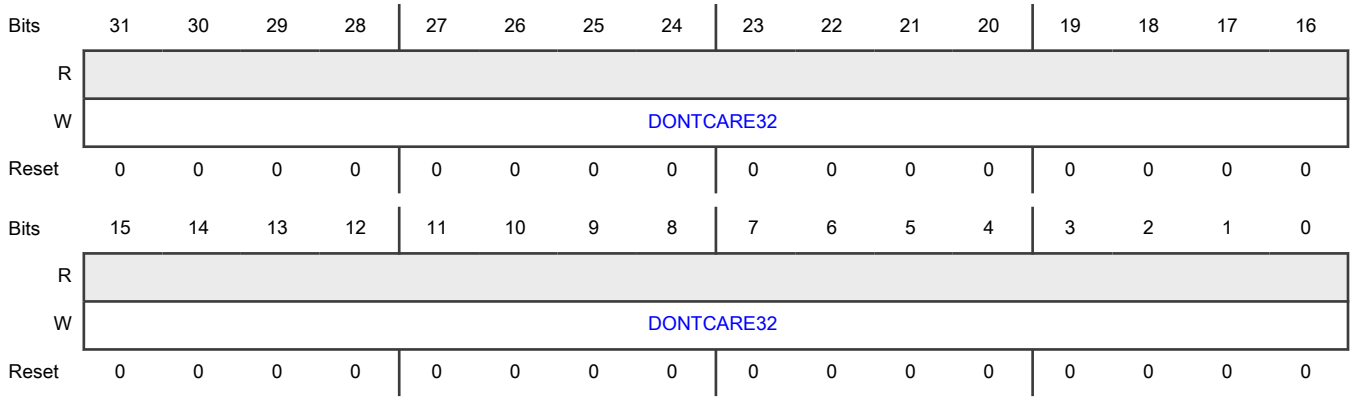
Offset

Register	Offset
SCTRM1	10Ch

Function

This virtual write-only register provides a security counter arithmetic function - register writes to this address decrement the security counter by 1 (-1). The security counter provides a mechanism to validate the control flow integrity of CPU code execution. This is one of four virtual registers that support a set of arithmetic operations including {+1, -1, +x, -x} where x is any 32-bit operand.

Diagram



Fields

Field	Function
31-0 DONTCARE32	Don't Care Data, 32 bits The entire contents of the write data word are ignored as register writes to this location decrement the security counter by 1, that, is, next-state SCTR = current-state SCTR - 1.

12.3.2.12 Security Counter Plus X Register (SCTRPX)

Offset

Register	Offset
SCTRPX	114h

Function

This virtual write-only register provides a security counter arithmetic function - register writes to this address increment the security counter by "x" (+x). The security counter provides a mechanism to validate the control flow integrity of CPU code execution. This is one of four virtual registers that support a set of arithmetic operations including {+1, -1, +x, -x} where x is any 32-bit operand.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA32															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA32	Data, 32 bits The entire contents of the write data word are added to the security counter, that, is, next-state SCTR = current-state SCTR + DATA32.

12.3.2.13 Security Counter Minus X Register (SCTRMX)

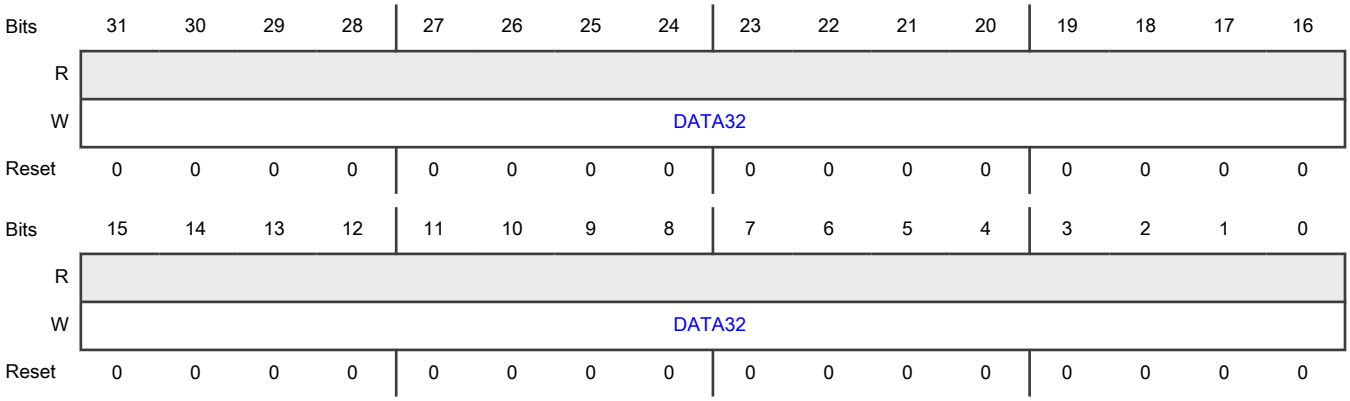
Offset

Register	Offset
SCTRMX	11Ch

Function

This virtual write-only register provides a security counter arithmetic function - register writes to this address decrement the security counter by "x" (-x). The security counter provides a mechanism to validate the control flow integrity of CPU code execution. This is one of four virtual registers that support a set of arithmetic operations including {+1, -1, +x, -x} where x is any 32-bit operand.

Diagram



Fields

Field	Function
31-0 DATA32	Data, 32 bits The entire contents of the write data word are subtracted to the security counter, that, is, next-state SCTR = current-state SCTR - DATA32.

12.3.2.14 On-Chip Memory Descriptor Register (OCMDR0)

Offset

Register	Offset
OCMDR0	400h

Function

This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provides configurable controls (where appropriate).

- Any access with a size other than 32 bits are terminated with an error.

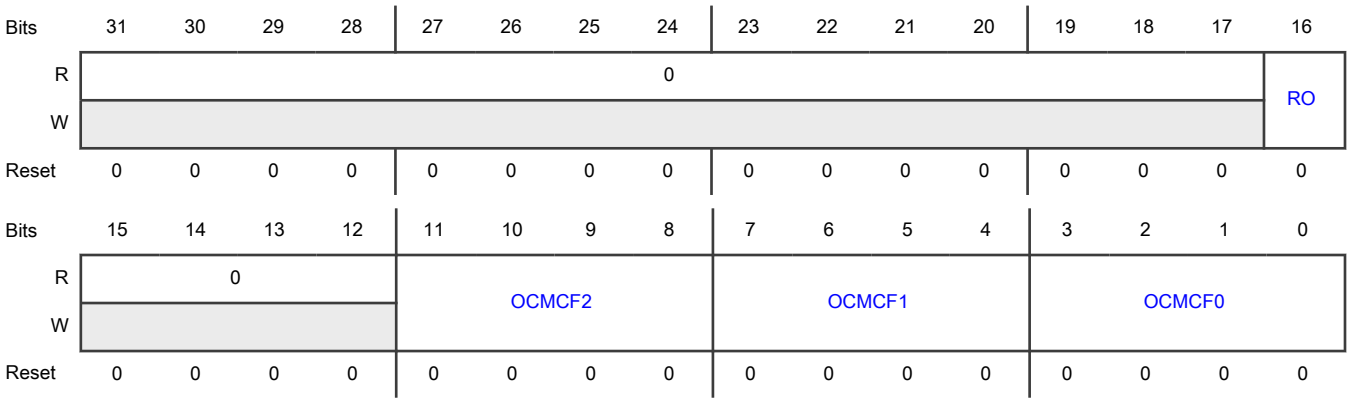
An on-chip memory has a corresponding SMSCM OCMDRn register when one of the following is true:

- The memory type is data flash
- The memory type is program flash
- The memory type is system RAM and ECC = true

The following table describes each on-chip memory type:

On-chip memory n	On-Chip Memory Type
On-chip memory 0	Program Flash
On-chip memory 2	System RAM
On-chip memory 3	System RAM
On-chip memory 5	System RAM

Diagram



Fields

Field	Function
31-17 —	Reserved
16 RO	Read-Only This register bit provides a mechanism to “lock” the configuration state defined by OCMDRn[11:0]. Once asserted, attempted writes to the OCMDRn[11:0] register are ignored until the next reset clears the flag. 0b - Writes to the OCMDRn[11:0] are allowed 1b - Writes to the OCMDRn[11:0] are ignored
15-12 —	Reserved
11-8	OCMEM Control Field 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
OCMCF2	The flash controller needs to be idle when writing to an OCMDRn register associated with Flash memory. Changing controller configuration while active can cause undesired results. The flash cache on this device is a small, read-only cache to help accelerate flash accesses. The 4 bits in CF2 configure and maintain this cache. CF2 = Control Field 2 - for flash cache • CF2[3] - disable flash cache • CF2[2] - disable instruction caching • CF2[1] - disable flash data caching • CF2[0] - clear flash cache CF2[3:1] • 000b - the flash cache is enabled and will cache both flash instruction and flash data fetches • 001b - the flash cache is enabled and will cache flash instruction fetches but not flash data fetches • 010b - the flash cache is enabled and will cache flash data fetches but not flash instruction fetches • 011b - the flash cache is enabled but will not be used (basically disabled) • 1xxb - the flash cache is diabled CF2[0] • 0b - do not clear the flash cache • 1b - clear the flash cache (negate all valid bits in the cache)
7-4 OCMCF1	OCMEM Control Field 1 The flash controller needs to be idle when writing to an OCMDRn register associated with Flash memory. This means no read/erase/execute/etc operations from the Flash memory should be made while writing to the OCMDRn register associated with that memory. Changing controller configuration while active can cause undesired results. CF1 = Control Field 1 - for flash controller CF1[3] LKIFR1 - Lock IFR1 • 0b = IFR1 space can be accessed. • 1b = IFR1 space cannot be accessed. This bit is sticky. Once set, it is cleared with reset. CF1[2] Reserved CF1[1] DFS - Disable Flash Speculate • 0b = Enable Flash Speculate • 1b = Disable Flash Speculate CF1[0] DDP - Disable Data Prefetch • 0b = Enable Data Prefetch • 1b = Disable Data Prefetch

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>CF1[1:0] bits controls whether prefetches (or speculative accesses) are initiated in response to instruction fetches or data references.

On new flash accesses, the flash controller reads 128-bit aligned data and stores it in a normal read buffer while bypassing the desired 32-bit to the input AHB bus. Subsequent accesses to the same 128-bits of flash data can be serviced from this buffer without re-reading the flash arrays and has buffers to hold a flash access. The flash controller also has a 128-bit speculation buffer. During idle cycles or cycles where the flash controller accesses are being serviced from the flash cache or the normal buffer, the flash controller will read the next sequential 128-bits of flash data and store it in the speculation buffer. Then, subsequent sequential accesses can hit this speculation buffer without accessing the flash. The following CF1[1:0] bits configure and maintain this speculation buffer.

<table>
<tr><th>CF1[1]
Flash Speculate Disable</th><th>CF1[0]
Data Prefetch Disable</th><th>Result</th></tr>
<tr><td>Disabled (1b)</td><td>Disabled (1b)</td><td rowspan="2">All speculation disabled and speculation buffer is cleared</td></tr>
<tr><td>Disabled (1b)</td><td>Enabled (0b)</td></tr>
<tr><td>Enabled (0b)</td><td>Disabled (1b)</td><td>Speculation for Instruction enabled and Speculation for Data disabled</td></tr>
<tr><td>Enabled (0b)</td><td>Enabled (0b)</td><td>Speculation for both Instruction and Data enabled</td></tr>
</table>

The reset state for CF1[1:0] is 00b - the speculation buffer is enabled and speculation will occur for both flash instruction accesses and flash data accesses.

Note, whenever the speculation buffer is disabled (CF1[1:0] = 1xb, the speculation buffer is cleared (its valid bit is negated). This provides a way to clear the speculation buffer. The speculation buffer is also cleared when the flash controller is reset.</td></tr>
<tr><td>3-0

OCMCF0</td><td>OCMEM Control Field 0

OCMCF0 is used to Program or Data Flash and System RAM ECC.

The flash controller needs to be idle when writing to an OCMDRn register associated with Flash memory. This means no read/erase/execute/etc operations from the Flash memory should be made while writing to the OCMDRn register associated with that memory. Changing controller configuration while active can cause undesired results.

CF0 = Control Field 0 - for ECC control functions.

CF0[3] DNCBED - Disable non-correctable bus errors on flash data fetches

• 0b = Bus error for non-correctable error on data fetch from flash

• 1b = Disable bus error response for non-correctable error on data fetch from flash

CF0[2] DNCBEI - Disable non-correctable bus errors on flash instruction fetches

• 0b = Bus error for non-correctable error on instruction fetch from flash</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 1b = Disable bus error response for non-correctable error on instruction fetch from flash CF0[1:0] - Reserved

12.3.2.15 On-Chip Memory Descriptor Register (OCMDR2 - OCMDR5)

Offset

Register	Offset
OCMDR2	408h
OCMDR3	40Ch
OCMDR5	414h

Function

This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provides configurable controls (where appropriate).

- Any access with a size other than 32 bits are terminated with an error.

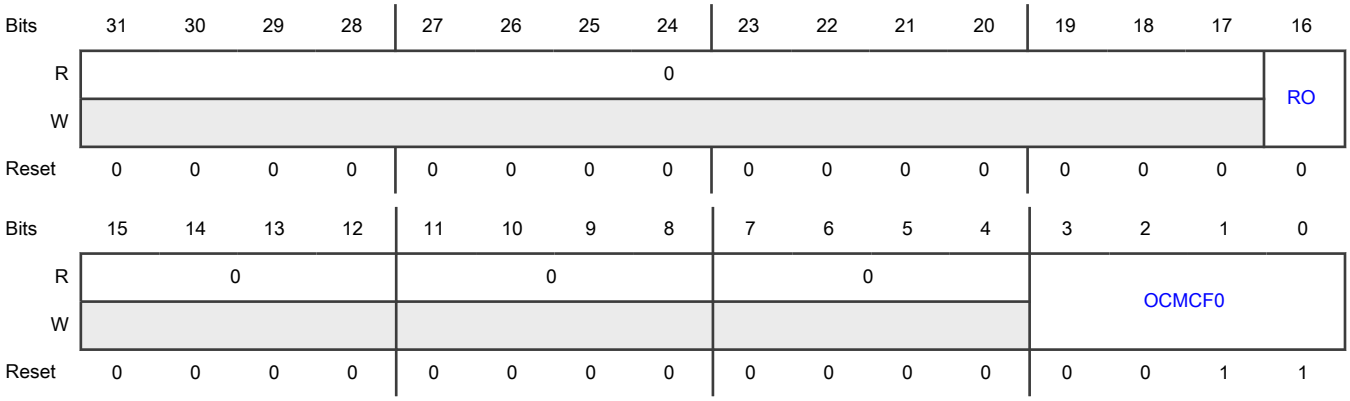
An on-chip memory has a corresponding SMSCM OCMDRn register when one of the following is true:

- The memory type is data flash
- The memory type is program flash
- The memory type is system RAM and ECC = true

The following table describes each on-chip memory type:

On-chip memory n	On-Chip Memory Type
On-chip memory 0	Program Flash
On-chip memory 2	System RAM
On-chip memory 3	System RAM
On-chip memory 5	System RAM

Diagram



Fields

Field	Function
31-17 —	Reserved
16 RO	Read-Only This register bit provides a mechanism to “lock” the configuration state defined by OCMDRn[11:0]. Once asserted, attempted writes to the OCMDRn[11:0] register are ignored until the next reset clears the flag. 0b - Writes to the OCMDRn[11:0] are allowed 1b - Writes to the OCMDRn[11:0] are ignored
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 OCMCF0	OCMEM Control Field 0 **OCMCF0 is used Program or Data Flash and System RAM ECC.** The flash controller needs to be idle when writing to an OCMDRn register associated with Flash memory. This means no read/erase/execute/etc operations from the Flash memory should be made while writing to the OCMDRn register associated with that memory. Changing controller configuration while active can cause undesired results. CF0 = Control Field 0 - for ECC control functions. CF0[3:2] - Reserved CF0[1] EERC - Enable ECC read check

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0b = Disable ECC on reads • 1b = Enable ECC on reads CF0[0] EERW - Enable ECC write generation • 0b = Disable ECC on writes • 1b = Enable ECC on writes

12.3.2.16 On-Chip Memory ECC Control Register (OCMECR)

Offset

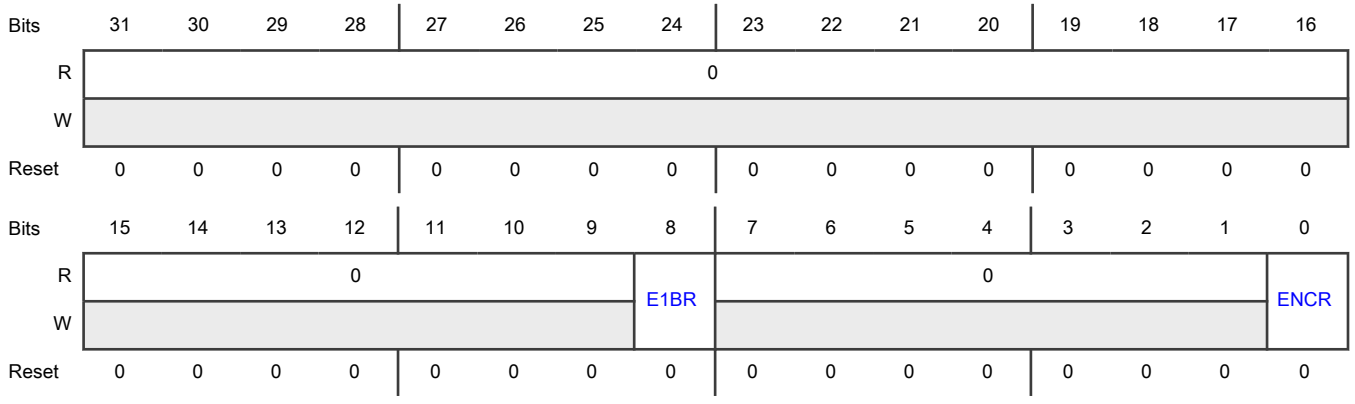
Register	Offset
OCMECR	480h

Function

The On-Chip Memory ECC Control Register is an 32-bit control register for specifying which types of memory errors are reported. In systems with ECC, the occurrence of a non-correctable error causes the current access to be terminated with an error condition. In many cases, this error termination is reported directly by the initiating bus master. However, there are certain situations where the occurrence of this type of non-correctable error is *not* reported by the master. Examples include speculative instruction fetches which are discarded due to a change-of-flow operation, and buffered operand writes. The ECC reporting logic in MSCM provides an optional error interrupt mechanism to signal all non-correctable memory errors. In addition to the interrupt generation, the MSCM captures specific information (memory address, attributes and data, bus master number, etc.) which may be useful for subsequent failure analysis.

Single bit memory corrections are performed on-the-fly, returning the corrected read data to the requesting bus master. For these single bit corrections, another configuration bit allows reporting of these events.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 E1BR	Enable RAM ECC 1 Bit Reporting 0b - 1-bit reporting disabled 1b - 1-bit reporting enabled
7-1 —	Reserved
0 ENCR	Enable RAM ECC Non-correctable Reporting 0b - Non-correctable reporting disabled 1b - Non-correctable reporting enabled

12.3.2.17 On-Chip Memory ECC Interrupt Register (OCMEIR)

Offset

Register	Offset
OCMEIR	488h

Function

The On-Chip Memory ECC Interrupt Register is a 32-bit control register for signaling which types of properly-enabled ECC events have occurred. The OCMEIR includes two bit maps to record the occurrence of individual ECC events, both 1-bit correctable and multi-bit non-correctable events.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALID	0			EELOC				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	E1BERRN								ENCERRN							
W	W1C								W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VALID	Valid ECC Error Location field The VALID bit is set whenever a bit in [15:0] is set. When all of the errors are cleared, the VALID bit is cleared. 0b - ECC Error Location field is not valid 1b - ECC Error Location field is valid
30-28 —	Reserved
27-24 EELOC	ECC Error Location 0000b - non-correctable on OCRAM0 0001b - non-correctable on OCRAM1 0010b - non-correctable on OCRAM2 0011b - non-correctable on OCRAM3 0100b - non-correctable on OCRAM4 0101b - non-correctable on OCRAM5 0110b - non-correctable on OCRAM6 0111b - non-correctable on OCRAM7 1000b - 1-bit correctable on OCRAM0 1001b - 1-bit correctable on OCRAM1 1010b - 1-bit correctable on OCRAM2 1011b - 1-bit correctable on OCRAM3 1100b - 1-bit correctable on OCRAM4 1101b - 1-bit correctable on OCRAM5 1110b - 1-bit correctable on OCRAM6 1111b - 1-bit correctable on OCRAM7
23-16 —	Reserved
15-8 E1BERRN	ECC 1-bit Error OCRAMn This field is the bit map of 1-bit correctable errors. **NOTE** An E1BERRN[n] bit is set when a non-correctable error has occured, the corresponding OCMDRn[1:0] = 11b, and the corresponding OCRAMn has the ECC feature. • E1BERRN[0] - OCRAM0 has a 1-bit correctable error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• E1BERRN[1] - OCRAM1 has a 1-bit correctable error • E1BERRN[2] - OCRAM2 has a 1-bit correctable error • E1BERRN[3] - OCRAM3 has a 1-bit correctable error • E1BERRN[4] - OCRAM4 has a 1-bit correctable error • E1BERRN[5] - OCRAM5 has a 1-bit correctable error • E1BERRN[6] - OCRAM6 has a 1-bit correctable error • E1BERRN[7] - OCRAM7 has a 1-bit correctable error W1C - Write-1-Clear individually clear each 1-bit correctable error.
7-0 ENCERRN	ECC Non-correctable Error OCRAMn This field is the bit map of non-correctable errors. **NOTE** An ENCERRN[n] bit is set when a non-correctable error has occured, the corresponding OCMDRn[1:0] = 11b, and the corresponding OCRAMn has the ECC feature. • ENCERRN[0] - OCRAM0 has a non-correctable error • ENCERRN[1] - OCRAM1 has a non-correctable error • ENCERRN[2] - OCRAM2 has a non-correctable error • ENCERRN[3] - OCRAM3 has a non-correctable error • ENCERRN[4] - OCRAM4 has a non-correctable error • ENCERRN[5] - OCRAM5 has a non-correctable error • ENCERRN[6] - OCRAM6 has a non-correctable error • ENCERRN[7] - OCRAM7 has a non-correctable error W1C - Write-1-Clear individually clear each non-correctable error.

12.3.2.18 On-Chip Memory Fault Address Register (OCMFAR)

Offset

Register	Offset
OCMFAR	490h

Function

The OCMFAR is a 32-bit read-only register for capturing the address of the last, properly-enabled ECC event in the on-chip memory. Depending on the state of the On-Chip Memory ECC Control Register, an ECC event in the on-chip memory causes the address, attributes and read data associated with the access to be loaded into the OCMFAR, OCMFTR and OCMFDR registers, and the appropriate flag in the On-Chip Memory ECC Interrupt Register to be asserted. Attempted writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EFADD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EFADD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EFADD	ECC Fault Address

12.3.2.19 On-Chip Memory Fault Attribute Register (OCMFTR)

Offset

Register	Offset
OCMFTR	494h

Function

The OCMFTR is a 32-bit read-only register for capturing the attributes of the last, properly-enabled ECC event in the on-chip memory. Depending on the state of the On-Chip Memory ECC Control Register, an ECC event in the on-chip memory causes the address, attributes and read data associated with the access to be loaded into the OCMFAR, OCMFTR and OCMFDR registers, and the appropriate flag in the On-Chip Memory ECC Interrupt Register to be asserted. Attempted writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								EFSYN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EFMST								EFW	EFMS			EFPRT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 EFSYN	On-Chip Memory ECC Fault Syndrome This read-only field specifies the checkbit syndrome from the last captured ECC event. For 32-bit memories, the syndrome is 7-bits, and EFSYN[7] == 0b. For 64-bit memories, the syndrome is 8-bits. The following describes the syndrome for a single-bit correctable error for a 32-bit RAM. For example, if there is a single-bit error in ram_rdata[21], then EFSYN[6:0] = 16h. If the EFSYN value is not found in the list below, a multi-bit, non-correctable error has been detected. • (EFSYN[6:0] == 00h) - no errors ram_rdata[31:0] • (EFSYN[6:0] == 61h) - 1-bit error on ram_rdata[0] • (EFSYN[6:0] == 51h) - 1-bit error on ram_rdata[1] • (EFSYN[6:0] == 19h) - 1-bit error on ram_rdata[2] • (EFSYN[6:0] == 45h) - 1-bit error on ram_rdata[3] • (EFSYN[6:0] == 43h) - 1-bit error on ram_rdata[4] • (EFSYN[6:0] == 31h) - 1-bit error on ram_rdata[5] • (EFSYN[6:0] == 29h) - 1-bit error on ram_rdata[6] • (EFSYN[6:0] == 13h) - 1-bit error on ram_rdata[7] • (EFSYN[6:0] == 62h) - 1-bit error on ram_rdata[8] • (EFSYN[6:0] == 52h) - 1-bit error on ram_rdata[9] • (EFSYN[6:0] == 4ah) - 1-bit error on ram_rdata[10] • (EFSYN[6:0] == 46h) - 1-bit error on ram_rdata[11] • (EFSYN[6:0] == 32h) - 1-bit error on ram_rdata[12] • (EFSYN[6:0] == 2ah) - 1-bit error on ram_rdata[13] • (EFSYN[6:0] == 23h) - 1-bit error on ram_rdata[14] • (EFSYN[6:0] == 1ah) - 1-bit error on ram_rdata[15] • (EFSYN[6:0] == 2ch) - 1-bit error on ram_rdata[16] • (EFSYN[6:0] == 64h) - 1-bit error on ram_rdata[17] • (EFSYN[6:0] == 26h) - 1-bit error on ram_rdata[18] • (EFSYN[6:0] == 25h) - 1-bit error on ram_rdata[19] • (EFSYN[6:0] == 34h) - 1-bit error on ram_rdata[20] • (EFSYN[6:0] == 16h) - 1-bit error on ram_rdata[21] • (EFSYN[6:0] == 15h) - 1-bit error on ram_rdata[22] • (EFSYN[6:0] == 54h) - 1-bit error on ram_rdata[23] • (EFSYN[6:0] == 0bh) - 1-bit error on ram_rdata[24]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• (EFSYN[6:0] == 58h) - 1-bit error on ram_rdata[25] • (EFSYN[6:0] == 1ch) - 1-bit error on ram_rdata[26] • (EFSYN[6:0] == 4ch) - 1-bit error on ram_rdata[27] • (EFSYN[6:0] == 38h) - 1-bit error on ram_rdata[28] • (EFSYN[6:0] == 0eh) - 1-bit error on ram_rdata[29] • (EFSYN[6:0] == 0dh) - 1-bit error on ram_rdata[30] • (EFSYN[6:0] == 49h) - 1-bit error on ram_rdata[31]
15-8 EFMST	On-Chip Memory ECC Fault Master Number This read-only field specifies the bus master responsible for initiating the last captured ECC event.
7 EFW	On-Chip Memory ECC Fault Write This read-only field specifies if the last captured ECC event was a write bus cycle. Since the ECC check only occurs on read operations, if the EFW bit is asserted, the captured ECC event was associated with the read-modify-write needed to generate the required check bits. This type of read-modify-write sequence only occurs on writes of less than 64 bits; for 64-bit writes, the destination check bits are calculated directly from the write data without the need for any read-modify-write sequence. 0b - Last captured ECC event was not a write bus cycle 1b - Last captured ECC event was a write bus cycle
6-4 EFMS	On-Chip Memory ECC Fault Master Size This read-only field specifies the access size of the last captured ECC event. The size encodings are: 000b - 8-bit size 001b - 16-bit size 010b - 32-bit size 011b - 64-bit size 100b - Reserved 101b - Reserved 110b - Reserved 111b - Reserved
3-0 EFPRT	On-Chip Memory ECC Fault Protection This read-only field specifies the protection field of the last captured ECC event. The protection encodings are: EFPRT[3] : Cacheable • 0 = non-cacheable • 1 = cacheable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	EFPRT[2] : Bufferable • 0 = non-bufferable • 1 = bufferable EFPRT[1] : Mode 0 • 0 = user mode • 1 = supervisor mode EFPRT[0] : Type 0 • 0 = I-Fetch • 1 = Data

12.3.2.20 On-Chip Memory ECC Fault Data High Register (OCMFDRH)

Offset

Register	Offset
OCMFDRH	498h

Function

The OCMFDR is a 64-bit read-only register for capturing the read data of the last, properly-enabled ECC event in the on-chip memory. OCMFDRH is the upper 32 bits. Depending on the state of the On-Chip Memory ECC Control Register (OCMECR), an ECC event in the on-chip memory causes the address, attributes and read data associated with the access to be loaded into the OCMFAR, OCMFTR and OCMFDR registers, and the appropriate flag in the On-Chip Memory ECC Interrupt Register to be asserted. The read data is captured after it has been processed by the ECC logic, that is, in the case of a correctable single-bit ECC event, the read data is the corrected value. The contents of OCMFTR[EFSYN] can be used to locate the original faulted data bit. Attempted writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EFDH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EFDH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EFDH	On-Chip Memory ECC Fault Data High This read-only field specifies the upper 32-bit read data word (data[63:32]) from the last captured ECC event. For ECC events that occur in 32-bit RAMs, this 32-bit field will return 32'h0.

12.3.2.21 On-Chip Memory ECC Fault Data Low Register (OCMFDRL)

Offset

Register	Offset
OCMFDRL	49Ch

Function

The OCMFDR is a 64-bit read-only register for capturing the read data of the last, properly-enabled ECC event in the on-chip memory. OCMFDRL is the lower 32 bits. Depending on the state of the On-Chip Memory ECC Control Register (OCMECR), an ECC event in the on-chip memory causes the address, attributes and read data associated with the access to be loaded into the OCMFAR, OCMFTR and OCMFDR registers, and the appropriate flag in the On-Chip Memory ECC Interrupt Register to be asserted. The read data is captured after it has been processed by the ECC logic, that is, in the case of a correctable single-bit ECC event, the read data is the corrected value. The contents of OCMFTR[EFSYN] can be used to locate the original faulted data bit. Attempted writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EFDL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EFDL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EFDL	On-Chip Memory ECC Fault Data Low This read-only field specifies the lower 32-bit read data word (data[31:0]) from the last captured ECC event.

12.3.2.22 Core Platform Control Register (CPCR)

Offset

Register	Offset
CPCR	C00h

Function

This register provides the ability to control the round robin arbitration mode in the various bus crossbars in the system.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															AXBS0_R...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-1 —	Reserved
0 AXBS0_RREN	AXBS0 Round Robin Enable When set, this forces the AXBS0 to round robin arbitration mode out of reset. To change the arbitration mode after reset is negated, the AXBS0 programming model must be used. Enabled at reset. 0b - AXBS0 in fixed priority arbitration mode at reset. 1b - AXBS0 in round robin arbitration mode at reset.

Chapter 13
Miscellaneous Control Module (MCM)

13.1 Chip-specific MCM information

Table 51. Reference links to related information

Topic	Related module	Reference
Full description	MCM	MCM
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing
Cache	LPCAC	Low Power Cache Controller (LPCAC)

13.1.1 Module instances

This device has one instance of the MCM module.

13.1.2 LMFATR[PEFSYN]

LMFATR[PEFSYN] field indicates the master that accesses the cache when the LMEM fault is detected.

- 0x0 - Core
- 0x1 - Debug tool
- Others - Reserved

13.1.3 ISCR[CWBEE]

In this device, the ISCR[CWBEE] must be configured as 0.

13.2 Overview

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

13.2.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision
- Error status and interrupts for the cache write buffer
- Error status and interrupts for the core's floating-point unit (FPU)

13.3 Functional description

This section describes the functions of the MCM module.

13.3.1 Interrupts

The MCM's interrupt is generated if any of the following is true:

- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC)
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC)

- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC)
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC)
- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC)
- FPU invalid operation interrupt is enabled (FIOCE) and an invalid occurs (FIOC)
- PC data parity error
- PC tag parity error
- Cache write buffer error

13.3.2 Determining source of the interrupt

To determine the exact source of the interrupt, check the interrupt status flags with the corresponding interrupt enable bits.

1. From MCM_ISCR[31:16] and MCM_ISCR[15:0]
2. Search the result for asserted flags, which indicate the exact interrupt sources

NOTE

Parity interrupts are determined by LMEPECR (interrupt enable) and LMPEIR (interrupt source).

13.4 Memory map/register descriptions

The memory map and register descriptions below describe the registers using byte addresses. The programming model for the Miscellaneous Control Module is accessed via the core's Private Peripheral Bus (PPB).

13.4.1 MCM register descriptions

13.4.1.1 MCM memory map

MCM base address: E008_0000h

Offset	Register	Width (In bits)	Access	Reset value
Ch	Core Platform Control (CPCR)	32	RW	See section
10h	Interrupt Status and Control (ISCR)	32	RW	0000_0000h
20h	Write Buffer Fault Address (FADR)	32	R	See section
24h	Store Buffer Fault Attributes (FATR)	32	R	See section
28h	Store Buffer Fault Data (FDR)	32	R	See section
34h	Core Platform Control 2 (CPCR2)	32	RW	0001_0040h
408h	Local Memory Descriptor 2 (LMDR2)	32	RW	8484_4000h
480h	LMEM Parity Control (LMPECR)	32	RW	0000_0000h
488h	LMEM Parity Interrupt (LMPEIR)	32	RW	0000_0000h
490h	LMEM Fault Address (LMFAR)	32	R	0000_0000h
494h	LMEM Fault Attribute (LMFATR)	32	R	0000_0000h
4A0h	LMEM Fault Data High (LMFDHR)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4A4h	LMEM Fault Data Low (LMFDLR)	32	R	0000_0000h

13.4.1.2 Core Platform Control (CPCR)

Offset

Register	Offset
CPCR	Ch

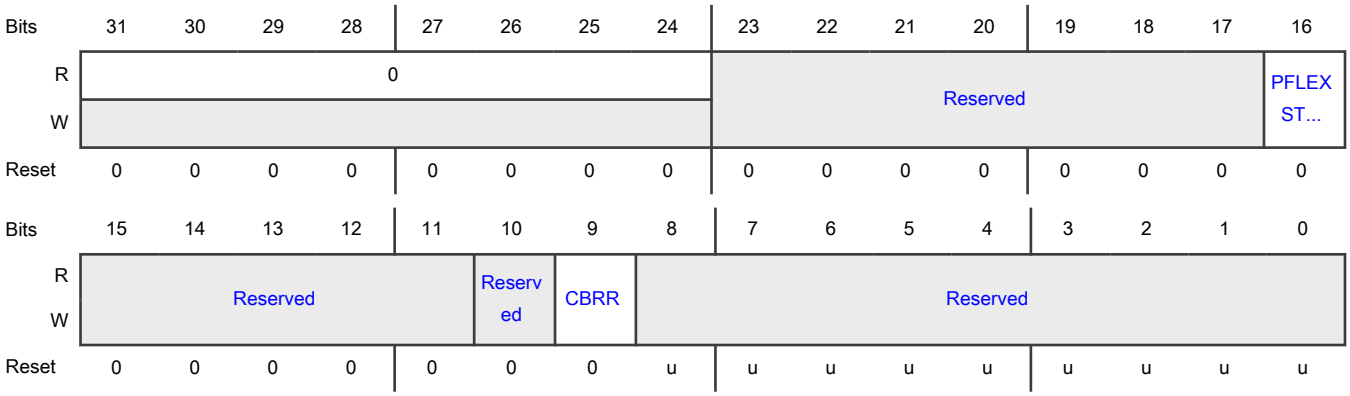
Function

The Core Platform Control Register (CPCR) configures the arbitration.

NOTE

Bits 8-0 are undefined after reset.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 —	Reserved
16 PFLEXSTALL	Flash Stall Enable This field configures whether to enable flash stall when the flash is busy.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flash stall is disabled when flash is busy. 1b - Flash stall is enabled when flash is busy.
15-11 —	Reserved
10 —	Reserved
9 CBRR	Crossbar Round-robin Arbitration Enable This field configures the crossbar slave ports to fixed-priority or round-robin arbitration. 0b - Fixed-priority arbitration 1b - Round-robin arbitration
8-0 —	Reserved

13.4.1.3 Interrupt Status and Control (ISCR)

Offset

Register	Offset
ISCR	10h

Function

The Interrupt Status and Control Register (ISCR) configures and reports status of core-related interrupt exception conditions. The individual event indicators are first checked with their exception enables and then logically summed to form an interrupt request sent to the core's NVIC.

Bits 15-8 are read-only indicator flags based on the processor's FPSCR register. Attempted writes to these bits are ignored. Once set, the flags remain asserted until software clears the corresponding FPSCR bit.

Bits 31-24 controls interrupt enablement. If an interrupt enablement control bit is set as 1, the MCM will generate an interrupt when the corresponding status bit is 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FIDCE	0		FIXCE	FUFCE	FOFCE	FDZCE	FIOCE	0		CPEE	CWBEE	0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIDC	0		FIXC	FUFC	FOFC	FDZC	FIOC	0		CPES	CWBER	0			0
W												W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FIDCE	FPU Input Denormal Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
30-29 —	Reserved
28 FIXCE	FPU Inexact Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
27 FUFCE	FPU Underflow Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
26 FOFCE	FPU Overflow Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
25 FDZCE	FPU Divide-by-zero Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
24 FIOCE	FPU Invalid Operation Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 CPEE	Cache Parity Error Enable This field enables the generation of an interrupt in response to the cache parity error. 0b - Disable error interrupt. 1b - Enable error interrupt.
20 CWBEE	Cache Write Buffer Error Enable This field enables the generation of an interrupt in response to a bus error termination reported on a system bus transfer initiated from the cache's write buffer. 0b - Disable error interrupt 1b - Enable error interrupt
19-16 —	Reserved
15 FIDC	FPU Input Denormal Interrupt Status This read-only field is a copy of the core's FPSCR[IDC] bit and indicates input denormalized number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IDC] bit. 0b - No interrupt 1b - Interrupt occurred
14-13 —	Reserved
12 FIXC	FPU Inexact Interrupt Status This read-only field is a copy of the core's FPSCR[IXC] bit and indicates an inexact number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IXC] bit. 0b - No interrupt 1b - Interrupt occurred
11 FUFC	FPU Underflow Interrupt status This read-only field is a copy of the core's FPSCR[UFC] bit and indicates an underflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[UFC] bit. 0b - No interrupt 1b - Interrupt occurred
10 FOFC	FPU Overflow Interrupt Status This read-only field is a copy of the core's FPSCR[OFC] bit and indicates an overflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[OFC] bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt 1b - Interrupt occurred
9 FDZC	FPU Divide-by-zero Interrupt Status This read-only field is a copy of the core's FPSCR[DZC] bit and indicates a divide-by-zero has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[DZC] bit. 0b - No interrupt 1b - Interrupt occurred
8 FIOC	FPU Invalid Operation Interrupt Status This read-only bit is a copy of the core's FPSCR[IOC] bit and indicates an illegal operation has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IOC] bit. 0b - No interrupt 1b - Interrupt occurred
7-6 —	Reserved
5 CPES	Cache Parity Error Status This field indicates whether a cache parity error is detected. This field can be set only when the corresponding enable bit (CPEE) is set. 0b - A cache parity error is not detected. 1b - A cache parity error is detected.
4 CWBER	Cache Write Buffer Error Status This field indicates whether a data transfer from the core's cache write buffer was terminated because of a bus error. This field can be set only when the corresponding enable bit (CWBEE) is set. The corresponding core fault address, attributes and write data are typically retrieved from the FADR, FATR, and FDR registers during the interrupt service routine before clearing the CWBER flag. 0b - No error 1b - Error occurred
3-1 —	Reserved
0 —	Reserved

13.4.1.4 Write Buffer Fault Address (FADR)

Offset

Register	Offset
FADR	20h

Function

The Write Buffer Fault Address Register (FADR) captures the faulting address when a properly-enabled cache write buffer error interrupt event is detected. The MCM logic captures a single cache write buffer bus error event; if a subsequent error is detected before the captured error information has been read from the corresponding registers and the MCM_ISCR[CWBER] indicator cleared, the MCM_FATR[BEOVR] flag is set. However, no additional information is captured.

The bits in this register are set by hardware when MCM_ISCR[CWBER] is asserted. Attempted writes have no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDRESS															
W	-															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDRESS															
W	-															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 ADDRESS	Fault address

13.4.1.5 Store Buffer Fault Attributes (FATR)

Offset

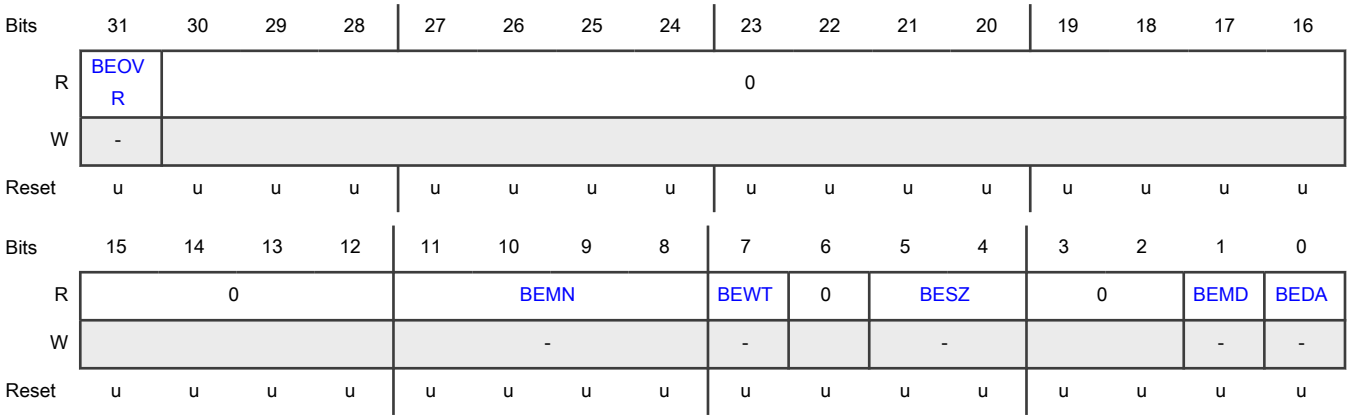
Register	Offset
FATR	24h

Function

The Store Buffer Fault Attributes Register (FATR) indicates the faulting attributes, when a properly-enabled cache write buffer error interrupt event is detected

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes have no effect.

Diagram



Fields

Field	Function
31 BEOVR	Bus Error Overrun This field indicates if another cache write buffer bus error is detected before system software has retrieved all the error information from the original event. The time window is from the detection of the original cache write buffer error termination to the ISCR[CWBER] is written with a 1 to clear it and rearm the capture logic. This bit is set by the hardware and cleared whenever software writes a 1 to the ISCR[CWBER] bit. 0b - No bus error overrun 1b - Bus error overrun occurred. The FADR and FDR registers and the other FATR bits will not be updated to reflect this new bus error.
30-12 —	Reserved
11-8 BEMN	Bus Error Master Number This field indicates the crossbar switch bus master number of the captured cache write buffer bus error. For this device, this value is always 0x1.
7 BEWT	Bus Error Write This field indicates the type of system bus access when the cach write buffer error was detected. Since this logic is monitoring data transfers from the cache write buffer, this bit is always a logical one, indicating a write operation. 0b - Read access 1b - Write access
6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 BESZ	Bus Error Size This field indicates the size of the cache write buffer access when the error was detected. 00b - 8-bit access 01b - 16-bit access 10b - 32-bit access 11b - Reserved
3-2 —	Reserved
1 BEMD	Bus Error Privilege level This field indicates the privilege level of the cache write buffer access when the error was detected. 0b - User mode 1b - Supervisor/privileged mode
0 BEDA	Bus Error Data Access Type This field indicates the type of cache write buffer access when the error was detected. This attribute is always logical one because only data access is performed on cache write buffer. 0b - Instruction 1b - Data

13.4.1.6 Store Buffer Fault Data (FDR)

Offset

Register	Offset
FDR	28h

Function

The Store Buffer Fault Data Register (FDR) indicates the captured faulting data, when a properly-enabled cache write buffer error interrupt event is detected.

The field in this register is set by hardware when the MCM_ISCR[CWBER] is asserted. For byte and halfword writes, only the accessed byte lanes contain valid data; the contents of the other bytes are undefined. Attempted writes have no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W	-															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W	-															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 DATA	Fault Data

13.4.1.7 Core Platform Control 2 (CPCR2)

Offset

Register	Offset
CPCR2	34h

Function

The Control Platform Control 2 Register (CPCR2) configures the cache related properties.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										0		0		LCCPWB	1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				CBCS				DCBC	FCCNA	DCCWB	0
W																CCBC
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17 LCCPWB	Limit Code Cache Peripheral Write Buffering 0b - Code cache peripheral write buffering is not limited: if write buffer is enabled, bufferable write is buffered. 1b - Code cache peripheral write buffering is limited: only bufferable and cachable write is buffered.
16 PCCMCTRL	Bypass Fixed Code Cache Map 0b - The fixed code cache map is not bypassed 1b - The fixed code cache map is bypassed
15-12 —	Reserved
11-8 —	Reserved
7-4 CBCS	Code Bus Cache Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB
3 DCBC	Disable Code Bus cache 0b - Enable code bus cache 1b - Disable code bus cache
2 FCCNA	Force Code Cache to No Allocation 0b - Force code cache to allocation 1b - Force code cache to no allocation

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 DCCWB	Disable Code Cache Write Buffer 0b - Enable code cache write buffer 1b - Disable code cache write buffer
0 CCBC	Clear Code Bus Cache 0b - No effect 1b - Clear code bus cache

13.4.1.8 Local Memory Descriptor 2 (LMDR2)

Offset

Register	Offset
LMDR2	408h

Function

The Local Memory Descriptor Register (LMDR] is an array of 32-bit generic on-chip memory descriptor registers that indicate static information on the attached memories as well as configurable controls (where appropriate).

NOTE

See chip specific section for the LMDRn registers mapping to the LMEMs

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	V	Reserved	Reserved	LMSZH	LMSZ				WY				DPW			RO
W																
Reset	1	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MT			Reserved	Reserved				PCPFE	Reserved	PCPME	Reserved	Reserved	Reserved	Reserved	
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
V	This field indicates the validity (presence) of the local memory. 0b - LMEMn is not present. 1b - LMEMn is present.
30 —	Reserved
29 —	Reserved
28 LMSZH	LMEM Size Hole This field is used for local memories that are not fully populated, that is, include a memory "hole" in the upper 25% of the address range. 0b - LMEMn is a power-of-2 capacity. 1b - LMEMn is a capacity of 0.75 × LMSZ.
27-24 LMSZ	LMEM Size This field indicates an encoded value of the local memory size. The capacity of the memory is expressed as Size [bytes] = $2^{(9+SZ)}$ where SZ is non-zero; a SZ = 0 indicates the memory is not present. 0000b - no LMEMn (0 KB) 0001b - 1 KB LMEMn 0010b - 2 KB LMEMn 0011b - 4 KB LMEMn 0100b - 8 KB LMEMn 0101b - 16 KB LMEMn 0110b - 32 KB LMEMn 0111b - 64 KB LMEMn 1000b - 128 KB LMEMn 1001b - 256 KB LMEMn 1010b - 512 KB LMEMn 1011b - 1024 KB LMEMn 1100b - 2048 KB LMEMn 1101b - 4096 KB LMEMn 1110b - 8192 KB LMEMn 1111b - 16384 KB LMEMn
23-20	Level 1 Cache Ways

Table continues on the next page...

Table continued from the previous page...

Field	Function
WY	0000b - No Cache 0010b - 2-Way Set Associative 0100b - 4-Way Set Associative 1000b - 8-Way Set Associative
19-17 DPW	LMEM Data Path Width This field indicates the width of the local memory. 000b-001b - Reserved 010b - LMEMn 32-bit wide 011b - LMEMn 64-bit wide 100b-111b - Reserved
16 RO	Read-Only This field provides a mechanism to "lock" the configuration state defined by LMDRn[7:0]. Once asserted, attempted writes to the corresponding LMDRn[7:0] are ignored until the next reset clears the flag. 0b - Writes to the corresponding LMDRn[7:0] are allowed. 1b - Writes to the corresponding LMDRn[7:0] are ignored.
15-13 MT	Memory Type This field indicates the type of the local memory. 000b - SRAM_L 001b - SRAM_U 010b - PC Cache 011b - PS Cache
12 —	Reserved
11-8 —	Reserved
7 PCPFE	PC Parity Fault Report Enable When the RO = 0b, this field is read only. 0b - PC parity fault report is disabled. 1b - PC parity fault report is enabled.
6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 PCPME	PC Parity Enable When the RO = 0b, this field is read only. 0b - PC parity is disabled. 1b - PC parity is enabled.
4 —	Reserved
3 —	Reserved
2 —	Reserved
1-0 —	Reserved

13.4.1.9 LMEM Parity Control (LMPECR)

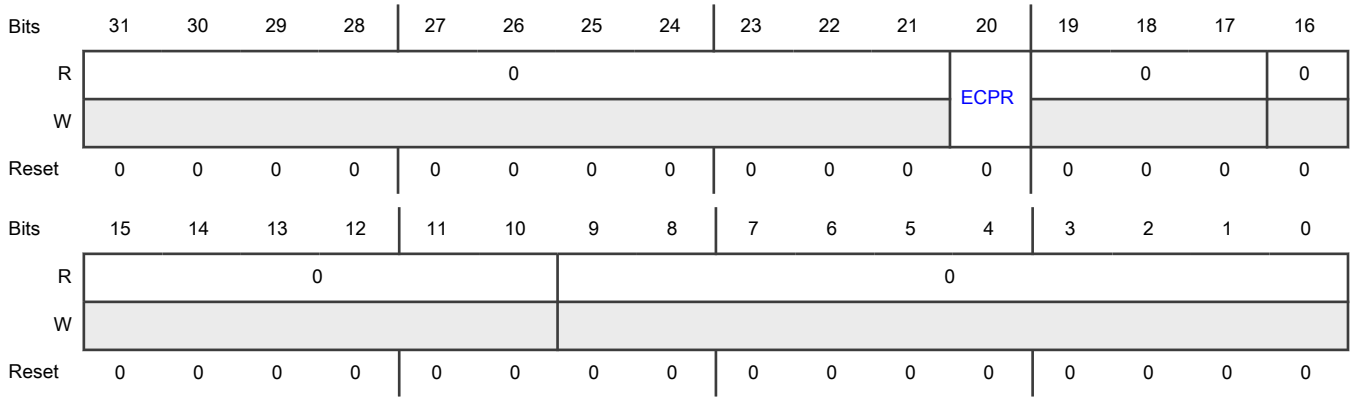
Offset

Register	Offset
LMPECR	480h

Function

The LMEM Parity Control Register (LMPECR) configures the LMEM parity reporting.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 ECPR	Enable Cache Parity Reporting LMPEIR[PE] is functional only when this bit field is set. 0b - Cache parity reporting is disabled 1b - Cache parity reporting is enabled
19-17 —	Reserved
16 —	Reserved
15-10 —	Reserved
9-0 —	Reserved

13.4.1.10 LMEM Parity Interrupt (LMPEIR)

Offset

Register	Offset
LMPEIR	488h

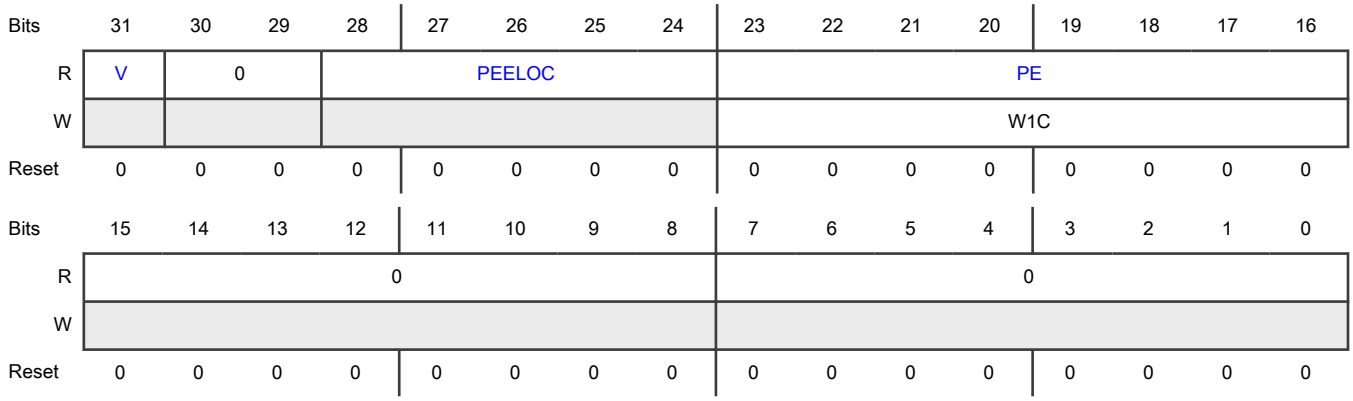
Function

The LMEM Parity Interrupt Register (LMPEIR) indicates the Parity error information.

NOTE

Write 1 to the error bit in this register can clear the related interrupt flag.

Diagram



Fields

Field	Function
31 V	Valid bit This field indicates whether the other fields in this register are valid.
30-29 —	Reserved
28-24 PEELOC	Error Location This field indicates the parity error location See the following for errors, all the others are reserved. 10100b - a PC Tag Parity Error 10101b - a PC Data Parity Error
23-16 PE	Parity Error This field indicates the parity error type. This field is functional only when the LMPECR[ECPR] is set. • LMPEIR[21] - PC Data Parity Error • LMPEIR[20] - PC Tag Parity Error • LMPEIR[19-16] - Reserved
15-8 —	Reserved
7-0 —	Reserved

13.4.1.11 LMEM Fault Address (LMFAR)

Offset

Register	Offset
LMFAR	490h

Function
The LMEM Fault Address Register (LMFAR) captures the faulting address where the LMEM fault is detected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EFADD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EFADD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EFADD	Fault Address

13.4.1.12 LMEM Fault Attribute (LMFATR)

Offset

Register	Offset
LMFATR	494h

Function
The LMEM Fault Attribute Register (LMFATR) indicates the attributes of the LMEM fault detected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OVR	0	0		0				PEFSYN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BKD	0							PEFW	PEFSIZE			PEFPRT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 OVR	Overrun 0b - There is sigle fault or no fault. 1b - There are multiple faults
30 —	Reserved
29-28 —	Reserved
27-24 —	Reserved
23-16 PEFSYN	Parity Fault Syndrome See chip-specific section for details.
15 BKD	Backdoor Access This field indicates the LMEM access fault is initiated by core access or backdoor access. 0b - Core access 1b - Backdoor access
14-8 —	Reserved
7 PEFW	Parity Fault Write This field indicates the parity fault is caused by read or write. 0b - Read fault 1b - Write fault

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 PEFSIZE	PEFSIZE Parity Fault Master Size 000b - 8-bit access 001b - 16-bit access 010b - 32-bit access 011b - 64-bit access others - Reserved
3-0 PEFPRT	Parity Fault Protection Signal Features of parity fault protection signal FATR[3] is Cacheable: 0=Non-cacheable, 1=Cacheable FATR[2] is Bufferable: 0=Non-bufferable, 1=Bufferable FATR[1] is Mode: 0=User mode, 1=Supervisor mode FATR[0] is Type: 0=I-Fetch, 1=Data

13.4.1.13 LMEM Fault Data High (LMFDHR)

Offset

Register	Offset
LMFDHR	4A0h

Function

The LMEM Fault Data High Register (LMFDHR), together with LMFDLR, indicates the read data which has failed in parity check.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PEFDH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PEFDH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PEFDH	PEFDH Parity Fault Data High

13.4.1.14 LMEM Fault Data Low (LMFDLR)

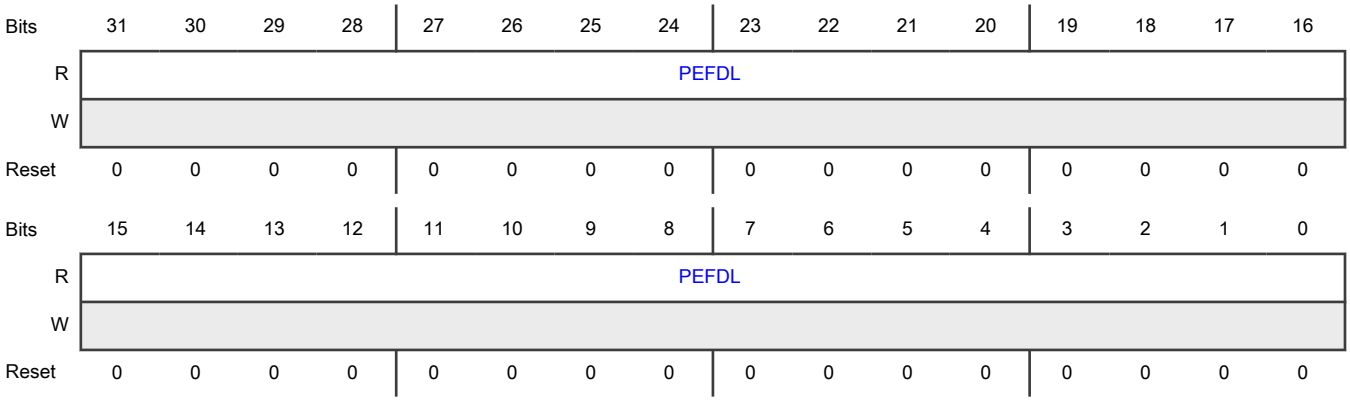
Offset

Register	Offset
LMFDLR	4A4h

Function

The LMEM Fault Data Low Register (LMFDLR), together with LMFDHR, indicates the read data which has failed in parity check.

Diagram



Fields

Field	Function
31-0 PEFDL	PEFDL Parity Fault Data Low

Chapter 14
Memories

14.1 Memory architecture

The memory architecture supports a wide range of applications and maximizes performance while maintaining low power consumption.

14.2 Memory hierarchy

The memory hierarchy is an arrangement of different types of memories with different capacities and operating speeds, to approximate the ideal memory behavior in a cost-efficient way. The following figure shows the memory hierarchical architecture implemented on this device.

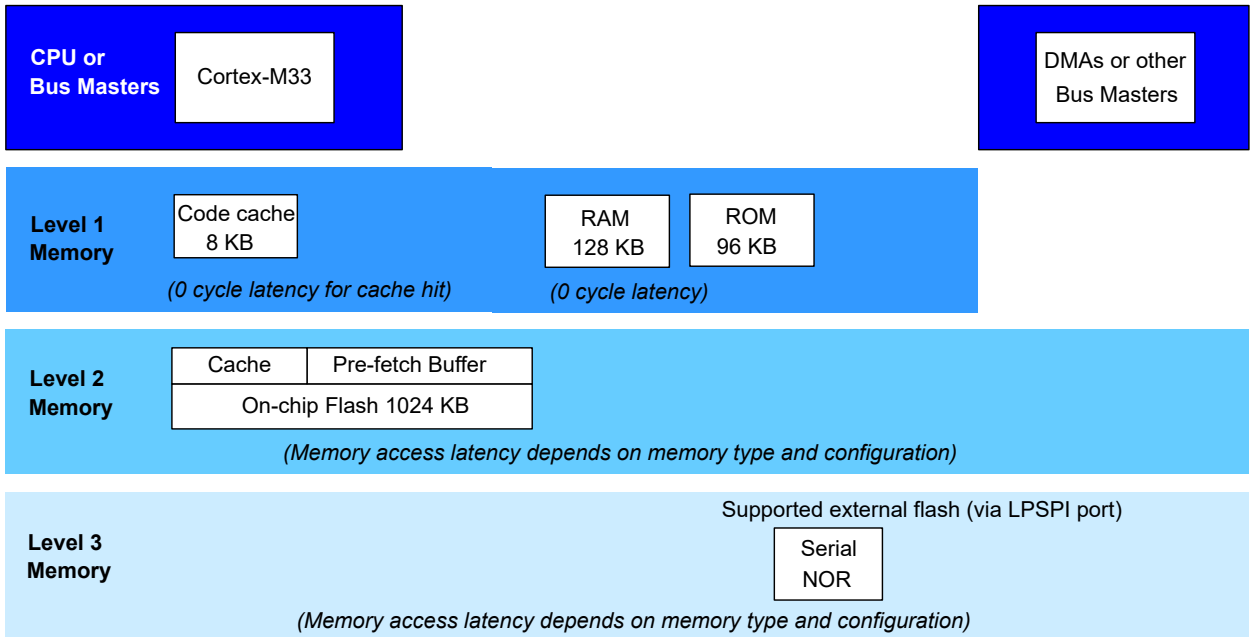


Figure 20. Memory hierarchy

14.3 TCM SRAMs

The core's local Tightly Coupled Memory (TCM) consists of two logical arrays, Code TCM (CTCM) and System TCM (STCM).

14.3.1 TCM SRAM configurations

- The CTCM is 16 KB
  - CTCM array 0: 8 KB
  - CTCM array 1: 8 KB
- The STCM is 112 KB
  - STCM array 0: 16 KB
  - STCM array 1: 16 KB

- STCM array 2: 32 KB
- STCM array 3: 32 KB
- STCM array 4: 8 KB
- STCM array 5: 8 KB

The address ranges of the TCMs are provided in the System memory map section

14.3.2 TCM SRAM retention

The TCM SRAMs can be retained in Sleep, Deep Sleep, and Power Down modes. Each instantiated block of TCM SRAM can be individually power gated. This allows the user to achieve the proper balance of power savings versus data retention.

Entering Deep Power Down mode will automatically power gate all of the SRAM instantiations, and all data is lost.

The last 8 KB RAM (0x2001_A000) is from independent power switch domain which can keep data retention in all power modes.

The first 16 KB of STCM (STCM0) is reserved for ROM use, and it will be initialized when waking up from Power Down mode

14.4 L1 Cache

A cache is a block of high-speed memory locations containing address information, which is commonly known as a tag, and the associated data. The purpose is to decrease the average time of a memory access to lower-level memory. The primary cache, or Level 1 (L1) cache, resides close to the CPU and is used for temporary storage of instructions and data organized in blocks of cache-line size.

This device implements one instance of L1 - the Code cache. The Code cache is accessible through the Code bus of Cortex-M33 core.

14.4.1 L1 cache principles

L1 caches operate on two principles of locality:

- Spatial locality - An access to one location is likely to be followed by accesses to adjacent locations, such as when executing sequential instructions or accessing a data structure. The spatial locality property is used to group several locations together using the same tag. This logical block is commonly known as a cache line.
- Temporal locality - An access to an area of memory is likely to be repeated within a short time period, such as when executing a code loop.

14.4.2 L1 cache features

Each L1 cache instance has following features:

- 8 KB cache array
- 32-bit wide SRAM for data storage
- Register based tags
- 8-way set associative
- 4-word (or 16-byte) lines
- Performs 4-beat, 32-bit wrapped burst to fetch 16 bytes of the cache line when a read misses
- Entire cache can be flushed at once

14.4.3 Cache policies

The following cache interaction policies with lower-level memories are applied to the Code cache:

- Write Through (WT) - accesses to address spaces with this cache mode are cacheable.

- A read miss in the cache causes a line read of a 16-byte aligned memory address containing the desired address in the lower-level memory. This miss data is loaded into the cache and is marked as valid and not modified.
- A read hit to a valid cache location returns data from the cache with no bus access to lower-level memory.
- A write miss bypasses the cache and writes to the lower-level memory; that means no allocate on write miss policy for write-through mode spaces.
- A write hit updates the cache hit line with the write data and writes to the desired address in the lower-level memory.

- Non-Cacheable (NC) - accesses to address spaces with this cache mode are not cacheable. These accesses bypass the cache and access the lower-level memory directly.

14.4.4 Cache control

In normal operation, caches are self-managing, with the updates occurring automatically on demand. Whenever the processor wants to access a cacheable location, the cache is checked. If the access is a cache hit, the access occurs immediately. Otherwise, a location is allocated and the cache line is loaded from lower-level memory.

The Code Cache doesn't have a programming model; all control comes from the Miscellaneous Control Module (MCM). The MCM has cache enable, cache clear, and cache force-no-allocate mode control per cache instance.

14.4.5 Low Power

The caches can be put in state retention or powered down in any of low power modes. The caches need to be flushed before being powered down. This affects the time it takes for the system to transition into the new power mode.

14.5 Read Only Memory (ROM)

The internal ROM memory is used to store the boot code and time-critical software library routines. After a reset, the Cortex-M33 processor starts its code execution from this memory.

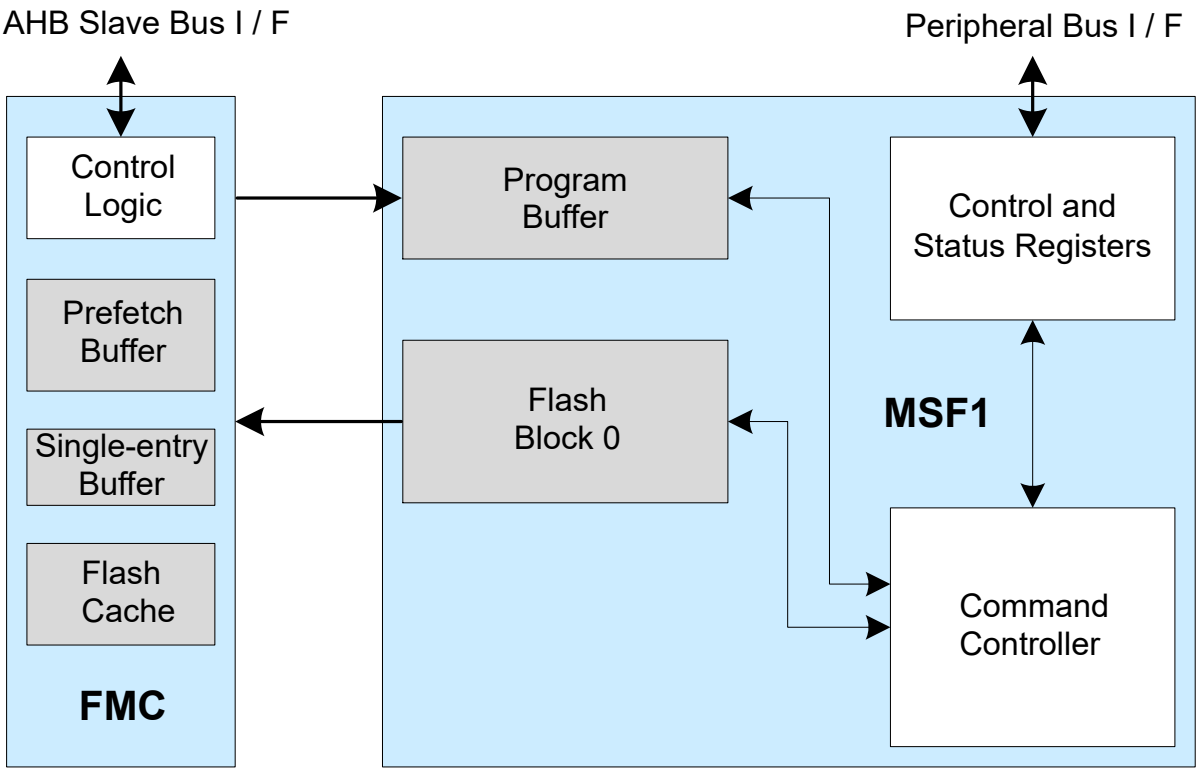
14.5.1 ROM size

The Boot ROM of this device is 96 KB.

14.6 Internal flash

This device embeds 1024 KB of flash.

The following figure shows the high-level block diagram of the on-chip flash.



14.6.1 Flash memory controller

The Flash Memory Controller (FMC) manages accesses performed by the bus masters of the system to the flash memory. The FMC accelerates flash memory transfers to allow program code execution at higher clock frequency than flash memory.

The FMC provides three separate mechanisms for accelerating read operations to the flash memory:

- A single-entry 128-bit buffer, which can store previously accessed flash memory
- A 128-bit prefetch buffer, which can prefetch the next 128-bit flash memory location
- A 64-Byte cache organized as a one set, four way associative cache with 128-bit (or 16-Byte) size entries.

Note the FMC has no visibility into flash memory erase and program cycles because the Flash Memory module manages them directly.

14.6.1.1 Single entry buffer

The width of the bus between the FMC and the flash blocks is 128-bit. A single read of the flash will return an entire bus width worth of data. Because the flash bus is wider than the master's bus, the Single-Entry Buffer is used to store data from the last flash read for quick access times on subsequent reads if the buffer is hit.

The FMC provides invalidation control for the Single-Entry Buffer.

14.6.1.2 Prefetch buffer

When speculative reads are enabled, the FMC immediately requests the next sequential address after a read completes. The next 128-bit memory location is read and placed in the Prefetch Buffer. The speculative prefetch mechanism improves performance by reducing or even eliminating wait states when accessing sequential code and/or data.

Speculative prefetching is programmable for each bank of memory. The FMC provides invalidation control for the Prefetch Buffer.

14.6.1.3 Flash cache

Cache memory stores already executed code. This code is immediately available for repeated execution without any wait states, if needed.

The FMC provides controls for flash replacement algorithm, lock per way, and invalidation per way. The ways are numbered 0-3 and the sets are numbered 0-3. The cache supports Least Recently Used (LRU) replacement algorithm per set across all 4 ways.

The cache entries, both data and tag/valid, can be read at any time.

Software is required to maintain memory coherence when any segment of the flash cache is programmed. For example, all buffer data associated with the reprogrammed flash should be invalidated. Accordingly, cache program visible writes must occur after a programming or erase event is completed and before the new memory image is accessed.

Chapter 15
ROM Bootloader

15.1 Introduction

ROM bootloader is the boot code resident in the internal read-only memory (ROM). The ROM bootloader begins its execution when the Cortex-M33 processor is released from reset. This chapter highlights the features supported by the bootloader and describes its execution flow logic.

15.2 Boot ROM

In this device, the ROM bootloader supports automated booting from internal flash and downloading image from serial interface (in-system programming via LPUART, LPI2C, LPSPI, CAN).

The ROM bootloader is in the memory region starting from the address 0x1480 0000. The following diagram shows the memory map in-use during ROM execution. The first 16 KB of Tightly Coupled Memory - System (STCM) (0x30000000 – 0x30004000) needs to be reserved for ROM bootloader execution and some ROM API execution. Please note that STCM has ECC enabled. To avoid STCM ECC error, users must always program word-aligned data to word-aligned address before the first time reading after power-on reset.

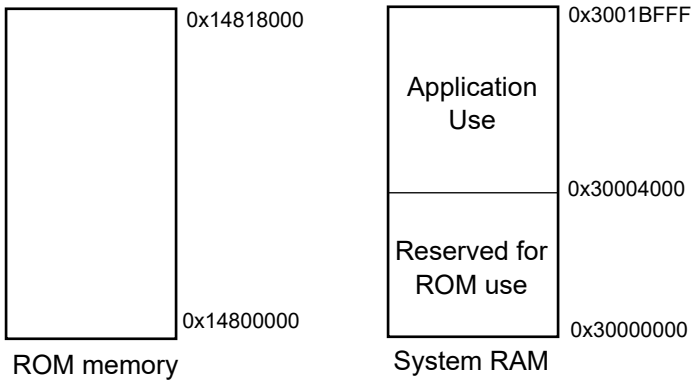


Figure 21. ROM bootloader memory usage

There are several alternate paths through the ROM bootloader:

- Normal boot path with security and TrustZone Mode (TZ-M) option.
- In-system programming (ISP) path.
- Debugger mailbox path which supports debug authentication.
- Low power wakeup path.

The decision about which path to take depends on fuses, boot configuration in user IFR, boot pin, debug request and low power wakeup configuration. Figure below shows the top-level boot flow.

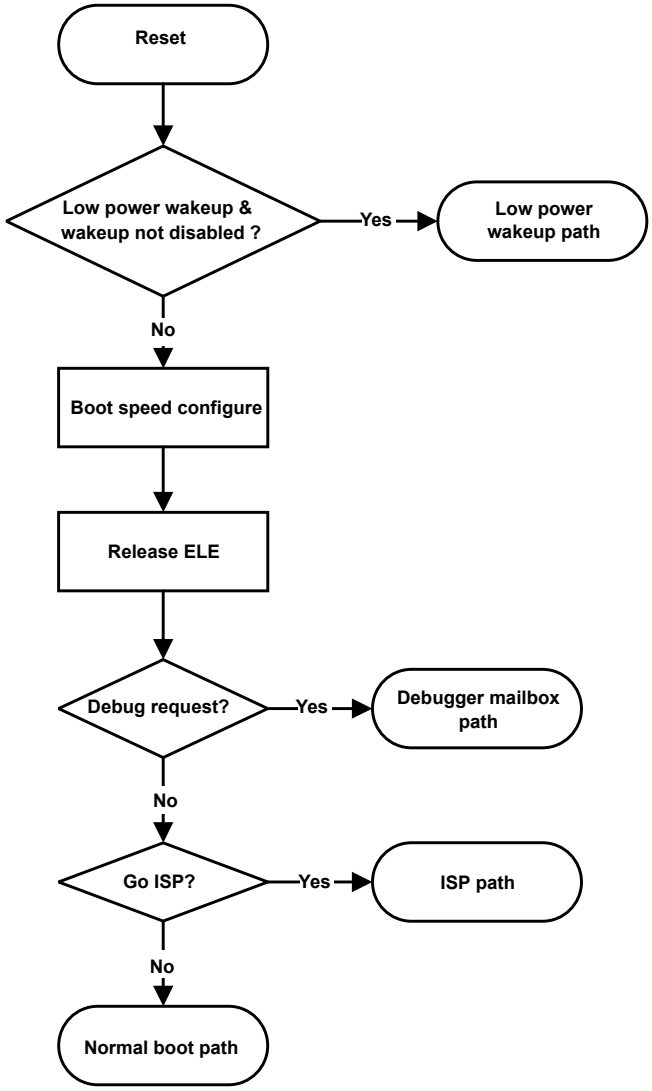


Figure 22. Top-level boot flow

15.2.1 Lifecycle and fuses

The table and diagram below describe the lifecycle states and transitions. ROM behaviour is different at different lifecycle state. The lifecycle fuse needs to be programmed correctly, otherwise ROM bootloader will not boot. The NA in the table means the feature is not available.

Table 52. Lifecycle states

LIFECYCLE	Lifecycle state [0:7]	CM33 image authentication required to boot?	Radio core (CM3) locked if radio authentication fails	Debug authentication required to debug CM33?	Debug authentication required to debug radio core (CM3)?	Able to access radio flash from CM33 after exiting ROM?
0000_0000	Blank	NA	NA	NA	NA	NA

Table continues on the next page...

Table 52. Lifecycle states (continued)

LIFECYCLE	Lifecycle state [0:7]	CM33 image authentication required to boot?	Radio core (CM3) locked if radio authentication fails	Debug authentication required to debug CM33?	Debug authentication required to debug radio core (CM3)?	Able to access radio flash from CM33 after exiting ROM?
0000_0111	OEM Open	No (Boot even if authentication fails)	Yes	No	Yes	No
0000_1111	OEM Secure World Closed	Yes	Yes	Yes[1]	Yes	No
0001_1111	OEM Closed	Yes	Yes	Yes	Yes	No
1001_1111	OEM Locked	Yes	Yes	NA	NA	No
0011_1111	OEM Return	NA	NA	No	Yes	No

1. For secure world only.

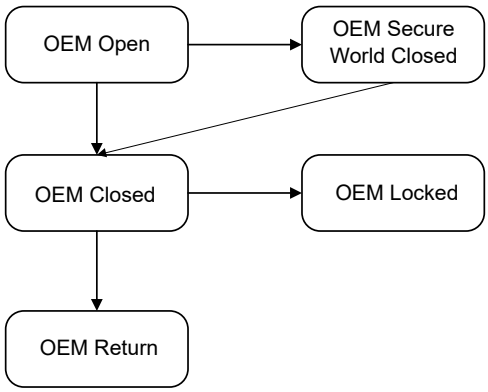


Figure 23. Lifecycle transitions

In addition to the lifecycle fuses, there are a handful of other fuse fields that will be used by the ROM bootloader during its operation. These fuse fields are shown in the table below.

Table 53. Additional fuse fields of interest

Name	Fuse index	Description
LIFECYCLE	0x0A	Lifecycle state. Corresponding product state can be determined from Table 52.
DBG_EN_LOCK	0x0B	Debug Enable Lock 0b - The debug access control registers are not write-locked when jumping to customer code. 1b - The debug access control registers are write-locked before jumping to customer code.
DBG_AUTH_DIS	0x0C	Debug Authentication Disabled

Table continues on the next page...

Table 53. Additional fuse fields of interest (continued)

Name	Fuse index	Description
		0b - Debug Authentication enabled. 1b - Debug Authentication disabled.
TZM_EN	0x0D	Trust Zone Mode Enable 0b - TZ-M is disabled by default. 1b - TZ-M is enabled.
SERIAL_DIS	0x11	Serial Download Disabled 0b - ISP path is enabled. 1b - ISP path is disabled.
WAKEUP_DIS	0x12	Wakeup Disabled 0b - Boot-ROM LP wakeup is enabled. 1b - Boot-ROM LP wakeup is disabled.
CUST_PROD_OEMFW_AUTH_PUK_REVOKE[3:0]	0x13	Key revocation indicators of CUST_PROD_OEMFW_AUTH_PUK.
DBG_AUTH_VU[15:0]	0x15	These Debug Authentication Vendor Usage fuses are used by customers to restrict debug certificates.
IMG_KEY_REVOKE[15:0]	0x16	These fuses are used to revoke image signing keys separately from root keys.
CUST_PROD_OEMFW_AUTH_PUK	0x1f	256-bit RoTKTH (customer trusted root key) used for CM33 main flash image authentication.
CUST_PROD_OEMFW_ENC_SK	0x20	256-bit encryption key used to protect confidentiality of OEM firmware, required for firmware updates using sb3 (SB3KDK).
DCFG_CC_SOCU_L1[16:0]	0x22	Debug authentication configuration for OEM1 customer.
DCFG_CC_SOCU_L2[16:0]	0x23	Debug authentication configuration for OEM2 customer.
SOC_VER_CNT	0x24	512-bits of version counter for various software components in SoC. Each software component version counter has dedicated fuse index 0x25 to 0x29.
CM33_S_VER_CNT	0x25	64-bit monotonic counter indicating the version of the secure OEM CM33 firmware. The number of set bits equals the version number.
CM33_NS_VER_CNT	0x26	256-bit monotonic counter indicating the version of the non-secure OEM CM33 firmware. The number of set bits equals the version number.
RADIO_VER_CNT	0x27	128-bit monotonic counter indicating the version of the radio firmware. The number of set bits equals the version number.
SNT_VER_CNT	0x28	32-bit monotonic counter indicating the version of the ELE loadable firmware. The number of set bits equals the version number.
CM33_BOOTLOADER_VER_CNT	0x29	Reserved for future usage of additional NXP software deliverable.

Table continues on the next page...

Table 53. Additional fuse fields of interest (continued)

Name	Fuse index	Description
GD_EN	0x30	• 0b - Core VDD glitch detection does not generate a hardware reset. • 1b - Boot ROM configures/enables Core VDD Glitch detection based reset.
CM33_S_VER_CNT_VIRTUAL	0x2A	64-bit virtual counter indicating the version of the secure OEM CM33 firmware. The integer value of this field indicates the number of set bits in CM33_S_VER_CNT and that number equals the version number.
CM33_NS_VER_CNT_VIRTUAL	0x2B	256-bit virtual counter indicating the version of the non-secure OEM CM33 firmware. The integer value of this field indicates the number of set bits in CM33_NS_VER_CNT and that number equals the version number.
RADIO_VER_CNT_VIRTUAL	0x2C	128-bit virtual counter indicating the version of the radio firmware. The integer value of this field indicates the number of set bits in RADIO_VER_CNT and that number equals the version number.
SNT_VER_CNT_VIRTUAL	0x2D	32-bit virtual counter indicating the version of the ELE loadable firmware. The integer value of this field indicates the number of set bits in SNT_VER_CNT and that number equals the version number.
CM33_BOOTLOADER_VER_CNT_VIRTUAL	0x2E	Reserved for future usage of additional NXP software deliverable.
HVD_EN	0x2F	• 0b - Core VDD high-voltage event does not generate hardware reset. • 1b - Boot ROM configures/enables Core VDD High voltage detection based reset.

ROM bootloader provides following ways to do the fuse reading and programming:

- nboot API (See #unique_433 for more details about fuse programming using nboot API),
- ISP fuse-program and fuse-read command,
- sbloader “programFuses” command in sb3.1 file

15.2.2 User IFR (IFR0) configuration

User IFR, known as IFR0 is reserved for software usage. First sector of User IFR is not able to be erased and is write-once. Other three sectors are erasable and programmable.

The allocation of User IFR pages is as follows:

Table 54. User IFR allocation

Sector	Start address	End address	Name	Description
0	0x02000000	0x2001FFF	ROMCFG	ROM Bootloader configurations
1	0x02002000	0x2003FFF	User	Reserved for customer usage

Table continues on the next page...

Table 54. User IFR allocation (continued)

Sector	Start address	End address	Name	Description
2	0x02004000	0x2005FFF	CMAC table	Used to save hashes of multiple boot components.
3	0x02006000	0x2007FFF	OTACFG	Used for Over-the-Air update

15.2.2.1 ROM bootloader configuration

Sector 0 of User IFR is ROM and ISP configuration, which provide configuration for ROM bootloader.

Table 55. ROM configuration fields

Offset	Size (bytes)	Configuration Field	Description
0x0000	1	Restore FLW Flag	Bit 0 – FLW state, whether enable dual image boot • 1 = Uninitialized • 0 = Normal, use specified mapping Bit 7:1 – Reserved
	7	Reserved	Reserved
	8	FLW region definition	Fields to be used for FLW configuration
0x0010	1	Configure NPX for Normal Boot	Bit 0 – Configure NPX to use PRINCE • 1 = Not configure NPX • 0 = Configure NPX Bit 1: • 0=Global lock enable • 1=Global lock disable Bit 2: • 0=System lock enable • 1=System lock disable Bit 3: • 0=Global decryption enable • 1=Global decryption disable Bit 4: • 0=Global encryption enable • 1=Global encryption disable Bit 7:5 – Reserved
	1	Sticky NPX Configuration for Waking up from Deep Power Down	Bit 0 – Configure NPX required: • 1 = Not required; ROM follows Low-power wakeup path • 0 = NPX config. required; ROM follows normal boot path Bit 7:1 – Reserved

Table continues on the next page...

Table 55. ROM configuration fields (continued)

Offset	Size (bytes)	Configuration Field	Description
	6	Reserved	Reserved
	56	NPX Region Definitions	Setting for NVM PRINCE XEX, the inline flash encryption IP module
0x0050	1	Boot Configuration	Bit 0 - Enable ISP or not • 0 = BOOT_CFG pin disabled • 1 = BOOT_CFG pin enabled Bit 2:1 – Boot speed • 00 = Normal boot (32 MHz) • 10 = Fast Boot (96 MHz) • 01, 11 = Normal boot (default) bit 7:3 – Reserved
	15	Reserved	Reserved
0x0070	16	Reserved	Reserved
0x0080	64	Boot Image Base Address	Start address of Boot image, should align with 32 KB. ROM Bootloader does search from 0x80 and the first not 0xFFFFFFFF 32-bit will be the boot address. So, it's better to program from 0xB0 to 0x80, users will have 4 chances to update boot address.
0x00C0	64	Reserved	Reserved
0x0100	1	Peripherals Enable	Bit 0 – LPUART1 peripheral for ISP • 0 = LPUART1 disabled • 1 = LUPART1 enabled Bit 1 – LPI2C1 peripheral for ISP • 0 = LPI2C1 disabled • 1 = LPI2C1 enabled Bit 2 – LPSPI1 peripheral for ISP • 0 = LPSPI1 disabled • 1 = LPSPI1 enabled Bit 3 – CAN peripheral for ISP • 0 = CAN disabled • 1 = CAN enabled Bit 7:4 – Reserved
	3	Reserved	Reserved
	2	Peripheral Detection Timeout	If 0xFFFF, defaults to no timeout.

Table continues on the next page...

Table 55. ROM configuration fields (continued)

Offset	Size (bytes)	Configuration Field	Description
			If not 0xFFFF, use this value as timeout in milliseconds for active peripheral detection.
	2	Reserved	Reserved
	1	I2C Slave Address	If 0xFF, defaults to 0x10 for LPI2C slave address. If not 0xFF, use this value as the 7-bit LPI2C slave address.
	1	Reserved	Reserved
	2	Reserved	Reserved
	2	CANTxID	TxID
	2	CANRxID	RxID
0x110	4	ELE Loadable FW Entry Address	ELE loadable firmware resides address
	12	Reserved	Reserved
0x120	4	Secure Hash based image verification feature (for faster subsequent boot)	0xFFFFFFFF – Disabled (ECDSA based verification will be used). If 0x48534148 (“HASH”) – Enabled.
	12	Reserved	Reserved

Restore FLW Flag and FLW Region Definitions fields are Flash Logic Window configurations used for dual image boot. When dual image boot is enabled, there are two boot images on the internal flash. ROM bootloader first tries to boot the boot image with latest version, if fails, ROM bootloader tries to boot the other boot image. To enable the dual image boot, Restore FLW Flag needs to be enabled, Boot Image Base Address field and FLW Region Definition fields need to be configured correctly. Boot Image Base Address indicates the start address of one boot image. FLW Region Definitions indicate the start address and the size of the alternative boot image. Here is the FLW Region Definition structure:

```
struct flw_region {
   uint32_t abase;
   uint32_t bcnt; };
```

The FLW region descriptors simply consist of FLW ABASE, the starting address of the alternative boot image, and, FLW BCNT, the size of the alternative boot image in 32 KB blocks.

If dual image boot is not enabled, ROM bootloader boots from Boot Image Base Address. If Boot Image Base Address is not configured, ROM bootloader will boot from the beginning of flash.

Configure NPX for Normal Boot, NPX Region Definitions fields are used for NPX feature.

Bit 0 of Boot Configuration field is used to enable ISP path for ROM bootloader. BOOT_CFG pin will be enabled if this bit is set. The pull-down on the BOOT_CFG pin is to ensure the device does not enter ISP by default. When BOOT_CFG pin is enabled and the BOOT_CFG pin is not pulled down, ROM bootloader will enter ISP path.

Boot Speed bits will be loaded by ROM bootloader to decide how to configure the power and clock. Bit 1:2 of Boot Configuration indicates the boot speed:

Table 56. Boot speed

Boot speed value in IFR0	Clock source	CPU_CLK/BUS_CLK/SLOW_CLK	DCDC & CORE_LDO voltage
11/00/01: Normal Boot (default)	FRO_192M	32/32/16 MHz	1.0 V
10: Fast Boot	FRO_192M	96/96/24 MHz	1.1 V

15.2.2.2 Token

Token needs to be programmed to the SECURE_PHANTOM_CONFIG fuse to change the ownership of radio firmware. By default, radio firmware is owned by NXP.

If developer at “OEM Open” lifecycle state requires an unrestricted access to radio flash for their own firmware development and debugging purposes, token is mandatory. Token contents must be programmed to SECURE_PHANTOM_CONFIG fuse, 0xddccbbaa (OEM1) or 0xaabbccdd (OEM2). If programmed token is valid, OEM_Enablement_Token fuse contents are used for NBU firmware authentication. Following a reset, specific decisions about locking radio core, blocking radio access, and debug authentication for CM3 core are made by boot ROM as described in Lifecycle states.

15.2.2.3 IFR0 Sector 2 (hash structure/cmac table maintenance)

This IFR0 dedicated page is used to store hash for boot images in a structured format as shown below. Note that, Buffer size of hash table to be reprogrammed must be at max. 48*4 bytes (to save flash programming time) but phrase-by-phrase programming can also be done as boot ROM progresses through various stages.

```
enum _cmac_table_index
{
kCmacIndexPrim = 0x0U,
kCmacIndexOEM = 0x1U,
kCmacIndexNBU = 0x2U,
kCmacIndexReserved = 0x3U,
kCmacIndexLast = 0x4U,
};

typedef struct{
uint32_t update_needed;
uint32_t reserved0[3];
uint8_t imgHash[NBOOT_CMAC_SIZE_IN_BYTES];
}cmac_info_t;

typedef struct {
cmac_info_t components[kCmacIndexLast]; /*!< Upto 4 components: 2 images (dual), 1 radio, 1 extended
bootloader (TBD) */
}cmac_table_t;
```

Table 57.

Offset	Size (bytes)	Field description	Description
0x0000	4	Hash update needed	0xFFFFFFFF - Yes 0x00000000 – No
	12	Reserved	Reserved

Table continues on the next page...

Table 57. (continued)

Offset	Size (bytes)	Field description	Description
0x0010	16	16 byte CMAC for application image	Hash of main flash primary boot image data at 0x0, or Hash of Image data at ABASE (if FLW is enabled)
0x0020	4	Hash update needed	0xFFFFFFFF - Yes 0x00000000 – No
	12	Reserved	Reserved
0x0030	16	16 byte CMAC for fallback image	Hash of fallback image data (if “Image 1 Base Address” field in IFR0 page0 is not 0xFFFFFFFF)
0x0040	4	Hash update needed	0xFFFFFFFF - Yes 0x00000000 – No
	12	Reserved	Reserved
0x0050	16	16 byte CMAC for radio image	Hash of radio (NBU) flash image data
0x0060	4	Reserved	Reserved
	12	Reserved	Reserved
0x0070	16	Reserved	Reserved

15.2.2.4 Over-the-air (OTA) update configuration

Table 58. OTA update configuration

Offset	Size (bytes)	Configuration Field	Description
0x00	4	Update available	0x746f4278u = Indicates update is available
	12	Reserved	Reserved
0x10	4	Update(sb3) dumped location	0x74784578 = sb3 is dumped in external flash
			Other values = sb3 is dumped in internal flash
0x14	4	Baud rate (in bps)	Value used to configure LPSPI NOR flash if update(sb3) is dumped in external flash.
0x18	4	Update(sb3) dump address	Start address (internal/external flash) where update(sb3) is dumped.
0x1c	4	Update(sb3) file bytes	Size of update(sb3) file in number of bytes.
0x20	4	FW update status	0x5ac3c35a = Indicates update was success 0x4412d283/0x2d61e1ac = Indicates failure to process sb3 file OR failure to erase/write update status to OTACFG page

Table continues on the next page...

Table 58. OTA update configuration (continued)

Offset	Size (bytes)	Configuration Field	Description
	12	Reserved	Reserved
0x30	16	Feature unlock key	To use FW update feature offered by ROM (instead of extended bootloader) 16-byte key must be programmed. updKey[16] = {0x61, 0x63, 0x74, 0x69, 0x76, 0x61, 0x74, 0x65, 0x53, 0x65, 0x63, 0x72, 0x65, 0x74, 0x78, 0x4d};

15.2.3 Normal boot path

ROM bootloader follows this path when device goes through a WARM reset or PoR if not getting request to go to another path. Boot ROM prepares the system for execution of application code residing in internal flash.

Please note that WDOG0 is disabled during normal boot path and enabled before jumping to the boot image. Meanwhile, the WDOG0 timeout value (TOVAL) has been set to longest (0xFFFF).

Figure 24 shows the flow of normal boot.

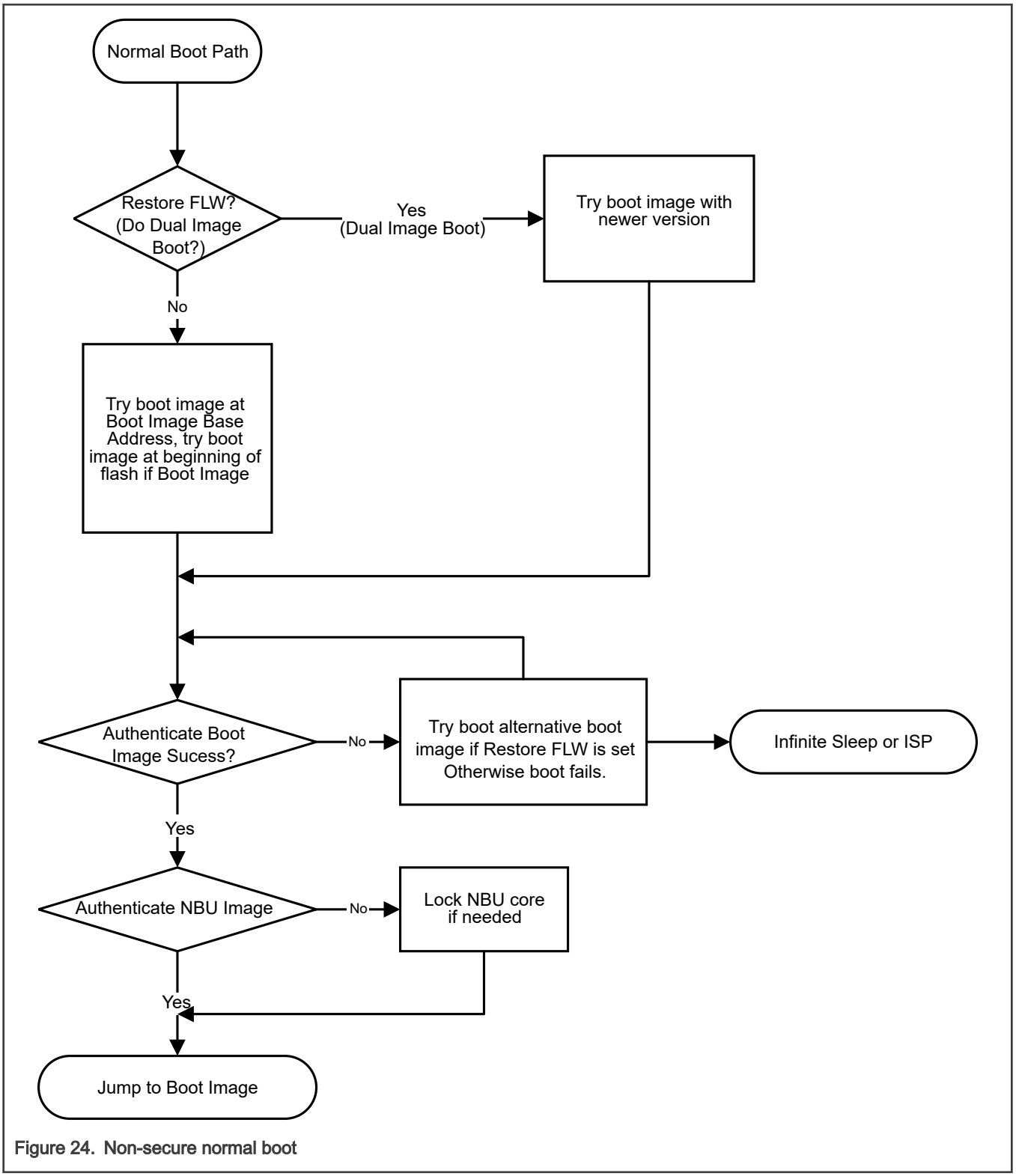


Figure 24. Non-secure normal boot

15.2.3.1 Dual image boot

Boot ROM supports dual image boot, useful in scenario where primary image fails to boot for any reason so secondary image (as a backup) can boot. This is achieved in conjunction with Flash Logical Window feature. It will allow developing for position independent code. That means same linker file can be used for creating multiple application images to be treated as primary (experimental or latest firmware image) and secondary (as a backup redundant image).

Use case example of dual boot:

1. “Restore FLW Flag” bit 0 is set to indicate ROM to use FLW feature.
2. Primary image is placed at address 0x10000 and “Boot Image Base Address” field of ROMCFG in user IFR0 is programmed accordingly.
3. Secondary image (fallback/backup) is placed at address 0x20000 and “ABASE” field in FLW region definitions of ROMCFG in user IFR0 is programmed accordingly.
4. Make sure BCNT field of FLW region definitions is set correctly to indicate image size as multiple of 32 KB block size.
5. Boot ROM will compare versions of primary and secondary image.
6. Boot ROM will try to boot latest versioned image. At the same time older versioned image base address is stored in ROM runtime context.
7. If both images have same version, then image located at address specified by “Boot Image Base Address” field of ROMCFG in user IFR0 is considered for boot.
8. For any reason (mainly authentication failure or rollback protection) latest versioned image fails to boot then boot ROM falls back to the older versioned image.

15.2.4 Faster subsequent boot

Boot ROM supports secure boot primarily using an ECDSA verification. And it can also support secure boot using hash (cmac) based image verification for faster subsequent boot.

- The user needs to:
  - Program IFR0 sector 0 at offset 0x120 to request hash based image verification.
  - Include an sbloader command to erase IFR0 sector 2 for every FW update.
- The ROM handles this request with the following steps:
  - First time after resetting to boot or trying to boot using jump/execute sbloader command after successful FW update, requires ROM to compute hash of image data.
  - Boot ROM computes hash (cmac) for multiple boot components, namely main flash image (application) and radio FW, and stores it in an IFR0 sector 2 in structured format. So it can be used to achieve faster boot time from next boot attempt.
  - But for very first time post FW update, after calculating hash of image data, boot ROM will still use ECDSA based image verification mechanism to perform secure boot.
  - For all subsequent FW updates, it is user’s responsibility to erase IFR0 sector 2. So that boot ROM understands it needs to calculate hash for multiple bootable components and store it in a structured format.
  - If no FW update is required/performed then triggering multiple resets could utilize faster secure boot sequence.
- If hash based image authentication fails
  - Images are still expected to have signature. So boot ROM would fall back to ECDSA based verification as a primary secure boot mechanism.

15.2.5 Debugger mailbox path

Debugger Mailbox (DM) is a register-based mailbox operating over Serial Wire Debug Port (SWD). It can be accessed via a Debug Access port (DM-AP) which is always enabled. The external world can send and receive data to/from the ROM bootloader via Debugger Mailbox. One important feature of Debugger Mailbox is that it supports NXP Debug Authentication Protocol, which allows users to enable debug access for CM33 when security is required. Meanwhile, users can utilize Debugger Mailbox to initiate a debugging session when there is no valid boot image.

The communication to the Debugger Mailbox is initiated by the debugger. It does so by setting a re-synchronization request and then resets the chip. ROM bootloader can observe the request and go to Debugger Mailbox Path.

15.2.5.1 Quick guide of using debugger mailbox

To make ROM bootloader enter the Debugger Mailbox, the debugger needs to select DM-AP and set the RESYNCH_REQ bit and the CHIP_RESET_REQ bit to 1 in the CSW register. Following a successful initial re-synchronization and reset, communication by the debugger to the device is achieved using 32-bit DM-AP command writes to the REQUEST register in the Debug Mailbox (DM-AP Commands are shown in table below) The debugger can read results of communications via the RETURN register. Response codes are shown in DM-AP return codes.

The example below shows how to enter debug mailbox and do the communication:

```
// Pseudo Code Syntax
// ------------------------------------------------------------------------
// WriteDP <register> <value>
// value = ReadDP <register>
// AP transactions presume the DM AP is selected
// WriteAP <register> <value>
// value = ReadAP <register> <value>
// ------------------------------------------------------------------------
// Read AP ID register to identify DM AP at index 2
WriteDP 2 0x020000F0
// The returned AP ID should be 0x002A0000
value = ReadAP 3
print "AP ID: ", value
// Select DM AP index 2
WriteDP 2 0x02000000
// Write DM RESYNC_REQ + CHIP_RESET_REQ
WriteAP 0 0x21
// Write DM-AP Command START DM-AP (1) to REQUEST register (1)
WriteAP 1 1
// Read the return code of START DM-AP command from RETURN register (2), 0x0 means success
ReadAP 2
// Write Get Lifecycle (2) to REQUEST register (1)
WriteAP 1 2
// Read the value of current lifecycle from RETURN register (2)
ReadAP 2
```

When doing debug authentication, Debug Auth Start command needs to be sent first and DAC needs to be read back. Then sending Debug Auth Response command with DAR. Pseudo code below shows how to use debug authentication commands. See Debug authentication for more details.

```
// Assume already in Debugger Mailbox
// Write Debug Auth Start (0x10) to REQUEST register (1)
WriteAP 1 0x10
// Read the data count of DAC
ReadAP 2
// Assume reading back 0x82000000, then the length of DAC is 0x200 word (4-byte)
// Send the ACK_TOKEN with the length of remaining data (word) to receive
WriteAP 1 0x0200A5A5
// Read DAC data
ReadAP 2
WriteAP 1 0x01FFA5A5
ReadAP 2
…
WriteAP 1 0x0001A5A5
ReadAP 2
// Get the whole DAC, generate DAR using DAC, assume length of DAR is 0x400 word (4-byte)
// Write Debug Auth Response (0x11) with DAR length to REQUEST register (1)
WriteAP 1 0x04000011
// Read the ACK_TOKEN from RETURN register (2), should be 0x0400A5A5
```

```
ReadAP 2
// Send DAR data, use 0x00000000 as an example of data
WriteAP 1 0x00000000
// Read the ACK_TOKEN, should be 0x03FFA5A5
ReadAP 2
// Send next DAR data
WriteAP 1 0x00000000
ReadAP 2
…
WriteAP 1 0x00000000
// Complete sending DAR when ACK_TOKEN is 0x0000A5A5
ReadAP 2
```

15.2.5.1.1 DM-AP commands

Commands for the DM-AP are listed below. These would be written to the REQUEST register. Please note that BulkErase/ISP Mode/Set FA Mode can only be issued after debug authentication is successful. These commands are not available in OEM Open.

Table 59. DM-AP commands

Command	ID	Parameter/Response	Description
Start DM-AP	0x01	*Parameters*: None *Response*: 32-bit status	Cause the device to enter DM-AP command mode. This must be done prior to Get Lifecycle command. This command is provided for backwards compatibility and does not need to be used for most commands.
Get Lifecycle	0x02	*Parameters*: None *Response*: 32-bit value	Return the value of current lifecycle.
Bulk Erase	0x03	*Parameters*: None *Response*: 32-bit status	Erase the entire CM33 Program Flash (IFR0 not included).
Exit DM-AP	0x04	*Parameters*: None *Response*: 32-bit status	Cause the device to exit DM-AP command mode. The device returns to normal boot path.
Enter ISP Mode	0x05	*Parameters*: dataWordCount: 0x1 data[0]: ISP mode enum. 0xffffffff – Auto detection 0x1 - LPUART 0x2 - LPI2C 0x4 - LPSPI 0x8 - CAN Others - Reserved *Response*: 32-bit status	Enter specified ISP path.

Table continues on the next page...

Table 59. DM-AP commands (continued)

Command	ID	Parameter/Response	Description
Set FA Mode	0x06	*Parameters*: data[]: FA Request *Response*: 32-bit status	Set the part permanently in Fault Analysis (FA or RMA) mode bit in PFR field, and return the part to NXP for FA/RMA testing. ROM erases customer sensitive assets (key codes) stored in PFR.
Start Debug Session	0x07	*Parameters*: None *Response*: 32-bit status	This command is used to indicate ROM the intention of connecting debugger. ROM bootloader enables debug access (if no debug authentication required) and enters while(1) loop.
Debug Auth Start	0x10	*Parameters*: dataWordCount: 0x0 *Response*: data[] : DAC	Start Debug Authentication Protocol. ROM responds to debugger with *DAC (Debug Authentication Challenge)* message.
Debug Auth Response	0x11	*Parameters*: data[] : DAR *Response*: 32-bit status	Debug Authentication Response.

15.2.5.1.2 DM-AP return codes

Return codes for DM-AP commands are listed below. These commands can be read from the RETURN register. See table below.

Table 60. DM-AP return codes

Return code	Descriptions
0x0000 0000	Command succeeded.
0x0010 0001	Debug mode not entered. This is returned if other commands are sent prior to the “Enter DM-AP” command.
0x0010 0002	Command not recognized. A command was received other than the ones defined above.
0x0010 0003	Command failed.

15.2.5.2 Debugger Mailbox protocol

ROM Bootloader implements debug mailbox protocol to interact with host debug systems over the SWD interface. The protocol has following features:

- Request/response based.
- Support for relatively large command and response data.
- All commands and responses are 32-bit word aligned.
- Supports data above 32 bits by using an ACK_TOKEN that moderates the transfer in 32-bit value chunks.
- Requests and responses use the same basic structure.

15.2.5.2.1 Mailbox registers

The registers in the debug mailbox are shown in Table below. These registers are readable by the CPU and are intended primarily to allow on-chip ROM routines to implement requests from an external debugger.

Table 61. Register overview: DBGMailbox (base address = 0x58000000)

Name	Access	Offset	Description	Reset value
CSW	R/W	0x000	Command and status word	0x0
REQ	R/W	0x004	Request from the debugger to the device	0x0
RETURN	R/W	0x008	Return value from the device to the debugger	0x0
ID	RO	0x0FC	Identification register	0x002A0000

15.2.5.2.1.1 Command and Status Word Register (CSW)

The CSW register contains command and status bits to facilitate communication between the debugger and the device.

Table 62. Command and Status Word register (CSW, offset = 0x000)

Bit	Symbol	Description	Reset Value
0	RESYNCH_REQ	The debugger sets this bit to request a re-synchronization.	0x0
1	REQ_PENDING	A request is pending for the debugger: a value is waiting to be read from the REQUEST register.	0x0
2	DBG_OR_ERR	When 1, a debug overrun has occurred: a REQUEST value has been overwritten by the debugger before it was read by the device.	0x0
3	AHB_OR_ERR	When 1, an AHB overrun has occurred: a RETURN value has been overwritten by the device before it was read by the debugger.	0x0
4	SOFT_RESET	Setting this write-only bit in the CSW register of the DM-AP by an external debugger resets the DM-AP state machine and its registers. While setting RESYNCH_REQ only resets the DP and CPU handshake state machine.	0x0
5	CHIP_RESET_REQ	This write-only bit causes the device (but not the DM-AP) to be reset by generating SYSRESET_REQ.	0x0
31:6	Reserved		-

15.2.5.2.1.2 Request value register

The REQUEST register is used by a debugger to send action requests to the device.

Table 63. Request value register (REQUEST, offset = 0x004)

Bit	Symbol	Description	Reset Value
31:0	REQUEST	Request value. Reads as 0 when no new request is present. Cleared by the device. Can be read back by the debugger in order to confirm communication.	0x0

15.2.5.2.1.3 Return value register

The RETURN register provides any response from the device to the debugger.

Table 64. Return value register (RETURN, offset = 0x008)

Bit	Symbol	Description	Reset Value
31:0	RET	Return value. This is any response from the device to the debugger. If no new data is present, a debugger read will be stalled until new data is available.	0x0

15.2.5.2.1.4 Identification register

The ID register provides an identification of the DM-AP interface.

Table 65. Identification register (ID, offset = 0x0FC)

Bit	Symbol	Description	Reset Value
31:0	ID	Identification value.	0x002A0000

15.2.5.2.2 Mailbox commands

15.2.5.2.2.1 Request

The first word transmitted in a request is a header word containing the command ID and number of following data words. Following the header are the number of 32-bit words specified in the header.

Table 66. Request register byte description

Word	Byte 0	Byte 1	Byte 2	Byte 3
0	commandID[7:0]	commandID[15:8]	dataWordCount[7:0]	dataWordCount[15:8]
1	*data...*			

The C structure definition for a request is as follows:

```
struct dm_request {
  uint16_t commandID;
  uint16_t dataWordCount;
  uint32_t data[]; };
```

15.2.5.2.2.2 Response

The first word transmitted in a response is a header word containing the command status and number of following data words.

- To support legacy command and response value, Bit_31 in header will be used to indicate that the response follows new protocol structure.

Table 67. Response register byte description

Word	Byte 0	Byte 1	Byte 2	Byte 3
0	commandStatus[7:0]	commandStatus[15:8]	dataWordCount[7:0]	dataWordCount[14:8] new_protocol[15]
1	*data...*			

The C structure definition for a response is as follows:

```
struct dm_response {
    uint16_t commandStatus;
    uint16_t dataWordCount;
    uint32_t data[];
};
```

15.2.5.2.2.3 ACK_TOKEN

- When command has parameters the debugger should wait for ACK_TOKEN (sent through DBG_MB_RETURN register) before sending next 32-bit value.
- Similarly when response packet has data to send back to debugger, ROM will wait for debugger to send ACK_TOKEN (sent through DBG_MB_REQUEST register) before sending next 32-bit value.
- Upper 16 bits are set by receiving end with number of remaining words expected.
- Lower 16 bits are always set to 0xA5A5.

Table 68. ACK_TOKEN register byte description

Word	Byte 0	Byte 1	Byte 2	Byte 3
0	0xA5	0xA5	remainCount[7:0]	remainCount[15:8]

The C structure definition for a ACK_TOKEN is as follows:

```
struct dm_ack_token {
    uint16_t token; /* always set to 0xA5A5 */
    uint16_t remainCount; /* count of remaining word */};
```

15.2.5.2.2.4 Error handling

If an overrun occurs from either side of the communication, the appropriate error flag is set in the CSW. The state machine hardware will prevent further communication in any direction once an overrun has occurred in that direction, so if such an error occurs, the debugger will need to start with a new re-synchronization request in order to clear the error flag.

15.2.5.3 Debug authentication

The fundamental principles of debugging, which require access to the system state and system information, conflict with the principles of security, which require the restriction of access to assets. Thus, many products disable debug access completely before deploying the product. This causes challenges for product design teams to do proper Return Material Analysis (RMA). To address these challenges, the ROM bootloader offers a debug authentication protocol as a mechanism to authenticate the debugger (an external entity) has the credentials approved by the product manufacturer before granting debug access to the device.

The debug authentication scheme is a challenge-response scheme and assures that debugger in possession of required debug credentials only can successfully authenticate over debug interface and access restricted parts of the device. This protocol is divided into steps as shown in description and figure below.

1. The debugger initiates the Debug Mailbox message exchange by setting the RESYNCH_REQ bit and CHIP_RESET_REQ bit in the CSW register of DM-AP.
2. The debugger waits (minimum 30ms) for the devices to restart and enter debug mailbox request handling loop.

3. The debugger sends Debug Authentication Start command (command code 0x10) to the device.
4. The device responds back with Debug Authentication Challenge (DAC) packet based on the debug access rights pre-configured in CMPA fields, which are collectively referred as Device Credential Constraints Configuration (DCFG_CC). The response packet also contains a 32 bytes random challenge vector.
5. The debugger responds to the challenge with a Debug Authentication Response (DAR) message by using an appropriate debug certificate, matching the device identifier in the DAC. The DAR packet contains the debug access permission certificate, also referred as Debug Credential (DC), and a cryptographic signature binding the DC and the challenge vector provided in the DAC.
6. The device on receiving the DAR, validates the contents by verifying the cryptographic signature of the message using the debugger's public key present in the embedded the Debug Credential (DC). On successful validation of DAR, the device enables access to the debug domains permitted in the DC.

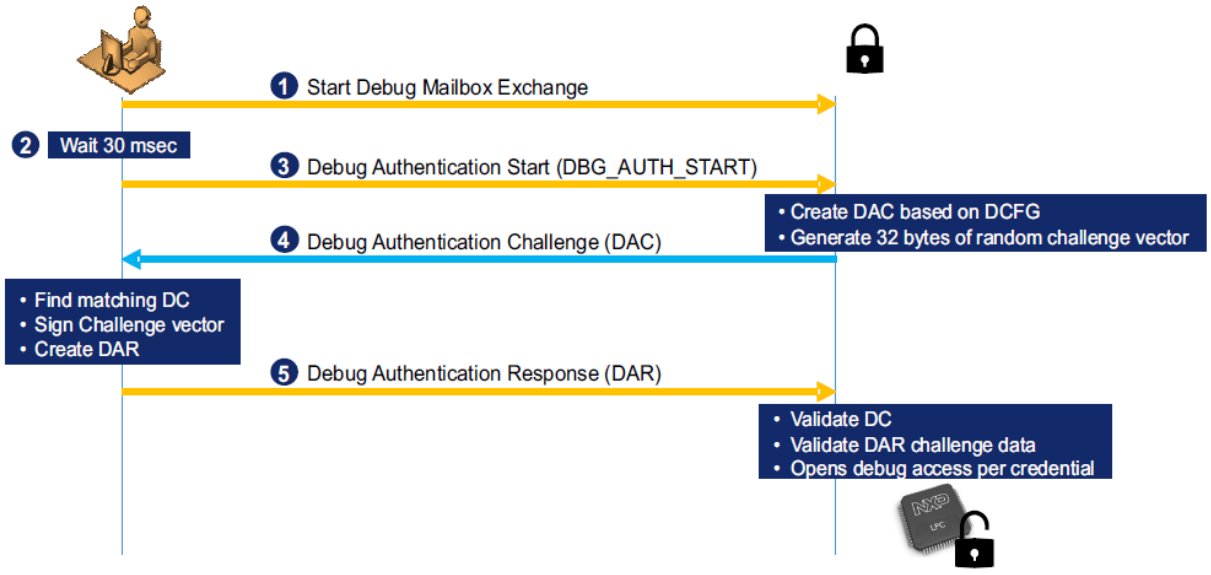


Figure 25. Debug authentication flow

15.2.5.3.1 Debug access control configuration

The ROM bootloader handles the device side of Debug authentication process. However, the debug access control rights and security policies can be configured by programming the following configuration fields which are collectively referred to as Device Configuration for Credential Constraints (DCFG_CC).

- DCFG_VER: This field controls the cryptographic primitives used during authentication.
- DCFG_ROTID: This field defines the Root of trust identifier (ROTID). The ROTID field is used to bind the devices to specific Certificate Authority (CA) keys issuing the debug credentials. These CA keys are also referred as Root of Trust (RoTK) keys.
- DCFG_UUID: Controls whether to enforce UUID check during Debug Credential (DC) validation. If this field is set only DC with matching device UUID can unlock the debug access.
- DCFG_CC_SOCU: This configuration field specifies access rights to various debug domains.
- DCFG_VENDOR_USAGE: This field can be used to define vendor specific debug policy use case such as DC revocations or department identifier. It is recommended to use field for revocation of already issued debug certificates.

15.2.5.3.1.1 Protocol Version (DCFG_VER)

This device supports Debug authentication protocol version 2.1, which are defined based on the secp384r1 curve.

NOTE

Both debug authentication certificates and image signing certificates use same Root of Trust keys (RoTK).

15.2.5.3.1.2 Root of Trust Identifier (DCFG_ROTID)

The Root of Trust Identifier used in debug authentication protocol is composed of two elements.

- A 256-bit cryptographic hash (SHA256 or leading 32 bytes of SHA384, according root certificates EC types, secp256r1->SHA256 or secp384r1->SHA384) over the Root of Trust Keys hash table.
- A 32-bit field containing revocation bits for the four Root of Trust keys in the table.

15.2.5.3.1.3 Enforce UUID checking (DCFG_UUID)

Controls whether to enforce UUID check during Debug Credential (DC) validation. If this field is set, then only DC containing a UUID attribute that is an exact match to the device can unlock the debug access.

This field can be set by programming UUID_CHECK (bit 16) in CC_SOCU.

This device-specific constraint, if enabled, is in addition to all the other constraints defined and enforced by the authentication protocol

15.2.5.3.1.4 Credential Constraints (DCFG_CC_SOCU)

The DCFG_CC_SOCU is a configuration that specifies debug access restrictions per debug domain. These access restrictions are also referred as constraint attributes in this section. The debug subsystem is sub-divided into multiple debug domains to allow finer access control. Table 70 shows debug domains and their corresponding control bit position in DCFG_CC_SOCU. Which is logically composed of two components:

- SOCU_PIN (bit 7-0 in CC_SOCU): A bitmask that specifies which debug domains are predetermined by device configuration.
- SOCU_DFLT (bit 15-8 in CC_SOCU): Provides the final access level for those bits that the SOCU_PIN field indicated are predetermined by device configuration.

The following table shows the restriction levels:

Table 69. Access restriction levels

Restriction Level	SOCU_PIN [n]	SOCU_DFLT [n]	Description
0	1	1	Access to the sub-domain is always enabled. This setting is provided for module use case scenario where DCFG_CC_SOCU_NS would be used to define further access restrictions before final deployment of the product.
1	0	0	Access to the sub-domain is disabled at start up. But the access can be enabled through debug authentication process by providing appropriate Debug Credential (DC) certificate.
2	0	1	Illegal setting. Part may lock-up if this setting is selected.
3	1	0	Access to the sub-domain is permanently disabled and cannot be reversed. This setting offers the highest level of restriction.

Debug domains supported by this device are shown in table below.

Table 70. CC_LIST_Table

Bit[1]	Symbol	Description
0	PIN_NIDEN	Controls non-Invasive debugging of TrustZone for Arm8-M defined non-secure domain of CM33.
1	PIN_DBGEN	Controls invasive debugging of TrustZone for Arm8-M defined non-secure domain of CM33.
2	PIN_SPNIDEN	Controls non-Invasive debugging of TrustZone for Arm8-M defined secure domain of CM33.
3	PIN_SPIDEN	Controls invasive debugging of TrustZone for Arm8-M defined secure domain of CM33.
4	PIN_NBU_DBGEN	Controls debugging of NBU (radio, CM3)
5	PIN_FA_CMD_EN	Controls whether DM-AP command, Set FA Mode (command code: 0x06), can be issued through after authentication.
6	PIN_ISP_CMD_EN	Controls whether ISP boot flow DM-AP command (command code: 0x05) can be issued after authentication.
7	PIN_ME_CMD_EN	Controls whether DM-AP command, Bulk Erase (command code: 0x02), can be issued through after authentication.
8	DFLT_NIDEN	Controls non-Invasive debugging of TrustZone for Arm8-M defined non-secure domain of CM33.
9	DFLT_DBGEN	Controls invasive debugging of TrustZone for Arm8-M defined non-secure domain of CM33.
10	DFLT_SPNIDEN	Controls non-Invasive debugging of TrustZone for Arm8-M defined secure domain of CM33.
11	DFLT_SPIDEN	Controls invasive debugging of TrustZone for Arm8-M defined secure domain of CM33.
12	DFLT_NBU_DBGEN	Controls debugging of NBU (radio, CM3)
13	DFLT_FA_CMD_EN	Controls whether DM-AP command, Set FA Mode (command code: 0x06), can be issued through after authentication.
14	DFLT_ISP_CMD_EN	Controls whether ISP boot flow DM-AP command (command code: 0x05) can be issued after authentication.
15	DFLT_ME_CMD_EN	Controls whether DM-AP command, Bulk Erase (command code: 0x02), can be issued through after authentication.
16	UUID_CHECK	Used for UUID matching required flag.
17:31	Reserved	Reserved

1. Bits [7:0] is SOCU pinned value, and bits [15:8] is SOCU default value.

15.2.5.3.1.5 DCFG_VENDOR_USAGE

This field can be used to define vendor specific debug policy use case such as Debug Credential (DC) certificate revocations or department identifier or model identifier. During Debug Authentication Response (DAR) processing the device checks that the value specified in Vendor Usage field of DC matches exactly the value programmed in DBG_AUTH_VU fields of device configurations. During debug authentication this value has to mach cc_vu value in debug credentials. For example user can define DBG_AUTH_VU for every product and then issue debug credential just for specific product.

15.2.5.3.2 Debug Authentication Challenge (DAC)

The debug authentication protocol begins with a DEBUG_AUTH_START request, issued by the debugger to the device, which replies by Authentication Challenge (DAC) message, issued by the device to the debugger. The DAC carries information about current device state to the debugger, inputs from DAC are necessary to find existing or request generation of new Debug Credential (DC). Protocol version 2.x uses ECC for digital signatures: minor version 1 (“2.1”) points to secp384r1.

Structure of DAC message is shown in following Figure.

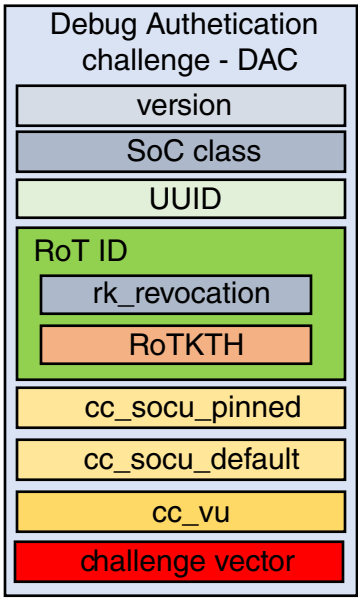


Figure 26. Debug Authentication Challenge (DAC)

- **version** – uint32_t, little endian

  Fixed to value 0x02000100u "2.1" (major =2, minor = 1) -> secp384r1 used as RoTKTH

- **SoC class** – uint32_t, little endian

  Fixed to 0x5u for this device, identify the device family

- **UUID** – uint32_t[4], little endian

  Unique ID of device.

- **RoT ID** – object

  Identifier of root of trust settings in device.

  - **rk_revocation** – uint32_t, little endian

    1 bit per RoTK (up to 4 RoTK can be defined for device, bit is 1 – RoTK is revoked (not usable for debug auth), bit is 0 – RoTK can be used.

    rk_revocation[bits 31:4] -> reserved for future use.

    rk_revocation[bit 3] -> bit value 0 = RoTK[3] OK / bit value 1 = RoTK[3] revoked

rk_revocation[bit 2] -> bit value 0 = RoTK[2] OK / bit value 1 = RoTK[2] revoked

rk_revocation[bit 1] -> bit value 0 = RoTK[1] OK / bit value 1 = RoTK[1] revoked

rk_revocation[bit 0] -> bit value 0 = RoTK[0] OK / bit value 1 = RoTK[0] revoked

- **RoTKTH** – uint8_t[32]

  RoTKTH (Root of Trust Key Table Hash) value is stored in device fuses. If RoTKTH size is 48 bytes (RoTKTH is SHA384 digest of the RoT Key Table), then this field value carries only leading 32 bytes of RoTKTH full value.

- **cc_socu_pinned** – uint32_t, little endian

  Actual value of CC_SOCU_PINNED in device, which is based on actual device lifecycle, DCFG_CC_SOCU_L1 and DCFG_CC_SOCU_L2 fuses combination. For bit representation of CC_SOCU_PINNED please refer to Table 69.

- **cc_socu_default** – uint32_t, little endian

  Actual value of CC_SOCU_DEFAULT, which is based on actual device lifecycle, DCFG_CC_SOCU_L1 and DCFG_CC_SOCU_L2 fuses combination. For bit representation please refer to Table 69.

- **cc_vu** – uint32_t, little endian

  Vendor usage value, custom value provided by vendor, stored in DBG_AUTH_VU fuse in device, value needs to be the same as value specified present later in debug certificate (DC), used for pairing of corresponding DC.

- **challenge vector** – uint8_t[32]

  Random number generated for each request to be signed in debug authentication response (DAR) to avoid reusing of DAR.

15.2.5.3.3 Debug Credential (DC)

Prior to generating debug credential, it is necessary to generate EC keypair (secp384r1) for debugger known as Debug Credential Key (DCK). The public key part of DCK (DCK_{public}) is used to represent the identity of the debugger through the creation of a DC, which binds that public key to some usage attributes (UUID, CC_SOCU), and is then authorized/signed by the vendor's Roots of Trust key ($RoTK_{private}$) using ECDSA algorithm. If RoTK is based on secp384r1then DCK also needs to be based on secp384r1. Combining of EC types is not allowed in DC.

The private part of DCK ($DCK_{private}$) will be used later for signing of Debug Authentication Response (DAR).

The DC structure is illustrated in the following figure.

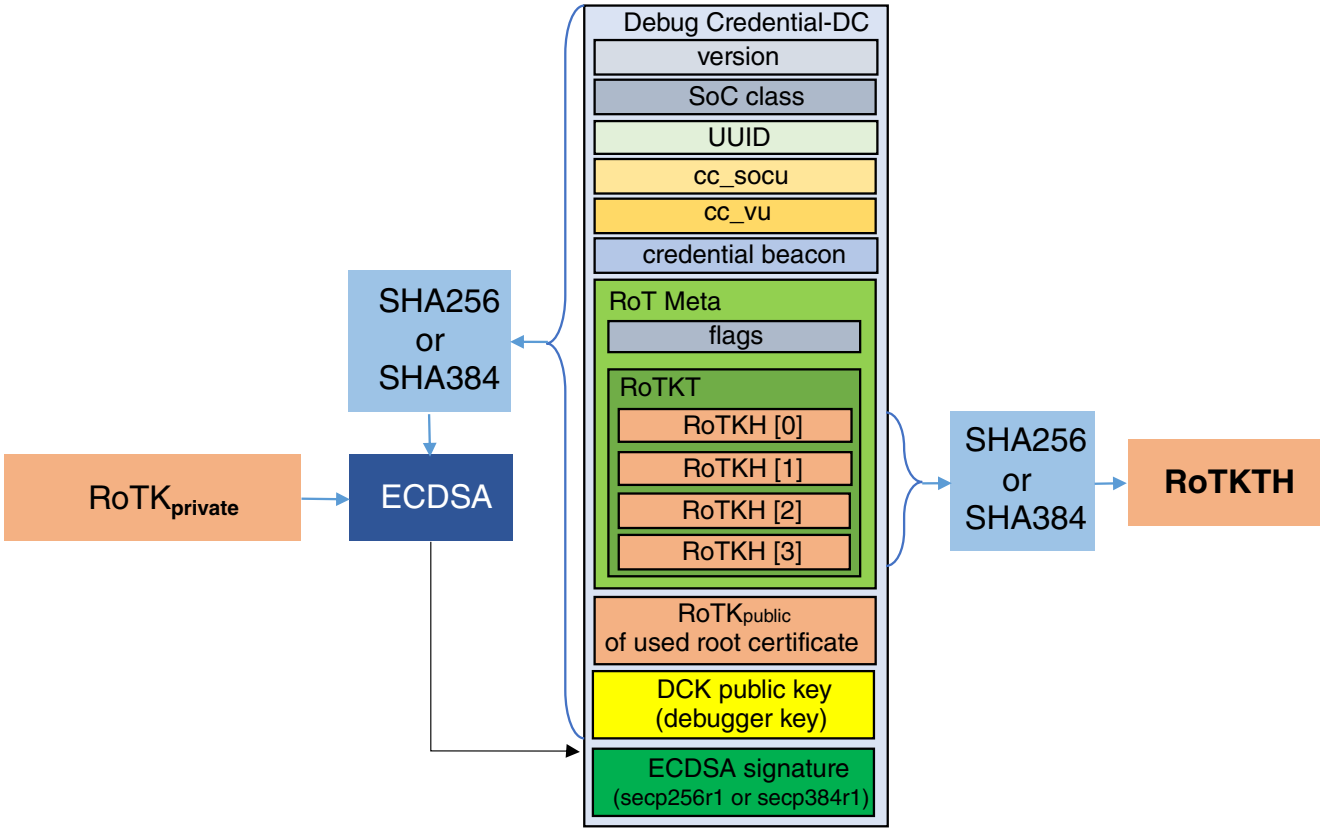


Figure 27. Debug Credential (DC)

The data structure is represented as a packed binary concatenation of its component fields as shown in the list below:

- **version** – uint32_t, little endian

  Fixed to value 0x02000100u “2.1” (major =2, minor = 1) -> secp384r1 used as RoTK

- **SoC class** – uint32_t, little endian

  Fixed to 0x5u, identify the device family.

- **UUID** – uint32_t[4], little endian

  Unique ID of device. Can be set to 0 when debug credential is used for SoC class matching (one debug credential can be used to open debug ports on all devices with same CTRK and same SoC class, UUID matching in device have to be disabled – default). If UUID matching in device is enabled, then UUID value needs to correspond with UUID value of the device received in DAC.

- **cc_socu** – uint32_t, little endian

  Proposed value of SOCU, will be validated and combined on device with current lifecycle, DCFG_CC_SOCU_L1 and DCFG_CC_SOCU_L2 fuses values to have final state of debug ports, which will be applied on device. Refer to Table 69 for Bit details.

- **cc_vu** – uint32_t, little endian

  Vendor usage value need to be matched with value of DBG_AUTH_VU in device fuses (received on DAC).

- **credential beacon** – uint32_t, little endian

  Value loaded to the beacon register after successful debug authentication. Value is used by application to control debug ports. Mask 0xFFFFu is applied to the value of credential beacon and is connected with authentication beacon value from DAR as following formula: beacon = (credentialBeacon & 0xFFFFu) | ((authenticationBeacon & 0xFFFFu) << 16). If final

beacon value is non zero, debug ports are not enabled by ROM bootloader itself and debug ports configuration is delegated to booted application, which can control debug ports according final beacon value available in register (e.g. third party secondary bootloader will open debug just before jump to loaded application, debug of secondary bootloader can be enabled only by special beacon value of credential beacon).

- **RoT meta** – object

  Device root of trust meta data object.

  — **flags** – uint32_t, little endian

  Configuration details of RoT meta object, needs to be matched with RoT ID received in DAC.

  ◦ **flags[bit 31]**: CA flag, always 1 in DC.

  ◦ **flags[bits 30:12]**: Reserved for future use.

  ◦ **flags[bits 11:8]**: Used root cert number [0-3] (specify RoTK used to DC signature), used only when more than 1 RoTK specified for device.

  ◦ **flags[bits 7:4]**: Number of records in RoTKT [1-4], when equal to 1, then RoTKT not present in DC.

  ◦ **flags[bits 3:0]**: Reserved for future use.

  — **RoTKT** – object

  Root of Trust keys Table, optional object (in case when is specified only one RoTK for device, RoTKT is not present, 1 to 4 RoTK can be specified for device), so if RoTKT is present in DC, then RoTKT table contains at least two RoTKH and maximally 4 RoTKH records.

  ◦ **RoTKH[0-3]** – uint8_t[32] or uint8_t[48]

  Root of Trust Key Hash is SHA256 or SHA384 of RoTKpublic. Hash algorithm is selected based on RoTK EC type (secp256r1 -> SHA256 or secp384r1 -> SHA384). Same RoTKs and RoTKTH values are shared between debug authentication, SB3.1 firmware update container and signed boot image.

- **$RoTK_{public}$** – uint8_t[64] or uint8_t[96]

  X and Y coordinates of public key of selected RoTK, which will be used for signing and verification of DC, field size depends on EC type (secp256r1 or secp384r1)

- **DCK_{public}** – uint8_t[64] or uint8_t[96]

  X and Y coordinates of public key of DCK, field size depends on EC type of DCK key (secp256r1 or secp384r1). DCK and RoTKs must be defined on same EC type, combination of EC types is not allowed.

- **ECDSA signature** – uint8_t[64] or uint8_t[96]

  SHA256 or SHA384 of all Debug Credential data (as described on previous figure) used for ECDSA signature. Hash algorithm selected according to EC type of RoTK (secp256r1 -> SHA256 or secp384r1 -> SHA384). Signature coordinates (r,s) stored in big-endian.

15.2.5.3.4 Debug Authentication Response (DAR)

Debugger will find existing or request generation of new corresponding DC according to the inputs from DAC message. The $DCK_{private}$ key is used as input for calculation of ECDSA signature of the corresponding DC including device UUID and Challenge vector from received DAC (Challenge vector is not part of DAR, but it is used as last input to hash of DAR). For signature used SHA256 or SHA384 hash algorithm based on DCK EC type (secp256r1 or secp384r1), if secp256r1-> SHA256, if secp384r1->SHA384. The DAR structure and generation process is described in following figure.

Because regular debuggers usually don't offer feature for ECDSA signing, some PC tool communicating with debugger is needed. NXP offers tool nxpcertgen.exe for generation of DC and nxpdebugmbox.exe for full debug authentication process. Usage of mentioned tools is described in separate user manual related to them.

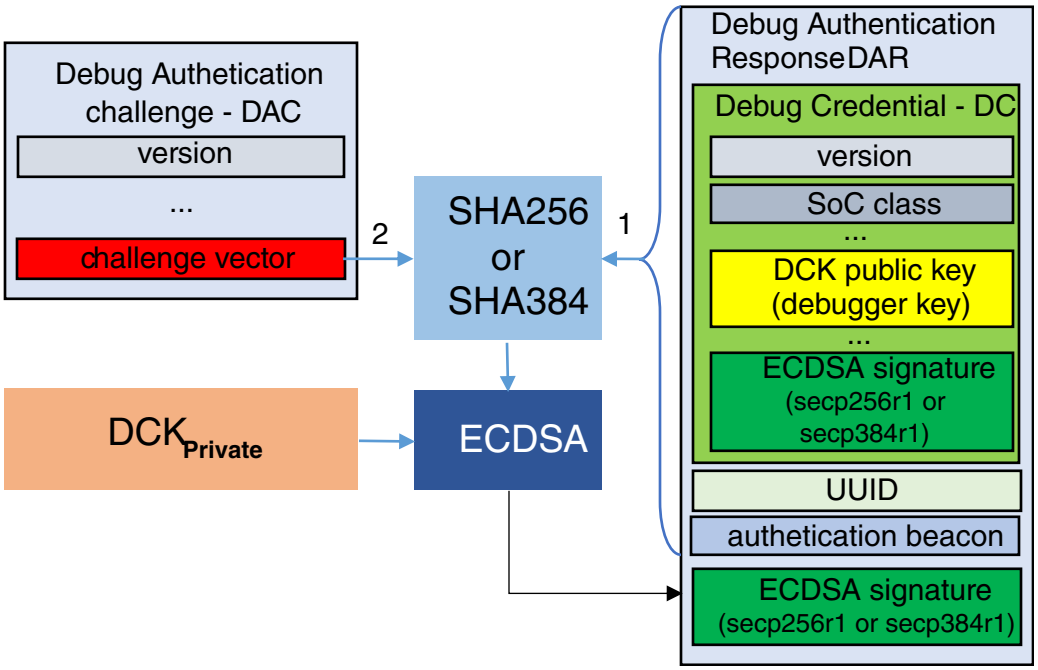


Figure 28. Debug Authentication Response (DAR)

- **Debug credential** – object

  Matched debug credential based on data received from DAC message.

- **authentication beacon** – uint32_t, little endian

  Value loaded to the beacon register after successful debug authentication. Value is used by running application to control debug ports. Mask 0xFFFFu is applied to the value and is connected with credential beacon value from DC as following formula: beacon. Beacon = (credentialBeacon & 0xFFFFu) | ((authenticationBeacon & 0xFFFFu) << 16). If final beacon value is non zero, debug ports are not enabled by ROM bootloader itself and debug ports configuration is delegated to booted application, which can control debug ports according final beacon value available in register (e.g. third party secondary bootloader will open debug just before jump to loaded application, debug of secondary bootloader can be protected by special beacon value).

- **UUID** – uint32_t[4], little endian

  Unique ID of device, must be same as UUID value provided by device in DAC message

15.2.5.3.5 Device processing the DAR

The device BootROM will process DAR received from debugger. As a part of the validation step, device will:

1. Verify DC: Validate DC version, SoCC, UUID, RoT, VU, and DC signature.
2. Verify that the DAR has a valid signature that binds it to the CV from the DAC.

If all the steps are successfully completed, it can be deduced that:

- The debugger possesses the private key corresponding to the vendor/RoT-signed credential.
- The credential satisfies the constraints specified in the device configuration.
- The response of the debugger to the challenge from the device is produced and signed in response to the challenge (because of its cryptographic dependency on the challenge vector). The response is not replayed from a previous authentication where a different challenge vector is used.

After completion of processing DAR, if authentication is successful, Debug Access will be granted. If authentication fails, no special response is issued but further debug access request will be ignored, and device will enter in a failure loop (infinite loop waiting for debug attach)

15.2.5.3.6 Debug authentication use cases

15.2.5.3.6.1 Return Material Analysis (RMA) use case

The diagram below shows RMA flow using debug authentication scheme, where a debug credential certificate is issued for each field technician.

1. Vendor generates RoT key pairs and programs the device with SHA256 hash of RoT public key hashes before shipping
2. Field technician generates his own key pair and provides public key to vendor for authorization.
3. Vendor attests the field technician's public key. In the debug credential certificate, vendor assigns the access rights.
4. End customer having issues with a locked product takes it to field technician.
5. Field technician uses his credentials to authenticate with device and un-locks the product for debugging.

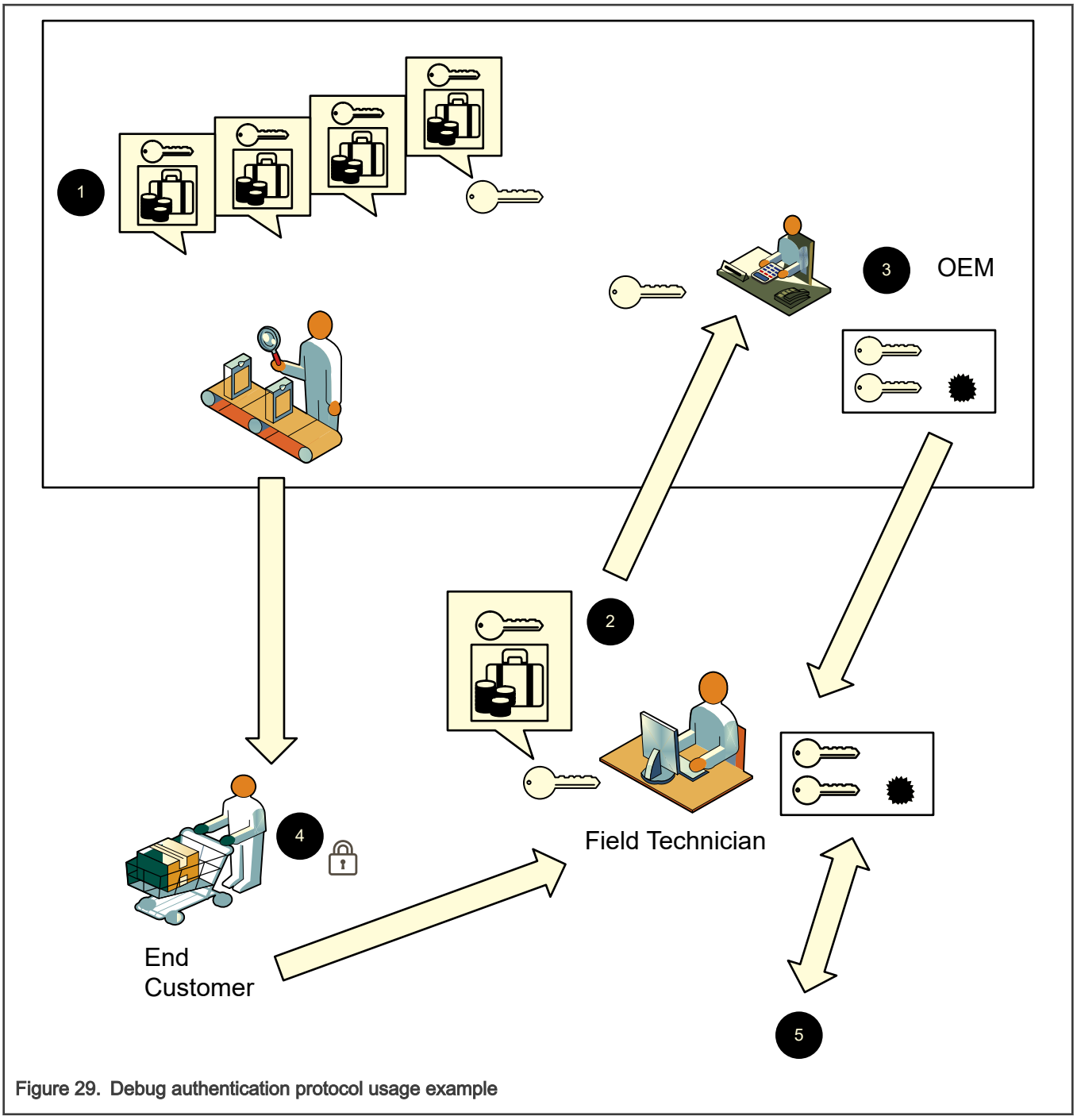


Figure 29. Debug authentication protocol usage example

15.2.5.3.6.2 Module use case with OEM tier1 and tier2 Lifecycle states

The CC_SOCU_PIN_NS & CC_SOCU_DFLT_NS is provided to allow module-maker and OEM using the module to implement tiered protection approach.

- The module maker who is referred as Tier-1 developer can develop secure code and define access rights to his module using CC_SOCU_PIN & CC_SOCU_DFLT.
- Configuration can be such that debug access to secure module is blocked but non-secure debug is always allowed.
- Once the module is ready, Tier-1 developer can release the module to OEM (a.k.a. tier-2 developer), but block debug access to secure mode and enable debug access to non-secure mode.

- Tier-2 developer can develop non-secure module and extend access rights configuration to that module using CC_SOCU_DFLT_NS and CC_SOCU_PIN_NS.

15.2.5.3.7 Fault Analysis (FA) mode

The ROM bootloader offers an FA Mode (Set FA Mode) command handler to enable to delete sensitive information (for example, Keys) before handling over the device to OEM for fault analysis. ROM allows Set FA Mode command only when corresponding flag, FA_CMD_EN, is set. Users must pass an FA request message to Set_FA_Mode command to enter FA mode.

Set FA Mode command will perform the following:

- Erase all OEM Keys in fuse
- Erase the internal flash
- Flush all temporary key registers
- Move the lifecycle to OEM Return

15.2.6 Low-power wakeup path

ROM bootloader goes to low-power wakeup path if the register SPC0->WAKEUP is set to a non-zero value, WAKEUP_DIS is 0 indicating wakeup path is not disabled and Sticky NPX Configuration for Waking up from Deep Power Down is not set. Please note that ROM bootloader does not configure/service/disable Watchdog for low power wakeup path. And ELE is not released for this path. Figure below shows the flow of low power wakeup path.

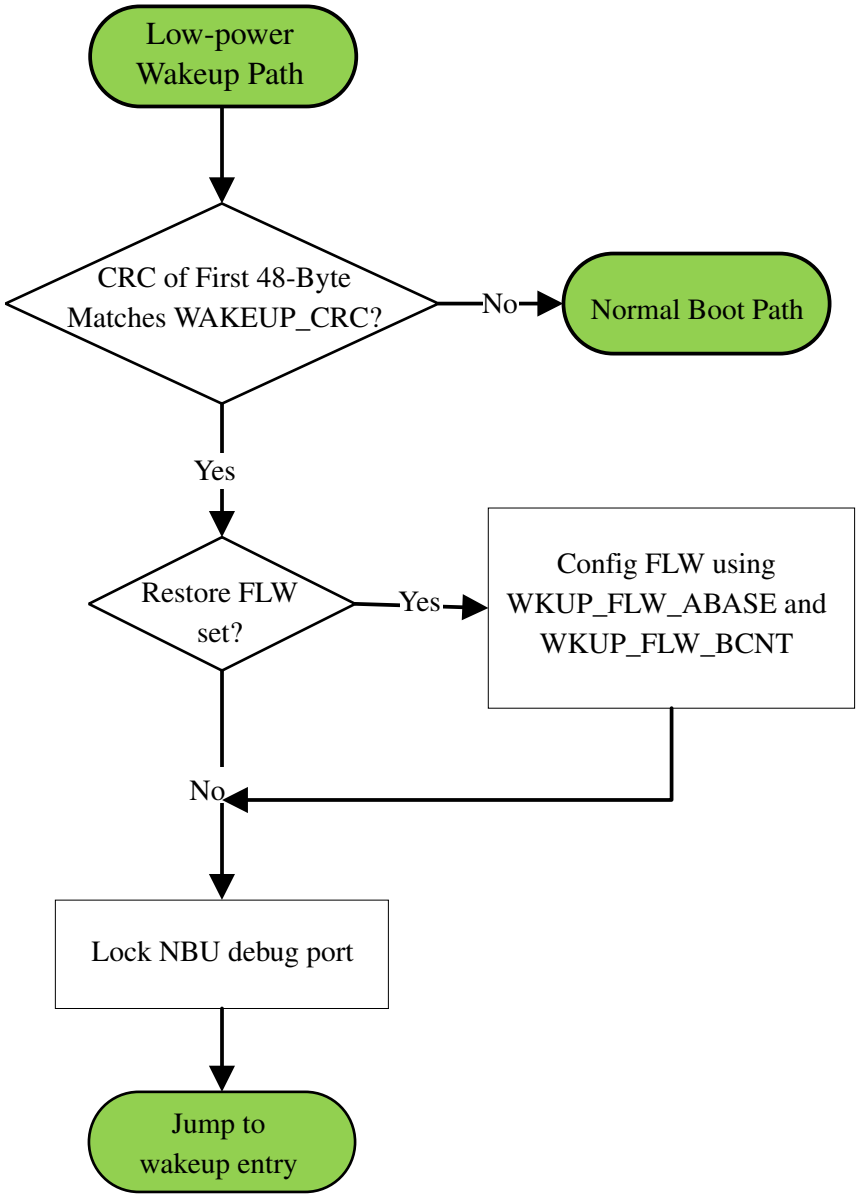


Figure 30. Low power wakeup flow

If WAKEUP_CRC in REGFILE1 (REG0 of REGFILE1) is set correctly, matching the CRC for first 48 bytes of wakeup image, ROM bootloader jumps to the wakeup process entry stored in SPC0->WAKEUP. Otherwise, ROM bootloader exits low power wakeup path and go to normal boot path. Please note that CM3 debug port is locked for low power wakeup path.

In addition, if Flash Logic Window needs to be configured for low power wakeup path, Restore FLW Flag, WAKEUP_FLW_ABASE (REG1 of REGFILE1) and WAKEUP_FLW_BCNT (REG2 of REGFILE1) need to be set correctly. ROM bootloader will configure the FLW using the value from WAKEUP_FLW_ABASE and WAKEUP_FLW_BCNT registers. Please note that Wini_applicatiohalting_executing_immediately_following_rom_AKEUP_FLW_BCNT has three fields, including Valid bit (bit 31), Lock bit (bit 30) and Block Count (bit 0 to bit 14).

15.2.7 Radio firmware update feature

Boot ROM can perform a firmware update remotely. It can be used for updating main flash (CM33) as well as radio flash (CM3) firmware.

Use case example of remote radio firmware update:

1. Main flash (CM33) contains a customer's application image along with NXP delivered OTA client.
2. NXP OTA client helps in fetching image over-the-air via wireless protocol.
3. Image blocks fetched over-the-air are dumped in either internal or external flash unused regions.
4. Once the entire image is received, OTA client must indicate and provide meta data information for update to be performed in User IFR0 OTACFG page. (See OTA update configuration).
5. Application is responsible of triggering a software reset upon which boot ROM uses data provided in OTACFG page to start firmware update.
6. Upon successful update, IFR0 OTACFG page is erased and “FW update status” is updated by boot ROM and reset is triggered to follow "Normal boot path".

If the external flash is used, LPSPI1 is configured to master mode and pin assignment is showed in table below.

Table 71. LPSPI1 pin assignment when external flash used and LPSPI1 configured to master mode

Pin name	Pin assignment	Alt
LPSPI1_SCK	PTB2	2
LPSPI1_SIN	PTB1	2
LPSPI1_SOUT	PTB3	2
LPSPI1_DATA2	PTB5	2
LPSPI1_DATA3	PTB4	2
LPSPI1_SS0	PTB0	2

15.2.8 Initiate a debug session

When ROM bootloader is executing (i.e., instructions are fetched from the ROM memory address range), the debug access port (AP) of CM33 is disabled irrespective of device life-cycle state or token settings. This mechanism is referred as 'Boot-ROM protection' in this document. Thus, the method to initiate a debug session will vary depending on the device state and intended debug scenario. The scenarios described in the rest of these section are as follows:

- Flash does not contain a valid, bootable image. If the flash does not contain a valid image, the ROM bootloader will proceed to ISP mode and wait to be connected via one of its serial interfaces; in ISP mode the debug interface is disabled.
- ISP mode initiated because the ISP pin was asserted on the device at reset.
- Connection to a device running a valid application, with the intent to update flash with a new application.
- Connection to a device running a valid application in flash, with the intent to debug without updating flash (also called a "debug attach").

15.2.8.1 Debug session with invalid flash image or ISP mode

When the device boots, there may be no valid image in the boot media, at which point the ROM bootloader enters the In-System Programming (ISP) path, and debug access is disabled for security reasons. Another scenario is where the device may be placed into ISP mode because the ISP has been asserted as the device leaves reset.

To start a debug session under these scenarios, users need to enter Debugger Mailbox and send the DM-AP command Start Debug Session. Upon receiving the command, the ROM bootloader ensures any unwanted peripheral interrupts are disabled and secrets managed before enabling debug access. After configuring debug access, the ROM enters a while (1) loop. For lifecycle which does not require debug authentication, once the Start Debug Session has been successfully executed, the AP for CM33 and CM3 will be accessible and can used to set breakpoints, etc. as with other Cortex-M devices. For lifecycle which requires debug authentication, DM-AP commands Debug Auth Start and Debug Auth Response need to be done successfully before sending Start Debug Session command to ensure AP for CM33 or CM3 can be accessible.

Example below shows how a debug session is initiated at lifecycle which debug authentication is not required:

```
// Pseudo Code Syntax
// ------------------------------------------------------------------------
// WriteDP <register> <value>
// value = ReadDP <register>
// AP transactions presume the DM AP is selected
// WriteAP <register> <value>
// value = ReadAP <register> <value>
// ------------------------------------------------------------------------
// Read AP ID register to identify DM AP at index 2
WriteDP 2 0x020000F0
// The returned AP ID should be 0x002A0000
value = ReadAP 3
print "AP ID: ", value
// Select DM AP index 2
WriteDP 2 0x02000000
// Write DM RESYNC_REQ + CHIP_RESET_REQ
WriteAP 0 0x21
// Write DM START_DBG_SESSION to REQUEST register (1)
WriteAP 1 7
// Poll RETURN register (2) for zero return
value = -1
while value != 0 {
        value = (ReadAP 2 & 0xFFFF)
}
print "DEBUG_SESSION_REQ: ", value
```

15.2.8.2 Debug session with valid application in flash

In this scenario, ROM bootloader configures debug access before jumping to the application in flash. If the device is at some lifecycle debug authentication is not required and debug has not been disabled by an application already in flash, the APs of CM33 and radio (CM3) are accessible, a debug session can be initialized without any other operation. If debug authentication is required, entering Debugger Mailbox, doing the debug authentication successfully and exiting Debugger Mailbox need to be done before ROM bootloader booting the application. By doing so, users can initialize a debug session after ROM bootloader jumping to the application in flash.

15.2.9 Clock

15.2.9.1 Clock gating

The heart of the clocking architecture is the System Clock Generator (SCG) module. The SCG controls multiple clock sources (FIRC/SIRC/SOSC/ROSC/CORECLK/BUSCLK/SLOWCLK) that can then be distributed to the Module Reset and Clock Control (MRCC) module. Clock gating of modules helps users to only consume power for modules that are needed for their end application. The clock to each module can be gated on or off via a programmable register.

Each peripheral has independent clock gating for both the peripheral interface clock and the peripheral functional clock. This clock gating is controlled via MRCC module. MRCC provides a clock select field, a clock disable, a divider of the selected clock source, and a peripheral reset bit that provides independent resetting of the peripheral. MRCC routes interface clocks and functional clocks for all peripherals.

The CPU_CLK, BUS_CLK, and SLOW_CLK are always enabled in the various RUN and Wait modes. If these clocks are not used in low-power stop modes, they will be disabled via the Core Mode Controller (CMC) and System Power Controller (SPC) modules.

15.2.9.2 Module/Peripheral clocking

Most peripheral clocks by default are disabled, ROM will un-gate the peripheral clocks as needed by the boot flow and also enables functional clock.

Here are some of the clocks used by ROM and their setup values:

- TRDC CPU_CLK
- CRC0 BUS_CLK
- LPSPI1 BUS_CLK
- LPUART1 BUS_CLK
- LPI2C1 BUS_CLK
- CAN BUS_CLK
- ELE CPU_CLK
- GPIO{B/C} CPU_CLK
- PORT{A/B} CPU_CLK

Chapter 16
ROM ISP

16.1 Overview

ROM bootloader provides in-system programming utility that operates over a serial connection on the MCUs. It enables quick and easy programming of MCUs through the entire product lifecycle, including application development, final product manufacturing, and beyond. Host-side command line tool is available to communicate with the bootloader. Users can utilize host tools to upload/download application code and do manufacturing via the bootloader.

When ROM bootloader enters ISP mode and it will auto-detect activity on the LPI2C/LPSPI/LPUART or CAN interface. The auto-detect looks for activity on the LPUART, LPI2C, LPSPI and CAN interface and selects the appropriate interface once a properly formed frame is received. If an invalid frame is received, the data is discarded and scanning resumes.

One method to make ROM bootloader go to ISP path is by pressing the BOOT_CONFIG pin (PTA4) since BOOT_CONFIG pin is enabled by default in User IFR (IFR0). See ROM bootloader configuration for details of config BOOT_CONFIG pin. In addition, ROM bootloader will go to ISP path when no boot image is found, PC and SP are not valid and some other boot failure conditions.

16.2 Available peripherals

Table below shows the peripheral instances and pin assignments used by ISP.

Table 72. Peripheral instances and pin assignments used by ISP

Peripheral instances	Pin name	Pin assignment	Alt
LPUART1	LPUART1_TX	PTC3	3
	LPUART1_RX	PTC2	3
LPSPI1	LPSPI1_PCS0	PTB0	2
	LPSPI1_SIN	PTB1	2
	LPSPI1_SCK	PTB2	2
	LPSPI1_SOUT	PTB3	2
CAN0	CAN0_TX	PTC4	3
	CAN0_RX	PTC5	3
LPI2C1	LPI2C1_SCL	PTB5	4
	LPI2C1_SDA	PTB4	4
Boot pin	BOOT_CONFIG	PTA4	Default

16.3 Available ISP commands

ISP commands availability is controlled by lifecycle. Table below shows the ISP commands available for each lifecycle. Details of several commands can be found in sections below.

Table 73. Available ISP commands for different lifecycle

Command	Description	At OEM Open	After OEM Open
Reset	Reset the device	Available	Available

Table continues on the next page...

Table 73. Available ISP commands for different lifecycle (continued)

Command	Description	At OEM Open	After OEM Open
get-property <tag>	Query about various properties and settings	Available	Available
set-property <tag> <value>	Change properties or options in ROM Bootloader	Available	Available
receive-sb-file <file>	Receive a file in Secure Binary (SB) format	Available	Available
flash-erase-region	Erase one or more sectors of the flash memory	CM33 only	NA
flash-erase-all [<memoryID>]	Erase the entire flash memory specified by memoryID	CM33 only	NA
read-memory <addr> <byte_count> [<file>]	Read memory at specified address	CM33 only	NA
write-memory <addr> [<file> | {{<hex-data>}}]	Write memory at specified address from file or string of hex values Note: When write to SRAM, make sure the length of file or hex-data is 4-byte aligned	CM33 only	NA
fill-memory <addr> <byte_count> <pattern> [word | short | byte]	Fill memory with pattern; pattern size can be word(default), short or byte	CM33 only	NA
fuse-program <index> [<file> | {{<hex-data>}}]	Program fuse at the specified index from file or string of hex values	Available	NA
fuse-read <index> <byte_count> [<file>]	Read fuse from the specified index	Available	NA
Execute <address> <arg> <stackpointer>	Jumps to code at the provided address and does not return to the ROM bootloader	Available	NA

16.3.1 Available properties

Table below shows the available properties (tag) for this device.

Table 74. Supported properties in GetProperty and SetProperty

Name	Writable	Tag value	Size in Bytes	Description
CurrentVersion	no	1	4	The current bootloader version.
AvailablePeripherals	no	2	4	The set of peripherals supported on this chip.
FlashStartAddress	no	3	4	Start address of program flash.
FlashSizeInBytes	no	4	4	Program flash size in bytes.
FlashSectorSize	no	5	4	The size of one sector of program flash in bytes.
FlashBlockCount	no	6	4	The number of blocks of the on-chip flash.
AvailableCommands	no	7	4	The set of commands supported by the bootloader.
CRCCheckStatus	no	8	4	The status of the application CRC check.

Table continues on the next page...

Table 74. Supported properties in GetProperty and SetProperty (continued)

Name	Writable	Tag value	Size in Bytes	Description
VerifyErase	yes	10	4	Controls whether the bootloader verifies erase to flash. The VerifyErase feature is enabled by default: 1 = Enable 0 = Disable
MaxPacketSize	no	11	4	Maximum supported packet size for the currently active peripheral interface.
ReservedRegions	no	12	4	List of memory regions reserved by the bootloader. Returned as value pairs (<startaddress-ofregion>, <end-addressof-region>).
RAMStartAddress	no	14	4	Start address of RAM
RAMSizeInBytes	no	15	4	RAM size in bytes.
SystemDeviceId	no	16	4	System Device ID
SecurityState	no	17	4	Security status
UniqueDeviceId	no	18	16	Unique device identification
BootStatus	no	20	4	Value of Boot Status Register
LoadableFWVersion	no	21	4	ELE loadable firmware version
FuseProgramVoltage	yes	22	4	Control the System LDO VDD Regulator Voltage Level. To program fuse, System LDO VDD regulator level needs to be regulated to Over Drive Voltage (2.5 V). The default System LDD VDO Regulator Voltage Level is regulated to Normal Voltage (1.8 V). 0 = System LDO VDO Regulator Voltage Level is related to Normal Voltage (1.8 V) 1 = System LDO VDD regulator level is regulated to Over Drive Voltage (2.5 V)
TargetVersion	no	24	4	Target version

16.3.2 Fuse reading and programming

A set of fuses can be read / programmed using fuse-program and fuse-read commands. The index of fuses can be found in Lifecycle and fuses.

Please note that -- set-property 22 1 needs to be used before -- fuse-program command to ensure fuse can be programmed successfully. And -- set-property 22 0 needs to be done after all the fuse programming completes.

16.3.3 Commands with limitation

At OEM Open lifecycle, flash-erase-region/ flash-erase-all/ read-memory/ write-memory/ fill-memory commands can be used to Program Flash (CM33) with no restriction. But for radio flash, the accessibility is controlled by Token. If Token is owned by OEM1,

these commands can be used to read/write/erase radio flash. Otherwise, these commands return error when trying to access radio flash.

Command flash-erase-all can have memoryID as a parameter, if no ID is specified or 0 is specified, the entire Program Flash will be erased. If 2 is specified as memoryID, the entire radio Program Flash will be erased. If 3 is specified as memoryID, the entire radio User IFR (IFR0) will be erased. See Table 96 for details about memory ID.

16.3.4 receive-sb-file

receive-sb-file command is used to do image update on CM33 flash or radio (CM3) flash when security is enforced. It receives a secure binary (SB) file, decrypt, authenticate and program the image to the target memory. CUST_PROD_OEMFW_ENC_SK fuse needs to be programmed correctly to ensure this command can be used successfully. Please note that -- set-property 22 1 needs to be used before receive-sb-file command if the sb file contains "programFuses" command. And -- set-property 22 0 needs to be done after receive-sb-file is done.

16.4 In-System Programming protocol

This section explains the general protocol for the packet transfers between the host and the ROM Bootloader. The description includes the transfer of packets for different transactions, such as commands with no data phase and commands with incoming or outgoing data phase. The next section describes various packet types used in a transaction.

Each command sent from the host is replied to with a response command.

Commands may include an optional data phase.

- If the data phase is incoming (from the host to the bootloader), it is part of the original command.
- If the data phase is outgoing (from the bootloader to host), it is part of the response command.

16.4.1 Command with no data phase

The protocol for a command with no data phase contains:

- Command packet (from the host)
- Generic response command packet (to host)

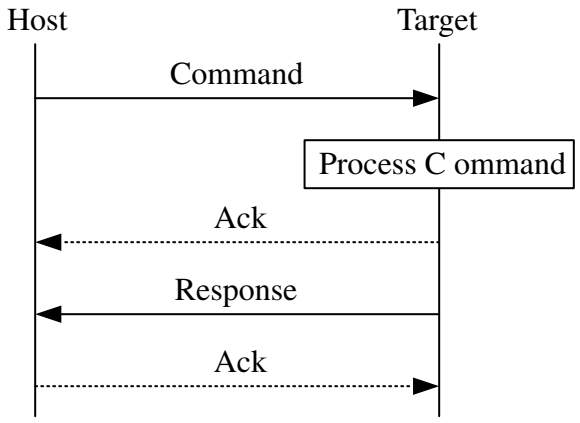


Figure 31. Command with no data phase

NOTE

In these diagrams, the ACK sent in response to a command or a data packet can arrive at any time before, during, or after the command or data packet has processed.

16.4.2 Command with incoming data phase

The protocol for a command with incoming data phase contains:

- Command packet (from host) (kCommandFlag_HasDataPhase set).
- Generic response command packet (to host).
- Incoming data packets (from the host).
- Generic response command packet.

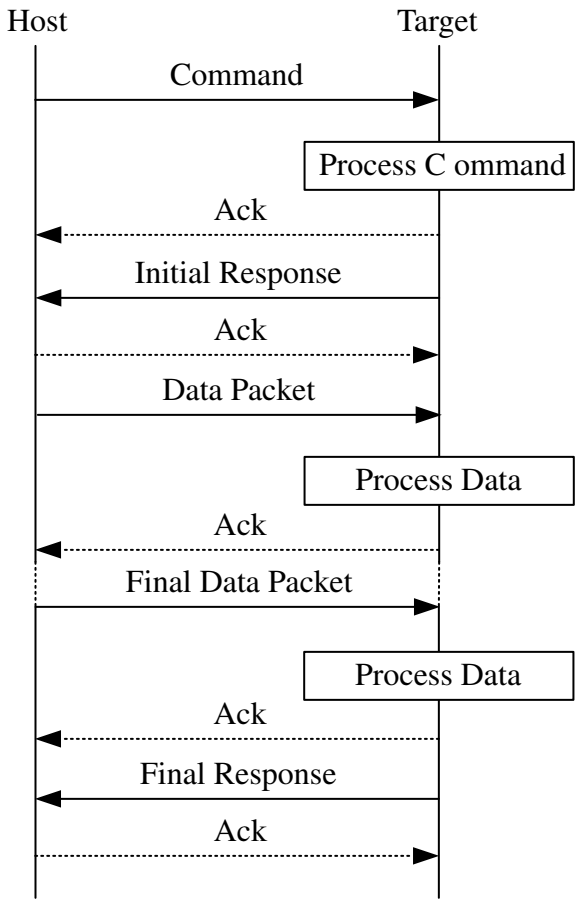


Figure 32. Command with incoming data phase

NOTE

- The host may not send any further packets while it is waiting for the response to a command.
- The data phase is aborted if the Generic Response packet prior to the start of the Data phase does not have a status of kStatus_Success.
- Data phases may be aborted by the receiving side by sending the final GenericResponse early with a status of kStatus_AbortDataPhase. The host may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet sent after the data phase includes the status of the entire operation.

16.4.3 Command with outgoing data phase

The protocol for a command with an outgoing data phase contains:

- Command packet (from the host).

- ReadMemory Response command packet (to host) (kCommandFlag_HasDataPhase set).
- Outgoing data packets (to host).
- Generic response command packet (to host).

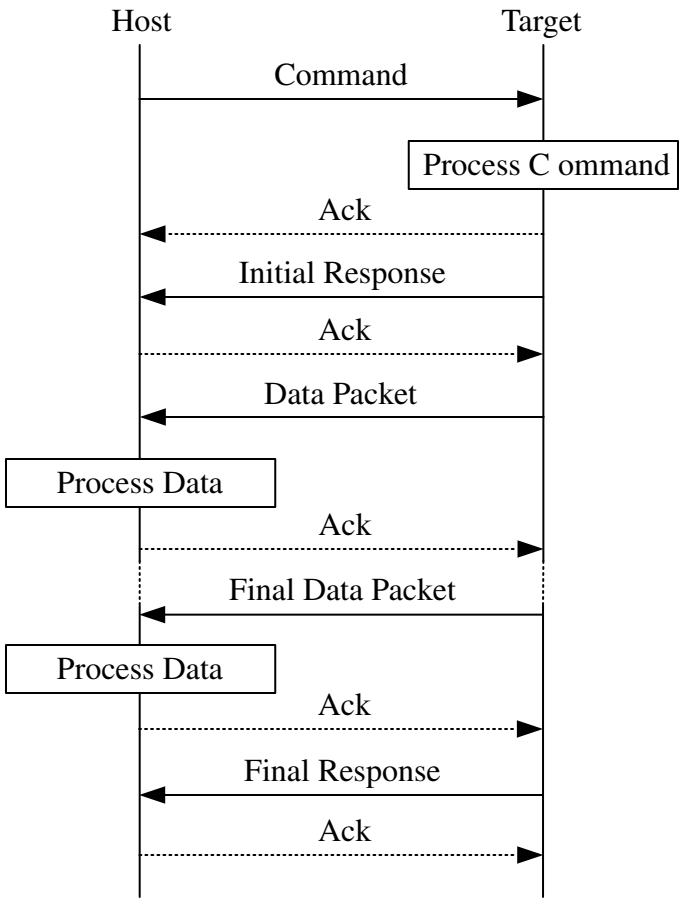


Figure 33. Command with outgoing data phase

NOTE

- The data phase is considered part of the response command for the outgoing data phase sequence. The host may not send any further packets while the host is waiting for the response to a command.
- The data phase is aborted if the ReadMemory Response command packet, prior to the start of the data phase, does not contain the kCommandFlag_HasDataPhase flag. Data phases may be aborted by the host sending the final Generic Response early with a status of kStatus_AbortDataPhase. The sending side may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet sent after the data phase includes the status of the entire operation.

16.5 ISP packet type

The bootloader device works in slave mode. All data communications are initiated by a host, which is either a PC or an embedded host. The bootloader device is the target, which receives a command or data packet. All data communications between host and target are packetized.

There are six types of packets used:

- Ping packet.

- Ping Response packet.
- Framing packet.
- Command packet.
- Data packet.
- Response packet.

All fields in the packets are in little-endian byte order.

16.5.1 Ping packet

The Ping packet is the first packet sent from a host to the target to establish a connection on the selected peripheral in order to run autobaud detection. The Ping packet can be sent from host to target at any time that the target is expecting a command packet. If the selected peripheral is LPUART, a ping packet must be sent before any other communications. For other serial peripherals, it is optional.

In response to a Ping packet, the target sends a Ping Response packet, discussed in later sections.

Table 75. Ping packet format

Byte #	Value	Name
0	0x5A	start byte
1	0xA6	ping

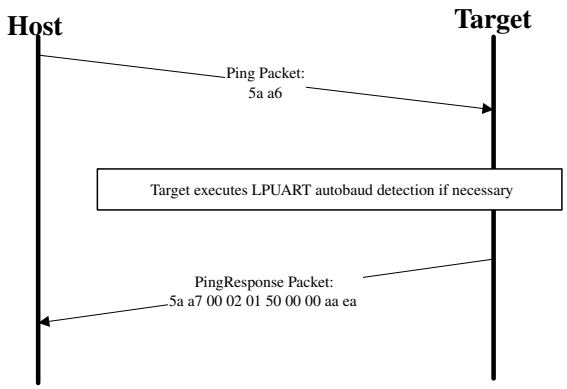


Figure 34. Ping packet protocol sequence

16.5.2 Ping response packet

The target sends a Ping Response packet back to the host after receiving a Ping packet. If communication is over an LPUART peripheral, the target uses the incoming Ping packet to determine the baud rate before replying with the Ping Response packet. Once the Ping Response packet is received by the host, the connection is established, and the host starts sending commands to the target.

Table 76. ping response packet format

Byte #	Value	Parameter
0	0x5A	Start byte
1	0xA7	Ping response code

Table continues on the next page...

Table 76. ping response packet format (continued)

Byte #	Value	Parameter
2	0x00	Protocol bugfix
3	0x03	Protocol minor
4	0x01	Protocol major
5	0x50	Protocol name = 'P' (0x50)
6	0x00	Options low
7	0x00	Options high
8	0xaa	CRC16 low
9	0xea	CRC16 high

The Ping Response packet can be sent from host to target any time the target expects a command packet. For the LPUART peripheral, it must be sent by the host when a connection is first established, in order to run outbound. For other serial peripherals, it is optional but recommended to determine the serial protocol version. The version number is in the same format as the bootloader version number returned by the GetProperty command.

16.5.3 Framing packet

The framing packet is used for flow control and error detection for the communications links that do not have such features built-in. The framing packet structure sits between the link layer and the command layer. It wraps command and data packets as well.

Every framing packet containing data sent in one direction results in a synchronizing response framing packet in the opposite direction.

The framing packet described in this section is used for serial peripherals including the LPUART, LPI2C, and LPSPI.

Table 77. Framing packet format

Byte #	Value	Parameter	Remark
0	0x5A	Start byte	
1		packetType	
2		length_low	Length is a 16-bit field that specifies the entire command or data packet size in bytes.
3		length_high	
4		crc16_low	This is a 16-bit field. The CRC16 value covers entire framing packet, including the start byte and command or data packets, but does not include the CRC bytes. See the CRC16 algorithm after this table.
5		crc16_high	
6 . . . n		Command or Data packet payload	

A special framing packet that contains only a start byte and a packet type is used for synchronization between the host and target.

Table 78. Special framing packet format

Byte #	Value	Parameter
0	0x5A	Start byte
1	0xA*n*	packetType

The Packet Type field specifies the type of the packet from one of the defined types (below):

Table 79. packetType field

packetType	Name	Description
0xA1	kFramingPacketType_Ack	The previous packet was received successfully; the sending of more packets is allowed.
0xA2	kFramingPacketType_Nak	The previous packet was corrupted and must be re-sent.
0xA3	kFramingPacketType_AckAbort	Data phase is being aborted.
0xA4	kFramingPacketType_Command	The framing packet contains a command packet payload.
0xA5	kFramingPacketType_Data	The framing packet contains a data packet payload.
0xA6	kFramingPacketType_Ping	Sent to verify the other side is alive. Also used for LPUART autobaud.
0xA7	kFramingPacketType_PingResponse	A response to Ping; contains the framing protocol version number and options.

16.5.4 CRC16 algorithm

This section provides the CRC16 algorithm.

The CRC is computed over each byte in the framing packet header, excluding the crc16 field itself, plus all of the payload bytes. The CRC algorithm is the XMODEM variant of CRC-16.

Table 80. Characteristics of the XMODEM variant

Variant	Values
Width	16
Polynomial	0x1021
Init value	0x0000
Reflect in	False
Reflect out	False
xor out	0x0000
Check result	0x31c3

The check result is computed by running the ASCII character sequence "123456789" through the algorithm.

```
uint16_t crc16_update(const uint8_t * src, uint32_t lengthInBytes)
{
   uint32_t crc = 0;
   uint32_t j;
   for (j=0; j < lengthInBytes; ++j)
   {
        uint32_t i;
    uint32_t byte = src[j];
     crc ^= byte << 8;
     for (i = 0; i < 8; ++i)
      {
         uint32_t temp = crc << 1;
         if (crc & 0x8000)
          {
              temp ^= 0x1021;
          }
          crc = temp;
      }
   }
   return crc;
}
```

16.5.5 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 81. Command packet format (32 bytes)

Command Header (4 bytes)				28 bytes for Parameters (Max 7 parameters)						
Tag	Flags	Rsvd	Param Count	Param1 (32-bit)	Param2 (32-bit)	Param3 (32-bit)	Param4 (32-bit)	Param5 (32-bit)	Param6 (32-bit)	Param7 (32-bit)
Byte 0	Byte 1	Byte 2	Byte 3	-	-	-	-	-	-	-

Table 82. Command header format

Byte #	Command Header Field	Remark
0	Command or Response tag	The command header is 4 bytes long, with these fields.
1	Flags	
2	Reserved. Should be 0x00.	
3	ParameterCount	

The header is followed by 32-bit parameters up to the value of the ParameterCount field specified in the header. Because a command packet is 32 bytes long, only 7 parameters can fit into the command packet.

Command packets are also used by the target to send responses back to the host. As mentioned earlier, command packets and data packets are embedded into framing packets for all the transfers.

Table 83. Command tags

Command Tag	Name	Remark
0x01	FlashEraseAll	The command tag specifies one of the commands supported by the bootloader. The valid command tags for the bootloader are listed here.
0x02	FlashEraseRegion	
0x03	ReadMemory	
0x04	WriteMemory	
0x05	FillMemory	
0x06	Reserved	
0x07	GetProperty	
0x08	ReceiveSbFile	
0x09	Execute	
0x0A	Reserved	
0x0B	Reset	
0x0C	SetProperty	
0x0D	Reserved	
0x0E	Reserved	
0x0F	Reserved	
0x10	Reserved	
0x11	Reserved	
0x12	Reserved	
0x13	Reserved	
0x14	FuseProgram	
0x15	Reserved	
0x16	Reserved	
0x17	FuseRead	

Table 84. Response tags

Response Tag	Name	Remark
0xA0	GenericResponse	The response tag specifies one of the responses the bootloader (target) returns to the host. The valid response tags are listed here.
0xA3	ReadMemoryResponse	
0xA7	GetPropertyResponse (used for sending responses to GetProperty command only)	
0xA3	ReadMemoryResponse (used for sending responses to ReadMemory command only)	
0xAF	FlashReadOnceResponse (used for sending responses to FlashReadOnce command only)	
0xB5	KeyProvisionResponse	

Flags: Each command packet contains a Flag byte. Only bit 0 of the flag byte is used. If bit 0 of the flag byte is set to 1, then data packets follow the command sequence. The number of bytes that are transferred in the data phase is determined by a command specific parameter in the parameters array.

ParameterCount: The number of parameters included in the command packet.

Parameters: The parameters are word-length (32 bits). With the default maximum packet size of 32 bytes, a command packet can contain up to 7 parameters.

16.5.6 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses include:

- GenericResponse
- GetPropertyResponse
- ReadMemoryResponse
- FlashReadOnceResponse
- KeyProvisionResponse

GenericResponse: After the bootloader has processed a command, the bootloader sends a generic response with status and command tag information to the host. The generic response is the last packet in the command protocol sequence. The generic response packet contains the framing packet data and the command packet data (with generic response tag = 0xA0) and a list of parameters (defined in the next section). The parameter count field in the header is always set to 2, for status code and command tag parameters.

Table 85. GenericResponse parameters

Byte #	Parameter	Description
0 - 3	Status code	The Status codes are errors encountered during the execution of a command by the target. If a command succeeds, then a kStatus_Success code is returned.
4 - 7	Command tag	The Command tag parameter identifies the response to the command sent by the host.

GetPropertyResponse: The GetPropertyResponse packet is sent by the target in response to the host query that uses the GetProperty command. The GetPropertyResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a GetPropertyResponse tag value (0xA7).

The parameter count field in the header is set to greater than 1, to always include the status code and one or many property values.

Table 86. GetPropertyResponse parameters

Byte #	Parameter
0 - 3	Status code
4 - 7	Property value
. . .	. . .
	Can be up to maximum 6 property values, limited to the size of the 32-bit command packet and property type.

ReadMemoryResponse: The ReadMemoryResponse packet is sent by the target in response to the host sending a ReadMemory command. The ReadMemoryResponse packet contains the framing packet data and the command

packet data, with the command/response tag set to a ReadMemoryResponse tag value (0xA3), the flags field set to kCommandFlag_HasDataPhase (1).

The parameter count set to 2 for the status code and the data byte count parameters shown below.

Table 87. ReadMemoryResponse parameters

Byte #	Parameter	Description
0 - 3	Status code	The status of the associated Read Memory command.
4 - 7	Data byte count	The number of bytes sent in the data phase.

FlashReadOnceResponse: The FlashReadOnceResponse packet is sent by the target in response to the host sending a FlashReadOnce command. The FlashReadOnceResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a FlashReadOnceResponse tag value (0xAF), and the flags field set to 0. The parameter count is set to 2 plus the number of words requested to be read in the FlashReadOnceCommand.

Table 88. FlashReadOnceResponse parameters

Byte #	Parameter
0 – 3	Status Code
4 – 7	Byte count to read
…	…
	Can be up to 20 bytes of requested read data.

The KeyProvisionResponse packet is sent by the target in response to the host sending a KeyProvision command. The KeyProvisionResponse packet contains the framing packet data and command packet data, with the command/response tag set to a KeyProvisionResponse tag value (0xB5), and the flags field set to kCommandFlag_HasDataPhase (1).

Table 89. KeyProvisionResponse parameters

Byte #	Parameter
0 – 3	Status Code
4 – 7	Data byte count

16.6 Bootloader command set

All bootloader commands follow the command packet format wrapped by the framing packet as explained in previous sections.

See Available ISP commands for different lifecycle for a list of commands supported by the bootloader.

16.6.1 GetProperty command

The GetProperty command is used to query the bootloader about various properties and settings. Each supported property has a unique 32-bit tag associated with it. The tag occupies the first parameter of the command packet. The target returns a GetPropertyResponse packet with the property values for the property identified with the tag in the GetProperty command.

Properties are the defined units of data that can be accessed with the GetProperty or SetProperty commands. Properties may be read-only or read-write. All read-write properties are 32-bit integers, so they can easily be carried in a command parameter.

The 32-bit property tag is the only parameter required for GetProperty command.

Table 90. Parameters for GetProperty Command

Byte #	Command
0 - 3	Property tag See Available properties for more details.
4 - 7	External Memory Identifier (only applies to get property for external memory)

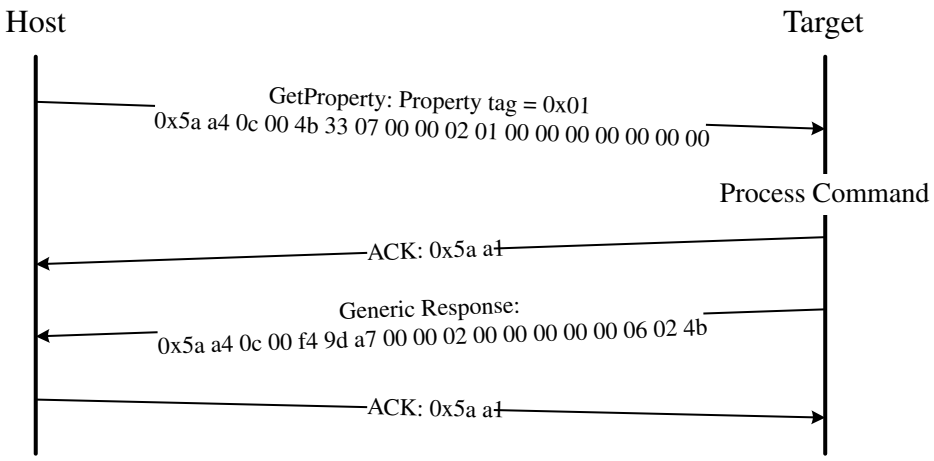


Figure 35. Protocol Sequence for GetProperty Command

Table 91. GetProperty Command Packet Format (Example)

GetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x4B 0x33
Command packet	commandTag	0x07 – GetProperty
	flags	0x00
	reserved	0x00
	parameterCount	0x02

Table continues on the next page...

Table 91. GetProperty Command Packet Format (Example) (continued)

	propertyTag	0x00000001 - CurrentVersion
	Memory ID	0x00000000 - Internal Flash

The GetProperty command has no data phase.

Response: In response to a GetProperty command, the target sends a

GetPropertyResponse packet with the response tag set to 0xA7. The parameter count indicates the number of parameters sent for the property values, with the first parameter showing status code 0, followed by the property value(s). The next table shows an example of a GetPropertyResponse packet.

Table 92. GetProperty Response Packet Format (Example)

GetPropertyResponse	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0c 0x00 (12 bytes)
	crc16	0x07 0x7a
Command packet	responseTag	0xA7
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	status	0x00000000
	propertyValue	0x0000014b - CurrentVersion

16.6.2 SetProperty command

The SetProperty command is used to change or alter the values of the properties or options of the bootloader. The command accepts the same property tags used with the GetProperty command. However, only some properties are writable--see Available properties. If an attempt to write a read-only property is made, an error is returned indicating the property is read-only and cannot be changed.

The property tag and the new value to set are the two parameters required for the SetProperty command.

Table 93. Parameters for SetProperty Command

Byte #	Command
0 - 3	Property tag See Available properties for more details.
4 - 7	Property value

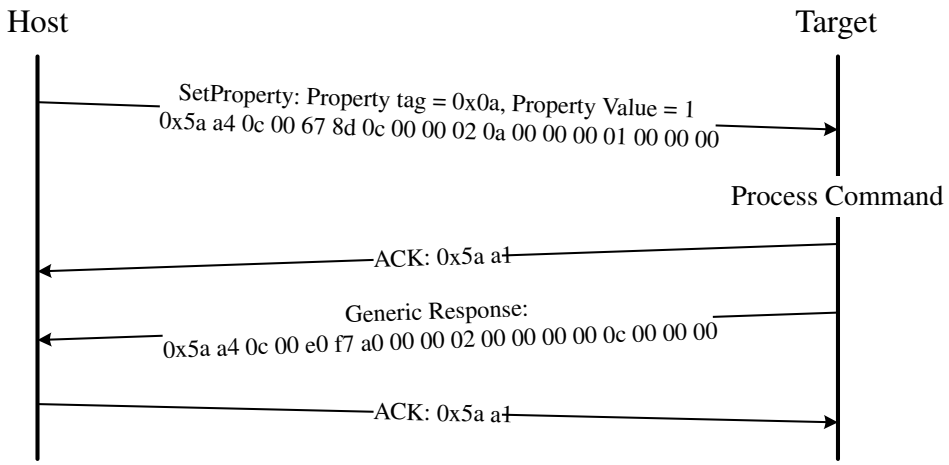


Figure 36. Parameters for SetProperty Command

Table 94. SetProperty Command Packet Format (Example)

SetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x67 0x8D
Command packet	commandTag	0x0C – SetProperty with property tag 10
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	propertyTag	0x0000000A - VerifyWrites
	propertyValue	0x00000001

The SetProperty command has no data phase.

Response: The target returns a GenericResponse packet with one of following status codes:

Table 95. SetProperty Response Status Codes

Status Code
kStatus_Success
kStatus_ReadOnly
kStatus_UnknownProperty
kStatus_InvalidArgument

16.6.3 FlashEraseAll command

The FlashEraseAll command performs an erase of the entire flash memory. If any flash regions are protected, then the FlashEraseAll command fails and returns an error status code. The Command tag for FlashEraseAll command is 0x01 set in the commandTag field of the command packet.

The FlashEraseAll command requires memory ID. If memory ID is not specified, the internal flash (memory ID =0) will be selected as default.

Table 96. Parameter for FlashEraseAll Command

Byte #	Parameter	
	Memory ID	Descriptions
0-3	0x000	Internal Flash
	0x002	Radio PFlash
	0x003	Radio User IFR

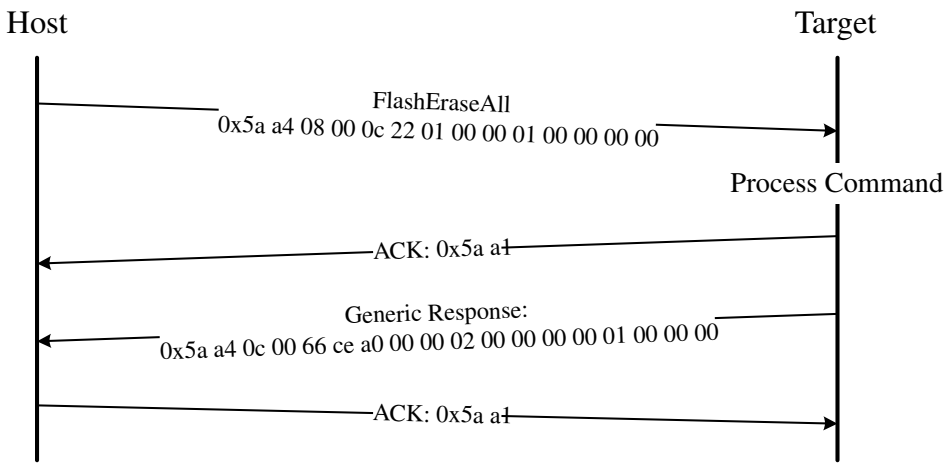


Figure 37. Protocol Sequence for FlashEraseAll Command

Table 97. FlashEraseAll Command Packet Format (Example)

FlashEraseAll	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x08 0x00
	crc16	0x0C 0x22
Command packet	commandTag	0x01 - FlashEraseAll
	flags	0x00
	reserved	0x00
	parameterCount	0x01
	Memory ID	refer to the above table

The FlashEraseAll command has no data phase.

Response: The target returns a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command or set to an appropriate error status code.

16.6.4 FlashEraseRegion command

The FlashEraseRegion command performs an erase of one or more sectors of the flash memory.

The start address, and number of bytes are the 2 parameters required for the FlashEraseRegion command. The start and byte count parameters must be 32-byte aligned ([3:0] = 0000), or the FlashEraseRegion command fails and returns kStatus_FlashAlignmentError (101). If the region specified does not fit in the flash memory space, the FlashEraseRegion command fails and returns kStatus_FlashAddressError (102). If any part of the region specified is protected, the FlashEraseRegion command fails and returns kStatus_MemoryRangeInvalid (10200).

Table 98. Parameters for FlashEraseRegion Command

Byte #	Parameter
0 - 3	Start address
4 - 7	Byte count
8 - 11	Memory ID

The FlashEraseRegion command has no data phase.

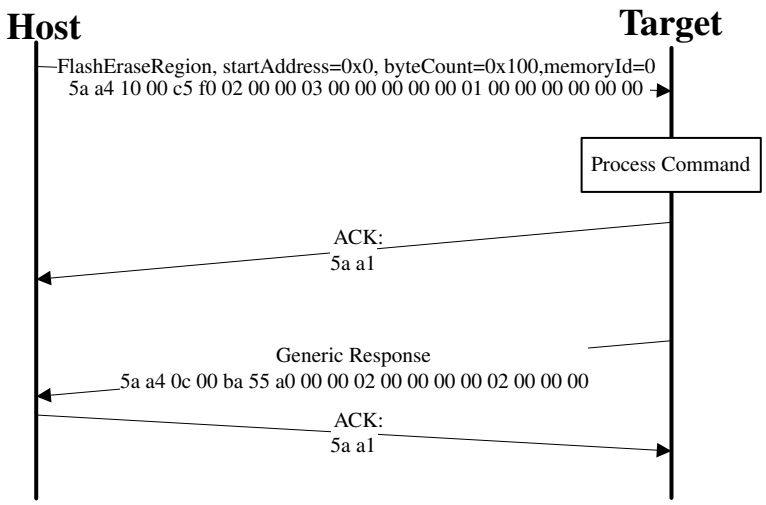


Figure 38. Protocol Sequence for FlashEraseRegion Command

Response: The target returns a GenericResponse packet with one of following error status codes.

Table 99. FlashEraseRegion Response Status Codes

Status Code
kStatus_Success (0)
kStatus_MemoryRangeInvalid (10200)
kStatus_FlashAlignmentError (101)
kStatus_FlashAddressError (102)
kStatus_FlashAccessError (103)
kStatus_FlashCommandFailure (105)

16.6.5 ReadMemory command

The ReadMemory command returns the contents of memory at the given address, for a specified number of bytes. This command can read any region of memory accessible by the CPU and not protected by security.

The start address, and number of bytes are the two parameters required for ReadMemory command. The memory ID is optional. Internal memory will be selected as default if memory ID is not specified.

Table 100. Parameters for read memory command

Byte	Parameter	Description
0-3	Start address	Start address of memory to read from
4-7	Byte count	Number of bytes to read and return to caller
8-11	Memory ID	Internal or external memory Identifier

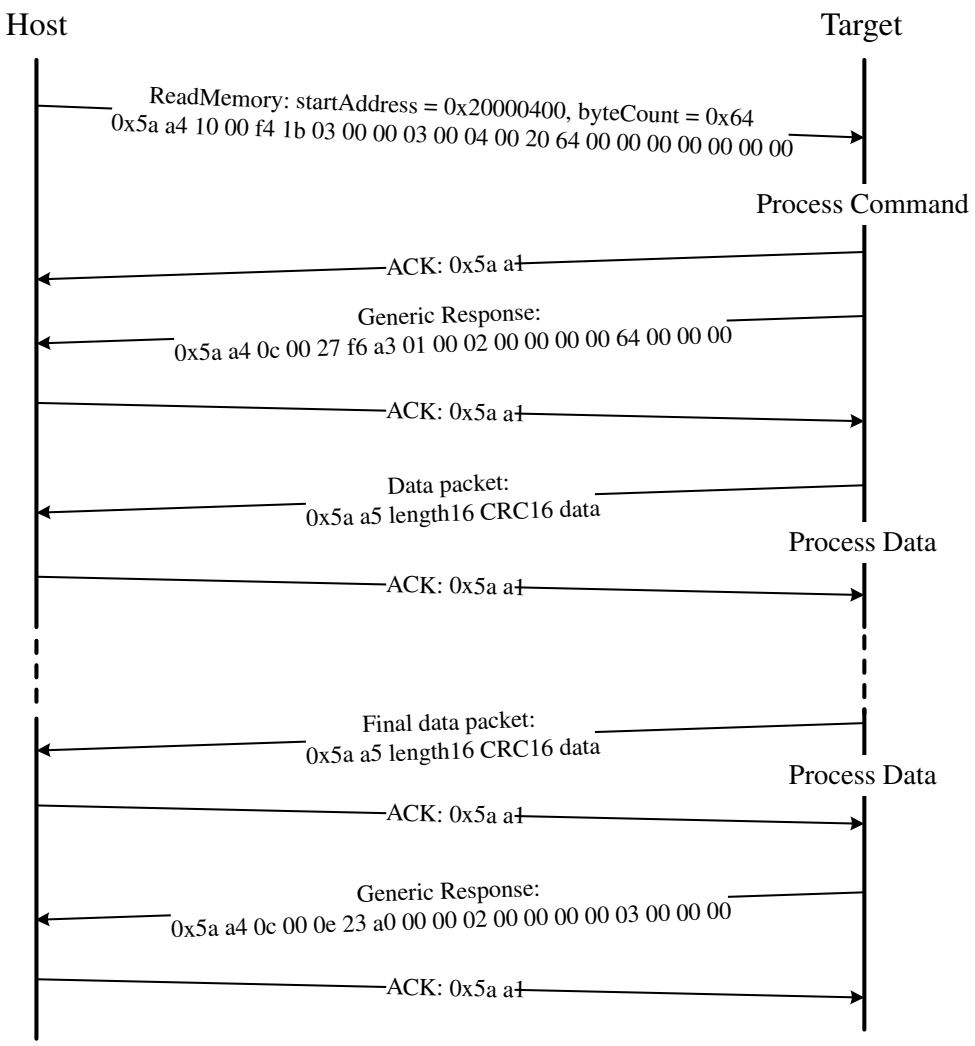


Figure 39. Command sequence for ReadMemory

Table 101. ReadMemory Command Packet Format (Example)

ReadMemory	Parameter	Value
Framing packet	Start byte	0x5A0xA4,
	packetType	kFramingPacketType_Command
	length	0x10 0x00
	crc16	0xF4 0x1B
Command packet	commandTag	0x03 - readMemory

Table continues on the next page...

Table 101. ReadMemory Command Packet Format (Example) (continued)

	flags	0x00
	reserved	0x00
	parameterCount	0x03
	startAddress	0x20000400
	byteCount	0x00000064
	memoryID	0x0

Data Phase: The ReadMemory command has a data phase. Because the target works in slave mode, the host needs to pull data packets until the number of bytes of data specified in the byteCount parameter of ReadMemory command are received by host.

Response: The target returns a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command or set to an appropriate error status code.

16.6.6 WriteMemory command

The WriteMemory command writes data provided in the data phase to a specified range of bytes in memory (flash or RAM). However, if flash protection is enabled, then writes to protected sectors fail.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll or FlashEraseRegion.
- First, any flash sector written to must have been previously erased with a FlashEraseAll or FlashEraseRegion command.
- Writing to flash requires the start address to be 16-byte aligned ([3:0] = 0000).
- The byte count is rounded up to a multiple of 4, and trailing bytes are filled with the flash erase pattern (0xff).
- If the VerifyWrites property is set to true, then writes to flash also performs a flash verify program operation.

When writing to RAM, the start address does not need to be aligned, and the data is not padded.

The start address and number of bytes are the 2 parameters required for WriteMemory command. The memory ID is optional. Internal memory will be selected as default if memory ID is not specified.

Table 102. Parameters for WriteMemory Command

Byte #	Command
0 - 3	Start address
4 - 7	Byte count
8 - 11	Memory ID

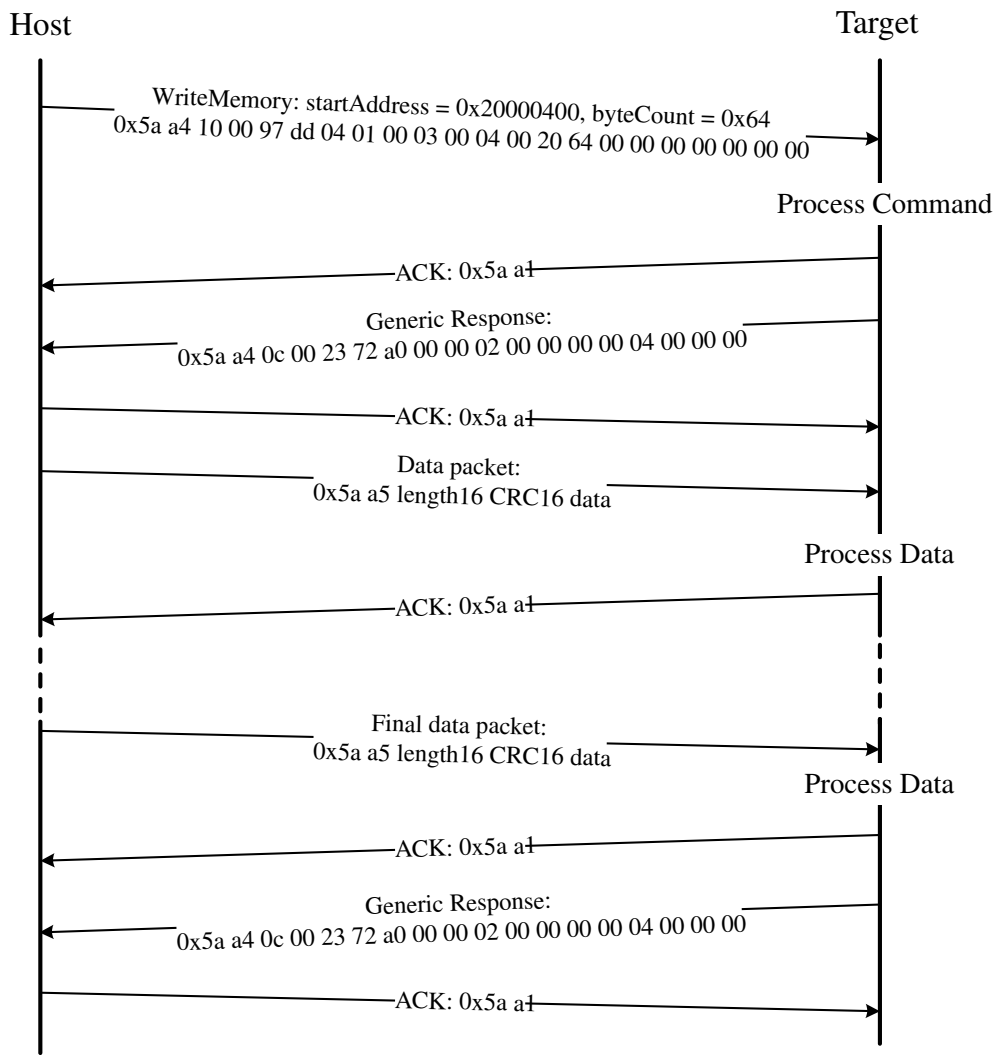


Figure 40. Protocol Sequence for WriteMemory Command

Table 103. WriteMemory Command Packet Format (Example)

WriteMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00

Table continues on the next page...

Table 103. WriteMemory Command Packet Format (Example) (continued)

	crc16	0x97 0xDD
Command packet	commandTag	0x04 - writeMemory
	flags	0x01
	reserved	0x00
	parameterCount	0x03
	startAddress	0x20000400
	byteCount	0x00000064
	memoryID	0x0

Data Phase: The WriteMemory command has a data phase; the host sends data packets until the number of bytes of data specified in the byteCount parameter of the WriteMemory command are received by the target.

Response: The target returns a GenericResponse packet with a status code set to kStatus_Success upon successful execution of the command, or to an appropriate error status code.

16.6.7 FillMemory command

The FillMemory command fills a range of bytes in memory with a data pattern. It follows the same rules as the WriteMemory command. The difference between FillMemory and WriteMemory is that a data pattern is included in FillMemory command parameter, and there is no data phase for the FillMemory command, while WriteMemory does have a data phase.

Table 104. Parameters for FillMemory Command

Byte #	Command
0 - 3	Start address of memory to fill
4 - 7	Number of bytes to write with the pattern • The start address should be 32-bit aligned. • The number of bytes must be evenly divisible by 4.
8 - 11	32-bit pattern

- To fill with a byte pattern (8-bit), the byte must be replicated 4 times in the 32-bit pattern.
- To fill with a short pattern (16-bit), the short value must be replicated 2 times in the 32-bit pattern.

For example, to fill a byte value with 0xFE, the word pattern is 0xFEFEFEFE; to fill a short value 0x5AFE, the word pattern is 0x5AFE5AFE.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll, or FlashEraseRegion command.
- First, any flash sector written to must have been previously erased with a FlashEraseAll or FlashEraseRegion command.
- Writing to flash requires the start address to be 16-byte aligned ([3:0] = 0000).
- If the VerifyWrites property is set to true, then writes to flash also performs a flash verify program operation.

When writing to RAM, the start address does not need to be aligned, and the data is not padded.

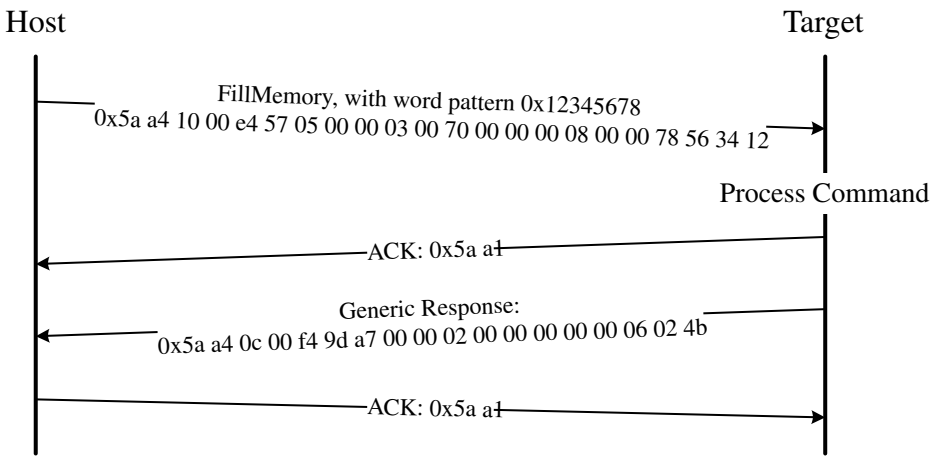


Figure 41. Protocol Sequence for FillMemory Command

Table 105. FillMemory Command Packet Format (Example)

FillMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0xE4 0x57
Command packet	commandTag	0x05 – FillMemory
	flags	0x00
	Reserved	0x00
	parameterCount	0x03
	startAddress	0x00007000
	byteCount	0x00000800
	patternWord	0x12345678

The FillMemory command has no data phase.

Response: upon successful execution of the command, the target (Kinetis bootloader) returns a GenericResponse packet with a status code set to kStatus_Success, or to an appropriate error status code.

16.6.8 Execute command

The execute command results in the bootloader setting the program counter to the code at the provided jump address, R0 to the provided argument, and a Stack pointer to the provided stack pointer address. Prior to the jump, the system is returned to the reset state.

The Jump address, function argument pointer, and stack pointer are the parameters required for the Execute command. If the stack pointer is set to zero, the called code is responsible for setting the processor stack pointer before using the stack.

Table 106. . Parameters for Execute Command

Byte #	Command
0 - 3	Jump address
4 - 7	Argument word
8 - 11	Stack pointer address

The Execute command has no data phase.

Response: Before executing the Execute command, the target validates the parameters and return a GenericResponse packet with a status code either set to kStatus_Success or an appropriate error status code.

16.6.9 Call command

The Call command executes a function that is written in memory at the address sent in the command. The address needs to be a valid memory location residing in accessible flash (internal or external) or in RAM. The command supports the passing of one 32-bit argument. Although the command supports a stack address, at this time the call still takes place using the current stack pointer. After execution of the function, a 32-bit return value is returned in the generic response message.

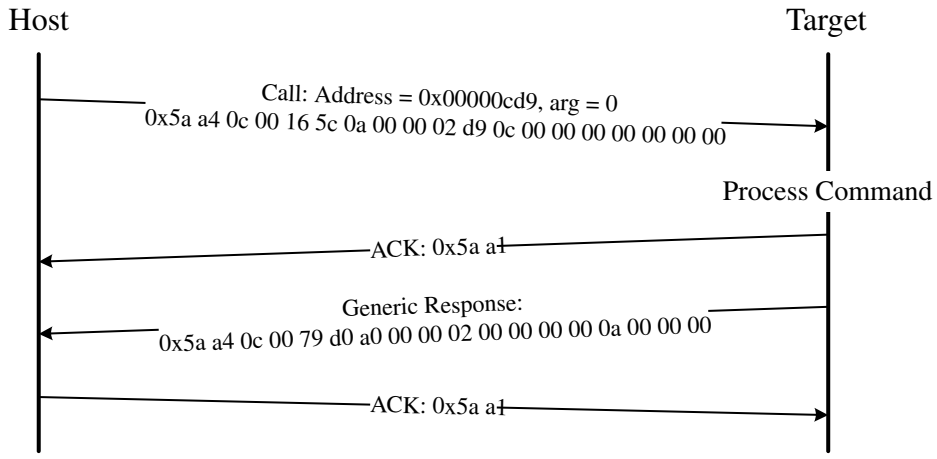


Figure 42. Protocol sequence for Call command

Table 107. Parameters for Call Command

Byte #	Command
0 - 3	Call address
4 - 7	Argument word
8 - 11	Stack pointer

Response: The target returns a GenericResponse packet with a status code either set to the return value of the function called or set to kStatus_InvalidArgument (105).

16.6.10 Reset command

The Reset command results in the bootloader resetting the chip.

The Reset command requires no parameters.

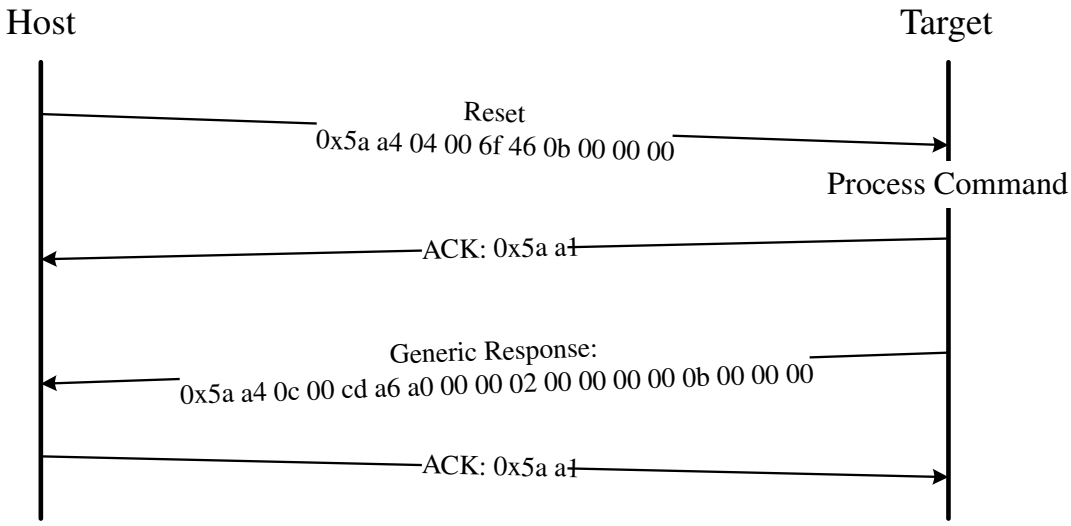


Figure 43. Protocol Sequence for Reset Command

Table 108. Reset Command Packet Format (Example)

Reset	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0x6F 0x46

Table continues on the next page...

Table 108. Reset Command Packet Format (Example) (continued)

Command packet	commandTag	0x0B - reset
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The Reset command has no data phase.

Response: The target returns a GenericResponse packet with status code set to kStatus_Success, before resetting the chip.

The reset command can also be used to switch boot from flash after successful flash image provisioning via ROM bootloader. After issuing the reset command, allow 5 seconds for the user application to start running from Flash.

16.6.11 ReceiveSBFile command

The Receive SB File command (ReceiveSbFile) starts the transfer of an SB file to the target. The command only specifies the size in bytes of the SB file that is sent in the data phase. The SB file is processed as it is received by the bootloader. See the Secure boot related sections for more details about the SB file.

Table 109. . Parameters for Receive SB File Command

Byte #	Command
0 – 3	Byte count

Data Phase: The Receive SB file command has a data phase; the host sends data packets until the number of bytes of data specified in the byteCount parameter of the Receive SB File command are received by the target.

Response: The target returns a GenericResponse packet with a status code set to the kStatus_Success upon successful execution of the command or set to an appropriate error code.

16.7 LPUART ISP

The bootloader integrates an autobaud detection algorithm for the LPUART peripheral, thereby providing flexible baud rate choices.

Autobaud feature: If LPUART*n* is used to connect to the bootloader, then the LPUART*n*_RX pin must be kept high and not left floating during the detection phase in order to comply with the autobaud detection algorithm. After the bootloader detects the ping packet (0x5A 0xA6) on LPUART*n*_RX, the bootloader firmware executes the autobaud sequence.

If the baudrate is successfully detected, then the bootloader sends a ping packet response [(0x5A 0xA7), protocol version (4 bytes), protocol version options (2 bytes) and crc16 (2 bytes)] at the detected baudrate. The Kinetis bootloader then enters a loop, waiting for bootloader commands via the LPUART peripheral.

NOTE

The data bytes of the ping packet must be sent continuously (with no more than 80 ms between bytes) in a fixed LPUART transmission mode (8-bit data, no parity bit and 1 stop bit). If the bytes of the ping packet are sent one-by-one with more than 80 ms delay between them, then the autobaud detection algorithm may calculate an incorrect baud rate. In this instance, the autobaud detection state machine should be reset.

Supported baud rates: The baud rate is closely related to the MCU core and system clock frequencies. Typical baud rates supported are 9600, 19200, 38400, 57600, 115200 and 230400.

NOTE

The peripheral baud rate mentioned here is just at ISP packet level, the actual download speed in ISP is lower than the baud rate because of overhead in the ISP protocol and packet handling.

Packet transfer: After autobaud detection succeeds, bootloader communications can take place over the LPUART peripheral. The following flow charts show:

- How the host detects an ACK from the target
- How the host detects a ping response from the target
- How the host detects a command response from the target

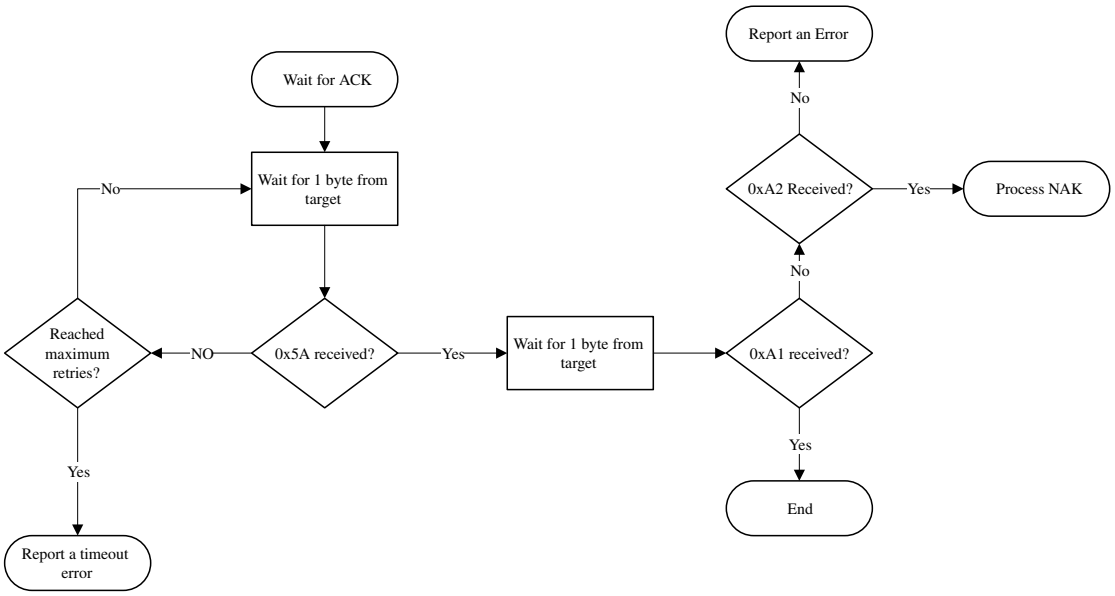


Figure 44. Host reads an ACK from target via LPUART

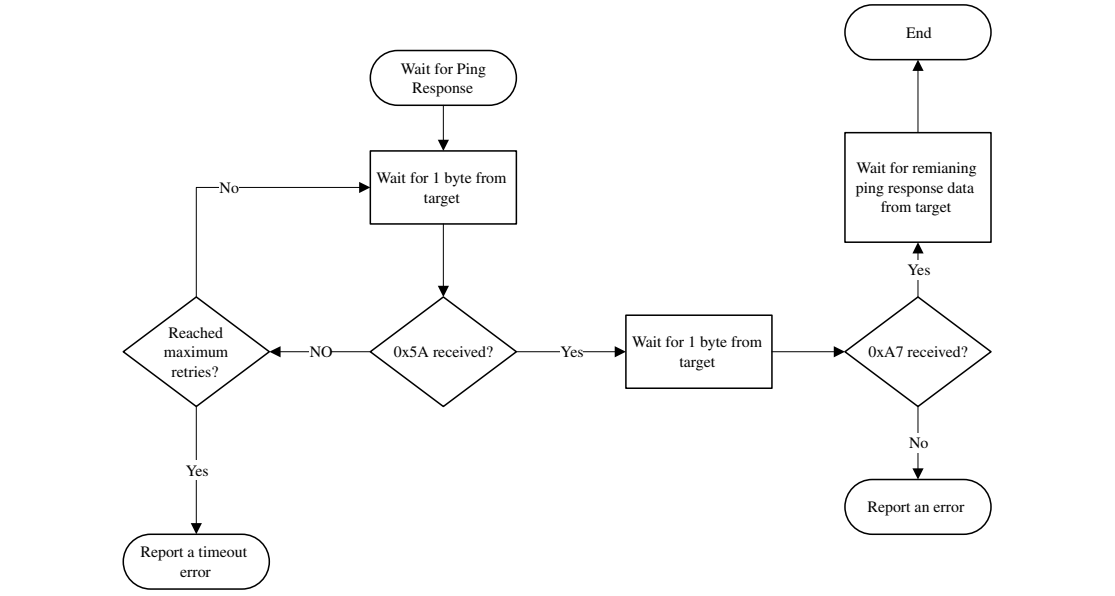


Figure 45. Host reads a ping response from target via LPUART

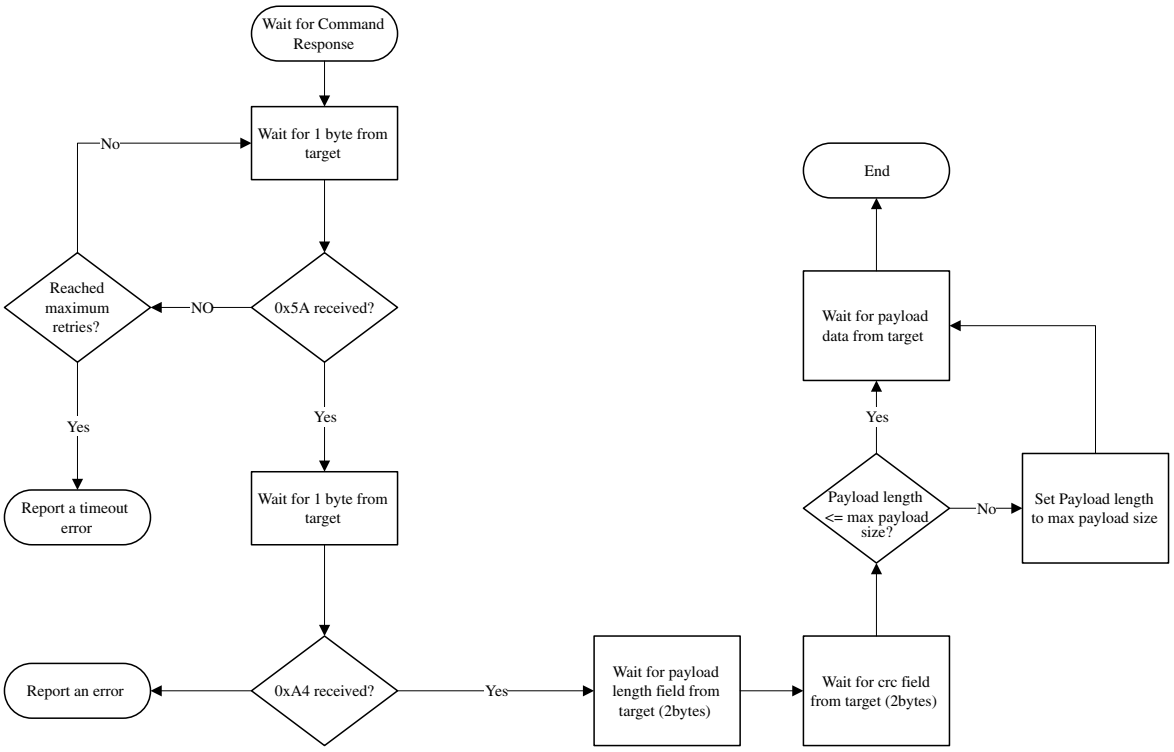


Figure 46. Host reads a command response from target via LPUART

For information on commands and formats, see ISP packet type

16.8 LPI2C ISP

The bootloader supports In-System Programming or serial boot via the LPI2C peripheral, where the LPI2C peripheral serves as the LPI2C slave. The bootloader uses 0x10 as the 7-bit LPI2C slave address and supports up to 400 kbit/s speed during transfer. The maximum supported LPI2C baud rate depends on the core clock frequency when the bootloader is running. The typical baud rate is 400 kbit/s with factory settings.

NOTE

The peripheral baud rate mentioned here is just at ISP packet level, the actual download speed in ISP is lower than the baud rate because of overhead in the ISP protocol and packet handling.

The LPI2C peripheral serves as an LPI2C slave device, hence each transfer should be started by the host, and each outgoing packet should be fetched by the host as well.

- An incoming packet is sent by the host with a selected LPI2C slave address and the direction bit is set to write.
- An outgoing packet is read by the host with a selected LPI2C slave address and the direction bit is set as read.
- 0x00 is sent as the response to host if the target is busy with processing or preparing data.

The following charts show the communication flow of the host reading the ACK packet and the corresponding responses from the target.

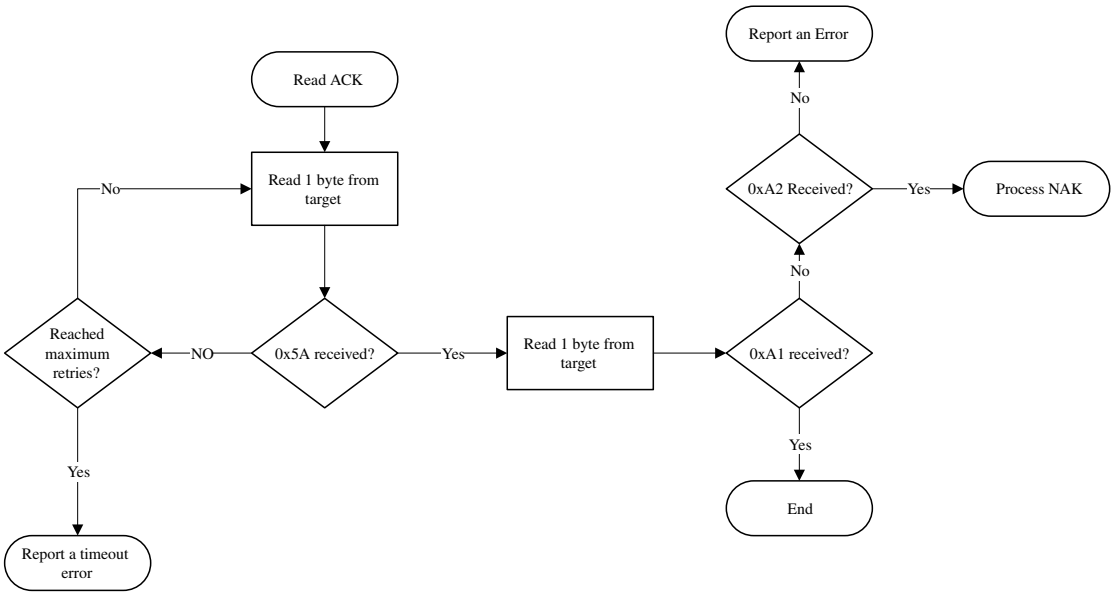


Figure 47. Host reads ACK packet from target via LPI2C

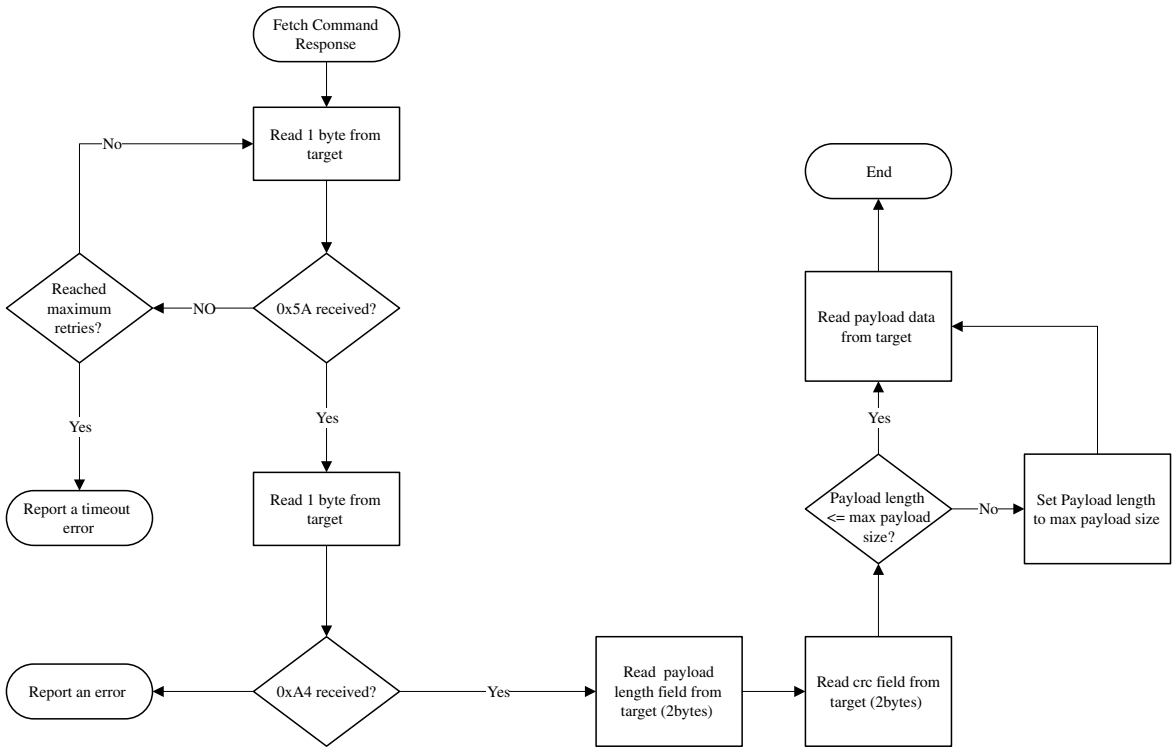


Figure 48. Host reads response from target via LPI2C

For information on commands and formats, see ISP packet type

16.9 LPSPI ISP

The bootloader supports In-System Programming or serial boot via the LPSPI peripheral.

The maximum supported baud rate of the LPSPI depends on the core clock frequency when the bootloader is running. The typical baud rate is 1 MHz with the factory settings. The actual baud rate is up to 4 MHz when the core is running at high-speed boot mode.

The LPSPI peripheral in the bootloader serves as an LPSPI slave device. Each transfer, therefore, must be started by the host, and each outgoing packet should be fetched by the host as well.

NOTE

The peripheral baud rate mentioned here is just at ISP packet level, the actual download speed in ISP is lower than the baud rate because of overhead in the ISP protocol and packet handling.

Compared to LPUART and LPI2C peripherals, the transfer on LPSPI is slightly different:

- The host receives 1 byte after it sends out any byte.
- Received bytes should be ignored when the host is sending out bytes to the target
- The host starts reading bytes by sending 0x00s to target
- The byte 0x00 is sent as a response to host if the target is under the following conditions:
  - Processing incoming packet
  - Preparing outgoing data
  - Received invalid data

The following flowcharts show how the host reads a ping response, an ACK and a command response from target via LPSPI without the nIRQ pin enabled.

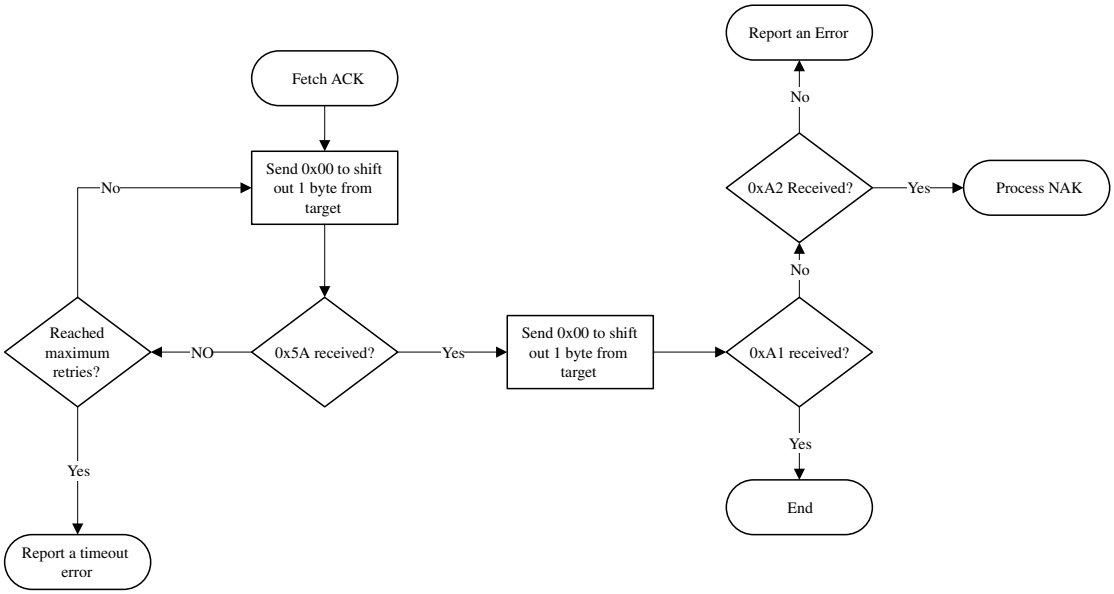


Figure 49. Host reads ACK from target via LPSPI

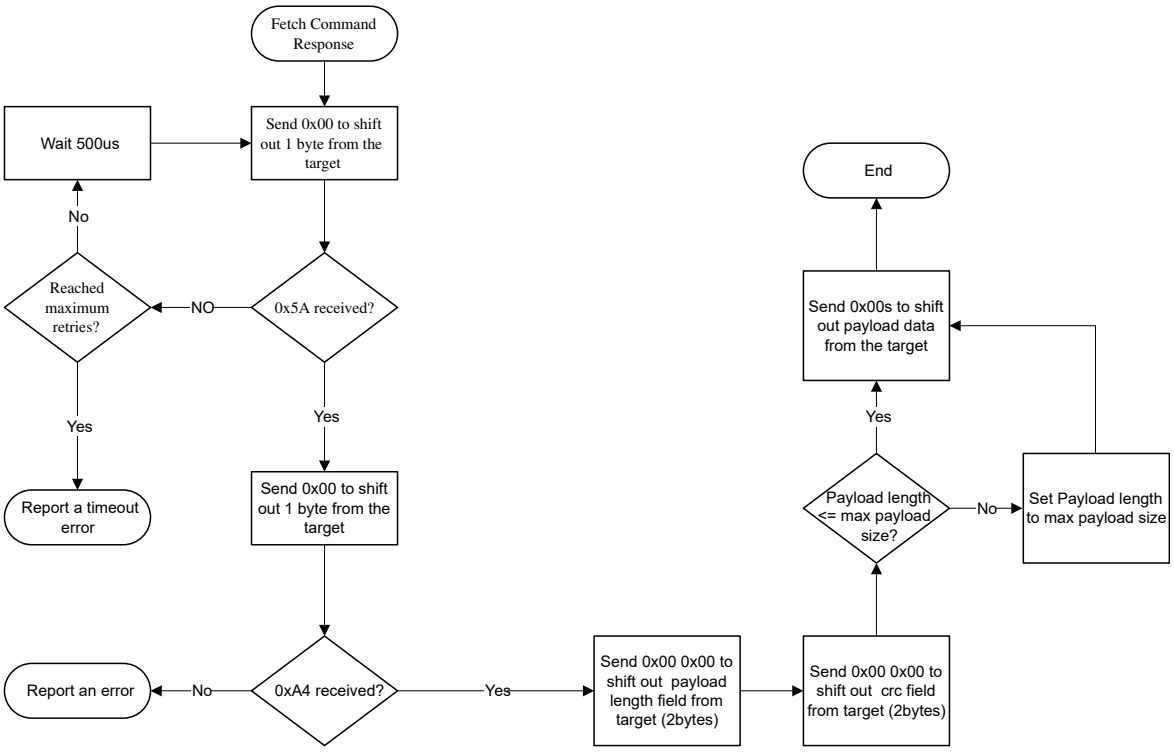


Figure 50. Host reads response from target via LPSPI

For information on commands and formats, see ISP packet type

16.10 CAN ISP

The bootloader supports loading data into flash via the FlexCAN peripheral. It supports four predefined speeds on FlexCAN transferring:

- 125 kHz
- 250 kHz
- 500 kHz
- 1 MHz

NOTE

The peripheral baud rate mentioned here is just at ISP packet level, the actual download speed in ISP is lower than the baud rate because of overhead in the ISP protocol and packet handling.

In host applications, the user can specify the speed for FlexCAN by providing the speed index as 0 through 4, which represents those 5 speeds (default is 1 MHz).

In bootloader, this supports the auto speed detection feature within supported speeds. In the beginning, the bootloader enters the listen mode with the initial speed (default speed 1 MHz). Once the host starts sending a ping to a specific node, it generates traffic on the FlexCAN bus. Because the bootloader is in a listen mode. It can check if the local node speed is correct by detecting errors. If there is an error, some traffic will be visible, but it may not be on the right speed to see the real data. If this happens, the speed setting changes and checks for errors again. No error means the speed is correct. The settings change back to the normal receiving mode to see if there is a package for this node. It then stays in this speed until another host is using another speed and try to communicate with any node. It repeats the process to detect a right speed before sending host timeout and aborting the request.

The host side should have a reasonable time tolerance during the auto speed detect period. If it sends as timeout, it means there is no response from the specific node, or there is a real error and it needs to report the error to the application.

The flow chart below shows the communication flow for how the host reads the ping packet, ACK, and response from the target.

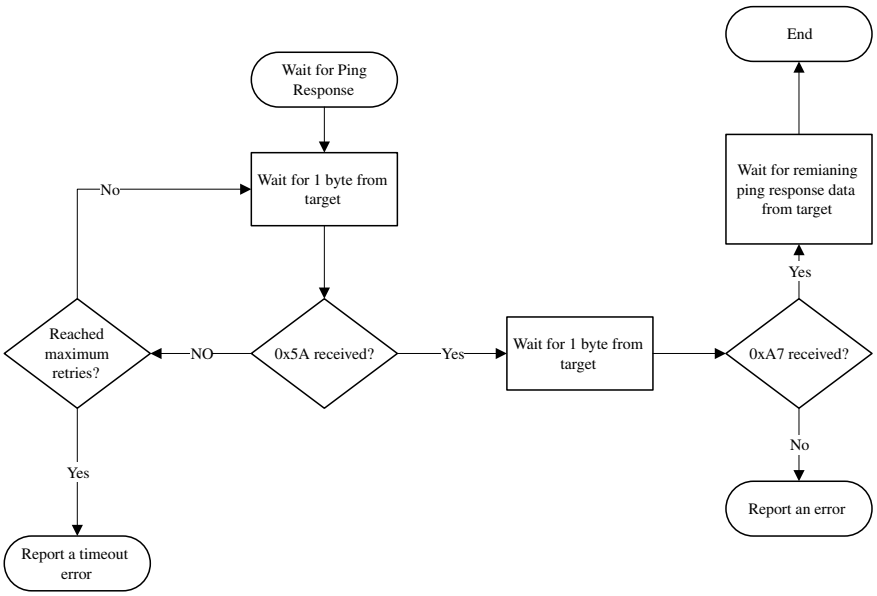


Figure 51. Host reads ping response from target via FlexCAN

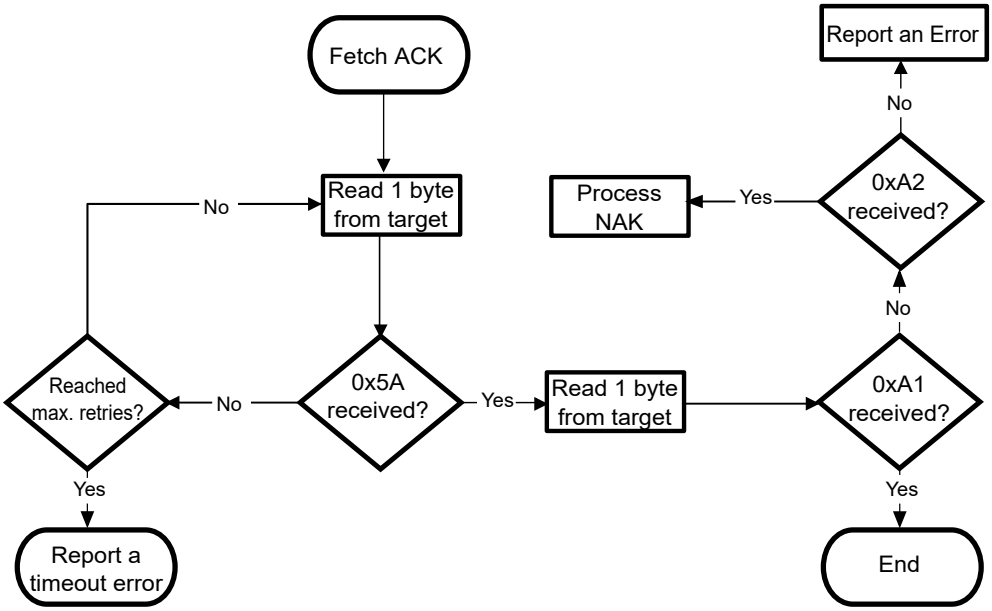


Figure 52. Host reads ACK packet from target via FlexCAN

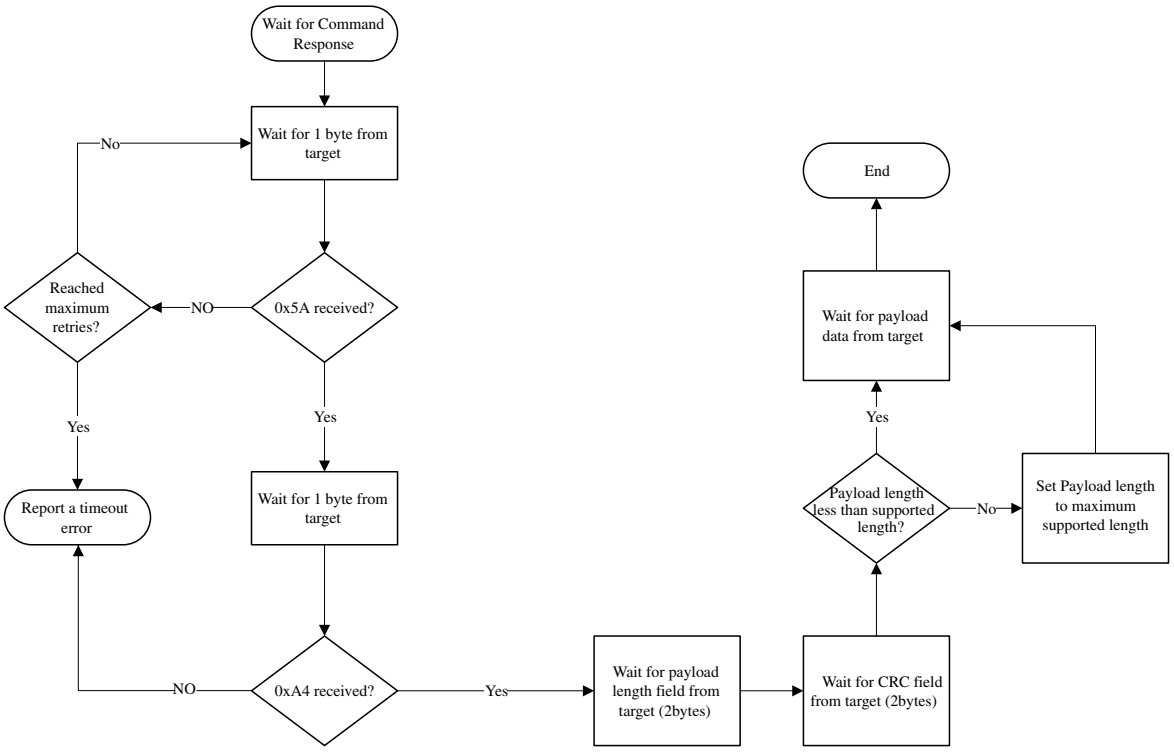


Figure 53. Host reads command response from target via FlexCAN

Chapter 17
ROM API

17.1 Overview

ROM bootloader provides several APIs for users. Disabling the interrupts before making any ROM API call is suggested, since API code does not deal with interrupts. Flash APIs are mainly used to manage flash including read, write and erase. Similarly, SPI NOR APIs are mainly used to manage SPI NOR including read, write and erase. nboot APIs are mainly used to do security related operations including image authentication and fuse programming. kb APIs are mainly used to process SB file. Please note that 5 KB of SRAM (0x30002000 - 0x30003400) needs to be reserved when calling flash api, SPI NOR API and kb APIs in user application.

The ROM API table locates at address 0x14816fe0. See Figure below for the ROM API layout.

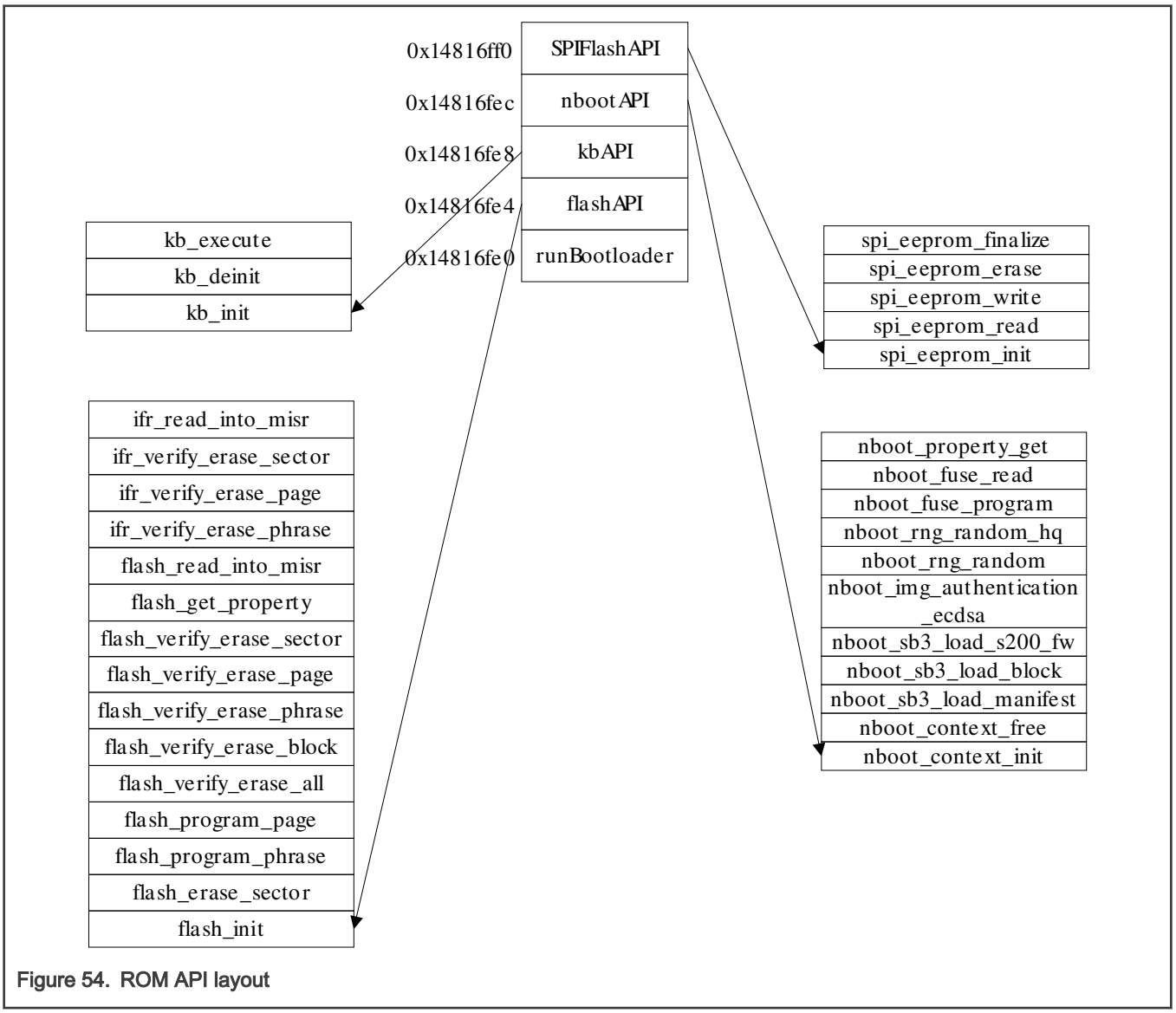


Figure 54. ROM API layout

17.2 Flash API

Flash APIs implement all the flash operation supported by CM33 FMU and RF-FMU1. Please note that operations to RF_FMU1 are only allowed when lifecycle is at OEM Open and Token indicates OEM1 ownership. Details of each API is described in this section.

17.2.1 flash_init

This API is used for initializing the flash controller and the flash_config context. It must be called before calling other flash APIs.

Prototype

```
status_t flash_init (flash_config_t *config);
```

Parameters

Table 110. flash_init parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.

flash_config_t is defined as following:

```
typedef struct _flash_mem_descriptor {
    uint32_t blockBase;
    uint32_t totalSize;
    uint32_t blockCount;
} flash_mem_desc_t;

typedef struct _flash_ifr_desc {
    uint32_t pflashIfr0Start;
    uint32_t pflashIfr0MemSize;
} flash_ifr_desc_t;

typedef struct _msf1_config {
    flash_mem_desc_t flashDesc;
    flash_ifr_desc_t ifrDesc;
} msf1_config_t;

typedef struct _flash_config {
    msf1_config_t msf1Config[2];
} flash_config_t;
```

Example:

```
typedef struct
{
    __I void (*runBootloader)(void *arg);           //!< Function to start the bootloader executing.
    __I FLASH_API_Type *flashApiBase;         //!< Internal Flash driver API.
    __I KB_API_Type *kbApiBase;               //!< Bootloader API.
__I NBOOT_API_Type *nbootApiBase;         //!< Image authentication API.
    __I SPI_FLASH_API_Type *spiflashApiBase;             //!< SPI Flash API.

} ROM_API_TYPE;
/** ROM API base address */
#define ROM_API_BASE                        (0x14816fe0u)
/** ROM API base pointer */
#define ROM_API                             ((ROM_API_TYPE*) ROM_API_BASE)
```

```
/** FLASH API base pointer */
#define FLASH_API                          (ROM_API->flashApiBase)
static flash_config_t flashConfig;
FLASH_API->flash_init(&flashConfig);
```

17.2.2 flash_erase_sector

This API is used for erasing specified flash sector (8 KB) in flash or User IFR(IFR0). This API will erase sectors which include the start address and (start + lengthInBytes - 1) address.

Prototype

```
status_t flash_erase_sector (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes, uint32_t key);
```

Parameters

Table 111. flash_erase_sector parameters

Parameter	Description
Config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the required flash memory to be erased. The starting address must be phrase-aligned (4-word or 16-byte).
lengthInBytes	The length, given in bytes (not words or long words) to be erased.
Key	Key is used to validate erase operation. Must be set to "kFLASH_ApiEraseKey"

Example:

```
#define FOUR_CHAR_CODE(a, b, c, d) (((d) << 24) | ((c) << 16) | ((b) << 8) | ((a)))
#define CM33_FMU                                    ((FMU_Type *)0x40020000u)
enum _flash_driver_api_keys
{
           kFLASH_ApiEraseKey = FOUR_CHAR_CODE('l', 'f', 'e', 'k')
};
result = FLASH_API->flash_erase_sector(&flashConfig, CM33_FMU, 0x0, 4, kFLASH_ApiEraseKey);
```

17.2.3 flash_program_phrase

This API is used for programming phrase-aligned data to a previously erased flash or User IFR phrase.

Prototype

```
status_t flash_program_phrase (flash_config_t *config, FMU_Type *base, uint32_t start,
                          uint32_t *src, uint32_t lengthInBytes);
```

Parameters

Table 112. flash_program_phrase parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1

Table continues on the next page...

Table 112. flash_program_phrase parameters (continued)

Parameter	Description
Start	The starting address of the required flash memory to be programmed. The starting address must be phrase-aligned (4-word or 16-byte).
Src	Pointer to the source buffer of data that is to be programmed into flash.
lengthInBytes	The length, given in bytes (not words or long words) to be programmed. The lengthInBytes must be phrase-aligned (4-word or 16-byte).

17.2.4 flash_program_page

This API is used for programming page aligned data to a previously erased flash or User IFR page.

Prototype

```
status_t flash_program_page (flash_config_t *config, FMU_Type *base, uint32_t start,
                        uint32_t *src, uint32_t lengthInBytes);
```

Parameters

Table 113. flash_program_page parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the required flash memory to be programmed. The starting address must be page-aligned (32-word or 128-byte).
Src	Pointer to the source buffer of data that is to be programmed into flash.
lengthInBytes	The length, given in bytes (not words or long words) to be programmed. The lengthInBytes must be page-aligned (32-word or 128-byte).

17.2.5 flash_verify_erase_all

This API is used for checking if all flash and IFR space are in the erased state..

Prototype

```
status_t flash_verify_erase_all (FMU_Type *base);
```

Parameters:

Table 114. flash_verify_erase_all parameters

Parameter	Description
FMU base	Base pointer to FMU or RF-FMU1

17.2.6 flash_verify_erase_block

This API is used for checking if a flash block is in the erased state.

Prototype

```
status_t flash_verify_erase_block (flash_config_t *config, FMU_Type *base, uint32_t blockaddr);
```

Parameters

Table 115. flash_verify_erase_block parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
blockaddr	Starting address of a flash block, needs to be block-aligned

17.2.7 flash_verify_erase_phrase

This API is used for checking if specified flash phrases are in the erased state.

Prototype

```
status_t flash_verify_erase_phrase (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes);
```

Parameters

Table 116. flash_verify_erase_phrase parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of required flash memory to be checked. The starting address must be phrase-aligned (4-word or 16-byte).
lengthInBytes	The length, given in bytes (not words or long words) to be checked. The lengthInBytes must be phrase-aligned (4-word or 16-byte).

17.2.8 flash_verify_erase_page

This API is used for checking if specified flash pages are in the erased state.

Prototype

```
status_t flash_verify_erase_page (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes);
```

Parameters

Table 117. flash_verify_erase_page parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of required flash memory to be checked. The starting address must be page-aligned (32-word or 128-byte).
lengthInBytes	The length, given in bytes (not words or long words) to be checked. The lengthInBytes must be page-aligned (32-word or 128-byte).

17.2.9 flash_verify_erase_sector

This API is used for checking if specified IFR0 sectors are in the erased state.

Prototype

```
status_t flash_verify_erase_sector (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes);
```

Parameters

Table 118. flash_verify_erase_sector parameters

Parameter	Description
Config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of required flash memory to be checked. The starting address must be sector-aligned (8192-byte).
lengthInBytes	The length, given in bytes (not words or long words) to be checked. The lengthInBytes must be sector-aligned (8192-byte).

17.2.10 flash_read_into_misr

This API is used for generating a signature based on the contents of the selected flash memory using an embedded MISR.

Prototype

```
status_t flash_read_into_misr (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t end,
uint32_t *seed, uint32_t *signature);
```

Parameters

Table 119. flash_read_into_misr parameters

Parameter	Description
Config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the selected flash region . The start address must be page-aligned (32-word or 128-byte).
End	The ending address of the selected flash region. Ending address must be flash phrase aligned (4-word or 16-byte) and the last phrase in a page.
Seed	MISR seed
Signature	Pointer to buffer expecting signature after successful generation

17.2.11 ifr_verify_erase_phrase

This API is used for checking if specified IFR0 phrases are in the erased state.

Prototype

```
status_t ifr_verify_erase_phrase (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes);
```

Parameters

Table 120. ifr_verify_erase_phrase parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the required IFR0 memory to be checked. The starting address must be phrase-aligned (4-word or 16-byte).
lengthInBytes	The length, given in bytes (not words or long words) to be checked. The lengthInBytes must be phrase-aligned (4-word or 16-byte).

17.2.12 ifr_verify_erase_page

This API is used for checking if specified IFR0 pages are in the erased state.

Prototype

```
status_t ifr_verify_erase_page (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes);
```

Parameters

Table 121. ifr_verify_erase_page parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the required flash memory to be checked. The start address must be page-aligned (32-word or 128-byte).
lengthInBytes	The length, given in bytes (not words or long words) to be checked. The lengthInBytes must be page-aligned (32-word or 128-byte).

17.2.13 ifr_verify_erase_sector

This API is used for checking if specified IFR0 sectors are in the erased state.

Prototype

```
status_t ifr_verify_erase_sector (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t
lengthInBytes);
```

Parameters

Table 122. ifr_verify_erase_sector parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the required flash memory to be checked. The start address must be sector-aligned(8192-byte).

Table continues on the next page...

Table 122. ifr_verify_erase_sector parameters (continued)

Parameter	Description
lengthInBytes	The length, given in bytes (not words or long words) to be checked. The lengthInBytes must be sector-aligned(8192-byte).

17.2.14 ifr_read_into_misr

This API is used for generating a signature based on the contents of the selected IFR0 space using an embedded MISR.

Prototype

```
status_t ifr_read_into_misr (flash_config_t *config, FMU_Type *base, uint32_t start, uint32_t end,
uint32_t *seed, uint32_t *signature);
```

Parameters

Table 123. ifr_read_into_misr parameters

Parameter	Description
Config	Pointer to flash_config_t data structure in memory to store driver runtime state.
FMU base	Base pointer to FMU or RF-FMU1
Start	The starting address of the selected flash region . The start address must be page-aligned (32-word or 128-byte).
End	The ending address of the selected flash region. Ending address must be flash phrase aligned (4-word or 16-byte) and the last phrase in a page.
Seed	MISR seed
Signature	Pointer to buffer expecting signature after successful generation

17.2.15 flash_get_property

This API returns the required flash property, which includes base address, block size, and other options.

Prototype

```
status_t flash_get_property (flash_config_t *config, flash_property_tag_t whichProperty, uint32_t
*value);
```

Parameters

Table 124. flash_get_property parameters

Parameter	Description
config	Pointer to flash_config_t data structure in memory to store driver runtime state.
whichProperty	The required property from the list of properties.
Value	Property value return to value pointer

Table 125. Property definition

Property Definition	Value
FMU_SectorSize	0x00
FMU_TotalSize	0x01
FMU_BlockSize	0x02
FMU_BlockCount	0x03
FMU_BlockBaseAddr	0x04
RF-FMU1_TotalSize	0x11
RF-FMU1_BlockSize	0x12
RF-FMU1_BlockCount	0x13
RF-FMU1_BlockBaseAddr	0x14

17.2.16 Flash API status code

Table 126. Flash API status code

Status	Code	Description
kStatus_FLASH_Success	0	The flash operation is successful
kStatus_FLASH_InvalidArgument	4	Invalid argument detected during executing a FLASH API.
kStatus_FLASH_SizeError	100	Invalid size detected during executing a FLASH API.
kStatus_FLASH_AlignmentError	101	Alignment error detected during executing a FLASH API.
kStatus_FLASH_AddressError	102	Address error detected during executing a FLASH API.
kStatus_FLASH_AccessError	103	Access error detected during executing a FLASH API.
kStatus_FLASH_ProtectionViolation	104	Command Protection Violation Flag which indicates an attempt was made to modify a protected area of flash memory during a command operation.
kStatus_FLASH_CommandFailure	105	Command failure detected during executing a FLASH API.
kStatus_FLASH_UnknownProperty	106	Unknown property for flash_get_property API.
kStatus_FLASH_EraseKeyError	107	Incorrect EraseKey for flash_erase API.
kStatus_FLASH_CommandAborOption	121	Operation is aborted

17.3 nboot API

The ROM API table is located at address 0x14816fe0 and contains absolute ROM API function addresses which can be called using function pointers. Only secure boot related functions are described in this chapter.

The main purpose of these APIs is to provide access to functions used and implemented in ROM to authenticate the application image and processing sb3 files.

17.3.1 nboot_context_init

This API is used for initializing the nboot context data structure. It must be called before calling other nboot APIs which performs authentication and sb3 processing.

Prototype:

```
nboot_status_t nboot_context_init(nboot_context_t *context);
```

Parameters

Table 127. nboot_context_init parameters

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.

17.3.2 nboot_context_free

This API is used to release context used by all nboot functions.

Prototype:

```
nboot_status_t nboot_context_free(nboot_context_t *context);
```

Parameters

Parameter	Description
context	Pointer to nboot_context_t data structure in memory used to store runtime state.

17.3.3 nboot_sb3_load_manifest

This API is used to verify nboot sb3.1 manifest. It loads keys into the key store so that they can be used for subsequent operations such as nboot_sb3_load_block. NBOOT context must be initialized by the API nboot_context_init before calling this API.

Prototype:

```
nboot_status_t nboot_sb3_load_manifest(nboot_context_t *context, uint32_t *manifest);
```

Parameters

Table 128. nboot_sb3_load_manifest parameters

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
manifest	Pointer to the input manifest buffer.

17.3.4 nboot_sb3_load_block

This API is used to verify and decrypt NBOOT SB3.1 block. Decryption is performed in-place. NBOOT context must be initialized by the API nboot_context_init before calling this API.

Prototype:

```
nboot_status_t nboot_sb3_load_block(nboot_context_t *context, uint32_t *block);
```

Parameters

Table 129. nboot_sb3_load_block parameters

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
block	Pointer to the input block

17.3.5 nboot_sb3_load_s200_fw

This API is used to verify and decrypt sb3.1 file with the ELE firmware. Decryption is performed to the ELE RAM and firmware automatically started after successful load operation. NBOOT context must be initialized by the API nboot_context_init before calling this API.

Prototype:

```
nboot_status_t nboot_sb3_load_s200_fw(nboot_context_t *context, uint32_t *sb3Data);
```

Parameters

Table 130. nboot_sb3_load_s200_fw parameters

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
block	Pointer to the sb3.1 block with the ELE firmware.

17.3.6 nboot_img_authenticate_ecdsa

This API is used to authenticate signed image with asymmetric cryptography.

Prototype:

```
nboot_status_t nboot_img_authenticate_ecdsa(nboot_context_t *context, uint8_t imageStartAddress[],
nboot_bool_t *isSignatureVerified);
```

Parameters

Table 131. nboot_img_authenticate_ecdsa

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
imageStartAddress	Pointer to start of the image in memory (32-bit word aligned)
isSignatureVerified	Pointer to memory holding function call result. typedef enum { kNBOOT_TRUE = 0x3C5AC33Cu, kNBOOT_TRUE256 = 0x3C5AC35Au, kNBOOT_TRUE384 = 0x3C5AC3A5u, kNBOOT_FALSE = 0x5AA55AA5u, kNBOOT_OperationAllowed = 0x3c5a33ccU,

Table continues on the next page...

Table 131. nboot_img_authenticate_ecdsa (continued)

Parameter	Description
	kNBOOT_OperationDisallowed = 0x5aa5cc33U, } nboot_bool_t;

17.3.7 nboot_rng_random

This API is used to get random number(s).

Prototype:

```
nboot_status_t nboot_rng_random(nboot_context_t *context, void *buf, size_t bufLen);
```

Parameters

Table 132. nboot_rng_random

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
Buf	Pointer to buffer in memory to store random number
bufLen	Buffer length in number of bytes.

17.3.8 nboot_rng_random_hq

This API is used to get high quality random number(s).

Prototype:

```
nboot_status_t nboot_rng_random_hq(nboot_context_t *context, void *buf, size_t bufLen);
```

Parameters

Table 133. nboot_rng_random_hq

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
Buf	Pointer to buffer in memory to store random number
bufLen	Buffer length in number of bytes.

17.3.9 nboot_fuse_program

This API is used to program a fuse word at a given address with new data. Before using this API, voltage must be regulated for over-drive and normalize voltage after operation is completed.

Prototype:

```
nboot_status_t nboot_fuse_program(nboot_context_t *context, uint32_t addr, uint32_t *data, uint32_t
systemClockFrequencyMHz);
```

Parameters

Table 134. nboot_fuse_program

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
Addr	Fuse index
Data	Pointer to data expected to be programmed in fuse
systemClockFrequencyMHz	Boot frequency

17.3.10 nboot_fuse_read

This API is used to read a fuse word.

Prototype:

```
nboot_status_t nboot_fuse_read(nboot_context_t *context,
                                    uint32_t addr,
                                    uint32_t *data,
                                    uint32_t systemClockFrequencyMHz);
```

Parameters

Table 135. nboot_fuse_read

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
Addr	Fuse index
Data	Pointer to data buffer expecting fuse contents after successful read
systemClockFrequencyMHz	Boot frequency

17.3.11 nboot_property_get

This API is used to read property. One of the important properties that can be read is the property that last authentication of signed image container has succeeded.

Prototype:

```
nboot_status_t nboot_property_get(nboot_context_t *context,
                                  uint32_t propertyId,
                                  uint8_t *destData,
                                  size_t *dataLen);
```

Parameters

Table 136. nboot_property_get parameters

Parameter	Description
context	Pointer to nboot_context_t data structure in memory to store runtime state.
propertyId	Property ID must be supported by nboot

Table continues on the next page...

Table 136. nboot_property_get parameters (continued)

Parameter	Description
destData	Pointer to data buffer for storing returned contents
dataLen	Data buffer length

Property IDs supported are as follows:

Table 137. Supported property IDs

Property	ID	Description
DICE_CDI	0x10	Returns 32 bytes of CDI value.
IMAGE_HASH	0x20	Return 64 bytes (SHA-512) value containing hash of the last authenticated image.
PSA_BOOT_SEED	0x30	Returns 32 bytes boot seed.
LAST_AUTH_STATE	0x40	Returns 4 bytes status of last authentication.
ELE_ROM_VERSION	0x50	Returns 8 bytes of ELE ROM version.
ELE_FW_VERSION	0x51	If ELE FW not loaded successfully, returns 0xFFFFFFFF as 1st word. Else returns 2 words, 1st word: version, 2nd word – commit id.
DTRK_ATTEST_PUBK	0x60	Returns 64 bytes value – public part of the DTRK_ATTEST private key
RADIO_IMG_OWNER	0x80	Returns 4 byte to indicate radio ownership as follows: 0x0 = NXP 0xAABBCCDD = OEM1 (Can be a radio firmware developer outside NXP based on token). 0xDDCCBBAA = OEM2
UUID	0x90	Returns UUID to serve as device unique identifier.

17.3.12 nboot API status code

The following table lists the nboot API status code.

Table 138. nboot API status code

Status	Code
kStatus_NBOOT_Success	0x5a5a5a5a
kStatus_NBOOT_Fail	0x5a5aa5a5
kStatus_NBOOT_InvalidArgument	0x5a5aa501
kStatus_NBOOT_RequestTimeout	0x5a5aa502
kStatus_NBOOT_ResourceBusy	0x5a5aa503
kStatus_NBOOT_OperationNotAvaialable	0x5a5aa5e5
kStatus_NBOOT_MemcpyFail	0x5a5a845a

17.4 kb API

This set of APIs provides interface to bootloader API functions. The main purpose of this APIs is to provide ROM functions that can be used to process SB file.

17.4.1 kb_init

This API is used to initialize bootloader and nboot context necessary to process sb3 file format.

Prototype:

```
status_t kb_init(void);
```

This API is expected to return kStatus_Success (0) on success and kStatus_Fail (1) on failure.

17.4.2 kb_deinit

This API is used to release nboot context and finalize sb3 file processing.

Prototype:

```
status_t kb_deinit(void)
```

17.4.3 kb_execute

This API is used to decrypt sb3 file and store signed image contents specified by loader command supported while generating sb3 image through Json configuration. If sb3 file to be processed includes sblaoder command “programFuses” then voltage must be regulated for over-drive and normalize voltage once operation is completed.

Prototype:

```
status_t kb_execute(const uint8_t *data, uint32_t dataLength, , uint32_t isUpdateExt);
```

Parameters

Table 139. kb_execute parameters

Parameter	Description
data	Pointer to start of sb file data in memory
dataLength	sb file data length in bytes
isUpdateExt	Indicator for update(sb) file start address is in internal or external flash. 0x74784578u = External flash Other values = Internal flash

17.5 SPI Flash API

17.5.1 spi_eeprom_init

This API is used to initialize SPI Flash with requested baud rate and reads JEDEC ID as sanity check.

Prototype:

```
status_t spi_eeprom_read(uint8_t *dest, uint32_t NoOfBytes, uint32_t address, bool
requestFastRead);
```

Parameters

Table 140. spi_eeprom_init parameters

Parameter	Description
baudRate	Value used to configure SPI NOR flash for subsequent operations.

17.5.2 spi_eeprom_read

This API is used to read SPI Flash memory contents.

Prototype:

```
status_t spi_eeprom_read(uint8_t *dest, uint32_t NoOfBytes, uint32_t address, bool
requestFastRead);
```

Parameters

Table 141. spi_eeprom_read parameters

Parameter	Description
dest	Pointer to destination buffer to store memory contents read from SPI flash.
NoOfBytes	Number of bytes to read.
address	Absolute start address of SPI Flash from which read operation should be prformed.
requestFastRead	true = Uses FAST READ (0Bh) false = Uses simple read READ (03h) • User must check AC chacracteristics of Flash memory model and should know SPI functional clock before requesting requestFastRead. Otherwise requestFastRead option can cause a bootleneck in data transaction speed as command buffer involves dummy byte.

17.5.3 spi_eeprom_write

This API is used to program data to SPI flash.

Prototype:

```
status_t spi_eeprom_write(uint8_t *data, uint32_t NoOfBytes, uint32_t address);
```

Parameters

Table 142. spi_eeprom_write parameter

Parameter	Description
data	Pointer to the buffer of data that is to be programmed into SPI flash.
NoOfBytes	Number of bytes to be programmed.
Address	Start address of the SPI Flash memory to be programmed.

17.5.4 spi_eeprom_erase

This API is used to erase SPI flash contents with specific erase size.

Prototype:

```
status_t spi_eeprom_erase(uint32_t address, eraseOptions_t option);
```

Parameters

Table 143. spi_eeprom_erase parameters

Parameter	Description
address	Start address of the SPI Flash memory to be erased.

option	Following are the options supported for erase operation:
	`typedef enum`
	`{`
	`  kSize_ErasePage = 0x1,`
	`  kSize_Erase4K = 0x2,`
	`  kSize_Erase32K = 0x3,`
	`  kSize_Erase64K = 0x4,`
	`  kSize_EraseAll = 0x5,`
	`} eraseOptions_t;`
	NOTE
	kSize_ErasePage option is not supported for all memory models. Example: Adesto flash can support page erase commands but not Micron.

17.5.5 spi_eeprom_finalize

This API is used de-initialize IOMUX and clock settings used for SPI Flash operations and resets all LPSPI logic and control registers.

Prototype:

```
void spi_eeprom_finalize(void);
```

17.5.6 SPI Flash API status code

The following table lists the SPI Flash API status code.

Table 144. SPI Flash API status code

Status	Code
kStatus_Success	0
kStatus_Fail	1

Chapter 18
Flash Memory Unit (FMU)

18.1 Chip-specific FMU information

Table 145. Reference links to related information

Topic	Related module	Reference
Full description	FMU	FMU
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

NOTE

For the SoC 32 KB IFR0, the first 8 KB is program once and never allows to be erased.

18.1.1 Module instances

This device has one instance of the FMU module, FMU0.

18.1.2 IFR map

For this device, refer to User IFR allocation for Table 176.

18.2 Overview

The flash module includes the following accessible memory regions:

- Program flash memory for vector space and code store
- Separate flash memory for parameter store

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The flash module includes a command controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash phrase or page is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that flash memory be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

18.2.1 Glossary

Block Checker — MCU IP that provides a mechanism to set access rights to the flash memory for read and write operations.

Command write sequence — A series of MCU writes to the Flash FCCOB register group and FSTAT[CCIF] that initiates and controls the execution of flash algorithms that are built into the flash module.

Endurance — The number of times that an aligned flash or IFR phrase can be erased and reprogrammed.

FCCOB (Flash Common Command Object Block) — A group of flash registers that are used to pass command, address, data, and any associated parameters to the command controller.

Flash block — A macro within the flash module which provides the nonvolatile memory storage with an aligned block address being the first address in the block.

Flash module — All flash blocks plus a flash management unit providing command control and an interface to MCU buses.

Flash page — 128 bytes of flash main memory with an aligned page having byte-address[6:0] = 00h; represents the largest portion of the flash memory that can be programmed in one operation.

Flash phrase — 16 bytes of flash main memory with an aligned phrase having byte-address[3:0] = 0000b; represents the smallest portion of the flash memory that can be programmed in one operation.

Flash sector — 8 Kbytes of flash main memory with an aligned sector having byte-address[12:0] = 0000h; represents the smallest portion of the flash memory that can be erased in one operation.

FLW — The Flash Logical Window module allows a specific logical address range to access a programmable physical address range in the flash memory with programmable FLW default settings stored in IFR space.

FMC — Flash Memory Controller module manages flash memory reads and writes.

IFR — Information flash region separate from the main flash memory array. Each flash block has 32 Kbytes of user IFR space. Sometimes referred to as IFR0.

IFR1 — Information flash region separate from the main memory array reserved for array trim, MCU trim, and test. Each flash array has 8 Kbytes of IFR1 space.

IFR page — 128 bytes of IFR or IFR1 space with an aligned IFR page having byte-address[6:0] = 00h; represents the largest portion of IFR space that can be programmed.

IFR phrase — 16 bytes of IFR or IFR1 space with an aligned IFR phrase having byte-address[3:0] = 0000b; represents the smallest portion of IFR space that can be programmed in one operation.

IFR sector — 8 Kbytes of IFR or IFR1 space with an aligned IFR sector having byte-address[12:0] = 0000h; represents the smallest portion IFR space that can be erased.

MISR — Multiple-input signature register used to generate a signature based on flash memory contents read.

MSF — Microcontroller Secure Flash.

NVM — Nonvolatile memory. The flash block is an NVM using NOR-type flash memory technology.

Program flash — Program flash memory provides nonvolatile storage for vectors and code store.

Retention — The length of time that data (erased or programmed) can be kept in the NVM without experiencing errors upon readout.

Security/Privilege protection levels — The Flash module supports 4 levels of security/privilege protection for flash commands:

1. Secure (Trusted) / Privileged
2. Secure (Trusted) / User
3. Non-secure (Non-Trusted) / Privileged
4. Non-secure (Non-Trusted) / User

18.2.2 Block diagram

The block diagram of the flash module is shown in the following figure.

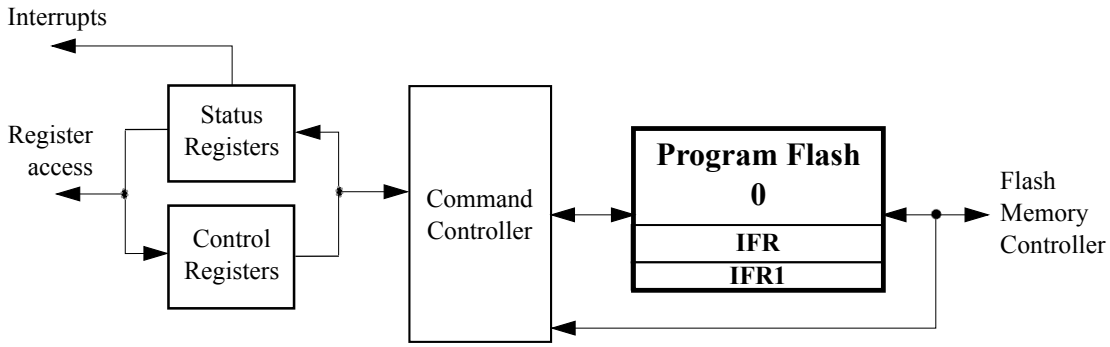


Figure 55. Flash block diagram

18.2.3 Features

The flash memory module provides the following features:

- Sector size of 8 Kbytes
- Single-bit error correction and double-bit error detection for each flash or IFR phrase
- Signature generation for flash contents
- Command-based flash program, erase, and verify operations
- Internal high-voltage supply generator for flash program and erase operations
- Optional interrupt generation upon flash command completion
- Optional interrupt generation upon double-bit error detection during flash reads from the FMC

NOTE

See the chip configuration details for the exact amount of flash memory available.

18.2.4 Modes of Operation

18.2.4.1 Sleep mode

If the CPU enters sleep mode while a flash command is executing, the flash module can wake the CPU via the command complete interrupt (see Interrupts).

18.2.4.2 Non-Active Power mode

If a flash command is active (CCIF = 0) when the MCU requests a non-active power mode, the command execution completes before the MCU is allowed to enter the non-active power mode.

18.2.4.3 Low Speed Active Power mode

While the MCU is in low speed active power mode (FCTRL[LSACTIVE]=1), the flash module will accept flash commands. While a flash command is in progress (FSTAT[CCIF]=0), the LSACTIVE bit is not writable.

18.3 External signal description

The flash module contains no signals that connect off-chip.

18.4 Functional description

The following sections describe functional details of the flash module.

18.4.1 Interrupts

The flash module can generate interrupt requests to the MCU upon the occurrence of various flash module events. These interrupt events and their associated status and control bits are shown in the following table.

Table 146. Flash Module Interrupt Sources

Flash Event	Readable Status Bit	Interrupt Enable Bit
Flash Command Complete	FSTAT[CCIF]	FCNFG[CCIE]
Flash ECC Fault Detected	FSTAT[DFDIF]	FCNFG[DFDIE]

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

18.4.2 Flash command operations

Flash command operations are typically used to modify flash memory contents. The next sections describe:

- Command write sequence used to set flash command parameters and launch execution
- Available flash commands
- Commands impacted by Block Checker and FMC access protection

18.4.2.1 Command write sequence

Flash commands are accomplished using a command write sequence illustrated in Figure 56. The command controller monitors the transaction protection level on writes during the command write sequence, performs various checks on the command (FCCOB) content, and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR, PVIOL, and CMDABT flags in the FSTAT register must be zero and the CCIF flag must be one to verify that any previous command has completed. The FAIL flag in the FSTAT register is cleared by the command controller when a command is launched unless the FAIL flag was set during initialization due to FMU parameters not being loaded from IFR1. If CCIF is clear, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

Attempts to launch a flash command while the radio is active are ignored.

While a flash command is executing, attempts to write to FCTRL[LSACTIVE] are ignored.

18.4.2.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the specific flash command. The contents of any FCCOB register not used by the specific flash command are ignored except for FCCOB1 (command options) where undefined bits should be cleared. The individual registers that make up the FCCOB data set can be written in any order. All writes to the FCCOB registers in a command write sequence must occur at the same transaction protection level. Any write to the FCCOB registers that violates this rule will be ignored.

Addresses loaded into FCCOB registers assume the flash and IFR spaces start at 0000_0000h.

18.4.2.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing the FSTAT[CCIF] bit by writing a '1' to it. The write to clear CCIF must occur at the same transaction protection level as the writes used to load

the FCCOB registers. Any write to clear CCIF that violates this rule will be ignored. The CCIF flag remains clear until the flash command completes.

The FSTAT register contains a blocking mechanism which prevents a new flash command from launching (unable to clear CCIF) if the previous command resulted in the setting of an access error (ACCERR), protection violation (PVIOL), or active command abort flag (CMDABT). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write to clear these flags, the second write to clear CCIF. The write to clear these flags can occur at any transaction protection level.

18.4.2.1.3 Command Execution and Error Reporting

Flash command processing contains the following steps:

1. The command controller performs a series of parameter checks, if applicable, which are unique to each command. If the parameter check fails, the FSTAT[ACCERR] flag is set. ACCERR reports invalid instruction codes or options, out of bounds or misaligned addresses, or missing FMU parameters. In the case FMU parameters were not loaded from IFR1 during initialization, the ACCERR flag sets and the FSTAT[FAIL] flag remains set. Command processing never proceeds to execution when the parameter check fails. Instead, command processing is terminated after setting the FSTAT[CCIF] flag. Note that for program and erase operations, PEWEN will not set if FMU parameters were not loaded and PERDY will not set if program or erase related writes to flash or IFR space violate command requirements.
2. If necessary, the command controller performs a protection check to see if the command is allowed to execute on the requested memory space. If the protection check fails, the FSTAT[PVIOL] flag is set. Command processing never proceeds to execution when the protection check fails. Instead, command processing is terminated after setting the FSTAT[CCIF] flag.
3. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to verify, may occur during the execution phase and are reported in the FAIL flag. A flash command may have access errors, protection violations, and run-time errors, but the run-time errors are not seen until all access errors and protection violations have been corrected.
4. If FCTRL[ABTREQ] sets during command execution, the operation will abort with the FSTAT[CMDABT] flag set.
5. Command execution results, if applicable, are reported back to the user via the FCCOB registers.
6. The command controller sets FSTAT[CCIF] signifying that the command has completed.

CAUTION

If the FAIL flag is set along with ACCERR, FMU parameters were not loaded from IFR1 during initialization. In this case, it is recommended that the flash module be reinitialized. If the issue persists, flash commands will continue to set the ACCERR and FAIL flags.

The flow for a generic command write sequence is illustrated in the following figure.

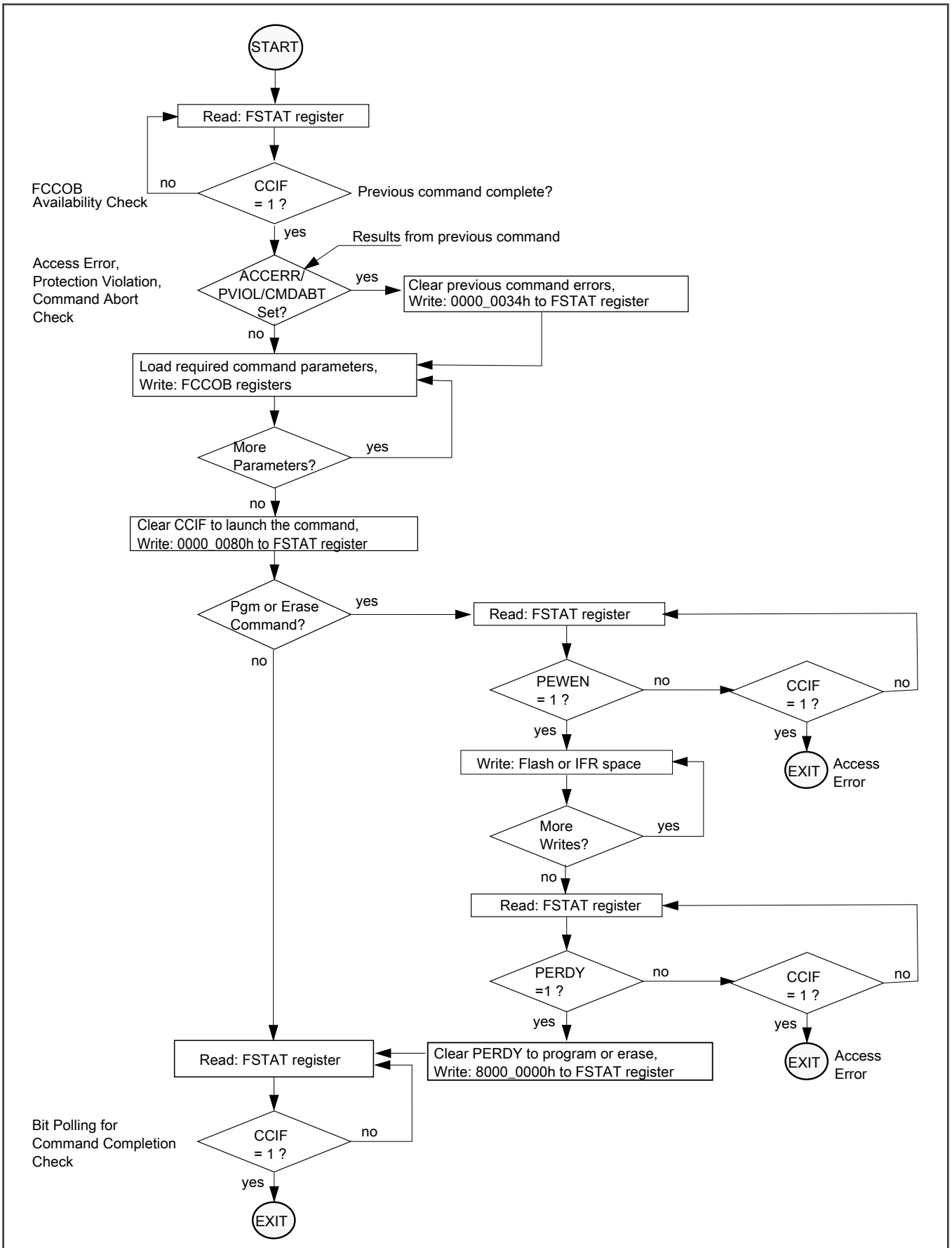


Figure 56. Generic Flash Command Write Sequence Flowchart

18.4.2.1.4 Aborting a Command Write Sequence or Command Operation

FSTAT[CMDP] sets at the start of a command write sequence, typically by writing to one of the FCCOB registers. With CMDP set, setting the FCTRL[ABTREQ] bit allows a user to abort a command write sequence and take control over a new command write sequence. The ABTREQ bit can only be set by a user with the same domain ID as indicated by FSTAT[CMDDID] and at the same or higher security/privilege protection level when compared to FSTAT[CMDPRT], the current owner of the command write sequence. While ABTREQ is set, CCIF cannot be cleared to launch a command.

If FSTAT[CCIF] is high when ABTREQ is set, the command write sequence will abort and ownership transferred to the user setting the ABTREQ bit. The FSTAT[CWSABT] flag will set and the CMDPRT field will be updated based on the write transaction that set the ABTREQ bit. The ABTREQ bit will then be cleared automatically.

If CCIF is low when ABTREQ is set, the command controller will abort execution of the current command operation, set FSTAT[CMDABT], and set CCIF. The CMDPRT field will be updated based on the write transaction that set the ABTREQ bit. The ABTREQ bit will then be cleared automatically. While CMDABT is set, CCIF cannot be cleared to launch a command. Aborting a program or erase operation may leave the flash or IFR space being modified in an indeterminate state.

NOTE

Aborting certain flash commands may take longer to abort than to execute. See the device data sheet for flash command timing specifications.

18.4.2.2 Flash commands

The following table summarizes the function of all flash commands. If any column is marked with an 'X', the flash command is relevant to that particular memory resource.

FCMD	Command	Program flash 0	IFR	Function
00h	Read 1s All (RD1ALL)	x	x	Verify that all flash and IFR space is erased
01h	Read 1s Block (RD1BLK)	x		Verify that a flash block is erased
02h	Read 1s Sector (RD1SCR)	x		Verify that a flash sector is erased
03h	Read 1s Page (RD1PG)	x		Verify that a flash page is erased
04h	Read 1s Phrase (RD1PHR)	x		Verify that a flash phrase is erased
05h	Read into MISR (RDMISR)	x		Generate MISR signature for range of flash pages
12h	Read 1s IFR Sector (RD1ISCR)		x	Verify that an IFR sector is erased
13h	Read 1s IFR Page (RD1IPG)		x	Verify that an IFR page is erased
14h	Read 1s IFR Phrase (RD1IPHR)		x	Verify that an IFR phrase is erased
15h	Read IFR into MISR (RDIMISR)		x	Generate MISR signature for range of IFR pages

Table continues on the next page...

Table continued from the previous page...

FCMD	Command	Program flash 0	IFR	Function
23h	Program Page (PGMPG)	x	x	Program data to a flash or IFR page
24h	Program Phrase (PGMPHR)	x	x	Program data to a flash or IFR phrase
40h	Erase All (ERSALL)	x	x	Erase all flash and IFR space if enabled
42h	Erase Sector (ERSSCR)	x	x	Erase a flash sector

18.4.2.3 Flash commands impacted by Block Checker and FMC access protection

The following flash commands are impacted by Block Checker and FMC access protection:

- Program Phrase (PGMPHR)
- Program Page (PGMPG)
- Erase Sector (ERSSCR)

During execution of these flash commands, a sequence of AHB writes to flash or IFR space are required with validity checked by the Block Checker or FMC. When writes are enabled by the FMU, violations detected by the FMC are reported as an access error in the FMU. The other violations are handled by the Block Checker or FMC.

Table 147. Flash Write Access Checks

Flash Write Violation Condition	Block Checker	FMC	FMU
Flash or IFR address out-of-range	X		
Flash or IFR write permission check fails	X		
Write to IFR1 space		X	
Write width not 32-bit		X	
Writes not enabled by FSTAT[PEWEN]		X	
First write in sequence not properly aligned per FSTAT[PEWEN]			X
Non-sequential write after first aligned write in sequence			X
Bus permissions for the AHB write do not match the bus permissions of the APB write that launched the flash command			X

18.4.3 Flash command descriptions

This section describes all flash commands that can be launched by a valid command write sequence. The flash module sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in an FCCOB field for the specific command. Refer to the error handling table provided for each command.

The ACCERR, PVIOL, and CMDABT flags in the FSTAT register should be cleared prior to starting a command write sequence. As described in Launch the Command by Clearing CCIF, a new command cannot be launched while these flags are set.

Do not attempt to read a flash block while a flash command is running (CCIF = 0) on that same block. The flash module may return a transfer error to the FMC resulting in a bus fault. The exception to this rule is during program operations where the flash block is still readable after the command is launched up until the time the FSTAT[PERDY] flag is cleared.

CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

18.4.3.1 Read 1s All Command

The Read 1s All command checks if all flash and IFR space are in the erased state.

Table 148. Read 1s All Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	00h (RD1ALL)

Upon clearing CCIF to launch the Read 1s All command, the command controller:

- verifies that all flash and IFR space are in the erased state

If all flash and IFR space are not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s All operation completes.

Table 149. Read 1s All Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.2 Read 1s Block Command

The Read 1s Block command checks if a flash block is in the erased state.

Table 150. Read 1s Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	01h (RD1BLK)
2	Block address

Upon clearing CCIF to launch the Read 1s Block command, the command controller:

- verifies that the selected flash block is in the erased state

If the selected flash block is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s Block operation completes.

Table 151. Read 1s Block Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]

Table continues on the next page...

Table 151. Read 1s Block Command Error Handling (continued)

Error Condition	Error Bit
Address is not valid	FSTAT[ACCERR]
Address is not block aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.3 Read 1s Sector Command

The Read 1s Sector command checks if a flash sector is in the erased state.

Table 152. Read 1s Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	02h (RD1SCR)
2	Sector address

Upon clearing CCIF to launch the Read 1s Sector command, the command controller:

- verifies that the selected flash sector is in the erased state

If the selected flash sector is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s Sector operation completes.

Table 153. Read 1s Sector Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Address is not sector aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.4 Read 1s Page Command

The Read 1s Page command checks if a flash page is in the erased state.

Table 154. Read 1s Page Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	03h (RD1PG)
2	Page address

Upon clearing CCIF to launch the Read 1s Page command, the command controller:

- verifies that the selected flash page is in the erased state

If the selected flash page is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s Page operation completes.

Table 155. Read 1s Page Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Address is not page aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.5 Read 1s Phrase Command

The Read 1s Phrase command checks if a flash phrase is in the erased state.

Table 156. Read 1s Phrase Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	04h (RD1PHR)
2	Phrase address

Upon clearing CCIF to launch the Read 1s Phrase command, the command controller:

- verifies that the selected flash phrase is in the erased state

If the selected flash phrase is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s Phrase operation completes.

Table 157. Read 1s Phrase Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Address is not phrase aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.6 Read into MISR command

The Read into MISR operation generates a signature based on the contents of the selected flash memory using an embedded MISR.

Table 158. Read into MISR Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	05h (RDMISR) [7:0]
2	Starting flash page address
3	Ending flash phrase address
4	Word 0 of MISR seed
5	Word 1 of MISR seed
6	Word 2 of MISR seed
7	Word 3 of MISR seed
Returned values	
4	Word 0 of MISR signature
5	Word 1 of MISR signature
6	Word 2 of MISR signature
7	Word 3 of MISR signature

Upon clearing CCIF to launch the Read into MISR command, the command controller:

1. initializes the MISR with the seed provided
2. processes flash data starting with the starting flash page address provided
3. continues until the ending flash phrase address has been processed (must be the last phrase in a page)
4. returns the resulting signature into the FCCOB register bank unless an uncorrectable ECC fault is detected

The CCIF flag sets after the Read into MISR operation completes. If the operation is aborted by setting FCTRL[ABTREQ], a signature will not be returned if the abort occurs prior to the operation writing the signature.

MISR polynomial is $X^{128} + X^{126} + X^{101} + X^{99} + 1$.

Table 159. Read into MISR Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Starting address is not flash page aligned	FSTAT[ACCERR]
Ending address is not flash phrase aligned	FSTAT[ACCERR]
Ending address is not the last phrase in a page	FSTAT[ACCERR]
Starting address is larger than the ending address	FSTAT[ACCERR]
Requested range crosses a flash block boundary	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Uncorrectable ECC error is detected	FSTAT[FAIL]

18.4.3.7 Read 1s IFR Sector Command

The Read 1s IFR Sector command checks if an IFR sector is in the erased state.

Table 160. Read 1s IFR Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	12h (RD1ISCR)
2	IFR sector address

Upon clearing CCIF to launch the Read 1s IFR Sector command, the command controller:

- verifies that the selected IFR sector is in the erased state

If the selected IFR sector is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s IFR Sector operation completes.

Table 161. Read 1s IFR Sector Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Address is not IFR sector aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.8 Read 1s IFR Page Command

The Read 1s IFR Page command checks if an IFR page is in the erased state.

Table 162. Read 1s IFR Page Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	13h (RD1IPG)
2	IFR page address

Upon clearing CCIF to launch the Read 1s IFR Page command, the command controller:

- verifies that the selected IFR page is in the erased state

If the selected IFR page is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s IFR Page operation completes.

Table 163. Read 1s IFR Page Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Address is not IFR page aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]

Table continues on the next page...

Table 163. Read 1s IFR Page Command Error Handling (continued)

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.9 Read 1s IFR Phrase Command

The Read 1s IFR Phrase command checks if an IFR phrase is in the erased state.

Table 164. Read 1s IFR Phrase Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	14h (RD1IPHR)
2	IFR phrase address

Upon clearing CCIF to launch the Read 1s IFR Phrase command, the command controller:

- verifies that the selected IFR phrase is in the erased state

If the selected IFR phrase is not in the erased state, the FSTAT[FAIL] flag is set. The CCIF flag sets after the Read 1s IFR Phrase operation completes.

Table 165. Read 1s IFR Phrase Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Address is not IFR phrase aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.10 Read IFR into MISR command

The Read IFR into MISR operation generates a signature based on the contents of the selected IFR space using an embedded MISR.

Table 166. Read IFR into MISR Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	15h (RDIMISR) [7:0]
2	Starting IFR page address
3	Ending IFR phrase address
4	Word 0 of MISR seed
5	Word 1 of MISR seed
6	Word 2 of MISR seed

Table continues on the next page...

Table 166. Read IFR into MISR Command FCCOB Requirements (continued)

FCCOB Number	FCCOB Contents
7	Word 3 of MISR seed
Returned values	
4	Word 0 of MISR signature
5	Word 1 of MISR signature
6	Word 2 of MISR signature
7	Word 3 of MISR signature

Upon clearing CCIF to launch the Read IFR into MISR command, the command controller:

1. initializes the MISR with the seed provided
2. processes IFR data starting with the starting IFR page address provided
3. continues until the ending IFR phrase has been processed (must be the last phrase in a page)
4. returns the resulting signature into the FCCOB register bank unless an uncorrectable ECC fault is detected

The CCIF flag sets after the Read IFR into MISR operation completes. If the operation is aborted by setting FCTRL[ABTREQ], a signature will not be returned if the abort occurs prior to the operation writing the signature.

MISR polynomial is $X^{128} + X^{126} + X^{101} + X^{99} + 1$.

Table 167. Read IFR into MISR Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Address is not valid	FSTAT[ACCERR]
Starting address is not IFR page aligned	FSTAT[ACCERR]
Ending address is not IFR phrase aligned	FSTAT[ACCERR]
Ending address is not the last phrase in a page	FSTAT[ACCERR]
Starting address is larger than the ending address	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Uncorrectable ECC error is detected	FSTAT[FAIL]

18.4.3.11 Program Page command

The Program Page operation programs 32 words of data to a previously erased flash or IFR page.

CAUTION

The page must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a page is not allowed.

Table 168. Program Page Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	23h (PGMPG) [7:0]

Upon clearing CCIF to launch the Program Page command, the command controller:

1. sets FSTAT[PEWEN] to enable writes to the flash and IFR space in the system memory map
2. waits for the user to write 32 consecutive words to the flash or IFR space with the first write being page aligned
3. clears FSTAT[PEWEN] field
4. sets FSTAT[PERDY]
5. waits for the user to clear FSTAT[PERDY]
6. programs the data into the targeted page written to in step 2
7. verifies bits to be programmed are programmed to the program verify level

The CCIF flag sets after the Program Page operation completes. The Program Page operation can be aborted by setting FCTRL[ABTREQ]. If the operation is aborted, the operation terminates with the FSTAT[CMDABT] flag set. Once CCIF is set, the Program Page command may be repeated (same addresses, same data) to complete the program operation without erasing the sector containing the targeted page. Invalid writes to flash or IFR space based on Block Checker or FMC access protection are trapped outside of the flash module.

CAUTION

The Program Page operation will not react to the early indicator for radio activity. Therefore, FSTAT[PERDY] must not be cleared if there is a concern about inadequate power available to support both radio activity and the Program Page operation.

Table 169. Program Page Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Protection level of the writes to flash or IFR space do not match the protection level of the command write sequence	FSTAT[ACCERR]
Initial address is not page aligned and subsequent addresses phrase aligned	FSTAT[ACCERR]
Number of writes to flash or IFR space is more than expected	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.11.1 Page programming

The standard process of programming one or more flash or IFR pages is as follows:

1. If required, launch the Erase Sector command to erase the flash or IFR sector to be programmed (repeat for additional sectors).
2. Launch the Program Page command to enable writes to the flash space.
3. After the command controller sets FSTAT[PEWEN], write 32 consecutive words to the flash space with the first word page aligned. The data written to the flash space is not readable until the program operation has successfully completed.
4. After the flash page has been written, the command controller will stall and set FSTAT[PERDY].
5. Clear PERDY allowing the command controller to resume and program the 32 words into the flash or IFR space written to in step 3.
6. Wait for CCIF to set indicating the program operation has completed.
7. To program additional flash or IFR pages in previously erased flash memory, repeat steps 2 through 6.

18.4.3.12 Program Phrase command

The Program Phrase operation programs 4 words of data to a previously erased flash or IFR phrase.

CAUTION

The phrase must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a phrase is not allowed.

Table 170. Program Phrase Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	24h (PGMPHR) [7:0]

Upon clearing CCIF to launch the Program Phrase command, the command controller:

1. sets FSTAT[PEWEN] to enable writes to the flash and IFR space in the system memory map
2. waits for the user to write 4 consecutive words to the flash or IFR space with the first write being phrase aligned
3. clears FSTAT[PEWEN] field
4. sets FSTAT[PERDY]
5. waits for the user to clear FSTAT[PERDY]
6. programs the data into the targeted phrase written to in step 2.
7. verifies bits to be programmed are programmed to the program verify level

The CCIF flag sets after the Program Phrase operation completes. The Program Phrase operation can be aborted by setting FCTRL[ABTREQ]. If the operation is aborted, the operation terminates with the FSTAT[CMDABT] flag set. Once CCIF is set and the CMDABT flag is cleared, the Program Phrase command may be repeated (same addresses, same data) to complete the program operation without erasing the sector containing the targeted phrase. Invalid writes to flash or IFR space based on Block Checker or FMC access protection are trapped outside of the flash module.

CAUTION

The Program Phrase operation will not react to the early indicator for radio activity. Therefore, FSTAT[PERDY] must not be cleared if there is a concern about inadequate power available to support both radio activity and the Program Phrase operation.

Table 171. Program Phrase Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Protection level of the writes to flash or IFR space do not match the protection level of the command write sequence	FSTAT[ACCERR]
Address is not phrase aligned	FSTAT[ACCERR]
Number of writes to flash or IFR space is more than expected	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.12.1 Phrase programming

The standard process of programming one or more flash or IFR phrases is as follows:

1. If required, launch the Erase Sector command to erase the flash or IFR sector to be programmed (repeat for additional sectors).
2. Launch the Program Phrase command to enable writes to the flash and IFR space.
3. After the command controller sets FSTAT[PEWEN], write 4 consecutive words to the flash or IFR space with the first word phrase aligned. The data written to the flash or IFR space is not readable until the program operation has successfully completed.
4. After the flash or IFR phrase has been written, the command controller will stall the operation and set FSTAT[PERDY].
5. Clear PERDY allowing the command controller to resume and program the 4 words into the flash or IFR space written to in step 3.
6. Wait for CCIF to set indicating the program operation has completed.
7. To program additional flash or IFR phrases in previously erased flash memory, repeat steps 2 through 6.

18.4.3.13 Erase All command

The Erase All operation erases all flash and IFR space if enabled by the MCU.

Table 172. Erase All Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	40h (ERSALL) [7:0]

Upon clearing CCIF to launch the Erase All command, the command controller:

1. erases flash block 0 sector-by-sector
2. verifies flash block 0 is erased; if not, skips IFR space in block 0
3. erases IFR space in block 0 sector-by-sector
4. verifies IFR space in block 0 is erased
5. clears FSTAT[CMDP] flag

The CCIF flag sets after the Erase All operation completes. If either the flash or IFR space is not in the erased state, FSTAT[FAIL] sets.

CAUTION

The Erase All operation will not react to the early indicator for radio activity. Therefore, the Erase All command must not be launched if there is a concern about inadequate power available to support both radio activity and the Erase All operation.

Table 173. Erase All Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Command is disabled by the MCU	FSTAT[PVIOL]
Operation is aborted	FSTAT[CMDABT]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.3.14 Erase Sector command

The Erase Sector operation erases a flash of IFR sector to prepare the sector for programming.

Table 174. Erase Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents
0	42h (ERSSCR) [7:0]

Upon clearing CCIF to launch the Erase Sector command, the command controller:

1. sets FSTAT[PEWEN] to enable writes to the flash and IFR space in the system memory map
2. waits for the user to write 4 consecutive words to the flash or IFR space with the first write being phrase (or sector) aligned
3. clears FSTAT[PEWEN] field
4. sets FSTAT[PERDY]
5. waits for the user to clear FSTAT[PERDY]
6. erases the selected flash or IFR sector
7. verifies the selected flash or IFR sector is erased

If the selected flash or IFR sector is not in the erased state, FSTAT[FAIL] is set. The CCIF flag sets after the Erase Sector operation completes. The Erase Sector operation can be aborted by setting FCTRL[ABTREQ]. If the operation is aborted, the operation terminates with the FSTAT[CMDABT] flag set. Once CCIF is set, the Erase Sector command may be repeated to complete the erase operation. Invalid writes to flash or IFR space based on Block Checker or FMC access protection are trapped outside of the flash module.

CAUTION

The Erase Sector operation will not react to the early indicator for radio activity. Therefore, the Erase Sector command must not be launched if there is a concern about inadequate power available to support both radio activity and the Erase Sector operation.

Table 175. Erase Sector Command Error Handling

Error Condition	Error Bit
FMU parameters not loaded from IFR1 during initialization	FSTAT[ACCERR]
Protection level of the writes to flash or IFR space do not match the protection level of the command write sequence	FSTAT[ACCERR]
Address is not flash or IFR phrase aligned	FSTAT[ACCERR]
Operation is aborted	FSTAT[CMDABT]
IFR Sector is protected (see FCNFG register)	FSTAT[PVIOL]
FMU parameters not loaded from IFR1 during initialization	FSTAT[FAIL]
Verify fails	FSTAT[FAIL]

18.4.4 Mass Erase

If enabled by the MCU, MCU resources can be used to launch a mass erase operation without accessing the flash registers.

To know how to enable/disable mass erase for your device, refer the chip-specific section.

The status of the mass erase request is reflected in the FCNFG[ERSREQ] bit. When launched, the mass erase operation occurs regardless of the state of the CMDP, ACCERR, PVIOL, or CMDABT flags in the FSTAT register but requires FSTAT[CCIF] to be set and the MCU to enable mass erase. When mass erase is launched with CCIF set, the CMDP, CCIF, CWSABT, PVIOL, and CMDABT flags in the FSTAT register clear. The ACCERR and FAIL flags also clear unless the FMU parameters were not loaded from IFR1 during initialization in which case the mass erase operation terminates without performing any erase of flash memory.

Upon successful launch of the mass erase operation, the command controller (same steps as the Erase All command):

1. erases flash block 0 sector-by-sector
2. verifies flash block 0 is erased; if not, skips IFR space in block 0
3. erases IFR space in block 0 sector-by-sector
4. verifies IFR space in block 0 is erased

The CCIF flag sets and the ERSREQ bit clears once the operation completes and the FSTAT[FAIL] flag sets if any verify step fails during the mass erase operation. This method of erasing the flash memory cannot be aborted by setting FCTRL[ABTREQ].

CAUTION

The mass erase operation occurs regardless of the state of the early indicator for radio activity and will not react to this early indicator during the operation. Therefore, mass erase must not be launched if there is a concern about inadequate power available to support both radio activity and the mass erase operation.

18.5 Initialization

On each reset, the command controller initializes the following based on IFR1 content:

1. FCTRL[RWSC]
2. FMU Parameters (FMUPARM)

The FSTAT[CCIF] flag is cleared throughout the initialization flow. The flash module also holds off all access to flash registers and flash memory during initialization. Completion of the initialization flow is marked by setting FSTAT[CCIF]. If an uncorrectable ECC error is detected during initialization, the FSTAT[FAIL] flag sets. The FAIL flag also sets if FMU parameters are not loaded from IFR1 during initialization. In this case, attempts to launch a flash command terminate with the ACCERR and FAIL flags set. The recommendation is to refrain from launching flash commands if the FAIL flag sets after initialization.

If a reset occurs while any flash command is in progress, that command is immediately aborted. The state of the flash memory or IFR space being programmed or erased is not guaranteed. Command operations do not automatically resume after exiting reset.

18.6 Memory map and registers

This section describes the flash memory map for IFR/IFR1 space and registers. See the MCU system memory map for the location of all flash memory space and registers. Data read from unimplemented memory space in the flash module is undefined.

18.6.1 IFR description

Each flash block contains IFR space separate from the main flash memory. IFR space is memory mapped (see the MCU system memory map). IFR space is divided into IFR pages programmable using the Program Page command. IFR pages are further divided into IFR phrases programmable using the Program Phrase command. Once programmed, IFR phrases can be reprogrammed after a successful erase using the Erase Sector command operation unless the IFR sector is protected by the FCNFG register.

Table 176. IFR map

Offset Byte Address	IFR Field Name	IFR Field Size (Bytes)	IFR Field Description
Block 0 IFR Sector 0			
0000h-1FFFh	USER	8,192	Available for customer usage.
Block 0 IFR Sector 1			
2000h-3FFFh	ROMCFG	8,192	ROM and ISP configuration. Refer to ROM Bootloader chapter for more information.

Table continues on the next page...

Table 176. IFR map (continued)

Offset Byte Address	IFR Field Name	IFR Field Size (Bytes)	IFR Field Description
Block 0 IFR Sector 2			
4000h-5FFFh	FLW0	8,192	FLW configuration set 0. Refer to ROM Bootloader chapter for more information.
Block 0 IFR Sector 3			
6000h-7FFFh	FLW1	8,192	FLW configuration set 1. Refer to ROM Bootloader chapter for more information.

Each flash block also contains IFR1 space separate from the main flash memory. IFR1 space in program flash block 0 is memory mapped for read-only access (see the MCU system memory map). IFR1 space cannot be erased or programmed by the user.

Table 177. IFR1 map

Offset Byte Address	IFR1 Field Name	IFR1 Field Size (Bytes)	IFR1 Field Description
Block 0 IFR1 Sector			
0000h-15FFh		5,632	Reserved
1600h-16FFh	FMUPARM	256	FMU Parameters
1700h-1FFFh		2,304	Reserved

18.6.2 Register descriptions

The flash module contains a set of memory-mapped control and status registers.

NOTE

While a command is running (FSTAT[CCIF]=0), register writes are allowed to FSTAT[PERDY,DFDIF,CWSABT], FCNFG, and FCTRL[ABTREQ,FDFD,RWSC].

18.6.2.1 Flash register descriptions

18.6.2.1.1 Flash memory map

FMU0 base address: 4002_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Flash Status Register (FSTAT)	32	RW	See section
4h	Flash Configuration Register (FCNFG)	32	RW	See section
8h	Flash Control Register (FCTRL)	32	RW	See section
10h	Flash Common Command Object Registers (FCCOB0)	32	RW	0000_0000h
14h	Flash Common Command Object Registers (FCCOB1)	32	RW	0000_0000h
18h	Flash Common Command Object Registers (FCCOB2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1Ch	Flash Common Command Object Registers (FCCOB3)	32	RW	0000_0000h
20h	Flash Common Command Object Registers (FCCOB4)	32	RW	0000_0000h
24h	Flash Common Command Object Registers (FCCOB5)	32	RW	0000_0000h
28h	Flash Common Command Object Registers (FCCOB6)	32	RW	0000_0000h
2Ch	Flash Common Command Object Registers (FCCOB7)	32	RW	0000_0000h

18.6.2.1.2 Flash Status Register (FSTAT)

Offset

Register	Offset
FSTAT	0h

Function

The FSTAT register reports the operational status of the flash module.

NOTE

When set, ACCERR, PVIOL, and CMDABT flags prevent the launch of any more commands until the flag is cleared (by writing a one to it).

CAUTION

When clearing the DFDIF flag, be careful not to use read-modify-write on the entire FSTAT register and inadvertently clear CCIF to launch a command.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERDY	0					PEWEN		0						SALV_US...	DFDIF
W	W1C															W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDDID				CMDP	0	CMDPRT		CCIF	CWSABT	ACCERR	PVIOL	0	CMDABT	0	FAIL
W									W1C	W1C	W1C	W1C		W1C		
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	u

Fields

Field	Function
31 PERDY	Program-Erase Ready Control/Status Flag The PERDY flag is set by the command controller when a program or sector erase command operation has successfully completed the write phase. The command controller will stall the command operation until the user clears the PERDY flag by writing a 1 to PERDY. Failure to clear PERDY will leave the command controller stalled but the flash or IFR space targeted by the command operation will remain readable by the FMC. If FCTRL[ABTREQ] is set while the PERDY flag is set, the command controller will clear PERDY, abort the command operation, and set both CCIF and CMDABT. 0b - Program or sector erase command operation not stalled 1b - Program or sector erase command operation ready to execute
30-26 —	Reserved
25-24 PEWEN	Program-Erase Write Enable Control During program or sector erase command operations, writes are enabled to flash or IFR space in the system memory map. For phrase programming or sector erase, writes are enabled for 4 consecutive words (16 bytes) with address phrase aligned. For page programming, writes are enabled for 8 consecutive phrases (128 bytes) with address page aligned. For programming, data written to erased flash or IFR space for program commands is not available until the program operation has successfully completed. PEWEN will not assert if FMU parameters were not loaded from IFR1 during initialization. 00b - Writes are not enabled 01b - Writes are enabled for one flash or IFR phrase (phrase programming, sector erase) 10b - Writes are enabled for one flash or IFR page (page programming) 11b - Reserved
23-18 —	Reserved
17 SALV_USED	Salvage Used for Erase operation This flag indicates salvage was used during the last Erase Sector, Erase All, or Mass Erase operation and is valid at the end of the operation. Salvage involves using ECC to pass erase verify allowing single-bit faults per phrase verified. This flag remains valid until any flash command or Mass Erase operation launches at which time the flag clears. Use of salvage does not guarantee that the erase operation succeeds but only that salvage was used on at least one phrase. 0b - Salvage not used during last operation 1b - Salvage used during the last erase operation
16 DFDIF	Double Bit Fault Detect Interrupt Flag The DFDIF flag indicates an uncorrectable ECC fault was detected during a valid flash read access from the FMC. The DFDIF flag is cleared by writing a 1 to DFDIF. Writing a 0 to DFDIF has no effect. 0b - Double bit fault not detected during a valid flash read access

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Double bit fault detected (or FCTRL[FDFD] is set) during a valid flash read access
15-12 CMDDID	Command domain ID While the CMDP flag is set, the CMDDID bits reflect the domain ID of the write transaction controlling a command write sequence or aborting a command operation. Writes to load an FCCOB register, clear CCIF, clear PERDY, or write FCTRL[LSACTIVE] must have the same domain ID as the write transaction that started the command write sequence. The CMDDID bits persist after the CMDP flag clears until the CMDP flag sets again at the start of a new command write sequence. Writes to clear the CWSABT, ACCERR, PVIOL, or CMDABT flags are not influenced by the state of the CMDDID bits.
11 CMDP	Command protection status flag The CMDP flag sets when a command write sequence starts with a write to load an existing FCCOB register or clear CCIF while ACCERR, PVIOL, and CMDABT are clear. Clearing the CWSABT, ACCERR, PVIOL, or CMDABT flags as part of a command write sequence will not set the CMDP flag. While the CMDP flag is set, the protection level of any subsequent write transaction to load an FCCOB register, clear CCIF, clear PERDY, or write FCTRL[LSACTIVE] must match the CMDPRT bits and the domain ID must match the CMDDID bits. The CMDP flag remains set until cleared when CCIF is set after a command operation or Power Down recovery completes unless the CMDABT flag is set. The CMDP flag is cleared even if the operation ends with ACCERR or PVIOL set. When the CMDP flag is cleared, the CMDPRT and CMDDID bits remain unchanged. The CMDP flag is also cleared by a mass erase request with CCIF set. 0b - Command protection level and domain ID are stale 1b - Command protection level (CMDPRT) and domain ID (CMDDID) are set
10 —	Reserved
9-8 CMDPRT	Command protection level While the CMDP flag is set, the CMDPRT bits reflect the secure/privilege protection level of the write transaction controlling a command write sequence or aborting a command operation. Writes to load an existing FCCOB register, clear CCIF, clear PERDY, or write FCTRL[LSACTIVE] must occur with the same protection level as indicated by the CMDPRT bits. Writes to clear the CWSABT, ACCERR, PVIOL, or CMDABT flags are not influenced by the state of the CMDPRT bits. If FCTRL[ABTREQ] is set while CCIF is high or low, the CMDPRT bits will change to reflect the security/privilege protection level of the write transaction that set the ABTREQ bit. 00b - Secure, normal access 01b - Secure, privileged access 10b - Nonsecure, normal access 11b - Nonsecure, privileged access
7 CCIF	Command Complete Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The CCIF flag indicates that a flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a flash command. The CCIF flag stays low until the command completes. The CCIF flag is cleared by a mass erase request with CCIF set. The CCIF flag is also cleared upon entry into Deep Sleep, Power Down, and Deep Power Down modes and stays low until the end of recovery from these non-active power modes. The CCIF flag is reset to 0 but is set to 1 at the end of flash initialization. 0b - Flash command, initialization, or power mode recovery in progress 1b - Flash command, initialization, or power mode recovery has completed
6 CWSABT	Command Write Sequence Abort Flag The CWSABT flag indicates whether a request to abort a command write sequence prior to command launch has been granted. If FCTRL[ABTREQ] is set while CCIF is high, the CWSABT flag will get set. Once CWSABT is set, ABTREQ will get cleared. While the CWSABT flag is set, ABTREQ cannot be set. The CWSABT flag is cleared by writing a 1 to CWSABT while CCIF is set or clear and ABTREQ is clear. Writing a 0 to the CWSABT flag has no effect. The CWSABT flag is cleared by a mass erase request with CCIF set. 0b - Command write sequence not aborted 1b - Command write sequence aborted
5 ACCERR	Command Access Error Flag The ACCERR flag indicates an illegal attempt was made to launch a flash command due to a violation of the command write sequence or by providing invalid command parameters. The ACCERR flag also sets during command execution if FMU parameters were not loaded from IFR1 during initialization. While the ACCERR flag is set, the CCIF flag cannot be cleared to launch a command. The ACCERR flag is cleared by writing a 1 to ACCERR while CCIF is set. Writing a 0 to the ACCERR flag has no effect. The ACCERR flag is cleared by a mass erase request with CCIF set but is not cleared during Power Down recovery. 0b - No access error detected 1b - Access error detected
4 PVIOL	Command Protection Violation Flag The PVIOL flag indicates an attempt was made to modify a protected area of flash memory during a command operation. While the PVIOL flag is set, the CCIF flag cannot be cleared to launch a command. The PVIOL flag is cleared by writing a 1 to PVIOL while CCIF is set. Writing a 0 to the PVIOL flag has no effect. The PVIOL flag is cleared by a mass erase request with CCIF set but is not cleared during Power Down recovery. 0b - No protection violation detected 1b - Protection violation detected
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 CMDABT	Command Abort Flag The CMDABT flag indicates FCTRL[ABTREQ] was set during a command operation while CCIF was clear. This event will result in the termination of the operation unless it occurs during the exit routine of the operation. Once CMDABT is set, ABTREQ will get cleared. While the CMDABT flag is set, ABTREQ cannot be set and the CCIF flag cannot be cleared to launch a command. The CMDABT flag is cleared by writing a 1 to CMDABT while CCIF is set and ABTREQ is clear. Writing a 0 to the CMDABT flag has no effect. The CMDABT flag is cleared by a mass erase request with CCIF set. 0b - No command abort detected 1b - Command abort detected
1 —	Reserved
0 FAIL	Command Fail Flag The FAIL flag is set if an error is detected during execution of a flash command, mass erase operation, or flash initialization. If the FAIL flag sets along with ACCERR after launching a flash command or mass erase operation, FMU parameters were not loaded from IFR1 during initialization and a clean POR where FMU parameters are successfully loaded is required to clear the FAIL flag. As a status flag, this bit cannot (and need not) be cleared by the user like some of the other flags in this register. The value of the FAIL flag for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the FAIL flag is cleared unless FMU parameters were not loaded from IFR1 during initialization. The FAIL flag is cleared by a mass erase request with CCIF set unless FMU parameters were not loaded from IFR1 during initialization. The FAIL flag is not cleared during Power Down recovery. 0b - Error not detected 1b - Error detected

18.6.2.1.3 Flash Configuration Register (FCNFG)

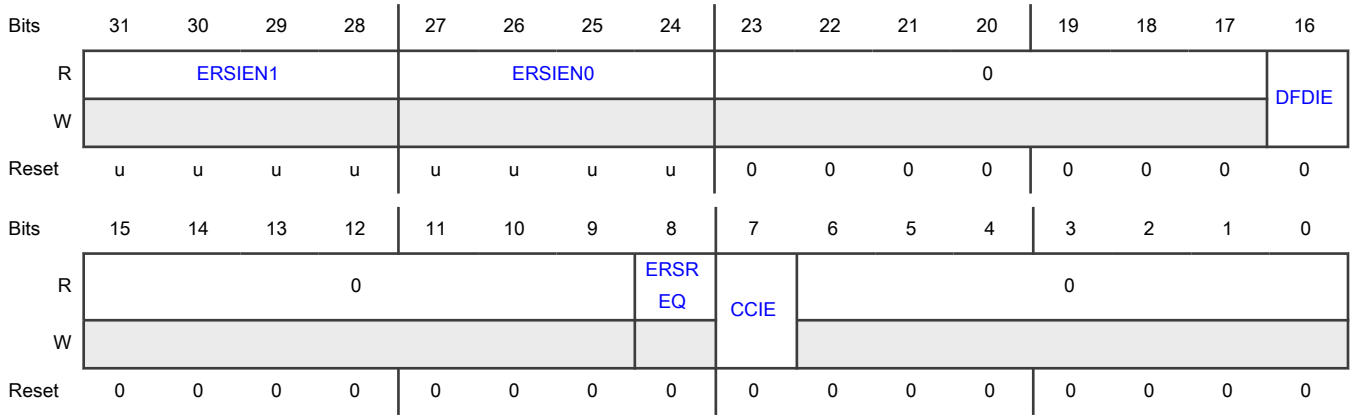
Offset

Register	Offset
FCNFG	4h

Function

This register provides information on the current functional state of the flash module.

Diagram



Fields

Field	Function
31-28 ERSIEN1	Erase IFR Sector Enable - Block 1 (for dual block configs) This field controls the ability to erase the IFR sectors in block 1 using the ERSSCR command. These bits are loaded from sideband signals during initialization. Note: These bits have no effect on single block configurations. Note: The ERSALL command and MCU mass erase request do not adhere to these restrictions. X equals 0, 1, 2, or 3. Any and all bits in the field are independently configured. 0000b - Block 1 IFR Sector X is protected from erase by ERSSCR command 0001b - Block 1 IFR Sector X is not protected from erase by ERSSCR command
27-24 ERSIEN0	Erase IFR Sector Enable - Block 0 This field controls the ability to erase the IFR sectors in block 0 using the ERSSCR command. These bits are loaded from sideband signals during initialization. Note: The ERSALL command and MCU mass erase request do not adhere to these restrictions. X equals 0, 1, 2, or 3. Any and all bits in the field are independently configured. 0000b - Block 0 IFR Sector X is protected from erase by ERSSCR command 0001b - Block 0 IFR Sector X is not protected from erase by ERSSCR command
23-17 —	Reserved
16 DFDIE	Double Bit Fault Detect Interrupt Enable The DFDIE bit controls interrupt generation when an uncorrectable ECC fault is detected during a valid flash read access from the platform flash controller. 0b - Double bit fault detect interrupt disabled 1b - Double bit fault detect interrupt enabled. An interrupt request is generated whenever the FSTAT[DFDIF] flag is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-9 —	Reserved
8 ERSREQ	Mass Erase Request This bit indicates whether a sideband request has been received by the command controller to execute the Mass Erase operation. ERSREQ is not directly writable but sets when a Mass Erase request is received by the flash module while CCIF is set. ERSREQ is cleared by the command controller when the operation completes. 0b - No request or request complete 1b - Request to run the Mass Erase operation
7 CCIE	Command Complete Interrupt Enable The CCIE bit controls interrupt generation when a flash command completes. 0b - Command complete interrupt disabled 1b - Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6-0 —	Reserved

18.6.2.1.4 Flash Control Register (FCTRL)

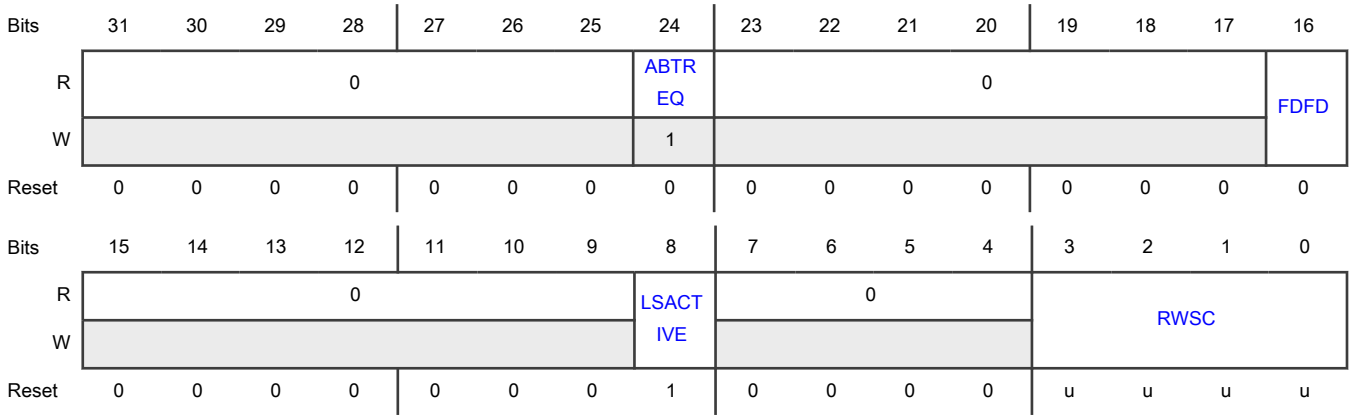
Offset

Register	Offset
FCTRL	8h

Function

The flash control register controls activity associated with flash memory reads and command operations.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ABTREQ	Abort Request With FSTAT[CMDP] set, setting the ABTREQ bit allows a user to abort a command write sequence or command operation. The ABTREQ bit can only be set by a user with the same domain ID as indicated by FSTAT[CMDDID] and the same or higher security/privilege protection level than is indicated by FSTAT[CMDPRT]. While ABTREQ is set, CCIF cannot be cleared to launch a command. If FSTAT[CCIF] is high when ABTREQ is set, the command write sequence will abort and ownership transferred to the user setting the ABTREQ bit. The FSTAT[CWSABT] flag will set and the CMDPRT field will be updated based on the write transaction that sets the ABTREQ bit. The ABTREQ bit will then be cleared automatically. If FSTAT[CCIF] is low when ABTREQ is set, the command operation will abort and ownership transferred to the user setting the ABTREQ bit. The FSTAT[CMDABT] flag will set and the CMDPRT field will be updated based on the write transaction that sets the ABTREQ bit. The ABTREQ bit will then be cleared automatically. While CWSABT or CMDABT are high, writes to ABTREQ will be ignored. 0b - No request to abort a command write sequence 1b - Request to abort a command write sequence
23-17 —	Reserved
16 FDFD	Force Double Bit Fault Detect The FDFD bit enables the user to emulate the setting of the FSTAT[DFDIF] flag to check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. 0b - FSTAT[DFDIF] sets only if a double bit fault is detected during a valid flash read access from the platform flash controller

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - FSTAT[DFDIF] sets during any valid flash read access from the platform flash controller. An interrupt request is generated if the DFDIE bit is set.
15-9 —	Reserved
8 LSACTIVE	Low speed active mode This bit controls entry into the low speed active mode. Flash memory reads are supported in low speed active mode. All flash commands are supported in low speed active mode but the LSACTIVE bit is not writable while a flash command is executing (CCIF=0) or during a non-active power mode. If the FSTAT[CMDP] is set, the protection level of any write transaction to the LSACTIVE bit must match the CMDPRT bits and the domain ID must match the CMDDID bits. Note: If a device does not support low speed active mode, the LSACTIVE bit will reset to 0b and will not be writable. 0b - Full speed active mode requested 1b - Low speed active mode requested
7-4 —	Reserved
3-0 RWSC	Read Wait-State Control These bits control the number of wait-states added to account for the ratio of system clock period to flash access time during full speed and low speed active power modes. Ratios greater than one require non-zero settings for the RWSC field for proper flash accesses. The required settings are documented in the device data sheet. 0h - no additional wait-states are added (single cycle access) 1h - 1 additional wait-state is added 2h - 2 additional wait-states are added ... Fh - 15 additional wait-states are added These bits are loaded from IFR1 during initialization.

18.6.2.1.5 Flash Common Command Object Registers (FCCOB0 - FCCOB7)

Offset

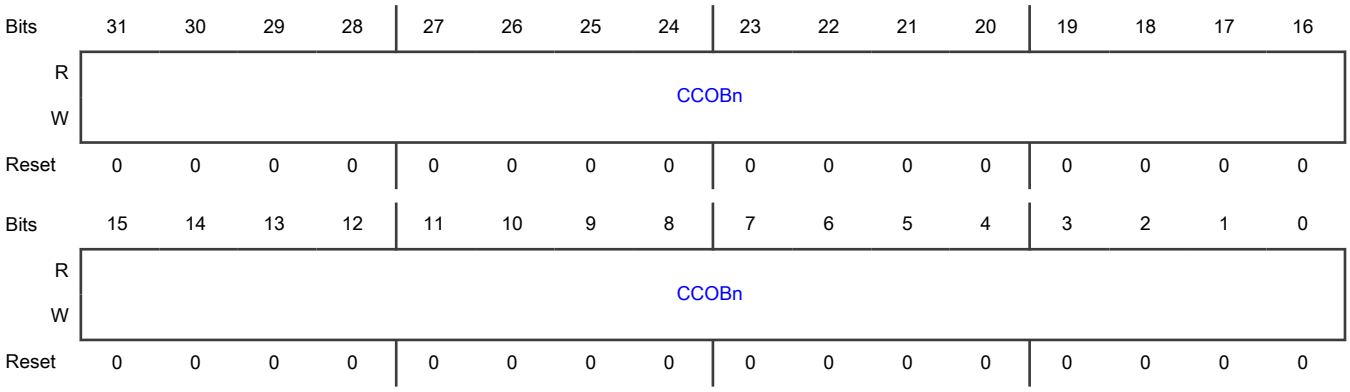
For a = 0 to 7:

Register	Offset
FCCOBa	10h + (a × 4h)

Function

The FCCOB register group provides fields for command codes and parameters. The individual words within the set append a 0-7 hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOB7.

Diagram



Fields

Field	Function
31-0 CCOBn	CCOBn The FCCOB register group provides a command code and relevant parameters to the command controller. The individual registers that compose the FCCOB data set can be written in any order, but you must provide all needed values, which vary from command to command. First, set up all required FCCOB fields and then initiate the command execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit, which locks all FCCOB parameter fields and they cannot be changed by the user until the command completes (CCIF returns to 1). No command buffering or queueing is provided; the next command can be loaded only after the current command completes. Some commands return information to the FCCOB registers. Any values returned to FCCOB are available for reading after the FSTAT[CCIF] flag returns to 1. The following table shows a generic flash command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific flash command, typically an address and/or data values.

NOTE

The command parameter table is written in terms of FCCOB Number (which is equivalent to the word number). This number is a reference to the FCCOB register name and is not the register address.

FCCOB Number[1]	Typical Command Parameter Contents [31:0]
0	FCMD [7:0] (flash command code, bits [31:8] are not writable)
1	FCMDOPT [7:0] (flash command options, bits [31:8] are not writable)
2	Flash address 1 (start)
3	Flash address 2 (end)
4	Data Word 0

Field	Function

FCCOB Number[1]	Typical Command Parameter Contents [31:0]
5	Data Word 1
6	Data Word 2
7	Data Word 3

1. Refers to FCCOB register name, not register address

Chapter 19
System Register File (REGFILE)

19.1 Chip-specific REGFILE information

Table 178. Reference links to related information

Topic	Related module	Reference
Full description	REGFILE	REGFILE
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

19.1.1 Module instances

This device has two instances of the REGFILE module, REGFILE0, and REGFILE1. REGFILE0 has no WAR or RAR registers while REGFILE1 has WAR and RAR registers.

19.1.2 Module reset

REGFILE0 retains its contents during low-voltage detect (LVD) events and is only reset during a power-on reset. REGFILE1 can be reset by every reset except wakeup from Power Down and Deep Power Down.

19.1.3 Low-power wakeup booth path

This device embeds a ROM bootloader code resident in the ROM area. Low-power wakeup boot flow requires the usage of REGFILE1 REG0 to store WAKEUP CRC data to verify the ROM wakeup word, therefore, REGFILE1 REG0 should not be used to store application data when low-power wakeup boot is enabled, but it should contain the valid WAKEUP CRC parameter to ensure proper boot sequence. See the Low_power wakeup path for more information about REGFILE1 REG0 usage during low-power boot flow.

19.2 Overview

The REGFILE module is used to store data, which is written to the Register File Registers, to be persistent.

19.3 Memory Map and Registers

19.3.1 REGFILE register descriptions

19.3.1.1 REGFILE memory map

REGFILE0 base address: 4002_1000h

REGFILE1 base address: 4002_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 1Ch	Register File Register a (REG0 - REG7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
100h	Write Access Register (WAR)	32	RW	0000_00FFh
104h	Read Access Register (RAR)	32	RW	0000_00FFh

19.3.1.2 Register File Register a (REG0 - REG7)

Offset

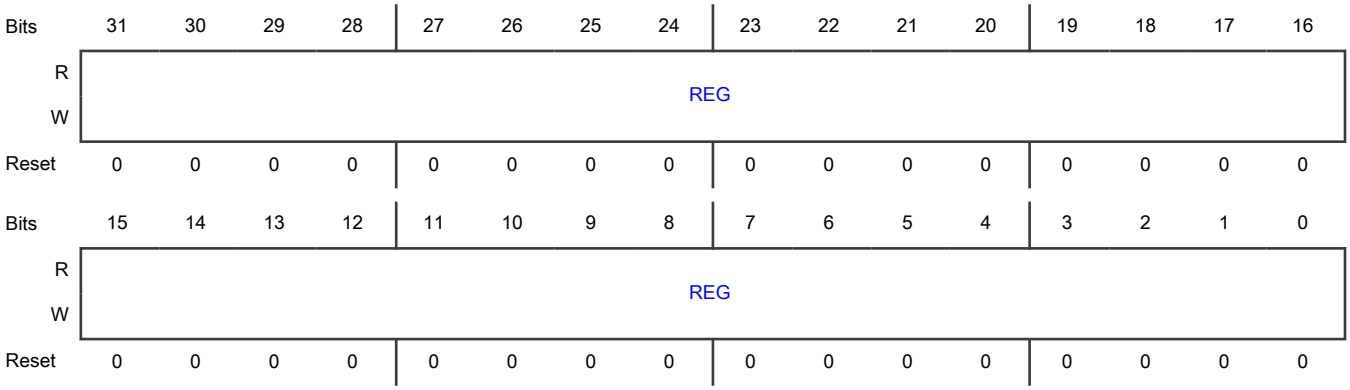
For a = 0 to 7:

Register	Offset
REGa	0h + (a × 4h)

Function

Each register can be accessed as 8-, 16-, or 32-bit.

Diagram



Fields

Field	Function
31-0 REG	Register File

19.3.1.3 Write Access Register (WAR)

Offset

Register	Offset
WAR	100h

Function

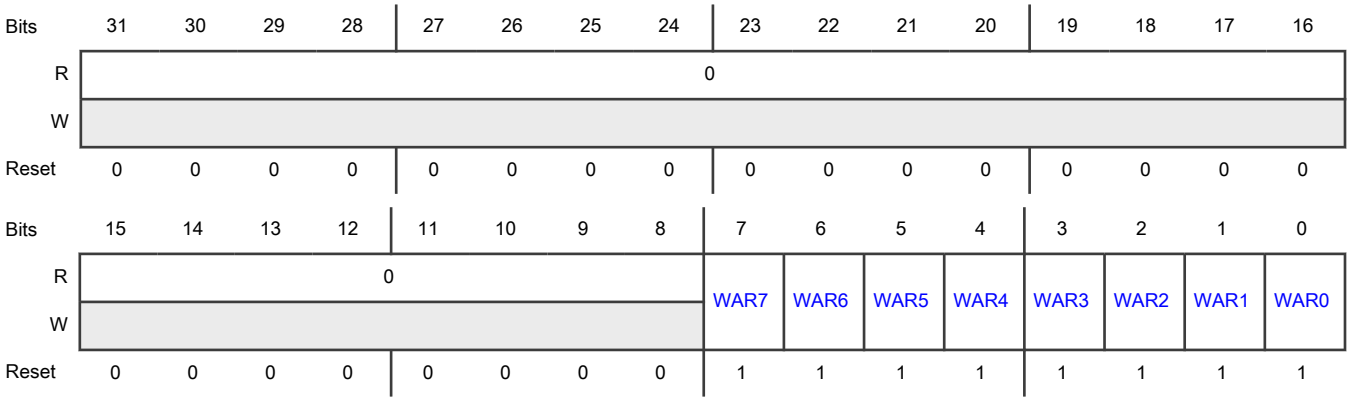
This register controls write access to the REG registers

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
REGFILE0	—	WAR
REGFILE1	WAR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 WARn	REGn Register Write Access 0b - Not allow to write to the REGn register and WARn field until next reset. 1b - Allow to write to the REGn register and WAR0 field.

19.3.1.4 Read Access Register (RAR)

Offset

Register	Offset
RAR	104h

Function

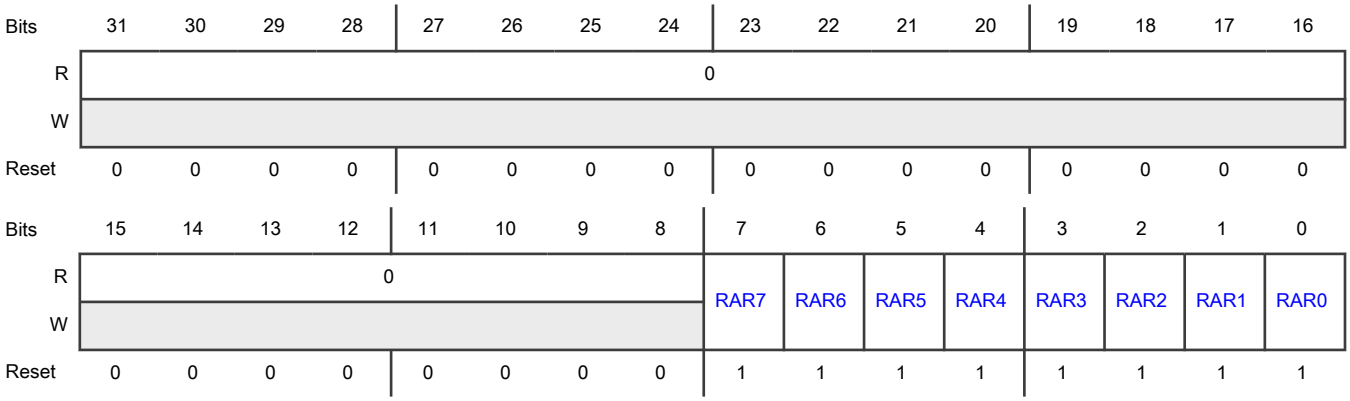
This register controls read access to the REG registers

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
REGFILE0	—	RAR
REGFILE1	RAR	—

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RARn	REGn Register Read Access 0b - Not allow to read the REGn register until next reset. Reading corresponding REGn register returns all 0. 1b - Allow to read the REGn register.

19.4 Functional description

REGFILE can be used to store data that must be persistent across the low power modes. Write the information into the Register File Registers and read the information back.

If a REGFILE instance supports Write Access Register (WAR) and Read Access Register (RAR), they can lock write and read operations on the Register File Registers until the next reset is performed.

NOTE

Not all the instances support WAR and RAR registers, refer to WAR and RAR registers for instances that support them.

19.4.1 Power mode

The REGFILE module is powered in all power modes unless it is in the VDD_SWITCH domain. In VDD_SWITCH domain, the REGFILE module is only powered down to Deep Power Down mode.

19.4.2 Reset

See chip-specific section for the reset of REGFILE module.

Chapter 20
Signal Muxing and Pinout

20.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. This chapter illustrates which of this device's signals are multiplexed on which external pin.

20.2 Pinout Table

48HVQFN	40HVQFN	Pin Name	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8	ALT9	ALT10	ALT11	Wakeup
2	1	PTB4		PTB4	LPSPI1_PCS3	LPUART1_CTS_b	LPI2C1_SDA	I3C0_SDA	TRGMUX0_IN0			FLEXIO0_D30			WUU0_P15
3	2	PTB5		PTB5	LPSPI1_PCS2	LPUART1_RTS_b	LPI2C1_SCL	I3C0_SCL	TRGMUX0_OUT0			FLEXIO0_D31			
4	3	VDD_IO_ABC	VDD_IO_ABC												
5	4	SWITCH_WAKEUP_B	SWITCH_WAKEUP_B												
6	5	VDD_SWITCH	VDD_SWITCH												
7	6	VOUT_SWITCH	VOUT_SWITCH												
8	7	PTA0		PTA0	CMP0_OUT	LPUART0_CTS_b	RF_GPO_11[1]	TPM0_CH4	FLEXIO0_D0	SWD_DIO					WUU0_P0
9	8	PTA1		PTA1	CMP1_OUT	LPUART0_RTS_b	RF_GPO_10[1]	TPM0_CH5	FLEXIO0_D1	SWD_CLK					
10	9	PTA4	ADC0_A10/	PTA4		RF_GPO_9[1]	TPM0_CLKIN	TRACE_SWO	FLEXIO0_D4	BOOT_CONFIG					WUU0_P2/RF_X

Table continues on the next page...

Table continued from the previous page...

48HVQFN	40HVQFN	Pin Name	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8	ALT9	ALT10	ALT11	Wake up
			CMP0_IN0												TAL_OUT_ENABLE[1]
11		PTA16	ADC0_A12	PTA16	LPSPI0_PCS0	EWM0_OUT_b	LPI2C0_SCLS	TPM0_CH4	LPUART0_RX	RF_GPO_8[1]		FLEXIO0_D5			RF_NOT_ALLOWED[1]
12	10	PTA17	ADC0_A13	PTA17	LPSPI0_SIN	EWM0_IN	LPI2C0_SDAS	TPM0_CH5	LPUART0_TX	RF_GPO_7[1]	RF_GPO_8[1]	FLEXIO0_D6		RF_EXT_XTAL_REQUEST/ RF_GPO_7[1]	WUU0_P3/ RF_NOT_ALLOWED[1]
13	11	PTA18	CMP1_IN1	PTA18	LPSPI0_SOUT	LPUART0_CTS_b	LPI2C0_SDA	TPM0_CH3	RF_GPO_0[1]				LPUART0_RX	SPC0_LPREQ	
14	12	PTA19	CMP1_IN0/	PTA19	LPSPI0_SCK	LPUART0_RTS_b	LPI2C0_SCL	TPM0_CH2	RF_GPO_1[1]						WUU0_P4
15	13	VDD_LDO_CORE	VDD_LDO_CORE												
16	14	VOUT_CORE/ VDD_CORE	VOUT_CORE/ VDD_CORE												
17	15	PTA20	ADC0_A14/ CMP0_IN3	PTA20	LPSPI0_PCS2	LPUART0_TX	EWM0_IN	TPM0_CH1	RF_GPO_2[1]		FLEXIO0_D7				
18	16	PTA21	ADC0_A15/ CMP0_IN2	PTA21	LPSPI0_PCS3	LPUART0_RX	EWM0_OUT_b	TPM0_CH0	RF_GPO_3[1]	RF_GPO_7[1]	FLEXIO0_D8	RF_GPO_10[1]			WUU0_P5
19	17	VSS_DCDC	VSS_DCDC												

Table continues on the next page...

Table continued from the previous page...

48HVQFN	40HVQFN	Pin Name	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8	ALT9	ALT10	ALT11	Wake up
20	18	DCDC_LX	DCDC_LX												
21	19	VDD_IO_D/VDD_DCDC	VDD_IO_D/VDD_DCDC												
22	20	VOUT_SYS/VDD_SYS	VOUT_SYS/VDD_SYS												
23	21	PTD0	ADC0_A5	PTD0		RESET_b									
24		PTD1	ADC0_B5	PTD1	SPC0_LPREQ	NMI_b	RF_GPO_4[1]								
25		PTD2	ADC0_A6	PTD2	LPTMR0_ALT3	TAMPER0	RF_GPO_5[1]								
26		PTD3	ADC0_B6	PTD3	LPTMR1_ALT3	TAMPER1	RF_GPO_6[1]		TRGMUX0_IN2						
27	22	PTD4	XTAL32K	PTD4	LPTMR0_ALT2	TAMPER2									
28	23	PTD5	EXTAL32K	PTD5	LPTMR1_ALT2										
29	24	VDD_ANA	VDD_ANA												
30	25	VREFO	VREFO												
49	41	VREFL[2]	VREFL												
31		XTAL_OUT	XTAL_OUT												
32	26	XTAL	XTAL												
33	27	EXTAL	EXTAL												

Table continues on the next page...

Table continued from the previous page...

48HVQFN	40HVQFN	Pin Name	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8	ALT9	ALT10	ALT11	Wakeup
34	28	VDD_RF	VDD_RF												
35	29	ANT_2P4GHZ[1,3]	ANT_2P4GHZ[1]												
36	30	VPA_2P4GHZ[1,3]	VPA_2P4GHZ[1]												
37		PTC0		PTC0	LPSPI1_PCS2	CAN0_TX[4]	I3C0_SDA	TPM1_CH0		LPI2C1_SCL		FLEXIO0_D16			WUU0_P7
38		PTC1		PTC1	LPSPI1_PCS3	CAN0_RX[4]	I3C0_SCL	TPM1_CH1		LPI2C1_SDA		FLEXIO0_D17			WUU0_P8
39	31	PTC2		PTC2	LPSPI1_SOUT	LPUART1_RX	LPI2C1_SCLS	TPM1_CH2		I3C0_PUR		FLEXIO0_D18			WUU0_P9
40	32	PTC3		PTC3	LPSPI1_SCK	LPUART1_TX	LPI2C1_SDAS	TPM1_CH3				FLEXIO0_D19			
41	33	VDD_CORE	VDD_CORE												
42	34	PTC4		PTC4	LPSPI1_SIN	CAN0_TX[4]	LPI2C1_SCL		TPM2_CH0[1]			FLEXIO0_D20			WUU0_P10
43	35	PTC5		PTC5	LPSPI1_PCS0	CAN0_RX[4]	LPI2C1_SDA	TPM1_CH4	TPM2_CH1[1]			FLEXIO0_D21			
44	[5]	PTC6	ADC0_A8	PTC6	LPSPI1_PCS1			TPM1_CH5				FLEXIO0_D22			WUU0_P11
45	36	PTC7		PTC7	TRGMUX0_IN3	TRGMUX0_OUT3	SFA0_CLK	TPM1_CLKIN	TPM2_CLKIN[1]	CLKOUT		FLEXIO0_D23			WUU0_P12/NMI_b/RF_NOT_ALLOWED[1]

Table continues on the next page...

Table continued from the previous page...

48HVQFN	40HVQFN	Pin Name	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8	ALT9	ALT10	ALT11	Wake up
46	37	PTB0	ADC0_B10	PTB0	LPSPI1_PCS0			TPM1_CH0				FLEXIO0_D26			WUU0_P13
47	38	PTB1	ADC0_B11	PTB1	LPSPI1_SIN			TPM1_CH1				FLEXIO0_D27			
48	39	PTB2	ADC0_B12	PTB2	LPSPI1_SCK	LPUART1_TX		TPM1_CH2				FLEXIO0_D28			
1	40	PTB3	ADC0_B13	PTB3	LPSPI1_SOUT	LPUART1_RX		TPM1_CH3				FLEXIO0_D29			WUU0_P14
49	41	VSS	VSS												

1. This signal is not available for the parts without Radio modules.
2. VREF shorts to VSS.
3. For the parts that have no radio modules, this pin is not connected.
4. This signal is not available for the parts without CAN module.
5. PTC6_WUU0_P11 pin signal available only as a wake up source for FlexCAN module on signal CAN0_RX from pin PTC5. Other configuration on PTC6 shall not be used.

20.3 Pinouts diagram

The below figure shows the pinout diagram for the devices supported by this document. Many signals may be multiplexed onto a single pin. To determine what signals can be used on which pin, see the previous section.

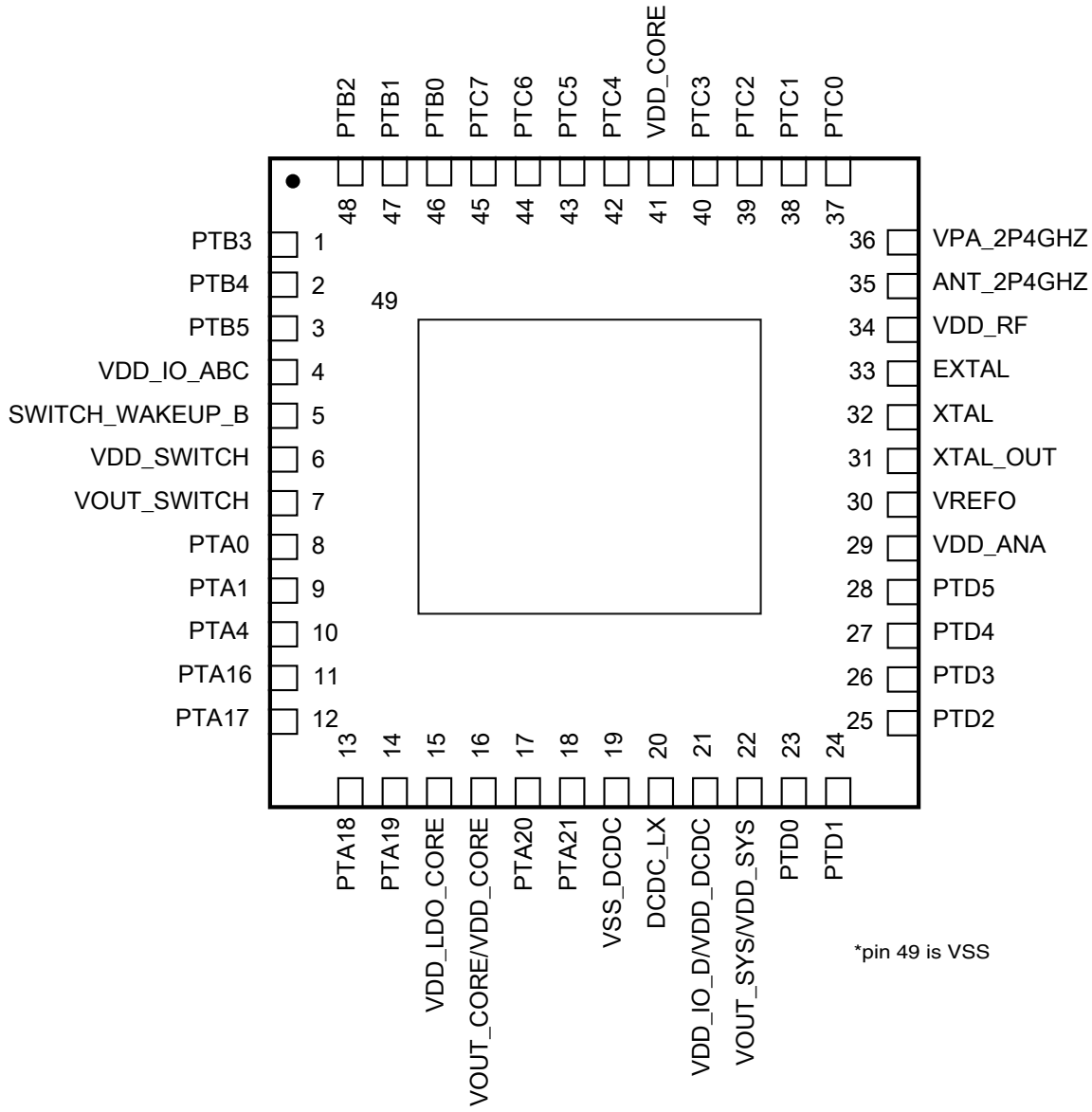
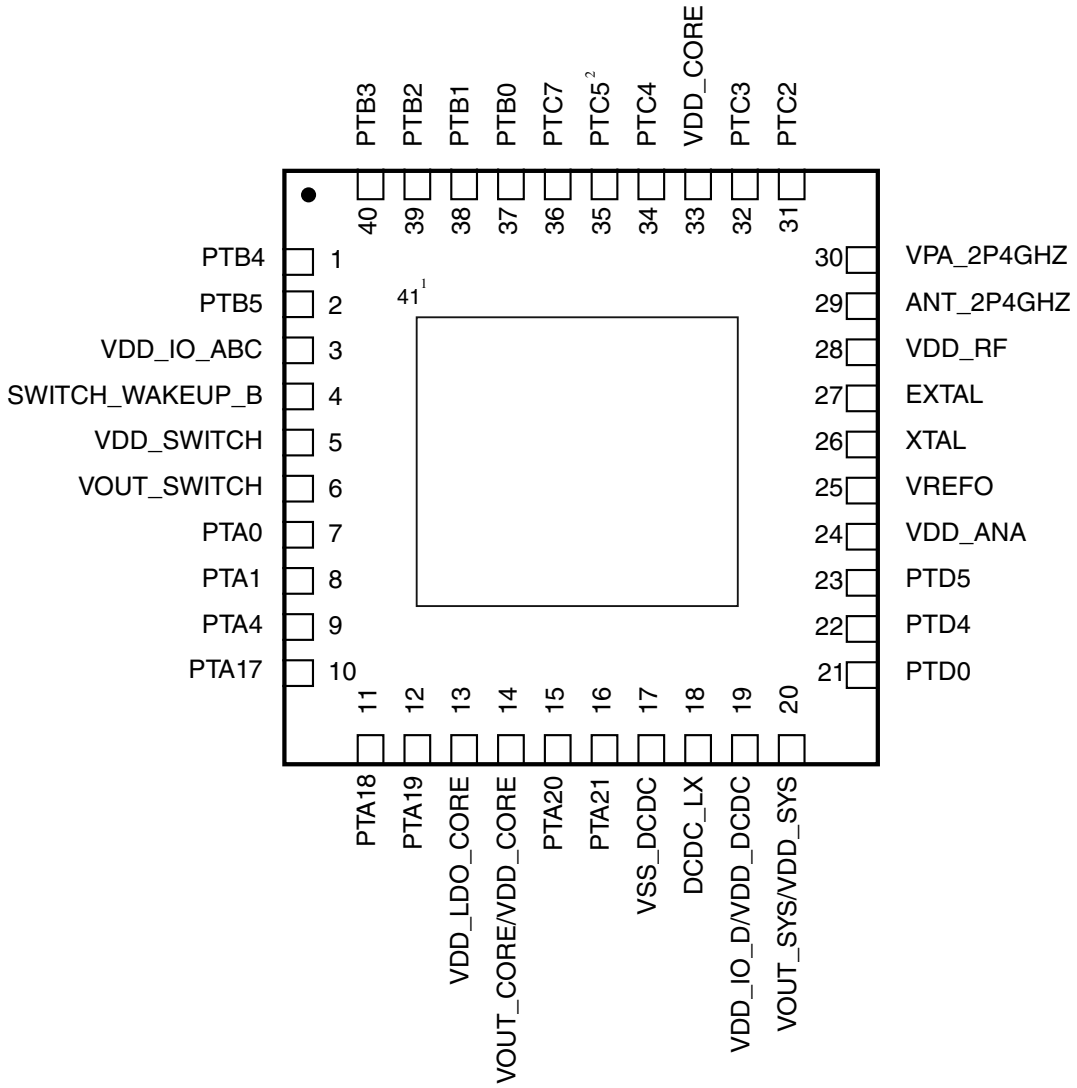


Figure 57. 48-pin HVQFN package pinout diagram



1. Pin 41 is VSS
2. PTC5 pin is internally bonded with PTC6 pin for wakeup purposes. See pinout table for details.

Figure 58. 40-pin HVQFN package pinout diagram

Chapter 21
Port Control (PORT)

21.1 Chip-specific PORT information

Table 179. Reference links to related information

Topic	Related module	Reference
Full description	PORT	PORT
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

21.1.1 Module instances

This device has four instances of the port module, PORTA, PORTB, PORTC, and PORTD.

21.1.2 Port B EFT detect clear (EDCR)

Port B and Port C pins share the same EFT detector clear control from PORTC_EDCR register. Any write to the PORTB_EDCR does not take effect.

21.1.3 Port voltage range (CONFIG[RANGE])

PORTA_CONFIG[RANGE] controls the voltage ranges of Port A, B, and C. Reading or writing PORTB_CONFIG[RANGE] and PORTC_CONFIG[RANGE] does not take effect.

21.2 Overview

PORT provides support for pad control functions. You can configure most functions independently for each pin, in the 32-bit port, and affect the pin regardless of its pin multiplexing state.

There exists a single instance of the PORT module for each port, and not all pins within each port are implemented on a specific chip.

21.2.1 Block diagram

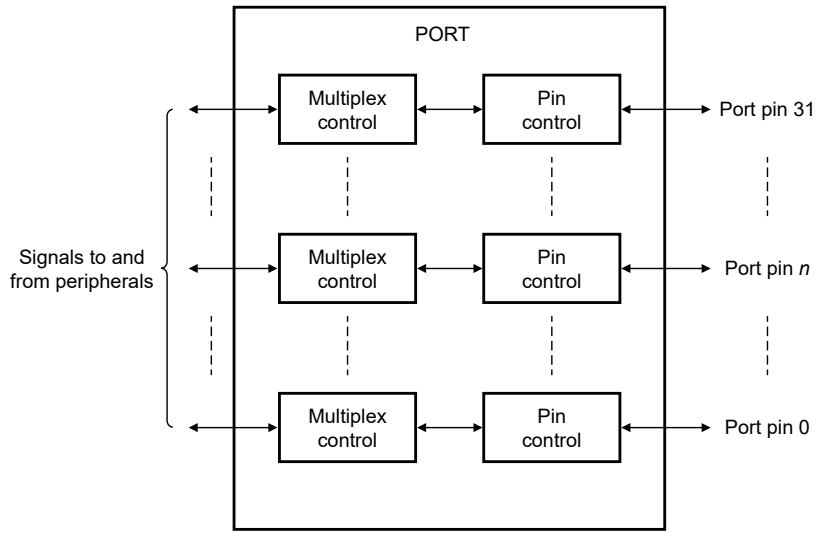


Figure 59. Block diagram

21.2.2 Features

- Individual pull control fields with pullup, pulldown, and pull-disable support
- Individual drive strength fields supporting high and low drive strengths
- Individual slew rate fields supporting fast and slow slew rates
- Individual PCR*n*[PFE] fields that enable and disable individual input passive filters on selected pins
- Individual PCR*n*[ODE] fields that enable and disable individual open drain outputs
- Individual electrical fast transient (EFT) detect with an EFT detect flag and associated interrupt
- Individual mux control fields supporting analog or pin disabled, GPIO, and up to 10 chip-specific digital functions

21.3 Functional description

21.3.1 Pin control

Each port pin has a corresponding Pin Control Register (PCR) associated with it that helps you configure the following functions for each pin within the 32-bit port:

- Pullup or pulldown enable
- Drive strength configuration
- Slew rate configuration
- Open drain enable
- Passive input filter enable on selected pins
- EFT detection
- Software configuration lock
- Pin multiplexing mode

These functions apply across all digital pin multiplexing modes, and individual peripherals do not override the configuration in PCR*n* unless otherwise noted. For example, if an I^2C function is enabled on a pin, it does not override the pullup or open drain configuration for that pin.

PCR*n*[LK] allows the configuration for each pin to be locked until the next system reset. When locked, writes to the lower half of that PCR*n* are ignored, although a bus error is not generated on an attempted write to a locked register.

The configuration of each PCR*n* is retained when the PORT module is disabled.

When you configure a pin in a digital pin multiplexing mode, the input buffer for that pin is enabled, allowing the pin state to be read via the corresponding GPIO.PDIR or allowing a pin interrupt or DMA request to be generated. If a pin is always floating when its input buffer is enabled, it can cause an increase in power consumption. This situation must be avoided. A pin can be floating because of an input pin that is not connected or an output pin that is tristated (output buffer is disabled).

Enabling the internal pull resistor (or implementing an external pull resistor) ensures that a pin does not float when its input buffer is enabled. The internal pull resistor is automatically disabled whenever the output buffer is enabled, allowing PCR*n*[PE] to remain 1. Configuring Pin Multiplexing mode to disabled or analog (PCR*n*[MUX] = 0) disables the pin's input buffer and results in lowest power consumption.

21.3.2 Global pin control

Global Pin Control Low (GPCLR) and Global Pin Control High (GPCHR) allow a single register write to update the lower 16 bits of PCR*n* for up to 16 pins, all with the same value. You cannot write to locked registers by using Global Pin Control Low (GPCLR) and Global Pin Control High (GPCHR).

Global Pin Control Low (GPCLR) and Global Pin Control High (GPCHR) enable you to quickly configure multiple pins within the same port and with the same peripheral function. These are write-only registers that always read as 0.

21.3.3 EFT detection

The EFT detect circuit is active whenever PORT is powered and EDCR[EDLC] and EDCR[EDHC] are 0.

The EFT high detect circuit for each pin is reset whenever EDCR[EDHC] = 1. Likewise, the EFT low detect circuit for each pin is reset whenever EDCR[EDLC] = 1 and that pin's EFT high detect is not asserted.

21.3.4 Interrupt

When EFT on a pin is detected, an interrupt request is generated to indicate the detection of EFT on that pin. This interrupt is enabled when EDIER[EDIE*n*] = 1. Writing 0 to these fields clears the interrupt request but does not affect EFT detection on the corresponding pin (EDFR[EDF*n*]).

21.4 Initialization

To initialize PORT, perform the following procedure:

1. Initialize the pin functions:
   - Initialize single pin functions by writing appropriate values to PCR*n*.
   - Initialize multiple pins (up to 16) with the same configuration by writing appropriate values to Global Pin Control Low (GPCLR) or Global Pin Control High (GPCHR).
2. Lock the configuration for a given pin, by writing 1 to PCR*n*[LK], so that it cannot be changed until the next reset.

21.5 Application information

21.5.1 Determine polarity of an asserted EFT detection

You can determine the polarity of an asserted EFT detect by performing this procedure:

1. Read EFT Detect Flag (EDFR).

2. Write 1 and then write 0 to EDCR[EDLC].
3. Read EFT Detect Flag (EDFR) again. Any field that has now become 0 indicates triggering of the EFT low detector only (no EFT high detector triggering).
4. Write 1 and then write 0 to EDCR[EDHC].
5. Read EFT Detect Flag (EDFR) again. Any field that has now become 0 indicates triggering of the EFT high detector. Any field that did not become 0 indicates triggering of both the EFT high and EFT low detectors.
6. Write 1 and then write 0 to both EDCR[EDLC] and EDCR[EDHC] to ensure that all detectors are reset.

21.6 Memory map and register definition

Any read or write access to the PORT memory space, outside the valid memory map, results in a bus error. All register accesses complete with zero wait states.

21.6.1 PORT register descriptions

21.6.1.1 PORT memory map

PORTA base address: 4004_2000h

PORTB base address: 4004_3000h

PORTC base address: 4004_4000h

PORTD base address: 4004_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0000h
10h	Global Pin Control Low (GPCLR)	32	W	0000_0000h
14h	Global Pin Control High (GPCHR)	32	W	0000_0000h
20h	Configuration (CONFIG)	32	RW	0000_0000h
40h	EFT Detect Flag (EDFR)	32	R	See section
44h	EFT Detect Interrupt Enable (EDIER)	32	RW	0000_0000h
48h	EFT Detect Clear (EDCR)	32	RW	0000_0000h
80h	Pin Control 0 (PCR0)	32	RW	See section
84h	Pin Control 1 (PCR1)	32	RW	See section
88h	Pin Control 2 (PCR2)	32	RW	See section
8Ch	Pin Control 3 (PCR3)	32	RW	0000_0000h
90h	Pin Control 4 (PCR4)	32	RW	See section
94h	Pin Control 5 (PCR5)	32	RW	0000_0000h
98h	Pin Control 6 (PCR6)	32	RW	0000_0000h
9Ch	Pin Control 7 (PCR7)	32	RW	0000_0000h
A0h	Pin Control 8 (PCR8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A4h	Pin Control 9 (PCR9)	32	RW	0000_0000h
C0h	Pin Control 16 (PCR16)	32	RW	0000_0000h
C4h	Pin Control 17 (PCR17)	32	RW	0000_0000h
C8h - CCh	Pin Control a (PCR18 - PCR19)	32	RW	0000_0000h
D0h - D4h	Pin Control a (PCR20 - PCR21)	32	RW	0000_0000h
D8h	Pin Control 22 (PCR22)	32	RW	0000_0000h

21.6.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Specifies the version number and feature number of the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the specification.
23-16	Minor Version Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
MINOR	Indicates the minor version number for the specification.
15-0 FEATURE	Feature Specification Number Indicates the feature set number. 0000_0000_0000_0000b - Basic implementation

21.6.1.3 Global Pin Control Low (GPCLR)

Offset

Register	Offset
GPCLR	10h

Function

Controls writes to the PCR15–PCR0 registers.

NOTE

This register supports only 32-bit writes and ignores any 16-bit or 8-bit writes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	GPWE 15	GPWE 14	GPWE 13	GPWE 12	GPWE 11	GPWE 10	GPWE 9	GPWE 8	GPWE 7	GPWE 6	GPWE 5	GPWE 4	GPWE 3	GPWE 2	GPWE 1	GPWE 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	GPWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 GPWEn	Global Pin Write Enable Configures the corresponding lower 16-bit field of PCR*n* to be updated with the value in the GPWD field. If a selected PCR is locked, the write to that register is ignored. 0b - Not updated

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Updated
15-0 GPWD	Global Pin Write Data Is written to PCR*n*[15:0] if GPWE*n* = 1.

21.6.1.4 Global Pin Control High (GPCHR)

Offset

Register	Offset
GPCHR	14h

Function

Controls writes to the PCR31–PCR16 registers.

NOTE

This register supports only 32-bit writes and ignores any 16-bit or 8-bit writes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	GPWE 31	GPWE 30	GPWE 29	GPWE 28	GPWE 27	GPWE 26	GPWE 25	GPWE 24	GPWE 23	GPWE 22	GPWE 21	GPWE 20	GPWE 19	GPWE 18	GPWE 17	GPWE 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	GPWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 GPWEn	Global Pin Write Enable Configures the corresponding lower 16-bit field of PCR*n* to be updated with the value in the GPWD field. If a selected PCR is locked, write to that register is ignored. 0b - Not updated 1b - Updated

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 GPWD	Global Pin Write Data Is written to PCR*n*[15:0] if GPWE*n* = 1.

21.6.1.5 Configuration (CONFIG)

Offset

Register	Offset
CONFIG	20h

Function

Configures the port voltage range.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															RANGE
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 RANGE	Port Voltage Range Configures the port voltage range. 0b - 1.71 V–3.6 V 1b - 2.70 V–3.6 V

21.6.1.6 EFT Detect Flag (EDFR)

Offset

Register	Offset
EDFR	40h

Function

Specifies whether an EFT event is detected. The EFT detect logic is active in all pin multiplexing modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	EDF22	EDF21	EDF20	EDF19	EDF18	EDF17	EDF16
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	EDF5	EDF4	EDF3	EDF2	EDF1	EDF0
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 Reserved31	Reserved 0b - Not supported 1b - Not supported
30 Reserved30	Reserved 0b - Not supported 1b - Not supported
29 Reserved29	Reserved 0b - Not supported 1b - Not supported
28 Reserved28	Reserved 0b - Not supported 1b - Not supported
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved27	0b - Not supported 1b - Not supported
26 Reserved26	Reserved 0b - Not supported 1b - Not supported
25 Reserved25	Reserved 0b - Not supported 1b - Not supported
24 Reserved24	Reserved 0b - Not supported 1b - Not supported
23 Reserved23	Reserved 0b - Not supported 1b - Not supported
22 EDF22	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin. **NOTE** The descriptions of the field settings vary by module instance.

Instance	Field value and description
PORTA	0b - No EFT event detected 1b - High or/and low EFT event detected
PORTB	0b - Not supported 1b - Not supported
PORTC	0b - Not supported 1b - Not supported
PORTD	0b - Not supported 1b - Not supported

21 EDF21	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>20
EDF20</td><td>EFT Detect Flag
Indicates whether high or low EFT is detected on the corresponding pin.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>19
EDF19</td><td>EFT Detect Flag
Indicates whether high or low EFT is detected on the corresponding pin.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>18
EDF18</td><td>EFT Detect Flag
Indicates whether high or low EFT is detected on the corresponding pin.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>17
EDF17</td><td>EFT Detect Flag
Indicates whether high or low EFT is detected on the corresponding pin.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>16
EDF16</td><td>EFT Detect Flag
Indicates whether high or low EFT is detected on the corresponding pin.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>15
Reserved15</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 Reserved14	Reserved 0b - Not supported 1b - Not supported
13 Reserved13	Reserved 0b - Not supported 1b - Not supported
12 Reserved12	Reserved 0b - Not supported 1b - Not supported
11 Reserved11	Reserved 0b - Not supported 1b - Not supported
10 Reserved10	Reserved 0b - Not supported 1b - Not supported
9 Reserved9	Reserved **NOTE** The descriptions of the field settings vary by module instance.

Instance	Field value and description
PORTA	0b - Not supported 1b - Not supported
PORTB	0b - Not supported 1b - Not supported
PORTD	0b - Not supported 1b - Not supported
PORTC	0b - No EFT event detected 1b - High or/and low EFT event detected

8 Reserved8	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
</table>
</td></tr>
<tr><td>7
Reserved7</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>6
Reserved6</td><td>Reserved
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
<tr><td>PORTD</td><td>0b - No EFT event detected
1b - High or/and low EFT event detected</td></tr>
</table>
</td></tr>
<tr><td>5</td><td>EFT Detect Flag</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDF5	Indicates whether high or low EFT is detected on the corresponding pin. 0b - No EFT event detected 1b - High or/and low EFT event detected
4 EDF4	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin. 0b - No EFT event detected 1b - High or/and low EFT event detected
3 EDF3	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin. 0b - No EFT event detected 1b - High or/and low EFT event detected
2 EDF2	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin. 0b - No EFT event detected 1b - High or/and low EFT event detected
1 EDF1	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin. 0b - No EFT event detected 1b - High or/and low EFT event detected
0 EDF0	EFT Detect Flag Indicates whether high or low EFT is detected on the corresponding pin. 0b - No EFT event detected 1b - High or/and low EFT event detected

21.6.1.7 EFT Detect Interrupt Enable (EDIER)

Offset

Register	Offset
EDIER	44h

Function

Configures whether to generate an interrupt when an EFT event is detected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	EDIE22	EDIE21	EDIE20	EDIE19	EDIE18	EDIE17	EDIE16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	EDIE5	EDIE4	EDIE3	EDIE2	EDIE1	EDIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 Reserved31	Reserved 0b - Not supported 1b - Not supported
30 Reserved30	Reserved 0b - Not supported 1b - Not supported
29 Reserved29	Reserved 0b - Not supported 1b - Not supported
28 Reserved28	Reserved 0b - Not supported 1b - Not supported
27 Reserved27	Reserved 0b - Not supported 1b - Not supported
26 Reserved26	Reserved 0b - Not supported 1b - Not supported
25 Reserved25	Reserved 0b - Not supported 1b - Not supported

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>24
Reserved24</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>23
Reserved23</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>22
EDIE22</td><td>EFT Detect Interrupt Enable
Configures the EFT detect interrupt for the corresponding pin.
NOTE
The descriptions of the field settings vary by module instance.
<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr><tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr><tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr><tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr></table></td></tr>
<tr><td>21
EDIE21</td><td>EFT Detect Interrupt Enable
Configures the EFT detect interrupt for the corresponding pin.
NOTE
The descriptions of the field settings vary by module instance.
<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr><tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr></table></td></tr>
</table>

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTC</td><td>0b - Not supported 1b - Not supported</td></tr><tr><td>PORTD</td><td>0b - Not supported 1b - Not supported</td></tr></table>
20 EDIE20	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. **NOTE** The descriptions of the field settings vary by module instance. <table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event</td></tr><tr><td>PORTB</td><td>0b - Not supported 1b - Not supported</td></tr><tr><td>PORTC</td><td>0b - Not supported 1b - Not supported</td></tr><tr><td>PORTD</td><td>0b - Not supported 1b - Not supported</td></tr></table>
19 EDIE19	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. **NOTE** The descriptions of the field settings vary by module instance. <table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event</td></tr><tr><td>PORTB</td><td>0b - Not supported</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td></td><td>1b - Not supported</td></tr><tr><td>PORTC</td><td>0b - Not supported 1b - Not supported</td></tr><tr><td>PORTD</td><td>0b - Not supported 1b - Not supported</td></tr></table>
18 EDIE18	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. **NOTE** The descriptions of the field settings vary by module instance. <table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event</td></tr><tr><td>PORTB</td><td>0b - Not supported 1b - Not supported</td></tr><tr><td>PORTC</td><td>0b - Not supported 1b - Not supported</td></tr><tr><td>PORTD</td><td>0b - Not supported 1b - Not supported</td></tr></table>
17 EDIE17	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. **NOTE** The descriptions of the field settings vary by module instance. <table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event</td></tr></table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>16
EDIE16</td><td>EFT Detect Interrupt Enable
Configures the EFT detect interrupt for the corresponding pin.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
</table>
</td></tr>
<tr><td>15
Reserved15</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>14
Reserved14</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>13</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>Reserved13</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>12
Reserved12</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>11
Reserved11</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>10
Reserved10</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>9
Reserved9</td><td>Reserved
NOTE
The descriptions of the field settings vary by module instance.
<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORTA</td><td>0b - Not supported
1b - Not supported</td></tr><tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr><tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr><tr><td>PORTC</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr></table></td></tr>
<tr><td>8
Reserved8</td><td>Reserved
NOTE
The descriptions of the field settings vary by module instance.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTD</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr>
</table>
</td></tr>
<tr><td>7
Reserved7</td><td>Reserved
0b - Not supported
1b - Not supported</td></tr>
<tr><td>6
Reserved6</td><td>Reserved
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>PORTA</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTB</td><td>0b - Not supported
1b - Not supported</td></tr>
<tr><td>PORTC</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr>
<tr><td>PORTD</td><td>0b - Interrupt not generated upon detection of the EFT event
1b - Interrupt generated upon detection of the EFT event</td></tr>
</table>
</td></tr>
<tr><td>5
EDIE5</td><td>EFT Detect Interrupt Enable
Configures the EFT detect interrupt for the corresponding pin.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event
4 EDIE4	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. 0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event
3 EDIE3	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. 0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event
2 EDIE2	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. 0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event
1 EDIE1	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. 0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event
0 EDIE0	EFT Detect Interrupt Enable Configures the EFT detect interrupt for the corresponding pin. 0b - Interrupt not generated upon detection of the EFT event 1b - Interrupt generated upon detection of the EFT event

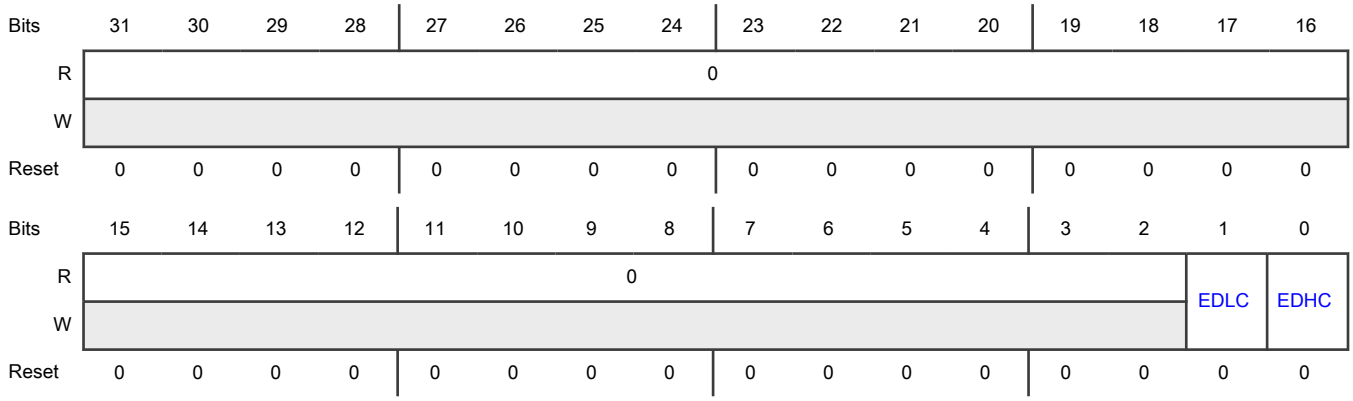
21.6.1.8 EFT Detect Clear (EDCR)

Offset

Register	Offset
EDCR	48h

Function
Clears EFT detectors. The EFT detect logic is cleared per port.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 EDLC	EFT Detect Low Clear Clears low EFT detectors. If this field = 1, all low EFT detectors for which the corresponding high EFT detectors are not asserted are cleared. 0b - Does not clear 1b - Clears
0 EDHC	EFT Detect High Clear Clears high EFT detectors. 0b - Does not clear 1b - Clears

21.6.1.9 Pin Control 0 (PCR0)

Offset

Register	Offset
PCR0	80h

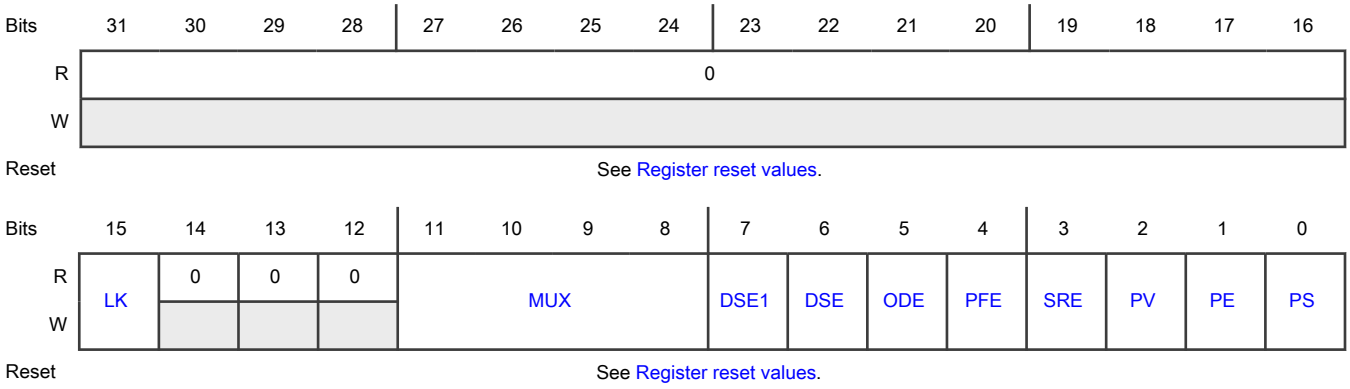
Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

Diagram



Register reset values

Register	Reset value
PCR0	PORTA: 0000_0703h PORTB,PORTC: 0000_0000h PORTD: 0000_0333h

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved.
Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die.
The corresponding pin is configured according to the following pin multiplexing slots:
0000b - Pin disabled (analog)
0001b - Alternative 1 (GPIO)
0010b - Alternative 2 (chip-specific)
0011b - Alternative 3 (chip-specific)
0100b - Alternative 4 (chip-specific)
0101b - Alternative 5 (chip-specific)
0110b - Alternative 6 (chip-specific)
0111b - Alternative 7 (chip-specific)
1000b - Alternative 8 (chip-specific)
1001b - Alternative 9 (chip-specific)
1010b - Alternative 10 (chip-specific)
1011b - Alternative 11 (chip-specific)</td></tr>
<tr><td>7
DSE1</td><td>Drive Strength Enable
Configures the drive strength on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• If this field = 0, normal drive strength is configured on the corresponding pin.
• If this field = 1, double drive strength is configured on the corresponding pin.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR0</td></tr><tr><td>PORTB</td><td>—</td><td>PCR0</td></tr><tr><td>PORTC</td><td>PCR0</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR0</td></tr></table>0b - Normal
1b - Double</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>6
DSE</td><td>Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
• When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR0</td><td>—</td></tr><tr><td>PORTB</td><td>PCR0</td><td>—</td></tr><tr><td>PORTC</td><td>PCR0</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR0</td></tr></table>
0b - Low
1b - High</td></tr>
<tr><td>5
ODE</td><td>Open Drain Enable
Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>See the chip's data sheet for filter characteristics.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR0</td></tr><tr><td>PORTB</td><td>—</td><td>PCR0</td></tr><tr><td>PORTC</td><td>PCR0</td><td>—</td></tr><tr><td>PORTD</td><td>PCR0</td><td>—</td></tr></table>
0b - Disables
1b - Enables</td></tr>
<tr><td>3
SRE</td><td>Slew Rate Enable
Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR0</td><td>—</td></tr><tr><td>PORTB</td><td>PCR0</td><td>—</td></tr><tr><td>PORTC</td><td>PCR0</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR0</td></tr></table>
0b - Fast
1b - Slow</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORTB</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORTC</td><td>PCR0</td><td>—</td></tr>
<tr><td>PORTD</td><td>PCR0</td><td>—</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pull resistor is not enabled on the corresponding pin.
• When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0b - Disables
1b - Enables</td></tr>
<tr><td>0
PS</td><td>Pull Select
Enables the internal pullup or pulldown resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCRn.PE field = 1.
• When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCRn.PE field = 1.
0b - Enables internal pulldown resistor
1b - Enables internal pullup resistor</td></tr>
</table>

21.6.1.10 Pin Control 1 (PCR1)

Offset

Register	Offset
PCR1	84h

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				DSE1	DSE	ODE	PFE	SRE	PV	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR1	PORTA: 0000_0702h PORTB–PORTD: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7 DSE1	Drive Strength Enable Configures the drive strength on each pin. The drive strength configuration is valid for all digital pin multiplexing modes. • If this field = 0, normal drive strength is configured on the corresponding pin. • If this field = 1, double drive strength is configured on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORTB</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORTC</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR1</td></tr>
</table>
0b - Normal
1b - Double
</td></tr>
<tr><td>6
DSE</td><td>
Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.

When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTB</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTC</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR1</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>5</td><td>Open Drain Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>ODE</td><td>Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.
See the chip's data sheet for filter characteristics.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR1</td></tr><tr><td>PORTB</td><td>—</td><td>PCR1</td></tr><tr><td>PORTC</td><td>PCR1</td><td>—</td></tr><tr><td>PORTD</td><td>PCR1</td><td>—</td></tr></table>0b - Disables
1b - Enables</td></tr>
<tr><td>3
SRE</td><td>Slew Rate Enable
Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTB</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTC</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR1</td></tr>
</table>
0b - Fast
1b - Slow
</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORTB</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORTC</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORTD</td><td>PCR1</td><td>—</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pull resistor is not enabled on the corresponding pin.
• When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0b - Disables</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.11 Pin Control 2 (PCR2)

Offset

Register	Offset
PCR2	88h

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				0	DSE	ODE	PFE	SRE	PV	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR2	PORTA: 0000_0040h PORTB–PORTD: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific)

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0111b - Alternative 7 (chip-specific)
1000b - Alternative 8 (chip-specific)
1001b - Alternative 9 (chip-specific)
1010b - Alternative 10 (chip-specific)
1011b - Alternative 11 (chip-specific)</td></tr>
<tr><td>7
—</td><td>Reserved</td></tr>
<tr><td>6
DSE</td><td>Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
• When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR2</td><td>—</td></tr><tr><td>PORTB</td><td>PCR2</td><td>—</td></tr><tr><td>PORTC</td><td>PCR2</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR2</td></tr></table>0b - Low
1b - High</td></tr>
<tr><td>5
ODE</td><td>Open Drain Enable
Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.
See the chip's data sheet for filter characteristics.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR2</td></tr><tr><td>PORTB</td><td>—</td><td>PCR2</td></tr><tr><td>PORTC</td><td>—</td><td>PCR2</td></tr><tr><td>PORTD</td><td>PCR2</td><td>—</td></tr></table>0b - Disables
1b - Enables</td></tr>
<tr><td>3
SRE</td><td>Slew Rate Enable
Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR2</td><td>—</td></tr><tr><td>PORTB</td><td>PCR2</td><td>—</td></tr></table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTC</td><td>PCR2</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR2</td></tr>
</table>
0b - Fast
1b - Slow
</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR2</td></tr>
<tr><td>PORTB</td><td>—</td><td>PCR2</td></tr>
<tr><td>PORTC</td><td>—</td><td>PCR2</td></tr>
<tr><td>PORTD</td><td>PCR2</td><td>—</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pull resistor is not enabled on the corresponding pin.
• When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0b - Disables
1b - Enables</td></tr>
<tr><td>0
PS</td><td>Pull Select
Enables the internal pullup or pulldown resistor.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.12 Pin Control 3 (PCR3)

Offset

Register	Offset
PCR3	8Ch

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				0	DSE	ODE	PFE	SRE	PV	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>6
DSE</td><td>Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
• When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR3</td><td>—</td></tr><tr><td>PORTB</td><td>PCR3</td><td>—</td></tr><tr><td>PORTC</td><td>PCR3</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR3</td></tr></table>
0b - Low
1b - High</td></tr>
<tr><td>5
ODE</td><td>Open Drain Enable
Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>See the chip's data sheet for filter characteristics.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR3</td></tr><tr><td>PORTB</td><td>—</td><td>PCR3</td></tr><tr><td>PORTC</td><td>—</td><td>PCR3</td></tr><tr><td>PORTD</td><td>PCR3</td><td>—</td></tr></table>
0b - Disables
1b - Enables</td></tr>
<tr><td>3
SRE</td><td>Slew Rate Enable
Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR3</td><td>—</td></tr><tr><td>PORTB</td><td>PCR3</td><td>—</td></tr><tr><td>PORTC</td><td>PCR3</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR3</td></tr></table>
0b - Fast
1b - Slow</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR3</td></tr>
<tr><td>PORTB</td><td>—</td><td>PCR3</td></tr>
<tr><td>PORTC</td><td>—</td><td>PCR3</td></tr>
<tr><td>PORTD</td><td>PCR3</td><td>—</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pull resistor is not enabled on the corresponding pin.
• When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0b - Disables
1b - Enables</td></tr>
<tr><td>0
PS</td><td>Pull Select
Enables the internal pullup or pulldown resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCRn.PE field = 1.
• When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCRn.PE field = 1.
0b - Enables internal pulldown resistor
1b - Enables internal pullup resistor</td></tr>
</table>

21.6.1.13 Pin Control 4 (PCR4)

Offset

Register	Offset
PCR4	90h

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				DSE1	DSE	ODE	PFE	SRE	PV	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR4	PORTA: 0000_0702h PORTB–PORTD: 0000_0000h

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7 DSE1	Drive Strength Enable Configures the drive strength on each pin. The drive strength configuration is valid for all digital pin multiplexing modes. • If this field = 0, normal drive strength is configured on the corresponding pin. • If this field = 1, double drive strength is configured on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR4</td></tr>
<tr><td>PORTB</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTC</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR4</td></tr>
</table>
0b - Normal
1b - Double
</td></tr>
<tr><td>6
DSE</td><td>
Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.

When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.

NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTB</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTC</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR4</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>5</td><td>Open Drain Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>ODE</td><td>Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.
See the chip's data sheet for filter characteristics.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR4</td></tr><tr><td>PORTB</td><td>PCR4</td><td>—</td></tr><tr><td>PORTC</td><td>PCR4</td><td>—</td></tr><tr><td>PORTD</td><td>PCR4</td><td>—</td></tr></table>
0b - Disables
1b - Enables</td></tr>
<tr><td>3
SRE</td><td>Slew Rate Enable
Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTB</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTC</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR4</td></tr>
</table>
0b - Fast
1b - Slow
</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR4</td></tr>
<tr><td>PORTB</td><td>PCR4</td><td>—</td></tr>
<tr><td>PORTC</td><td>—</td><td>PCR4</td></tr>
<tr><td>PORTD</td><td>PCR4</td><td>—</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pull resistor is not enabled on the corresponding pin.
• When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0b - Disables</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.14 Pin Control 5 (PCR5)

Offset

Register	Offset
PCR5	94h

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				DSE1	DSE	ODE	PFE	SRE	PV	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7	Drive Strength Enable

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>DSE1</td><td>Configures the drive strength on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• If this field = 0, normal drive strength is configured on the corresponding pin.
• If this field = 1, double drive strength is configured on the corresponding pin.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR5</td></tr><tr><td>PORTB</td><td>PCR5</td><td>—</td></tr><tr><td>PORTC</td><td>PCR5</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR5</td></tr></table>
0b - Normal
1b - Double</td></tr>
<tr><td>6
DSE</td><td>Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
• When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR5</td><td>—</td></tr><tr><td>PORTB</td><td>PCR5</td><td>—</td></tr><tr><td>PORTC</td><td>PCR5</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR5</td></tr></table></td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
</table>
0b - Low
1b - High</td></tr>
<tr><td>5
ODE</td><td>Open Drain Enable
Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.
See the chip's data sheet for filter characteristics.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTA</td><td>—</td><td>PCR5</td></tr>
<tr><td>PORTB</td><td>PCR5</td><td>—</td></tr>
<tr><td>PORTC</td><td>PCR5</td><td>—</td></tr>
<tr><td>PORTD</td><td>PCR5</td><td>—</td></tr>
</table>
0b - Disables
1b - Enables</td></tr>
<tr><td>3</td><td>Slew Rate Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>SRE</td><td>Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>PCR5</td><td>—</td></tr><tr><td>PORTB</td><td>PCR5</td><td>—</td></tr><tr><td>PORTC</td><td>PCR5</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR5</td></tr></table>
0b - Fast
1b - Slow</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTA</td><td>—</td><td>PCR5</td></tr><tr><td>PORTB</td><td>—</td><td>PCR5</td></tr><tr><td>PORTC</td><td>—</td><td>PCR5</td></tr><tr><td>PORTD</td><td>PCR5</td><td>—</td></tr></table>
0b - Low
1b - High</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pull resistor is not enabled on the corresponding pin. • When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input. 0b - Disables 1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.15 Pin Control 6 (PCR6)

Offset

Register	Offset
PCR6	98h

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORTA	—	PCR6
PORTB	—	PCR6

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PORTC	PCR6	—
PORTD	PCR6	—

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				0	DSE	ODE	PFE	SRE	PV	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved.
Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die.
The corresponding pin is configured according to the following pin multiplexing slots:
0000b - Pin disabled (analog)
0001b - Alternative 1 (GPIO)
0010b - Alternative 2 (chip-specific)
0011b - Alternative 3 (chip-specific)
0100b - Alternative 4 (chip-specific)
0101b - Alternative 5 (chip-specific)
0110b - Alternative 6 (chip-specific)
0111b - Alternative 7 (chip-specific)
1000b - Alternative 8 (chip-specific)
1001b - Alternative 9 (chip-specific)
1010b - Alternative 10 (chip-specific)</td></tr>
<tr><td>7
—</td><td>Reserved</td></tr>
<tr><td>6
DSE</td><td>Drive Strength Enable
Configures drive strength, low or high, on each pin.
The drive strength configuration is valid for all digital pin multiplexing modes.
• When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
• When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTC</td><td>PCR6</td><td>—</td></tr><tr><td>PORTD</td><td>—</td><td>PCR6</td></tr></table>
0b - Low
1b - High</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>5
ODE</td><td>Open Drain Enable
Enables open drain output on each pin.
The open drain configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the open drain output is disabled on the corresponding pin.
• When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
0b - Disables
1b - Enables</td></tr>
<tr><td>4
PFE</td><td>Passive Filter Enable
Enables passive input filter on each pin.
The passive filter configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the passive input filter is disabled on the corresponding pin.
• When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input.
See the chip's data sheet for filter characteristics.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>PORTC</td><td>—</td><td>PCR6</td></tr><tr><td>PORTD</td><td>PCR6</td><td>—</td></tr></table>
0b - Disables
1b - Enables</td></tr>
<tr><td>3
SRE</td><td>Slew Rate Enable
Configures the slew rate feature, fast or slow, on each corresponding pin.
The slew rate configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTC</td><td>PCR6</td><td>—</td></tr>
<tr><td>PORTD</td><td>—</td><td>PCR6</td></tr>
</table>
0b - Fast
1b - Slow
</td></tr>
<tr><td>2
PV</td><td>Pull Value
Selects high or low internal pull resistor value.
The pull value configuration is valid for all digital pin multiplexing modes.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>PORTC</td><td>—</td><td>PCR6</td></tr>
<tr><td>PORTD</td><td>PCR6</td><td>—</td></tr>
</table>
0b - Low
1b - High
</td></tr>
<tr><td>1
PE</td><td>Pull Enable
Enables the internal pull resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pull resistor is not enabled on the corresponding pin.
• When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0b - Disables
1b - Enables</td></tr>
<tr><td>0
PS</td><td>Pull Select
Enables the internal pullup or pulldown resistor.
This configuration is valid for all digital pin multiplexing modes.
• When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCRn.PE field = 1.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.16 Pin Control a (PCR7 - PCR17)

Offset

Register	Offset
PCR7	9Ch
PCR8	A0h
PCR9	A4h
PCR16	C0h
PCR17	C4h

Function

Configures pin control features on each pin.

NOTE

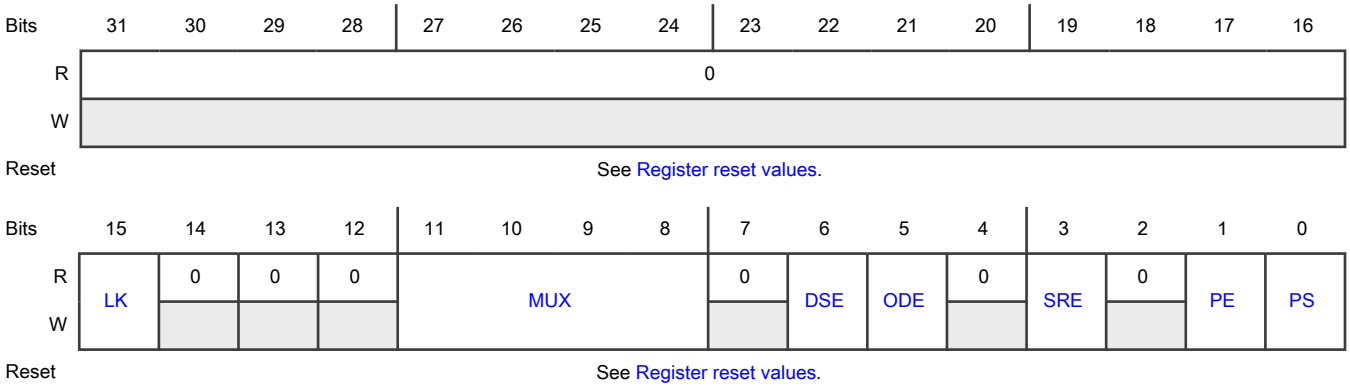
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORTA	PCR16–PCR17	PCR7–PCR9
PORTB	—	PCR7–PCR9 PCR16–PCR17
PORTC	PCR7–PCR9	PCR16–PCR17
PORTD	—	PCR7–PCR9 PCR16–PCR17

Diagram



Register reset values

Register	Reset value
PCR7–PCR9	0000_0000h
PCR10–PCR15	Register not supported
PCR16–PCR17	0000_0000h

Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7 —	Reserved
6 DSE	Drive Strength Enable Configures drive strength, low or high, on each pin. The drive strength configuration is valid for all digital pin multiplexing modes. • When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output. • When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output. 0b - Low 1b - High
5 ODE	Open Drain Enable Enables open drain output on each pin. The open drain configuration is valid for all digital pin multiplexing modes. • When this field = 0, the open drain output is disabled on the corresponding pin. • When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables 1b - Enables
4 —	Reserved
3 SRE	Slew Rate Enable Configures the slew rate feature, fast or slow, on each corresponding pin. The slew rate configuration is valid for all digital pin multiplexing modes. 0b - Fast 1b - Slow
2 —	Reserved
1 PE	Pull Enable Enables the internal pull resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pull resistor is not enabled on the corresponding pin. • When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input. 0b - Disables 1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.17 Pin Control a (PCR18 - PCR19)

Offset

Register	Offset
PCR18	C8h
PCR19	CCh

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORTA	PCR18–PCR19	—
PORTB	—	PCR18–PCR19
PORTC	—	PCR18–PCR19
PORTD	—	PCR18–PCR19

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				DSE1	DSE	ODE	PFE	SRE	0	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7 DSE1	Drive Strength Enable Configures the drive strength on each pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The drive strength configuration is valid for all digital pin multiplexing modes. • If this field = 0, normal drive strength is configured on the corresponding pin. • If this field = 1, double drive strength is configured on the corresponding pin. 0b - Normal 1b - Double
6 DSE	Drive Strength Enable Configures drive strength, low or high, on each pin. The drive strength configuration is valid for all digital pin multiplexing modes. • When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output. • When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output. 0b - Low 1b - High
5 ODE	Open Drain Enable Enables open drain output on each pin. The open drain configuration is valid for all digital pin multiplexing modes. • When this field = 0, the open drain output is disabled on the corresponding pin. • When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output. 0b - Disables 1b - Enables
4 PFE	Passive Filter Enable Enables passive input filter on each pin. The passive filter configuration is valid for all digital pin multiplexing modes. • When this field = 0, the passive input filter is disabled on the corresponding pin. • When this field = 1, the passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input. See the chip's data sheet for filter characteristics. 0b - Disables 1b - Enables
3 SRE	Slew Rate Enable Configures the slew rate feature, fast or slow, on each corresponding pin. The slew rate configuration is valid for all digital pin multiplexing modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Fast 1b - Slow
2 —	Reserved
1 PE	Pull Enable Enables the internal pull resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pull resistor is not enabled on the corresponding pin. • When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input. 0b - Disables 1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.18 Pin Control a (PCR20 - PCR21)

Offset

Register	Offset
PCR20	D0h
PCR21	D4h

Function

Configures pin control features on each pin.

NOTE

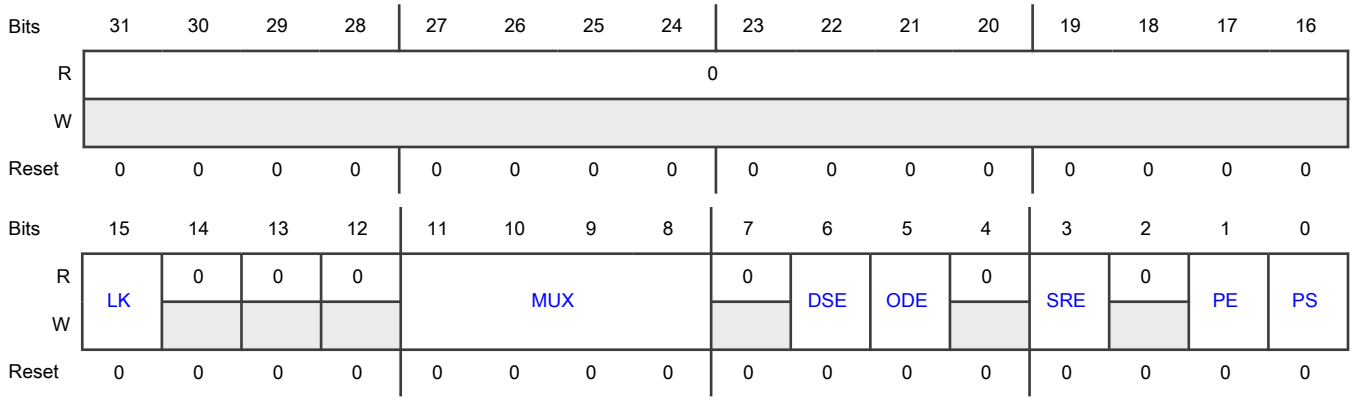
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORTA	PCR20–PCR21	—
PORTB	—	PCR20–PCR21
PORTC	—	PCR20–PCR21
PORTD	—	PCR20–PCR21

Diagram



Fields

Field	Function
31-16 —	Reserved
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific) 0100b - Alternative 4 (chip-specific) 0101b - Alternative 5 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific)
7 —	Reserved
6 DSE	Drive Strength Enable Configures drive strength, low or high, on each pin. The drive strength configuration is valid for all digital pin multiplexing modes. • When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output. • When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output. 0b - Low 1b - High
5 ODE	Open Drain Enable Enables open drain output on each pin. The open drain configuration is valid for all digital pin multiplexing modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• When this field = 0, the open drain output is disabled on the corresponding pin. • When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output. 0b - Disables 1b - Enables
4 —	Reserved
3 SRE	Slew Rate Enable Configures the slew rate feature, fast or slow, on each corresponding pin. The slew rate configuration is valid for all digital pin multiplexing modes. 0b - Fast 1b - Slow
2 —	Reserved
1 PE	Pull Enable Enables the internal pull resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pull resistor is not enabled on the corresponding pin. • When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input. 0b - Disables 1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor 1b - Enables internal pullup resistor

21.6.1.19 Pin Control 22 (PCR22)

Offset

Register	Offset
PCR22	D8h

Function

Configures pin control features on each pin.

NOTE

Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORTA	PCR22	—
PORTB	—	PCR22
PORTC	—	PCR22
PORTD	—	PCR22

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	0	0	0	MUX				0	DSE	ODE	0	SRE	0	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 LK	Lock Register Locks this PCR. When a PCR*n* is locked, its fields cannot be updated until the next reset. 0b - Locks 1b - Does not lock
14 —	Reserved
13 —	Reserved
12 —	Reserved
11-8 MUX	Pin Multiplex Control Configures the multiplexing slots on each pin. Not all pins support all pin multiplexing slots. Unimplemented pin multiplexing slots are reserved. Unimplemented pin multiplexing slots can result in different behaviors, if the unimplemented slot is phantomed (because the module is phantomed on that die) versus unimplemented on the die. The corresponding pin is configured according to the following pin multiplexing slots: 0000b - Pin disabled (analog) 0001b - Alternative 1 (GPIO) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific)
7 —	Reserved
6 DSE	Drive Strength Enable Configures drive strength, low or high, on each pin. The drive strength configuration is valid for all digital pin multiplexing modes. • When this field = 0, low drive strength is configured on the corresponding pin, if the pin is configured as a digital output. • When this field = 1, high drive strength is configured on the corresponding pin, if the pin is configured as a digital output. 0b - Low 1b - High

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 ODE	Open Drain Enable Enables open drain output on each pin. The open drain configuration is valid for all digital pin multiplexing modes. • When this field = 0, the open drain output is disabled on the corresponding pin. • When this field = 1, the open drain output is enabled on the corresponding pin, if the pin is configured as a digital output. 0b - Disables 1b - Enables
4 —	Reserved
3 SRE	Slew Rate Enable Configures the slew rate feature, fast or slow, on each corresponding pin. The slew rate configuration is valid for all digital pin multiplexing modes. 0b - Fast 1b - Slow
2 —	Reserved
1 PE	Pull Enable Enables the internal pull resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pull resistor is not enabled on the corresponding pin. • When this field = 1, the internal pull resistor is enabled on the corresponding pin, if the pin is configured as a digital input. 0b - Disables 1b - Enables
0 PS	Pull Select Enables the internal pullup or pulldown resistor. This configuration is valid for all digital pin multiplexing modes. • When this field = 0, the internal pulldown resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. • When this field = 1, the internal pullup resistor is enabled on the corresponding pin, if the corresponding PCR*n*.PE field = 1. 0b - Enables internal pulldown resistor

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables internal pullup resistor

Chapter 22
General-purpose Input and Output (GPIO)

22.1 Chip-specific GPIO information

Table 180. Reference links to related information

Topic	Related module	Reference
Full description	GPIO	GPIO
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

22.1.1 Module instances

This device has four instances of the GPIO module, GPIOA, GPIOB, GPIOC, and GPIOD.

22.1.2 GPIO interrupts

For each GPIO module, two interrupt channels have been assigned. Refer to Interrupt channel assignments for more details.

22.2 Overview

The GPIO module communicates to the processor core via a zero wait-state interface for maximum pin performance.

22.2.1 Block diagram

The following figure shows the block diagram for the GPIO module.

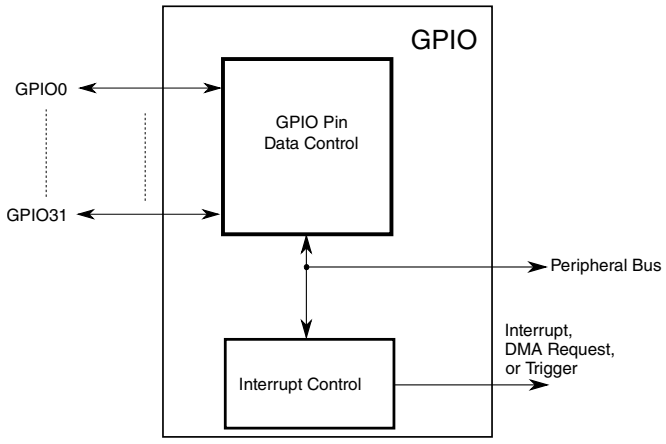


Figure 60. GPIO block diagram

22.2.2 Features

The GPIO module includes the following features:

- Port Data Input (PDIR) register displays the logic value on each pin when the pin is configured for any digital function provided the corresponding Port Control and Interrupt module for that pin are enabled.

- Port Data Output (PDOR) register with corresponding set/clear/toggle registers controls output data of each pin when the pin is configured for the GPIO function.
- Port Data Direction (PDDR) register controls the direction of each pin when the pin is configured for the GPIO function.
- Port Input Disable (PIDR) register controls the disable of the input for each general-purpose pin.
- Pin interrupts
  - Interrupt flag and enable registers for each pin are functional in all digital pin muxing modes.
  - Support for interrupt, peripheral trigger, or DMA request configured per pin.
  - Support for edge sensitive (rising or falling, or both) or level sensitive (low, high) configured per pin.
  - Asynchronous wake-up in Low-Power modes.
  - GPIO module generates a total of 2 interrupts, 2 output triggers and 2 DMA requests.
  - Each pin can be used to generate a single interrupt, output trigger or DMA request.
- Protection registers
  - Each pin is configured for Secure or Non-Secure and Privilege/Non-Priviliege access.
  - Each interrupt, trigger and DMA request domain is configured for Secure or Non-Secure and Privilege/Non-Priviliege access.

22.3 Functional description

22.3.1 Low-Power modes

The GPIO module can be configured to exit the Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

22.3.2 Debug mode

The GPIO module remains functional in debug mode.

22.3.3 General-purpose input

The logic state of each pin is available via the Port Data Input registers (PDIR) if,

- the corresponding bit in the Port Input Disable Register (PIDR) is clear, and
- the pin is configured for a digital function.

22.3.4 General-purpose output

The logic state of each pin is controlled via the Port Data Output Registers(PDOR) and Port Data Direction Registers(PDDR), provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

If	Then
A pin is configured for the GPIO function and the corresponding PDDR bit is clear.	The pin is configured as an input.
A pin is configured for the GPIO function and the corresponding PDDR bit is set.	The pin is configured as an output and the logic state of the pin is equal to the corresponding PDOR.

For efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers allows one or more outputs within one port to be set, cleared, or toggled from a single register write, eliminating the need for a read-modify-write operation to prevent changing pins accidentally.

22.3.5 External interrupts

The external interrupt capability of the GPIO module is available in all digital pin muxing modes.

GPIO module generates a total of 2 interrupts, 2 output triggers and 2 DMA requests. Each pin can be individually configured for interrupt, output trigger or DMA request.

- Each output implements a separate interrupt status flag register for that domain
- Each output generates a single interrupt
- Each output generates a single DMA request
- Each output generates a single peripheral trigger output

Each pin can be individually configured for any of the following external interrupt modes:

Table 181. Available pin configuration for external interrupt

Signal conditions	Software polling using flags	Peripheral triggers	Interrupts	DMA requests
Rising-edge	Yes	—	Yes	Yes
Falling-edge	Yes	—	Yes	Yes
Rising- and falling-edge	Yes	—	Yes	Yes
High-level	—	Yes	Yes	—
Low-level	—	Yes	Yes	—

The interrupt status flag is set when the configured edge or level is detected on the pin. Unless the GPIO module is in a Low-Power mode, the input is first synchronized to the system clock to detect the configured level or edge transition.

The GPIO module generates a pin interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that output. The interrupt negates after the interrupt status flags for all enabled interrupts are cleared by writing a logic 1 to the ISF flag in either the ISFR or ICRn registers.

The GPIO module generates a DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that output. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

In Low-Power mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wake-up signal to exit the Low-Power mode.

The GPIO module generates a peripheral trigger output that asserts if any pin configured for active high trigger is logic one, or any pin triggered for active low trigger is logic zero. The peripheral trigger output asynchronously updates from the value on the configured pins.

22.3.6 Global interrupt control

The two global interrupt control registers (GICLR and GICHR) allow a single register write to update the upper 16-bit of the Interrupt Control Register for up to 16 pins, all with the same value.

The global interrupt control registers allow software to quickly configure multiple pins within the same port with the same interrupt configuration.

The global interrupt control registers are write-only registers and always read as 0.

22.3.7 Access protection

The GPIO module implements access protection registers (PCNS and PCNP) for each pin as follows:

Access	Description
Secure	Pins configured for Secure access can only be written or read by software in Secure state.
Non-Secure	Pins configured for Non-Secure access can only be written or read by software in Non-Secure state.
Privilege	Pins configured for Privilege access can only be written by software in Privilege state.
Non-Privilege	Pins configured for Non-Privilege access can be written by software in both Privilege and Non-Privilege states.

The GPIO module implements access protection registers (ICNS and ICNP) for each interrupt, trigger output and DMA request as follows:

Access	Description
Secure	Outputs configured for Secure access can only be configured by software in Secure state.
Non-Secure	Outputs configured for Non-Secure access can only be configured by software in Non-Secure state.
Privilege	Outputs configured for Privilege access can only be configured by software in Privilege state.
Non-Privilege	Outputs configured for Non-Privilege access can be configured by software in both Privilege and Non-Privilege states.

The access protection registers (PCNS, PCNP, ICNS and ICNP) can only be written by software in secure-privilege state and can be optionally locked until the next reset.

Configuring a pin interrupt/trigger output/DMA request requires the following access permissions:

- Access permission to configure that pin.
- Access permission to configure to the desired interrupt/trigger output/DMA request, or writing IRQC=0.
- Access permission to configure with the existing interrupt/trigger output/DMA request, or currently IRQC=0.
- The interrupt configuration is not locked.

22.4 External signals

Table 182. GPIO external signals

Signal	Description		Direction
GPIO31 - GPIO0	General-purpose input/output		I/O
	State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.	
	Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.	

Table continues on the next page...

Table 182. GPIO external signals

Signal	Description		Direction
		Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.	

NOTE

Not all pins within each GPIO are implemented on each device. See the chapter on signal multiplexing for the number of GPIO pins within each port available in the device.

22.5 Initialization

To initialize the GPIO module:

1. Initialize GPIO pins for output function
   a. Configure the output logic value for each pin by using PDOR register.
   b. Configure the direction for each pin by using PDDR register.
2. Initialize interrupt function
   - Write to ICRn for the corresponding pins and desired configuration.
   - If the pin is previously used for a different function, first write 0x0100_0000 to ICRn register to disable the previous function and clear the flag.

22.6 Application information

The GPIO module has the following applications:

- Read the state of a single pin
  - Perform a byte read of PnDR register.
- Update the state of a single pin
  - Perform a byte write to PnDR register.
- Read the state of multiple pins
  - Read PDIR register.
- Update the state of multiple pins
  - Write to PDOR register bits.
  - Write to PSOR register to set PDOR register bits.
  - Write to PCOR register to clear PDOR register bits.
  - Write to PTOR register to toggle PDOR register bits.

22.7 Memory map and register definition

The GPIO registers support 8-bit, 16-bit, or 32-bit accesses. Any read or write access to the GPIO memory space that is outside the valid memory map results in a bus error.

NOTE

For simplicity, each GPIO port's register appears with the same width of 32 bits, corresponding to 32 pins. The actual number of pins per port (and therefore the number of usable control bits per port register) is chip-specific. See the Signal Multiplexing for exact control bits for each port.

22.7.1 GPIO register descriptions

22.7.1.1 GPIO memory map

GPIOA base address: 4801_0000h

GPIOB base address: 4802_0000h

GPIOC base address: 4803_0000h

GPIOD base address: 4004_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0201_0001h
4h	Parameter (PARAM)	32	R	0000_0002h
Ch	Lock (LOCK)	32	RW	0000_0000h
10h	Pin Control Non-Secure (PCNS)	32	RW	0000_0000h
14h	Interrupt Control Non-Secure (ICNS)	32	RW	0000_0000h
18h	Pin Control Non-Privilege (PCNP)	32	RW	0000_0000h
1Ch	Interrupt Control Non-Privilege (ICNP)	32	RW	0000_0000h
40h	Port Data Output Register (PDOR)	32	RW	0000_0000h
44h	Port Set Output Register (PSOR)	32	W	0000_0000h
48h	Port Clear Output Register (PCOR)	32	W	0000_0000h
4Ch	Port Toggle Output Register (PTOR)	32	W	0000_0000h
50h	Port Data Input Register (PDIR)	32	R	0000_0000h
54h	Port Data Direction Register (PDDR)	32	RW	0000_0000h
58h	Port Input Disable Register (PIDR)	32	RW	0000_0000h
60h - 7Fh	Pin Data Register a (P0DR - P31DR)	8	RW	00h
80h - FCh	Interrupt Control Register a (ICR0 - ICR31)	32	RW	0000_0000h
100h	Global Interrupt Control Low Register (GICLR)	32	W	0000_0000h
104h	Global Interrupt Control High Register (GICHR)	32	W	0000_0000h
120h - 124h	Interrupt Status Flag Register (ISFR0 - ISFR1)	32	RW	0000_0000h

22.7.1.2 Version ID (VERID)

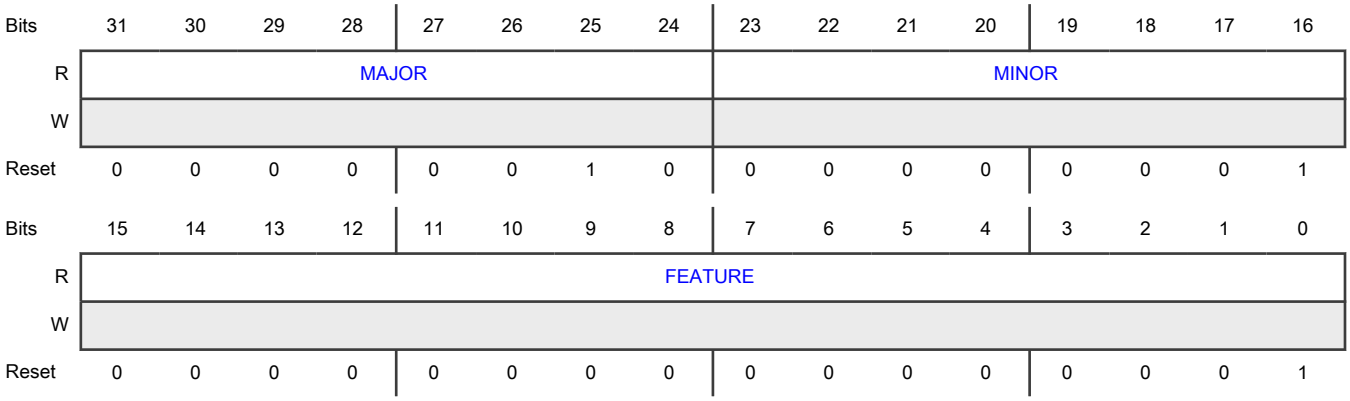
Offset

Register	Offset
VERID	0h

Function

Indicates the version ID number of each GPIO.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000_0000_0000_0000b - Basic implementation. 0000_0000_0000_0001b - Protection registers implemented.

22.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Indicates the interrupt number of each GPIO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												IRQNUM			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-4 —	Reserved
3-0 IRQNUM	Interrupt Number This field indicates the number of Interrupt/Trigger/DMA request domains.

22.7.1.4 Lock (LOCK)

Offset

Register	Offset
LOCK	Ch

Function

Locks the Non-Secure and Non-Privilege access protection registers. This register can be written only by software in Secure-Privilege state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												ICNP	PCNP	ICNS	PCNS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 ICNP	Lock ICNP This field locks ICNP register. When set, the ICNP register is not writable until the next reset. 0b - ICNP register is writable by software in Secure-Privilege state. 1b - ICNP register is not writable until the next reset.
2 PCNP	Lock PCNP This field locks PCNP register. When set, the PCNP register is not writable until the next reset. 0b - PCNP register is writable by software in Secure-Privilege state. 1b - PCNP register is not writable until the next reset.
1 ICNS	Lock ICNS This field locks ICNS register. When set, the ICNS register is not writable until the next reset. 0b - ICNS register is writable by software in Secure-Privilege state. 1b - ICNS register is not writable until the next reset.
0 PCNS	Lock PCNS This field locks PCNS register. When set, the PCNS register is not writable until the next reset. 0b - PCNS register is writable by software in Secure-Privilege state. 1b - PCNS register is not writable until the next reset.

22.7.1.5 Pin Control Non-Secure (PCNS)

Offset

Register	Offset
PCNS	10h

Function

Configures Secure or Non-Secure access protection for each pin. This register can only be written by software in Secure-Privilege state if it is not locked (LOCK[PCNS] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NSE31	NSE30	NSE29	NSE28	NSE27	NSE26	NSE25	NSE24	NSE23	NSE22	NSE21	NSE20	NSE19	NSE18	NSE17	NSE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NSE15	NSE14	NSE13	NSE12	NSE11	NSE10	NSE9	NSE8	NSE7	NSE6	NSE5	NSE4	NSE3	NSE2	NSE1	NSE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NSEn	Non-Secure Enable This field configures the Secure or Non-Secure access protection for each pin. 0b - The pin is configured for Secure access. Read or write access to the corresponding pin's registers and bit fields is only allowed by software in Secure state. When the corresponding pin's registers are accessed by software in Non-Secure state, all bits in the registers related to that pin are read zero and write ignored. 1b - The pin is configured for Non-Secure access. Read or write access to the corresponding pin's registers and bit fields is only allowed by software in Non-Secure state. When the corresponding pin's registers are accessed by software in Secure state, all bits in the registers related to that pin are read zero and write ignored.

22.7.1.6 Interrupt Control Non-Secure (ICNS)

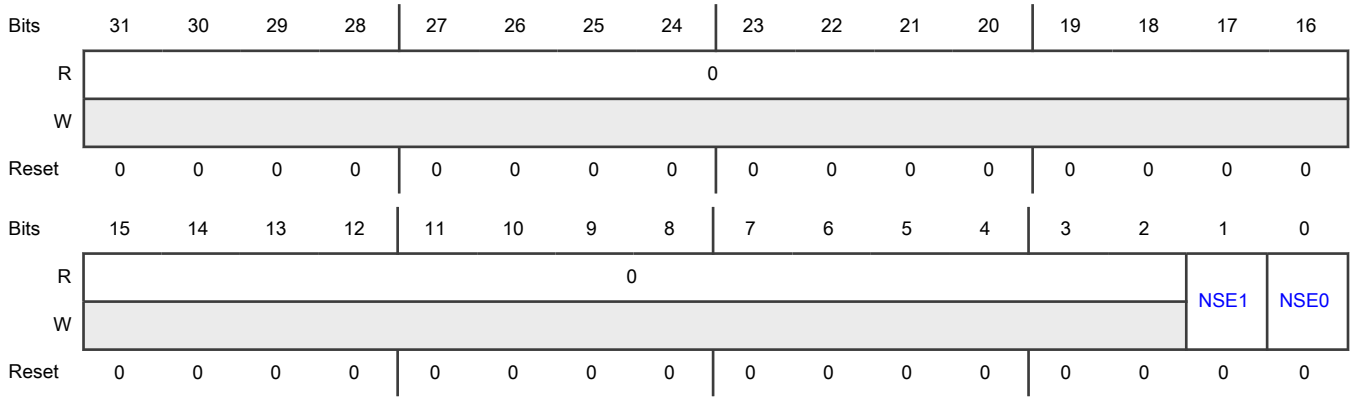
Offset

Register	Offset
ICNS	14h

Function

Configures Secure/Non-Secure access protection for each interrupt, output trigger or DMA request. This register can only be updated by software in Secure-Privilege state if it is not locked (LOCK[ICNS] = 0).

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 NSEn	Non-Secure Enable This field configures Secure or Non-Secure access protection for each interrupt, output trigger or DMA request. 0b - The interrupt, output trigger or DMA request is configured for Secure access. Only software in Secure state can configure a pin to use the corresponding interrupt, output trigger or DMA request or reconfigure a pin that is already configured to use the corresponding interrupt, output trigger or DMA request. 1b - The interrupt, output trigger or DMA request is configured for Non-Secure access. Only software in Non-Secure state can configure a pin to use the corresponding interrupt, output trigger or DMA request or reconfigure a pin that is already configured to use the corresponding interrupt, output trigger or DMA request.

22.7.1.7 Pin Control Non-Privilege (PCNP)

Offset

Register	Offset
PCNP	18h

Function

Configures either for Privilege or for both (Privilege and Non-Privilege) access protection for each pin. This register can only be updated by software in Secure-Privilege state if it is not locked (LOCK[PCNP] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NPE31	NPE30	NPE29	NPE28	NPE27	NPE26	NPE25	NPE24	NPE23	NPE22	NPE21	NPE20	NPE19	NPE18	NPE17	NPE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NPE15	NPE14	NPE13	NPE12	NPE11	NPE10	NPE9	NPE8	NPE7	NPE6	NPE5	NPE4	NPE3	NPE2	NPE1	NPE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NPEn	Non-Privilege Enable This field configures Privilege/Non-Privilege access protection for each pin. 0b - The pin is configured for Privilege access. Write access to the corresponding pin's registers and bit fields is allowed only by software in Privilege state. When the corresponding pin's registers and bit fields are accessed by software in Non-Privilege state, all bits related to that pin in this GPIO are readable but write ignored. 1b - The pin is configured for Non-Privilege access, Read or write access to the corresponding pin's registers is allowed by software in both Privilege or Non-Privilege state.

22.7.1.8 Interrupt Control Non-Privilege (ICNP)

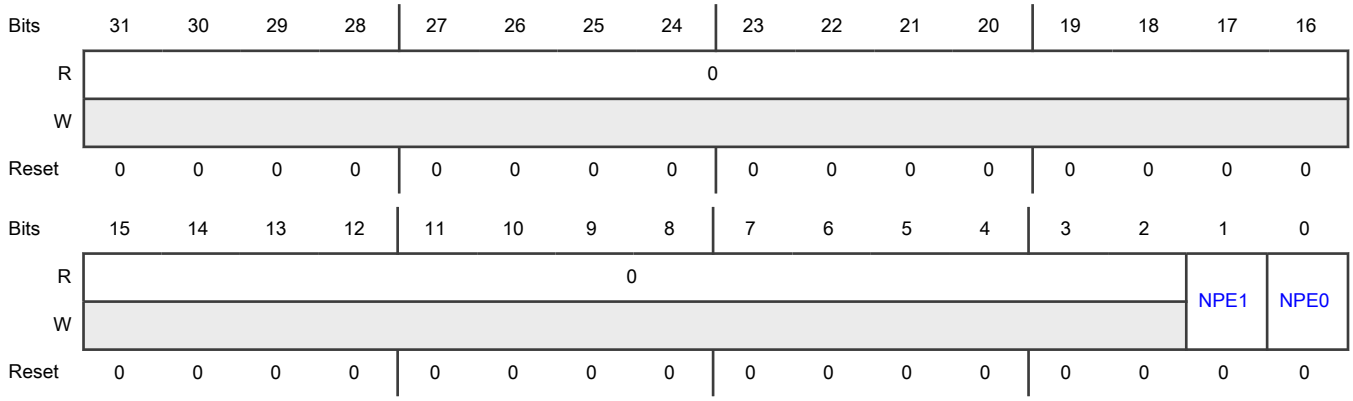
Offset

Register	Offset
ICNP	1Ch

Function

Configures Privilege/Non-Privilege access protection for each interrupt/trigger output/DMA request. This register can only be updated by software in Secure-Privilege state if it is not locked (LOCK[ICNP] = 0).

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 NPEn	Non-Privilege Enable This field configures Privilege/Non-Privilege access protection for each interrupt/trigger output/DMA request. 0b - The pin is configured for Privilege access. Only software in Privilege state can configure a pin to use the corresponding interrupt/trigger output/DMA request or reconfigure a pin that is already configured to use the corresponding interrupt/trigger output/DMA request. 1b - The pin is configured for Non-Privilege access. Software in either Privilege or Non-Privilege state can configure a pin to use the corresponding interrupt/trigger output/DMA request or reconfigure a pin that is already configured to use the corresponding interrupt/trigger output/DMA request.

22.7.1.9 Port Data Output Register (PDOR)

Offset

Register	Offset
PDOR	40h

Function

Configures the logic levels that are driven on each general-purpose output pin.

NOTE

Do not modify the pin configuration registers associated with pins that are not available in your selected package. These unbonded pins are default set to Disbale state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDO31	PDO30	PDO29	PDO28	PDO27	PDO26	PDO25	PDO24	PDO23	PDO22	PDO21	PDO20	PDO19	PDO18	PDO17	PDO16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDO15	PDO14	PDO13	PDO12	PDO11	PDO10	PDO9	PDO8	PDO7	PDO6	PDO5	PDO4	PDO3	PDO2	PDO1	PDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDOn	Port Data Output This field configures the logic level on the pin if it is configured for general-purpose output. **NOTE** Reading the fields for unbonded pins returns an undefined value. 0b - Logic level 0 is driven on pin, if the pin is configured for general-purpose output. 1b - Logic level 1 is driven on pin, if the pin is configured for general-purpose output.

22.7.1.10 Port Set Output Register (PSOR)

Offset

Register	Offset
PSOR	44h

Function

Configures whether to set the corresponding fields of the PDOR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTSO 31	PTSO 30	PTSO 29	PTSO 28	PTSO 27	PTSO 26	PTSO 25	PTSO 24	PTSO 23	PTSO 22	PTSO 21	PTSO 20	PTSO 19	PTSO 18	PTSO 17	PTSO 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTSO 15	PTSO 14	PTSO 13	PTSO 12	PTSO 11	PTSO 10	PTSO 9	PTSO 8	PTSO 7	PTSO 6	PTSO 5	PTSO 4	PTSO 3	PTSO 2	PTSO 1	PTSO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PTSOn	Port Set Output Writing to this field updates the content of the corresponding field in the PDOR as follows: 0b - Corresponding field of PDOR[PDOn] does not change. 1b - Corresponding field of PDOR[PDOn] is set to logic 1.

22.7.1.11 Port Clear Output Register (PCOR)

Offset

Register	Offset
PCOR	48h

Function

Configures whether to clear the corresponding fields of PDOR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTCO 31	PTCO 30	PTCO 29	PTCO 28	PTCO 27	PTCO 26	PTCO 25	PTCO 24	PTCO 23	PTCO 22	PTCO 21	PTCO 20	PTCO 19	PTCO 18	PTCO 17	PTCO 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTCO 15	PTCO 14	PTCO 13	PTCO 12	PTCO 11	PTCO 10	PTCO 9	PTCO 8	PTCO 7	PTCO 6	PTCO 5	PTCO 4	PTCO 3	PTCO 2	PTCO 1	PTCO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PTCOn	Port Clear Output Writing to this field updates the content of the corresponding field in the PDOR as follows: 0b - Corresponding field of PDOR[PDOn] does not change. 1b - Corresponding field of PDOR[PDOn] is cleared to logic 0.

22.7.1.12 Port Toggle Output Register (PTOR)

Offset

Register	Offset
PTOR	4Ch

Function
Configures whether to toggle the corresponding fields of PDOR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTTO 31	PTTO 30	PTTO 29	PTTO 28	PTTO 27	PTTO 26	PTTO 25	PTTO 24	PTTO 23	PTTO 22	PTTO 21	PTTO 20	PTTO 19	PTTO 18	PTTO 17	PTTO 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	PTTO 15	PTTO 14	PTTO 13	PTTO 12	PTTO 11	PTTO 10	PTTO 9	PTTO 8	PTTO 7	PTTO 6	PTTO 5	PTTO 4	PTTO 3	PTTO 2	PTTO 1	PTTO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PTTOn	Port Toggle Output Writing to this field updates the content of the corresponding field in the PDOR as follows: 0b - Corresponding field of PDOR[PDOn] does not change. 1b - Corresponding field of PDOR[PDOn] is set to the inverse of its current logic state.

22.7.1.13 Port Data Input Register (PDIR)

Offset

Register	Offset
PDIR	50h

Function

Captures the logic levels of each general-purpose input pin.

NOTE

Do not modify the pin configuration registers associated with the pins that are not available in your selected package. These unbonded pins are default set to Disable state for lowest power consumption.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI31	PDI30	PDI29	PDI28	PDI27	PDI26	PDI25	PDI24	PDI23	PDI22	PDI21	PDI20	PDI19	PDI18	PDI17	PDI16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI15	PDI14	PDI13	PDI12	PDI11	PDI10	PDI9	PDI8	PDI7	PDI6	PDI5	PDI4	PDI3	PDI2	PDI1	PDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDIn	Port Data Input The field indicates the logic level of the pin that is configured for use by digital function. Reads 0 at the unimplemented pins or pins that are not configured for a digital function. 0b - Pin logic level is logic 0, or is not configured for use by digital function. 1b - Pin logic level is logic 1.

22.7.1.14 Port Data Direction Register (PDDR)

Offset

Register	Offset
PDDR	54h

Function

Configures the individual port pins for input or output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDD31	PDD30	PDD29	PDD28	PDD27	PDD26	PDD25	PDD24	PDD23	PDD22	PDD21	PDD20	PDD19	PDD18	PDD17	PDD16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDD15	PDD14	PDD13	PDD12	PDD11	PDD10	PDD9	PDD8	PDD7	PDD6	PDD5	PDD4	PDD3	PDD2	PDD1	PDD0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDDn	Port Data Direction This field configures individual port pins for input or output. 0b - Pin is configured as general-purpose input for the GPIO function. 1b - Pin is configured as general-purpose output for the GPIO function.

22.7.1.15 Port Input Disable Register (PIDR)

Offset

Register	Offset
PIDR	58h

Function

Disables the input for each general-purpose pin, which prevents the value from being reported in the PDIR register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24	PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDn	Port Input Disable This field is used to disable a pin for general-purpose input 0b - Pin is configured for general-purpose input provided, the pin is configured for any digital function. 1b - Pin is disabled for general-purpose input.

22.7.1.16 Pin Data Register a (P0DR - P31DR)

Offset

For a = 0 to 31:

Register	Offset
PaDR	60h + (a × 1h)

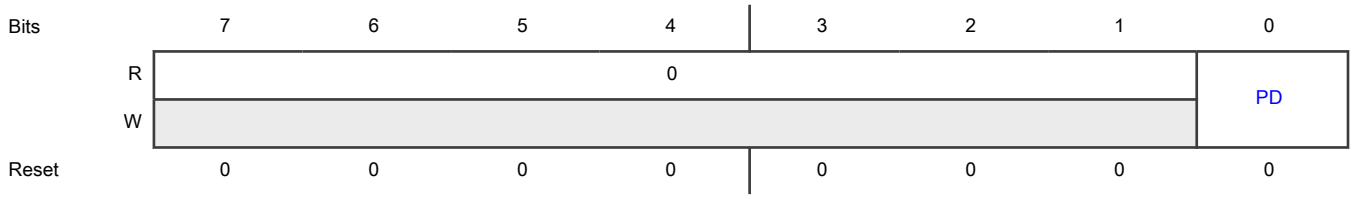
Function

Configures the data feature of a pin. Pin that is unimplemented or not configured for a digital function reads as zero.

NOTE

Do not modify PaDR registers associated with the pins that are not available in your selected package. These unbonded pins are default set to Disable state for lowest power consumption.

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PD	Pin Data (input and output) Writing to this bit field of PaDR register will update corresponding bit a in the PDOR register, reading this bit field of PaDR register will return value in the corresponding bit a of the PIDR register. 0b - Pin logic level is logic zero or not configured for use by digital function. 1b - Pin logic level is logic one.

22.7.1.17 Interrupt Control Register a (ICR0 - ICR31)

Offset

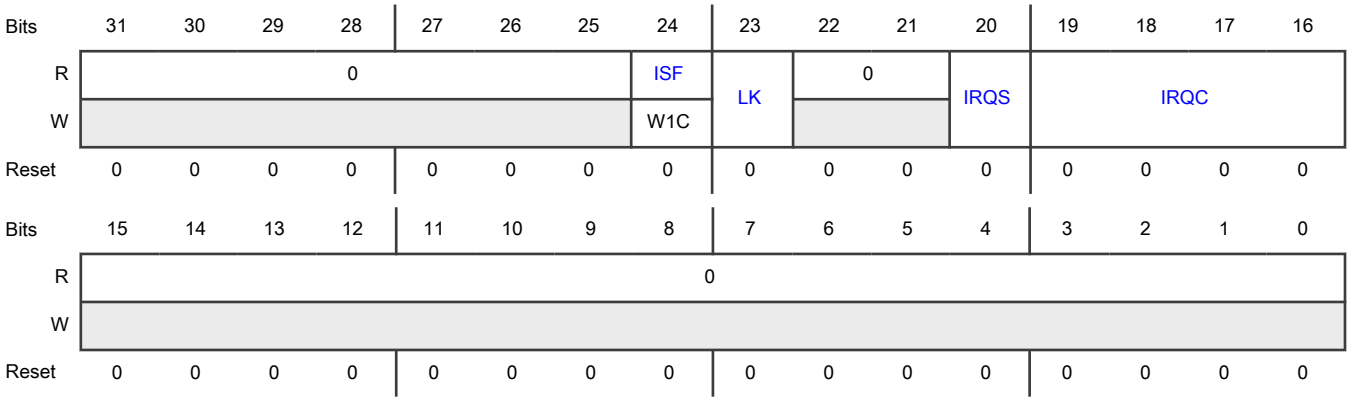
For a = 0 to 31:

Register	Offset
ICRa	80h + (a × 4h)

Function

Configures interrupt features on each pin.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ISF	Interrupt Status Flag This field indicates whether the configured interrupt is detected. The pin interrupt configuration is valid in all digital pin muxing modes. The bit fields in the ISFR register have the same function. The interrupt status flag can be cleared with either register bit. 0b - Configured interrupt is not detected. 1b - Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23 LK	Lock Register This field is used to lock ICR[23:0], and the locked field cannot be updated until the next system reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt configuration by ICR[23:0] is not locked and can be updated. 1b - Interrupt configuration by ICR[23:0] is locked and cannot be updated until next system reset.
22-21 —	Reserved
20 IRQS	Interrupt Select This field configures the interrupt/trigger output/DMA request. 0b - Interrupt/trigger output/DMA request 0. 1b - Interrupt/trigger output/DMA request 1.
19-16 IRQC	Interrupt Configuration The pin interrupt configuration is valid in all digital pin muxing modes. When changing the interrupt configuration, it is recommended to first disable the interrupt status flag and then write the new configuration. The corresponding pin is configured to generate interrupt, trigger or DMA request as follows: 0000b - Interrupt Status Flag (ISF) is disabled. 0001b - ISF flag and DMA request on rising edge. 0010b - ISF flag and DMA request on falling edge. 0011b - ISF flag and DMA request on either edge. 0100b - Reserved. 0101b - ISF flag sets on rising edge. 0110b - ISF flag sets on falling edge. 0111b - ISF flag sets on either edge. 1000b - ISF flag and Interrupt when logic 0. 1001b - ISF flag and Interrupt on rising-edge. 1010b - ISF flag and Interrupt on falling-edge. 1011b - ISF flag and Interrupt on either edge. 1100b - ISF flag and Interrupt when logic 1. 1101b - Enable active high trigger output, ISF flag on rising edge. Pin state is ORed with other enabled triggers to generate the output trigger, for use by other peripherals. 1110b - Enable active low trigger output, ISF flag on falling edge. Pin state is inverted and ORed with other enabled triggers to generate the output trigger, for use by other peripherals. 1111b - Reserved.
15-0 —	Reserved

22.7.1.18 Global Interrupt Control Low Register (GICLR)

Offset

Register	Offset
GICLR	100h

Function
Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GIWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	GIWE 15	GIWE 14	GIWE 13	GIWE 12	GIWE 11	GIWE 10	GIWE 9	GIWE 8	GIWE 7	GIWE 6	GIWE 5	GIWE 4	GIWE 3	GIWE 2	GIWE 1	GIWE 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 GIWD	Global Interrupt Write Data Write value that is written to upper 16 bits of the Interrupt Control Registers that are selected by GIWE.
15-0 GIWEn	Global Interrupt Write Enable This field selects whether the upper 16-bit field of the Interrupt Control Register ICR(n) will be updated with the value in GIWD. 0b - Upper 16-bit of corresponding Interrupt Control Register is not updated with the value in GIWD. 1b - Upper 16-bit of corresponding Interrupt Control Register is updated with the value in GIWD.

22.7.1.19 Global Interrupt Control High Register (GICHR)

Offset

Register	Offset
GICHR	104h

Function

Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GIWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	GIWE 31	GIWE 30	GIWE 29	GIWE 28	GIWE 27	GIWE 26	GIWE 25	GIWE 24	GIWE 23	GIWE 22	GIWE 21	GIWE 20	GIWE 19	GIWE 18	GIWE 17	GIWE 16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 GIWD	Global Interrupt Write Data Write value that is written to upper 16 bits of the Interrupt Control Registers that are selected by GIWE.
15-0 GIWEn	Global Interrupt Write Enable This field selects whether the upper 16-bit field of the Interrupt Control Register ICR(n) will be updated with the value in GIWD. 0b - Upper 16-bit of corresponding Interrupt Control Register is not updated with the value in GIWD. 1b - Upper 16-bit of corresponding Interrupt Control Register is updated with the value in GIWD.

22.7.1.20 Interrupt Status Flag Register (ISFR0 - ISFR1)

Offset

Register	Offset
ISFR0	120h
ISFR1	124h

Function

The Interrupt Status Flag Register (ISFR) indicates whether the related configured interrupt is detected on each pin. The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Interrupt Control Register, and each flag can be cleared in either location.

There is a separate Interrupt Status Flag Register for each interrupt, trigger, or DMA request domain. Each status flag is only visible in the register that corresponds to that flag's domain as configured in IRCa[IRQS].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ISF31	ISF30	ISF29	ISF28	ISF27	ISF26	ISF25	ISF24	ISF23	ISF22	ISF21	ISF20	ISF19	ISF18	ISF17	ISF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISF15	ISF14	ISF13	ISF12	ISF11	ISF10	ISF9	ISF8	ISF7	ISF6	ISF5	ISF4	ISF3	ISF2	ISF1	ISF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ISFn	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt on each pin of the same number. 0b - Configured interrupt is not detected on the pin of the same number. 1b - Configured interrupt is detected on the pin of the same number. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

Chapter 23
Debug

23.1 Introduction

This chapter describes the debug architecture of this device.

Features:

- The Arm CoreSight™ Architecture is adapted for software trace and debug.
- Serial wire debug (SWD) and Debug Access Port (DAP) to support debugger tools.
- Instrumentation Trace Macrocell (ITM) is supported for the Cortex-M33 core and Radio Cortex-M3 core.
- Data Watchpoint and Trace (DWT) and breakpoint unit (BPU) are supported for the Cortex-M33 and Radio Cortex-M3 core subsystem.
- Cross trigger interface (CTI) are supported for the Cortex-M33 core and radio Cortex-M3 sbsystem.
- 1-bit SWO trace port for efficiently accessing Cortex-M33 trace information from the system.

The following figure shows the system level debug architecture.

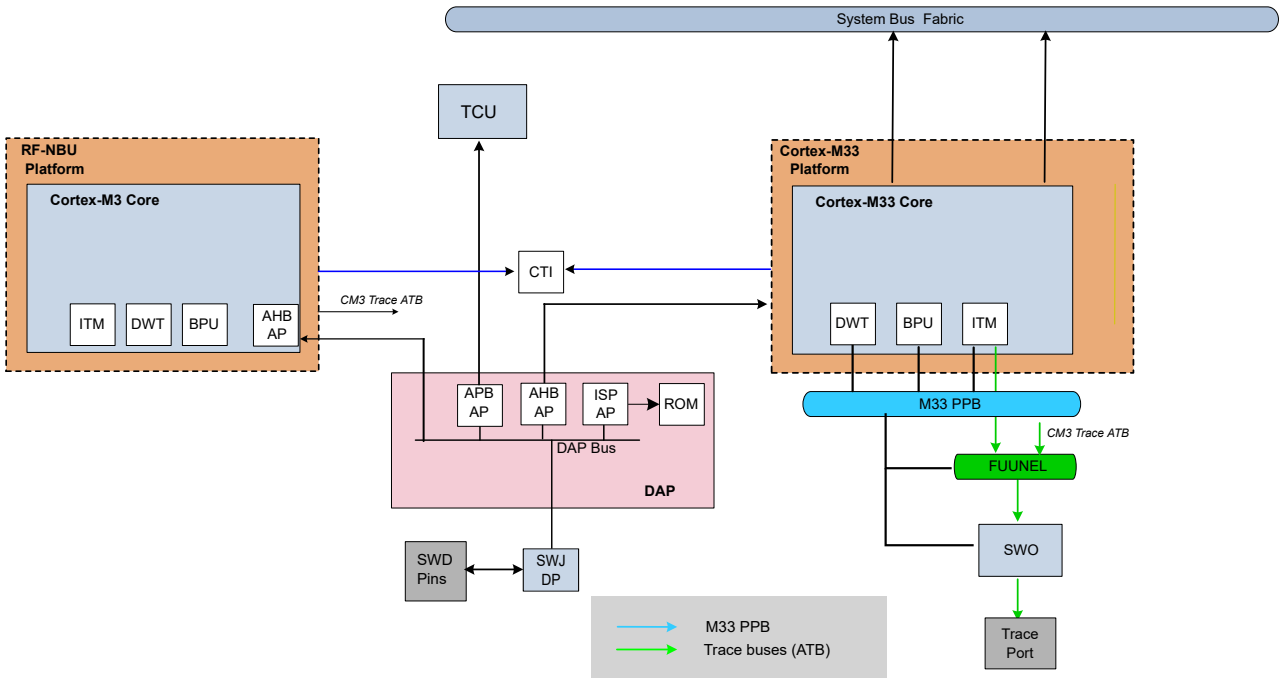


Figure 61. Debug architecture

23.2 Test and debug port connectivity

The following figure shows the connectivity of test and debug port on this device.

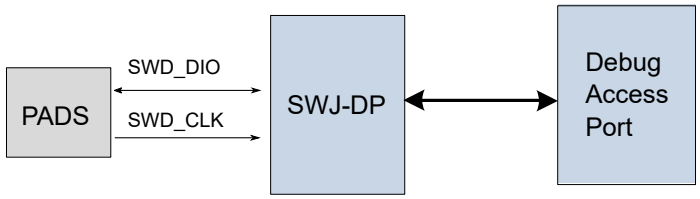


Figure 62. Debug port connectivity

The SWD signals are summarized in the following table.

Table 183. SWD signals summary

Pin Name	SWD		Internal pull up/down
	Type	Description	
SWD_DIO	I/O	Data	Pull-up
SWD_CLK	I	Clock	Pull-down

23.2.1 Serial wire debug (SWD)

The serial wire debug is a 2-pin electrical interface with a clock (SWD_CLK) and a single bi-directional data pin (SWD_DIO) to provide debug and test functionality.

23.2.2 Debug access port

Debug Access Port (DAP) is a standard Arm CoreSight component. The DAP provides multiple master driving ports, all of which are accessible and controlled through a single external interface port to provide system-wide debug. The DAP Instruction Register codes are listed in the following table.

Table 184. DAP IR Codes

Code	DAP IR
4'b1000	ABORT
4'b1010	DPACC
4'b1011	APACC
4'b1110	IDCODE

The DAP offers AHB and APB master interfaces to access system buses. It also exports the internal DAP bus to allow to extend the access ports as per the system requirement. For more information on DAP TAP, refer to Arm Debug Interface v5 Architecture specification on arm.com.

The exported DAP bus is used to host AHB AP, APB AP, MDM AP and ISP AP. The APB-AP implements registers which are used for Test mode control on the device. The MDM-AP hosts system level debug status and control registers (refer to Debug Status and Control Registers) which can be used for low power and other miscellaneous control and status. The ISP AP supports software debug feature at platform level. The selection of different access ports in the DAP is done based on the APSEL value set in the SELECT register of SWJ-DP.

The following figure shows the configuration of the DAP on this device.

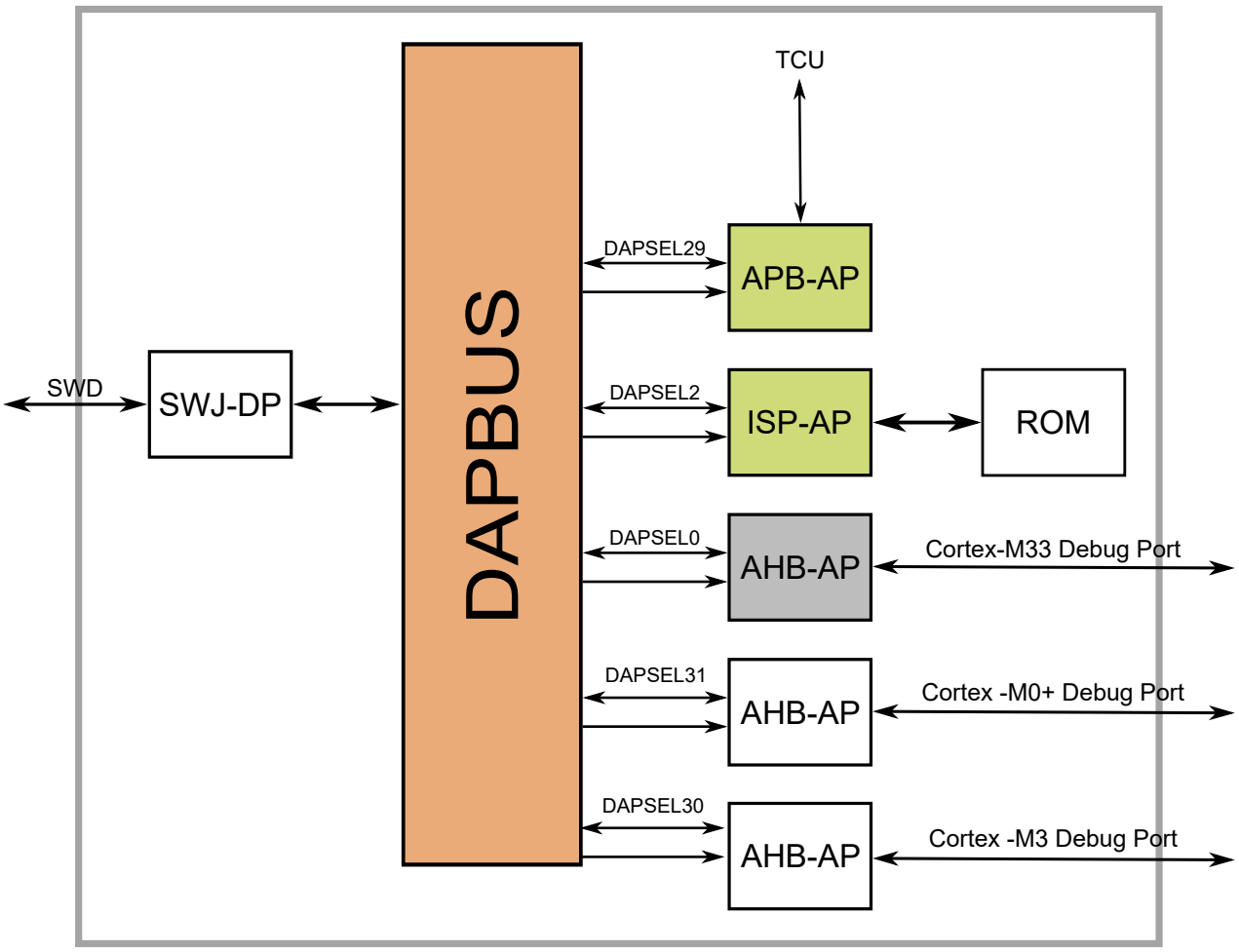


Figure 63. Debug Access Port configuration

23.3 Debug ROM tables

The debug ROM tables hold information about different debug components and help identify them. The following ROM table is available:

1. CM33 ROM: This table resides in the CM33 core and contains entries for CM33 debug components

23.4 Cortex M33 debug

This section covers the debug mechanisms associated with the CM33 processor.

23.4.1 CM33 software debug and trace

The following figure shows the data flow of software debug and trace information for the Cortex-M33 core on this device.

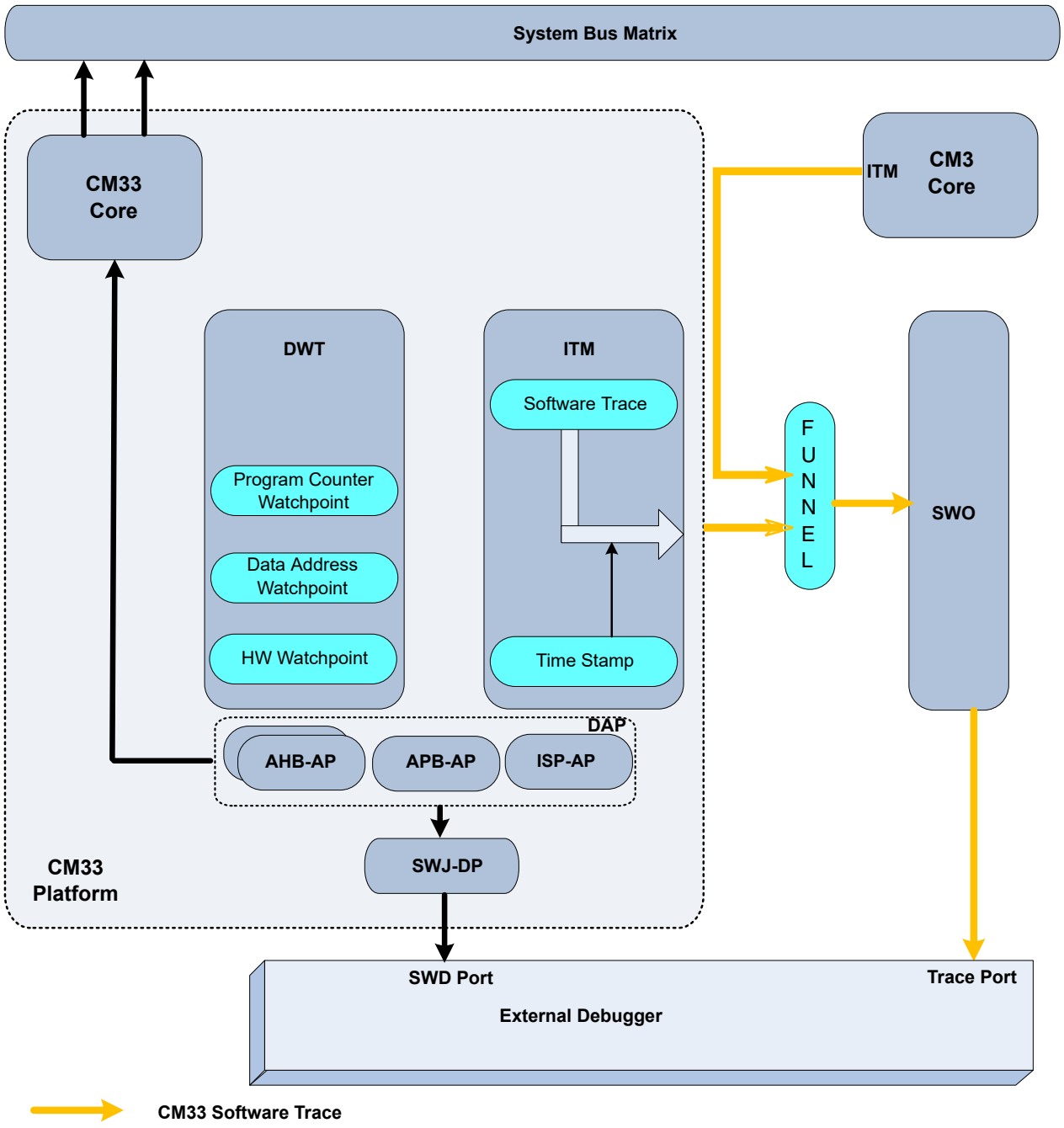


Figure 64. Software debug and trace flow

23.4.2 Data Watchpoint and Trace

The CM33 Data Watchpoint and Trace (DWT) module generates trace information in the core. It has 4 independent watchpoints which can be programmed to compare data, address, or to function as a hardware watchpoint. The DWT can be configured to generate PC samples at defined intervals and to generate interrupt event information. The CM33 DWT also provides periodic requests for protocol synchronization to the ITM and the SWO.

23.4.3 Breakpoint Unit

The CM33 Breakpoint Unit (BPU) module comprises up to 8 instruction address comparators. It provides breakpoint functionality on all instructions fetched across the entire address range in which code can be located.

23.4.4 Instrumentation Trace Macrocell (ITM)

The ITM generates trace information as packets. There are multiple sources that can generate packets. If multiple sources generate packets at the same time, the ITM arbitrates the order in which packets are output. The sources in decreasing order of priority are:

- Software Trace: Application software can write console messages directly to ITM stimulus ports, and output them to the host as trace packets.
- Hardware trace: The DWT generates these packets, and the ITM outputs them.
- Timestamping: ITM can generate timestamp packets that are inserted in to the trace stream, to help the host debugger to find out the timing of events. Timestamps are generated relative to packets. The ITM contains a 21-bit counter to generate the timestamp.

Trace data from ITM will be forwarded to SWO and streamed out via the trace port.

23.5 Cryptographic Acceleration Unit V3 (CAU3) debug

This section covers the debug mechanisms associated with the CAU3 security module.

23.6 Cross Trigger Interface (CTI)

Table 185. CTI input triggers (CM33)

Trigger bit	Description	Connection	Acknowledge, handshake
7	--	--	--
6	--	--	--
5	--	--	--
4	--	--	--
3	--	--	--
2	--	--	--
1	--	--	--
0	Processor halted	Processor to CTI	Pulsed

Table 186. CTI output triggers (CM33)

Trigger bit	Description	Connection	Acknowledge, handshake
7	--	--	--
6	--	--	--
5	--	--	--
4	--	--	--

Table continues on the next page...

Table 186. CTI output triggers (CM33) (continued)

Trigger bit	Description	Connection	Acknowledge, handshake
3	Interrupt request 1	CTI to system	Acknowledged by writing to the CTINTACK register in ISR
2	Interrupt request 0		
1	Processor restart	CTI to processor	Processor restarted
0	Processor debug request		Acknowledged by the debugger writing to the CTINTACK register

Table 187. CTI input triggers (CM3)

Trigger Bit	Source Signal	Source Module	Description
7	--	--	--
6	ETMTRIGGER[2]	Processor DWT	Processor DWT trigger
5	ETMTRIGGER[1]	Processor DWT	Processor DWT trigger
4	ETMTRIGGER[0]	Processor DWT	Processor DWT trigger
3	--	--	--
2	--	--	--
1	--	--	--
0	HALTED	Processor core	Processor core's output indicating that it has entered the debug mode.

Here the "Source Signal" is the output port of the "Source Module". They are connected to the input triggers of the CTI.

Table 188. CTI output triggers (CM3)

Trigger Bit	Destination Signal	Destination Module	Description
7	DBGRESTART	Processor core	Request Processor core to quit the debug mode
6	--	--	--
5	--	--	--
4	--	--	--
3	cti_irq	NVIC	CTI interrupt
2	cti_irq	NVIC	CTI interrupt
1	--	--	--
0	EDBGRQ	Processor core	External request Processor core to enter the debug mode

Here the "Destination Signal" is the input port of the "Destination Module". The output triggers of the CTI are connected to the input port of the destination modules.

23.7 Low power debug

Low power debug is controlled by DBGCTL[SOD] field in the CMC. When set, the debugger is disabled (debug power up acknowledge negates) when the CM33 sleeps and the device fully enters the low power mode. When clear, the clocks to the CM33 remain enabled when the CM33 sleeps and debug is attached (debug power up request is asserted) to maintain the debug connection, and the low power mode is not completely entered. Software can modify DBGCTL[SOD] before each low power entry. There are two modes supported:

- Clock gated mode
- Power gated mode

23.7.1 Clock gated modes

- If the DAP CXXXPWRUPREQ is high and the system attempts to enter STOP mode, the DAP clock and the Core clock continue to run to support core register access and trace, in this case the debug module will have access to core registers but not to modules which are clock gated.
- If the device is in STOP mode (core clock gated), the debugger can assert EDBGRQ (CM33 EDBGRQ) in dap_ctrl_halt_req register, this will wake up the core and will move it into a halted state.

23.7.2 Power gated modes

During power gated modes, the debugger will loose connection and it cannot gather any data in the duration when device stays in low power mode. If the device is expected to go into low power mode and it is desired to gain control of device when it exits low power modes.

Chapter 24
Reset

24.1 Introduction

The following reset sources exist in this device:

Table 189. Reset sources

Reset type	Source
Power	• System Power-On Reset (POR) • System Low-Voltage Detect (LVD) • System High-Voltage Detect (HVD)
External	• External Pin Reset (RESET_b) • Wake-Up Unit (WUU)
Internal	• Watchdog (WDOG) timer • Software (SW) • System Clock Generator (SCG) • Reset Acknowledge timeout (RSTACK) • Low Power Acknowledge (LPACK) • Lockup (LOCKUP) • Debug Access Port (DAP) • EdgeLock Secure Enclave (ELE) • Tamper

Each system reset source has a corresponding bit that is maintained in the System Reset Status register through reset. This information can be used to take appropriate action when coming out of a reset.

24.2 Power reset sources

To ensure predictable operation of the device, the supply voltages must remain in their specified ranges during execution. Because this is not possible during power-up and power-down, the device implements multiple voltage monitors to detect when the supply is outside an acceptable range. When this occurs, the voltage monitor asserts a reset to halt execution and prevent unexpected behavior. The voltage monitors detect power-on events as well as low and high voltage events.

24.2.1 Power-on reset (POR)

The POR voltage monitor is used to ensure that the voltage applied to the system is high enough for other analog modules such as the bias circuits and low voltage monitors to operate correctly. When voltage is initially applied to the device or when the supply voltage drops below the POR falling threshold, the POR monitor triggers the POR reset condition to the device.

The POR reset condition asserts POR reset to all power domains. A POR reset sets the appropriate bits in the CMC registers so that software can determine that a POR reset has occurred.

24.2.2 Low-voltage detect (LVD)

The LVD monitors are used to ensure that the voltage supplies to the device are high enough for the logic and memories to operate correctly. When voltage is initially applied to the device, the POR voltage monitor holds the LVD monitors in reset until the voltages are high enough. The LVD monitors continue to hold the device in reset until all monitoring voltages have risen above the respective default LVD Low thresholds (VLVDL). The LVD monitors are enabled by default.

When enabled, the LVD monitor generates a reset when the voltage drops below the V_{LVD} threshold. The voltage threshold at which the LVD reset asserts is programmable. This ensures that the device halts operation and remains in reset while the monitored voltage is not within the desired range.

The LVD monitors are available in the Active, Sleep, Deep Sleep and Power Down modes. Except VDD_VSYS LVD, the other LVD monitors are disabled in Deep Power Down modes. See Module operation in low power modes for the available modes.

The reset sources triggered by individual LVD monitors are combined into a single LVD reset. The LVD reset condition asserts a POR reset in all power domains with only the POR/LVD/HVD detection logic remaining operational. An LVD reset sets the appropriate bits in the CMC registers so that software can determine that an LVD reset has occurred.

24.2.3 High-voltage detect (HVD)

The HVD monitors are used to ensure that the voltage supplies to the device are within operation range for the logic and memories using those power supplies. The HVD monitors are disabled by default, but when an HVD circuit is enabled, the HVD monitor generates a reset when the voltage rises above the V_{HVD} threshold. This ensures that the device halts operation and remains in reset while the monitored voltage is not within the desired range.

The HVD monitors are available in the Active, Sleep, Deep Sleep and Power Down modes. Except VDD_VSYS HVD, the other HVD monitors are disabled in Deep Power Down modes. See Module operation in low power modes for the available modes.

The reset source triggered by individual HVD monitors are combined into a single HVD reset. The HVD reset condition asserts a reset to all power domains with only the POR/LVD/HVD detection logic remaining operational. An HVD reset sets the appropriate bits in the CMC registers so that software can determine that an HVD reset has occurred.

24.3 External reset sources

External resets are provided so that the device can be reset or awakened from very low power states. This allows the device to start at the correct time from a known state.

24.3.1 External pin reset (RESET_b)

The RESET_b pin is a bi-directional open-drain pin with internal pull-up resistor. The RESET_b pin is functional in two modes:

- During active and low power modes, the RESET_b pin can be asserted externally to force the device into Pin reset condition.
- During reset, the RESET_b drives low until the device has completed hardware initialization, at which point the RESET_b pin is released. If the RESET_b pin is asserted externally, then the device will remain in reset until the RESET_b input is pulled high.

The RESET_b pin implements a digital filter that software can configure to filter out glitches on the RESET_b pin that are less than 1-32 CMC clock cycles. The filter is bypassed in low power modes if the CMC clock is disabled. The CMC reset filter clock source is the SLOW_CLK.

The RESET_b is multiplexed on PTD0 pin on this device. RESET_b is the default function of this GPIO pin. Asserting RESET_b wakes the device from any mode.

The PIN reset condition triggers the CMC to assert a WARM reset to all power domains. A PIN reset sets the appropriate fields in the CMC registers so that software can determine that a PIN reset has occurred.

24.3.2 Wake-up unit (WAKEUP)

The Wake-Up Unit (WUU) provides for a number of external pins and on-chip peripherals to wake the device from Power Down or Deep Power Down modes. See the WUU chapter for a list of external and internal reset sources connected to the WUU.

On a wakeup from Power Down or Deep Power Down modes, the power management logic triggers a WAKEUP reset in the power domains that had been powered off.

The WAKEUP reset condition triggers the CMC to assert a POR reset to the power domains that were powered down. The system voltage domain is not affected by a WAKEUP reset. A WAKEUP reset sets the appropriate fields in the CMC registers so that software can determine that a WAKEUP reset has occurred.

24.4 Internal reset sources

Internal resets are provided so that the device can be reset when certain erroneous conditions are detected. This allows the device to recover from the erroneous conditions and restart from a known state.

24.4.1 Watchdog (WDOG) timer

The WDOG reset is generated by one of the WDOG modules.

The WDOG monitors the operation of the system by receiving periodic communication from software, known as servicing or refreshing the WDOG. If this periodic servicing does not occur, then the Watchdog triggers a WDOG reset condition.

The WDOG reset condition triggers the CMC to assert a WARM reset to all power domains. A WDOG reset sets the appropriate fields in the CMC registers so that software can determine that a WDOG reset has occurred.

24.4.2 Software (SW)

During execution, if the software detects erroneous operation, the software can initiate a reset to restart the device from a known state.

The SW reset condition triggers the CMC to assert a WARM reset to all power domains. A SW reset sets the appropriate fields in the CMC registers so that software can determine that a SW reset has occurred.

24.4.3 System Clock Generation (SCG)

The SCG module contains a clock monitor, which, if enabled, detects a loss of the external clock. A loss of external clock triggers the SCG reset condition.

The SCG reset condition will assert a WARM reset in all power domains. The CMC_SRS[SCG] and CMC_SRS[WARM] status bits are set on a SCG reset condition.

The SCG clock monitor must be enabled for a loss of clock to be detected. For details on enabling the clock monitor, see the SCG chapter. To prevent unexpected reset events, clock monitors must be disabled before entering any low-power modes that the external clocks are not present.

24.4.4 Reset Acknowledge timeout (RSTACK)

The reset state machine includes a timeout counter that is triggered by the reset state machine not progressing within 8192 cycles of the FRO-6MHz/DIV6 clock, this will trigger the RSTACK reset condition.

The RSTACK reset condition triggers the CMC to assert a WARM reset to all power domains. An RSTACK reset sets the appropriate fields in the CMC registers so that software can determine that an RSTACK reset has occurred.

24.4.5 Low Power Acknowledge timeout (LPACK)

The low power entry state machine includes a timeout counter that is triggered if a module does not acknowledge entry into a low power mode after 8192 cycles of the FRO-6MHz/DIV6 clock, this will trigger the LPACK reset condition.

The LPACK reset condition triggers the CMC to assert a WARM reset to all power domains. An LPACK reset sets the appropriate fields in the CMC registers so that software can determine that an LPACK reset has occurred.

24.4.6 Lock up (LOCKUP)

The Cortex-M33 core will enter the lockup state as a result of certain illegal operations, this will trigger the LOCKUP reset condition.

The LOCKUP reset condition triggers the CMC to assert a WARM reset to all power domains. A LOCKUP reset sets the appropriate fields in the CMC registers so that software can determine that a LOCKUP reset has occurred.

24.4.7 Debug Port (DAP)

Any debug access port that can initiate a reset request from a connected debugger triggers the DAP reset condition.

The DAP reset condition triggers the CMC to assert a WARM reset to all power domains. A DAP reset sets the appropriate fields in the CMC registers so that software can determine that a DAP reset has occurred.

24.4.8 EdgeLock secure enclave (ELE)

Internal EdgeLock enclave resets are provided so that the security module can be reset when security violation or certain erroneous conditions are detected. This allows security module to recover from the erroneous conditions. The reset sources in the security module available in this device are as follows.

- Security violation
- Security Watchdog timer
- Security software
- Security CPU lockup
- Security debug
- eFuse loading error

Whenever the security module detects a security violation or an erroneous condition, this will trigger the EdgeLock enclave reset condition.

The EdgeLock enclave reset condition will trigger the Core Mode Controller (CMC) to assert a WARM reset to all power domains. An EdgeLock enclave reset will set the appropriate bits in the CMC registers so that software can determine that an EdgeLock enclave reset has occurred.

24.4.9 Tamper detect (TAMPER)

Whenever a tamper detection module detects a tamper event, this will trigger the TAMPER reset condition.

The TAMPER reset condition triggers the CMC to assert a WARM reset to all power domains. A TAMPER reset sets the appropriate fields in the CMC registers so that software can determine that a TAMPER reset has occurred.

24.4.10 Radio reset request

Radio reset request could happen when NBU CM3 meet timeout or NBU system hang. It should be configured in advance to decide reset Radio subsystem or whole chip.

How to handle reset is under discussion. Will share more when available

24.5 Reset type

This section describes the different resets that can be generated on this device and their respective reset sequences. Reset conditions on this device are categorized into two types, POR and WARM resets.

24.5.1 POR reset

Each power domain generates a POR that is used to reset the entire device, including debug logic and mode control logic. The POR for each power domain can assert as a result of the following events.

- Initial power-up of MCU
- Low or High Voltage Detect
- Power domain wakeup from Power Down or Deep Power Down (varies per domain)

A POR does not guarantee the contents of on-chip SRAM, except for a Power Down wakeup and then only for the SRAM that is configured to retain state.

24.5.1.1 POR reset sequence

The following steps are performed as part of the POR (or LVD/HVD) sequence:

1. RESET_b pin is driven low and internal reset signals assert.
2. POR and LVD signals negate and clocks are enabled in their default configuration.
3. Internal reset remains asserted for 15 clock cycles.
4. Internal reset to Flash and Fuse controller negate and their initialization sequences commence.
5. Initialization sequences for Flash and Fuse complete.
6. Internal trim registers are loaded and RESET_b pin is tri-stated.
7. Reset state machine waits for RESET_b pin input to pull high or drive high externally.
8. Internal reset signals negate.
9. Core exits reset and fetches the initial program counter and stack pointer from Boot ROM.

When the device wakes up from Deep Power Down or Power Down mode, the reset sequence is slightly different from the POR reset sequence. They are described in the subsequent sections.

24.5.1.2 Deep Power Down (DPD) reset sequence

The following steps are performed as part of the Deep Power Down wakeup sequence:

1. Internal reset signals assert.
2. Wakeup signal negates and clocks are enabled in their default configuration.
3. Internal reset remains asserted for 15 clock cycles.
4. Internal reset to Flash and Fuse controller negate and their initialization sequences commence.
5. Initialization sequences for Flash and Fuse complete.
6. Internal trim registers are loaded.
7. Internal reset signals negate.
8. Core exits reset and fetches the initial program counter and stack pointer from Boot ROM.

The RESET_b pin is not driven low by a Deep Power Down reset sequence unless the RESET_b pin was used as the wake up source.

A wakeup from Deep Power Down via the RESET_b pin will follow the warm reset sequence.

24.5.1.3 Power Down (PD) reset sequence

The following steps are performed as part of the Power Down wakeup sequence:

1. Internal reset signals assert.
2. Clocks are enabled in their default configuration.
3. Internal reset remains asserted for 15 clock cycles.
4. Internal trim registers are loaded.
5. Internal reset signals negate.
6. Core exits reset and fetches the initial program counter and stack pointer from Boot ROM.

The RESET_b pin is not driven low by a Power Down reset sequence unless the RESET_b pin was used as the wake up source.

A wakeup from Power Down via the RESET_b pin will follow the warm reset sequence.

24.5.2 Warm reset

A WARM reset re-initializes the device to a known state once it has already been running for a period of time.

A WARM reset resets most of the logic in each power domain. Warm resets are divided into fatal reset sources and non-fatal reset sources. See the CMC chapter for which reset sources are fatal resets.

Non-fatal reset sources can be configured to generate an interrupt instead of the WARM reset. If software can clear the non-fatal reset source (including status flag) within 8192 cycles of the FRO-6MHz/DIV6 clock then the WARM reset is averted. Non-fatal resets retain the contents of on-chip SRAM.

Fatal reset sources cannot be configured to generate an interrupt, and do not guarantee the contents of on-chip SRAM.

The following steps are performed as part of the warm reset sequence:

1. RESET_b pin drives low and internal reset signals assert.
2. Clocks are enabled in their default configuration.
3. Internal reset remains asserted for 15 clock cycles.
4. Internal reset to Flash and Fuse controller negate and their initialization sequences commence.
5. Initialization sequences for Flash and Fuse complete.
6. Internal trim registers are loaded and RESET_b pin is tri-stated.
7. Reset state machine waits for RESET_b pin input to pull high or drive high externally.
8. Internal reset signals negate.

Core exits reset and fetches the initial program counter and stack pointer from Boot ROM.

Chapter 25
Clocking

25.1 Introduction

The heart of the clocking architecture is the System Clock Generator (SCG) module. The SCG controls multiple clock sources that are distributed to the main CPU platform, the memory modules and the peripheral modules. The peripheral modules have an *interface* clock that is used by the CPU/DMA to interface with each module's registers. When applicable, modules may also have *functional* clocks to provide the main timing function of the module's application. For example, functional clocks can be used to source the baud rate for a serial communications peripheral or to clock the counter of a timer peripheral.

Each module's interface clock comes directly from the SCG. The SCG provides functional clock options via the Module Reset and Clock Control (MRCC) module. Software configures the MRCC module to select a module's functional clock. The MRCC module also provides functional clock dividers as needed.

The following diagram shows the clock sources and clock trees.

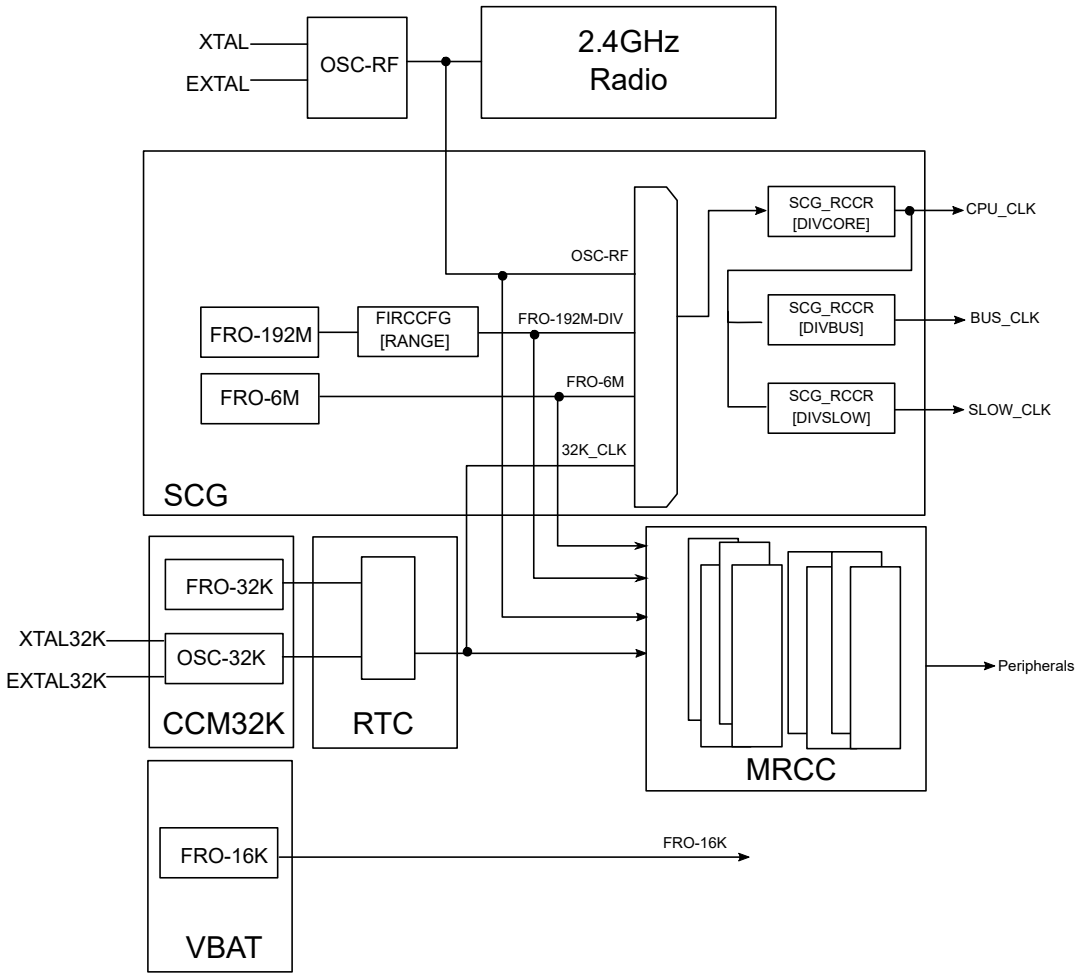


Figure 65. Device clock sources and trees

The CPU_CLK can be sourced directly from the following:

- FRO-192M-DIV (96 MHz) – general default
- FRO-6M (~6 MHz) – optional default

- 32K_CLK
- OSC-RF

Each of the above clock sources can be divided via the SCG_RCCR[DIVCORE] divider to avoid clocking above the max CPU frequency or to reduce the CPU frequency for lower power consumption.

Other major clock trees are fed from the CPU_CLK for the on-chip peripheral and memory interfaces that are required to be either equal to the CPU_CLK frequency or an integer divide or multiple of the CPU_CLK.

The BUS_CLK and the SLOW_CLK are both sourced from the CPU_CLK. Each of this clock tree has its own independent clock divider to provide frequencies that are an integer divide of the CPU_CLK.

Modules that are required to function at a faster frequency than the CPU_CLK obtain a functional clock via the MRCC.

25.2 Clock definitions

The following table describes the clocks in the device.

Table 190. Clock definitions

Clock name	Description
CPU_CLK[1]	The main CPU clock. The CPU_CLK can be provided by one of the clock sources within the SCG, and also from the clock sources not specifically configured by the SCG. The selected clock source is fed through the DIVCORE divider stage prior to clocking the CPU. The CPU_CLK can be gated off, while other clock trees are allowed to function. The maximum frequency of the CPU_CLK is 96 MHz.
BUS_CLK[1]	The BUS_CLK provides synchronization between the CPU and the peripherals. The BUS_CLK is an integer divide or equal to the CPU_CLK. It is fed via the SCG_RCCR[DIVBUS] divider stage. The BUS_CLK can be gated off, while other clock trees are allowed to function. The maximum frequency of the BUS_CLK is 96 MHz.
SLOW_CLK	The SLOW_CLK provides synchronization between the CPU and peripherals that operate at lower frequencies. The SLOW_CLK is an integer divide of the CPU_CLK. The SLOW_CLK is fed from the CPU_CLK via the SCG_RCCR[DIVSLOW] divider stage. The SLOW_CLK can be gated off, while other clock trees are allowed to function. The maximum frequency of the BUS_CLK is 24 MHz.
OSC-32K	Oscillator with external crystal resonator of 32 kHz
OSC-RF	External crystal oscillator (26 MHz or 32 MHz) used by the Radio Frequency modules
FRO-32K	32 kHz Free Running Oscillator clock reference
32K_CLK	32 kHz clock source from either OSC-32K or FRO-32K
FRO-6M	6 MHz Free Running Oscillator clock source
FRO-192M-DIV	192 MHz Free Running Oscillator clock source with dividers that can be used for high speed operation

1. The maximum frequency of this clock is different depending on VDD_CORE, see data sheet for details.

25.3 MRCC clock outputs

The SCG provides output clock trees for peripheral modules that require functional clocks. Functional clocks are used as the fundamental timing for applications of a module, providing asynchronous clocking from the CPU main clock. This allows modules to operate at different rates compared to the CPU platform. Examples of how functional clocks are used include:

- Clocking the reference counter of a timer module
- Sourcing the baud rate for the external communication of a serial communications peripheral

Each module that needs functional clocking has its own MRCC register that provides a clock select field, a clock disable, a divider of the selected clock source, and a peripheral reset bit that provides independent resetting of the peripheral.

25.4 Clock gating

Clock gating of modules helps consume power only for modules needed for their end application. The clock to each module can be gated on or off via a programmable register.

Each peripheral has independent clock gating for both the peripheral interface clock and the peripheral functional clock. This clock gating is controlled via the peripheral's MRCC register. Clearing MRCC[CC] disables both the interface clock and functional clock of a peripheral.

The CPU_CLK, BUS_CLK, and SLOW_CLK are always enabled in the Active modes. If these clocks are not used in low-power modes, they are disabled via the CMC and SPC modules.

25.5 Module/Peripheral clocking

The following tables detail the various clocks associated with a specific module/peripheral.

Table 191. Module/Peripheral clocking options

Peripheral/Module	Interface clock	Module/Peripheral condition out of POR/ reset condition [1]	External/Alternative input clocks	External output clocks
CPU and system modules				
EWM0	SLOW_CLK	Disabled	—	—
WDOG0	SLOW_CLK	Enabled[2]	—	—
WDOG1	SLOW_CLK	Disabled	—	—
WUU0	SLOW_CLK	Disabled	—	—
M33 CPU	CPU_CLK	Enabled	—	—
eDMAx	CPU_CLK	Disabled	—	—
TRDC	CPU_CLK	Disabled[3]	—	—
SCG0	SLOW_CLK	Enabled	—	Outputs from SCG: CPU_CLK, BUS_CLK, SLOW_CLK, 32K_CLK, FRO-6M, FRO-192M-DIV, OSC-RF
MRCC	SLOW_CLK	Disabled	—	—
Debug (SWD)	CPU_CLK	Disabled/Enabled if SWD signals applied	SWD_CLK	—
TRGMUX0	SLOW_CLK	Disabled	—	—
CMC0	SLOW_CLK	—	—	—
SFA0	BUS_CLK	Disabled	—	—
SPM0	CPU_CLK	Disabled	—	—

Table continues on the next page...

Table 191. Module/Peripheral clocking options (continued)

Peripheral/Module	Interface clock	Module/Peripheral condition out of POR/ reset condition [1]	External/Alternative input clocks	External output clocks
Memory modules				
ROM	CPU_CLK	Enabled	—	—
TCRAM	CPU_CLK	Enabled	—	—
Cache RAM	CPU_CLK	Enabled	—	—
Flash	CPU_CLK[4]	Enabled	—	—
FMU0	CPU_CLK	Enabled	—	—
Security modules				
CRC0	BUS_CLK	Disabled	—	—
EdgeLock enclave	CPU_CLK	disabled	—	—
Communication modules				
FlexIO0	BUS_CLK	Disabled	—	—
LPSPI0	SLOW_CLK	Disabled	LPSPI0_CLK	LPSPI0_CLK
LPSPI1	BUS_CLK	Disabled	LPSPI1_CLK	LPSPI1_CLK
LPUART0	SLOW_CLK	Disabled	—	—
LPUART1	BUS_CLK	Disabled	—	—
SEMA42	BUS_CLK	Disabled	—	—
CAN0	BUS_CLK	Disabled	—	—
I3C0	BUS_CLK	Disabled	I3C0_SCL	I3C0_SCL
LPI2C0	SLOW_CLK	Disabled	LPI2C0_SCL	LPI2C0_SCL
LPI2C1	BUS_CLK	Disabled	LPI2C1_SCL	LPI2C1_SCL
Radio modules				
DSB	CPU_CLK	Disabled	—	—
RFMC	SLOW_CLK	Disabled	—	—
RF2.4G	CPU_CLK	Disabled	—	—
RF-NBU	CPU_CLK	Disabled	—	—
Timer modules				
LPTMR0	SLOW_CLK	Disabled	LPTMR0_ALT0,1,2,3	LPTMR0_ALT0,1,2,3
LPTMR1	SLOW_CLK	Disabled	LPTMR1_ALT0,1,2,3	LPTMR1_ALT0,1,2,3
LPIT0	BUS_CLK	Disabled	—	—
TPM0	SLOW_CLK	Disabled	TPM0_CLKIN	—
TPM1	BUS_CLK	Disabled	TPM1_CLKIN	—

Table continues on the next page...

Table 191. Module/Peripheral clocking options (continued)

Peripheral/Module	Interface clock	Module/Peripheral condition out of POR/ reset condition [1]	External/Alternative input clocks	External output clocks
TSTMR	SLOW_CLK	Enabled	—	—
HMI modules				
GPIOA/B/C	CPU_CLK	Disabled	—	—
GPIOD	SLOW_CLK	Enabled	—	—
PORTA/B/C	SLOW_CLK	Disabled	—	—
PORTD	SLOW_CLK	Enabled	—	—
Analog modules				
ADC-GP0	BUS_CLK	Disabled	—	—
CMP-GP0	SLOW_CLK	Disabled	—	—
CMP-GP1	SLOW_CLK	Disabled	—	—
VREF	SLOW_CLK	Disabled	—	—
SPC0	SLOW_CLK	Enabled	—	—
Battery domain				
RTC	SLOW_CLK	Enabled	—	RTC_CLKOUT

1. This is the expected status of the module out of reset.
2. Safety to have a WDOG enabled on reset, app code can disable, but provides a mechanism to protect against runaway after reset.
3. BootROM will enable if TZ-M is required, app code will enable if other policy attributes are required.
4. The flash read cycles could be multiple of CPU clock cycle which is configured by FMU_FCTRL[RWSC]

Table 192. Modules/Peripherals with MRCC functional clocking

Peripheral/ Module	Interface clock	Default status	Reset control	Clock gate	Functional alt0 clock	Functional alt1 clock	Functional alt2 clock	Functional alt3 clock	Functional alt4 clock	Functional alt5 clock
CPU and system modules										
EWM0	SLOW_CLK	Disabled	Y	Y	—	—	—	—	—	—
WDOG0	SLOW_CLK	Enabled	—	Y	—	—	—	—	—	—
WDOG1	SLOW_CLK	Disabled	—	Y	—	—	—	—	—	—
WUU0	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
CM33 CPU	CPU_CLK	N/A	—	—	—	—	—	—	—	—
eDMAx	CPU_CLK	Disabled	Y	Y	—	—	—	—	—	—
TRDC	CPU_CLK	N/A	—	—	—	—	—	—	—	—
SCG0	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
MRCC	SLOW_CLK	N/A	—	—	—	—	—	—	—	—

Table continues on the next page...

Table 192. Modules/Peripherals with MRCC functional clocking (continued)

Peripheral/ Module	Interface clock	Default status	Reset control	Clock gate	Functional alt0 clock	Functional alt1 clock	Functional alt2 clock	Functional alt3 clock	Functional alt4 clock	Functional alt5 clock
Debug (SWD)	CPU_CLK	N/A	—	—	—	—	—	—	—	—
TRGMUX0	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
TCU	Test Clock	N/A	—	—	—	—	—	—	—	—
CMC0	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
SFA0 (was DTX)	BUS_CLK	Disabled	Y	Y	—	—	—	—	—	—
SEMA	BUS_CLK	Disabled	Y	Y	—	—	—	—	—	—
SPM0	CPU_CLK	Disabled	—	Y	—	—	—	—	—	—
Memory modules										
ROM	CPU_CLK	Enabled	—	Y	—	—	—	—	—	—
TCRAM	CPU_CLK	Enabled	—	Y	—	—	—	—	—	—
Flash	CPU_CLK[1]	Enabled	—	Y	—	—	—	—	—	—
FMU0	CPU_CLK	N/A	—	—	—	—	—	—	—	—
Security modules										
CRC0	BUS_CLK	Disabled	Y	Y	—	—	—	—	—	—
EdgeLock enclave	CPU_CLK	Disabled	Y	Y	—	—	—	—	—	—
Communication modules										
CAN0	BUS_CLK	Disabled	Y	Y	—	—	—	FRO-192M-DIV	OSC-RF	—
FlexIO0	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
LPSPI0	SLOW_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
LPSPI1	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
LPUART0	SLOW_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	32K_CLK
LPUART1	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	32K_CLK
I3C0	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
LPI2C0	SLOW_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—

Table continues on the next page...

Table 192. Modules/Peripherals with MRCC functional clocking (continued)

Peripheral/ Module	Interface clock	Default status	Reset control	Clock gate	Functional alt0 clock	Functional alt1 clock	Functional alt2 clock	Functional alt3 clock	Functional alt4 clock	Functional alt5 clock
LPI2C1	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
Radio modules										
DSB	CPU_CLK	Disabled	Y	Y	—	—	—	—	—	—
RFMC	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
RF2.4G	CPU_CLK	N/A	—	—	—	—	—	—	—	—
RF-NBU	CPU_CLK	N/A	—	—	—	—	—	—	—	—
Timers modules										
LPTMR0	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
LPTMR1	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
LPIT0	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
TPM0	SLOW_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	32K_CLK
TPM1	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	32K_CLK
TSTMR	SLOW_CLK	Enabled	—	Y	—	—	—	—	—	—
Battery domain										
RTC	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
ips2dgo_vsys	SLOW_CLK	—	—	—	—	—	—	—	—	—
HMI modules										
GPIOA/B/C	CPU_CLK	Disabled	Y	Y	—	—	—	—	—	—
GPIOD	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
PORTA/B/C	SLOW_CLK	Disabled	Y	Y	—	—	—	—	—	—
PORTD	SLOW_CLK	N/A	—	—	—	—	—	—	—	—
Analog modules										
ADC0	BUS_CLK	Disabled	Y	Y	—	—	FRO-6M	FRO-192M-DIV	OSC-RF	—
LPCMP0	SLOW_CLK	Disabled	Y	Y	—	—	—	—	—	—
LPCMP1	SLOW_CLK	Disabled	Y	Y	—	—	—	—	—	—
VREF	SLOW_CLK	Disabled	Y	Y	—	—	—	—	—	—
SPC0	SLOW_CLK	N/A	—	—	—	—	—	—	—	—

1. The flash read cycles could be multiple of CPU clock cycle which is configured by FMU_FCTRL[RWSC]

Chapter 26
System Clock Generator (SCG)

26.1 Chip-specific SCG information

Table 193. Reference links to related information

Topic	Related module	Reference
Full description	SCG	SCG
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

26.1.1 Module instances

This device has one instance of the SCG module, SCG0.

26.1.2 Clock naming

Each device that has an SCG module generally has a few differences from SCG modules in different devices. This section describes those differences.

Signal name differences

Table 194. Device signal names = SCG signal names

Device signal		SCG chapter signal	
Free-Running Oscillator – 192 MHz	FRO_192M	FIRC	Fast internal reference clock
Free-Running Oscillator – 6 MHz	FRO_6M	SIRC	Slow internal reference clock
FRO-32K or OSC-32K is selected to become the 32K_CLK (See note below)	32K_CLK	ROSC	Real-time clock oscillator clock
OSC-RF	OSC_RF	RFOSC, SOSC	External RF clock, system oscillator clock
Not available		LPFLL	Low power FLL clock

NOTE

Separate Power Domain: The FRO-32K and the OSC-32K clocks are generated in a separate module CCM32K (32 kHz Clock Control Module) within a different power domain. The different power domain can be powered independently, allowing the RTC module (also in this separate power domain) to be clocked by one of these clock sources (FRO-32K or OSC-32K). The input to the SCG from this separate power domain is called 32K_CLK. The OSC-RF is generated in the RF module within a different power domain. The input to the SCG from this power domain is called OSC-RF.

26.1.3 SCG clock section CLKOUTCNFG[CLKOUTSEL]

The SCG can select the system clock from the following source by CLKOUTCNFG[CLKOUTSEL]:

- 0001b - System OSC (SOSC_CLK)

- 0010b - Slow IRC (SIRC_CLK)
- 0011b - Fast IRC (FIRC_CLK)
- 0100b - RTC OSC (ROSC_CLK)

NOTE

The SCG slow clock cannot be selected as SCG system clock.

26.2 Introduction

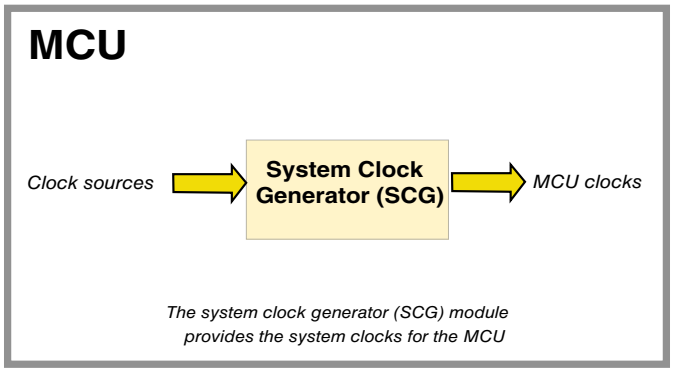


Figure 66. System Clock Generator (SCG) in MCU

The system clock generator (SCG) module provides the system clocks for the MCU. The SCG takes clock inputs from a variety of sources and generates the types of clocks that the MCU requires.

- Available clock source inputs for the SCG include:
  - System oscillator clock (SOSC)
  - Slow internal reference clock (SIRC)
  - Fast internal reference clock (FIRC)
  - Real-Time Oscillator (ROSC)

Clock dividers are available for:

- core clock (DIVCORE)
- peripheral interface clock (DIVSLOW)
- bus clock (DIVBUS)

26.2.1 Features

Key features of the SCG module are:

- 2 Internal reference clock (IRC) generators:
  - High frequency free-running oscillator (FRO_192M) with trim capability
  - Low frequency running oscillator (FRO_6M) with trim capability
  - Either of the free-running oscillators (FRO_192M, FRO_6M) can be selected as the clock source for the system clocks

— The external clock source (RFOSC) can also be used as a source clock for the systems clocks.

- System Crystal Oscillator:
  — Can be selected as the clock source for the system clocks
- Real Time Oscillator Clock (ROSC):
  — Can be selected as the clock source for the device. See the chip specific sections to ascertain which signal the RTC input maps to.
- Clock monitor with reset and interrupt request capability for SOSC, ROSC clocks

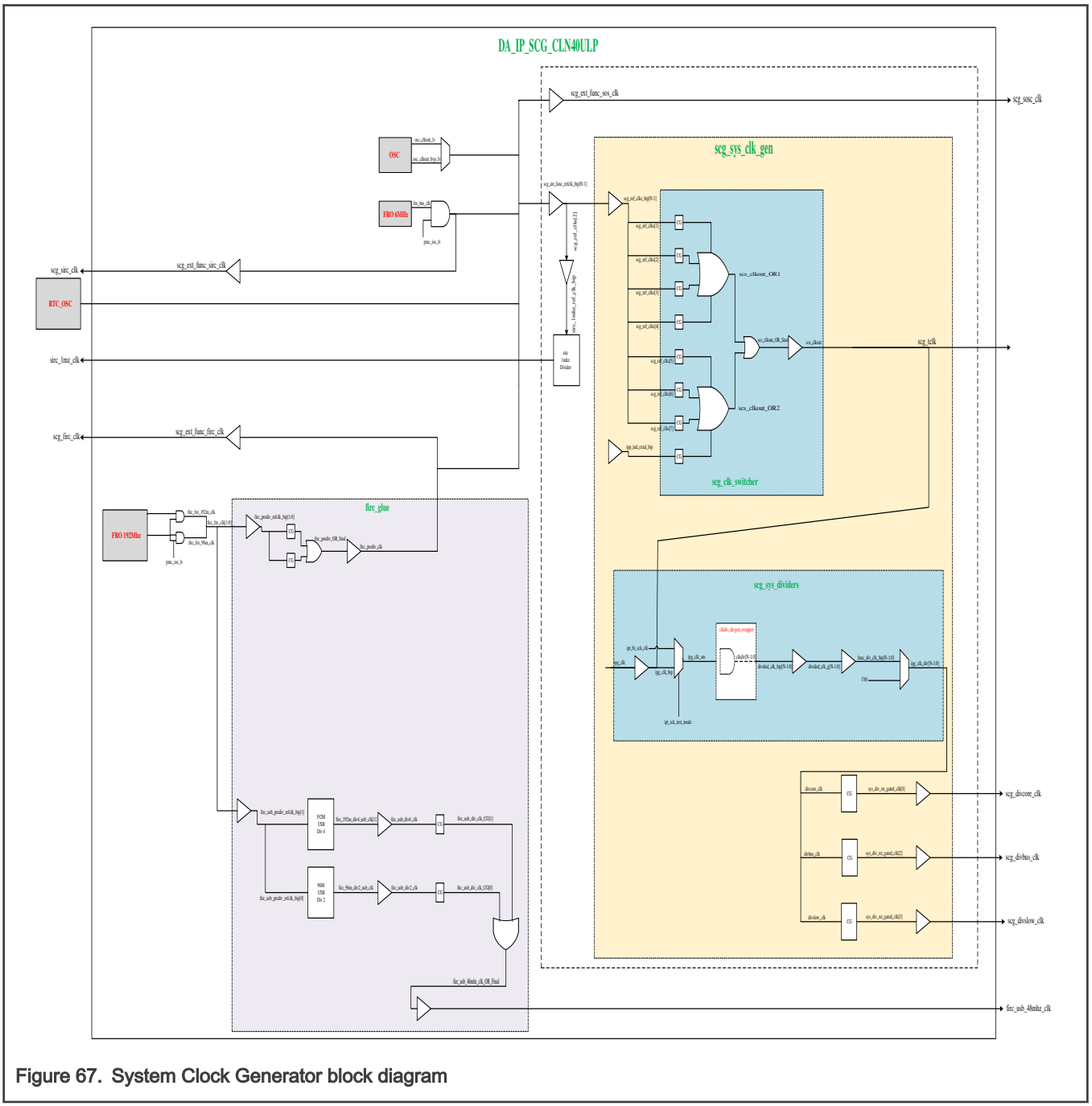


Figure 67. System Clock Generator block diagram

NOTE

To identify the oscillators used in your specific device, see the chip-specific SCG information or clocking chapters.

26.3 Functional description

26.3.1 SCG Clock Mode Transitions

The following figure shows the valid clock mode transitions supported by SCG.

Slow IRC (SIRC) boot mode is not supported on this device.

Figure 68. SCG Valid Mode Transitions

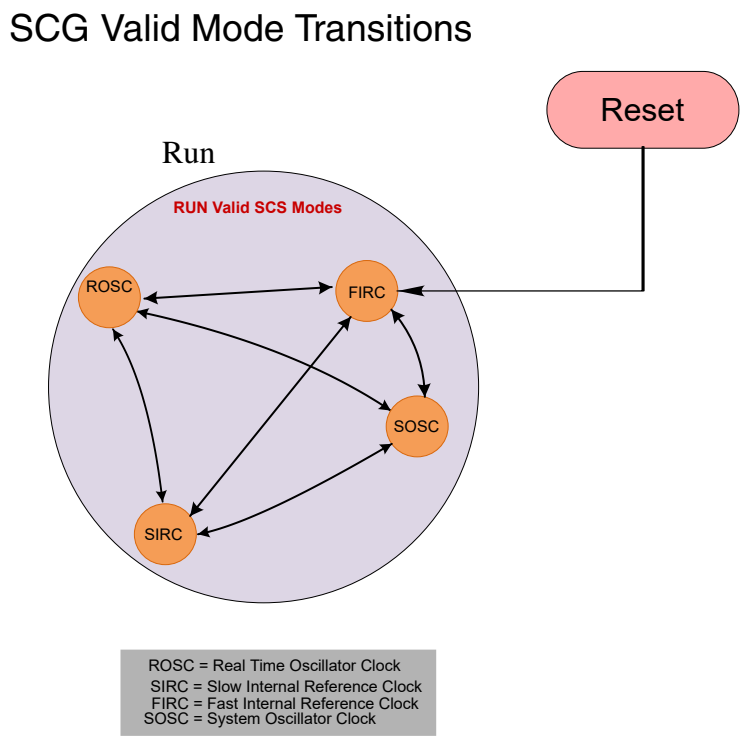


Figure 69. SCG Valid Mode Transitions

The SCG will restrict programming into invalid clock modes and writes to System Clock Source (SCS) bits will be ignored. When a transition between run modes or a transition into wait mode occurs, the SCG completes the switch to the clock mode as defined in the Run Clock Control Register (RCCR). When completed, the system switches to the selected run/wait mode.

The modes of operation listed in the following table are the valid modes for this implementation of the SCG.

Table 195. SCG modes of operation

Mode	Description
System Oscillator Clock (SOSC)	System Oscillator Clock (SOSC) mode is entered when all the following conditions occur: • RUN MODE:

Table continues on the next page...

Table 195. SCG modes of operation (continued)

Mode	Description
	— 0001 is written to RCCR[SCS] • SOSCEN = 1 • SOSCVLD = 1 In SOSC mode, SCSCLKOUT and system clocks are derived from the external System Oscillator Clock (SOSC).
Slow Internal Reference Clock (SIRC)	Slow Internal Reference Clock (SIRC) mode is entered when all the following conditions occur: • RUN MODE: — 0010 is written to RCCR[SCS] • SIRCVLD = 1 In SIRC mode, SCSCLKOUT and system clocks are derived from the slow internal reference clock.
Fast Internal Reference Clock (FIRC)	Fast Internal Reference Clock (FIRC) mode is the default clock mode of operation and is entered when all the following conditions occur: • RUN MODE: — 0011 is written to RCCR[SCS] • FIRCEN = 1 • FIRCVLD = 1 In FIRC mode, SCSCLKOUT and system clocks are derived from the fast internal reference clock. Two frequency range settings are available for FIRC clock as described in the FIRC[RANGE] register definition. Changes to FIRC range settings will be ignored when FIRC clock is enabled.
Real Time Clock Oscillator (RTCOSC)	Real Time Clock Oscillator (RTCOSC) mode is entered when all the following conditions occur: • RUN MODE: — 0100 is written to RCCR[SCS] • ROSCVLD = 1 In RTCOSC mode, SCSCLKOUT and systems clocks are derived from the external RTC Oscillator Clock (RTCOSC).
Stop	Entered whenever the MCU enters a Stop state. The power modes are chip-specific. For power mode assignments, see the chapter that describes how modules are configured and SCG behaviour during Stop recovery. Entering Stop mode, all SCG clock signals are static, including SCG system clocks (DIVCORE, DIVBUS, DIVSLOW). There are exceptions; the following clocks can continue to run and stay enabled in the following cases: SIRC is available in SLEEP mode when all the following conditions become true: • SIRCCSR[SIRCSTEN] = 1 FIRC is available in SLEEP mode when all the following conditions become true: • FIRCCSR[FIRCEN] = 1

Table continues on the next page...

Table 195. SCG modes of operation (continued)

Mode	Description
	• FIRCCSR[FIRCSTEN] = 1 SOSC is available in SLEEP mode when all the below conditions are true. • SOSCCSR[SOSCEN] = 1 • SOSCCSR[SOSCSTEN] = 1

26.4 Memory Map/Register Definition

This section includes the memory map and register definition.

The SCG registers can only be written when in supervisor mode. Write accesses when in user mode will result in a bus transfer error. Read accesses may be performed in both supervisor and user mode.

26.4.1 SCG register descriptions

- For any writeable SCG registers, only 32-bit writes are allowed. 8-bit or 16-bit writes will result in transfer errors.

26.4.1.1 SCG memory map

SCG0 base address: 4001_E000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0100_0000h
4h	Parameter Register (PARAM)	32	R	9800_001Eh
10h	Clock Status Register (CSR)	32	R	0302_0001h
14h	Run Clock Control Register (RCCR)	32	RW	0302_0001h
20h	SCG CLKOUT Configuration Register (CLKOUTCNFG)	32	RW	0000_0000h
100h	System OSC Control Status Register (SOSCCSR)	32	RW	See section
200h	Slow IRC Control Status Register (SIRCCSR)	32	RW	0000_0000h
300h	Fast IRC Control Status Register (FIRCCSR)	32	RW	0300_0001h
308h	Fast IRC Configuration Register (FIRCCFG)	32	RW	0000_0002h
30Ch	Fast IRC Trim Configuration Register (FIRCTCFG)	32	RW	0000_0000h
318h	Fast IRC Status Register (FIRCSTAT)	32	RW	See section
400h	RTC OSC Control Status Register (ROSCCSR)	32	RW	See section

26.4.1.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

Note: Writing to this register will result in a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 VERSION	SCG Version Number

26.4.1.3 Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Note: Writing to this register will result in a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DIVPRES					0										
W																
Reset	1[1]	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CLKPRES							
W																
Reset	0	0	0	0	0	0	0	0	0[2]	0	0	1	1	1	1	0

1. The reset value is controlled by which SCG System Dividers are used by the SoC.
2. The reset value is controlled by which SCG Clock Sources are used by the SoC. Please reference the Reference manual clocking chapter.

Fields

Field	Function
31-27 DIVPRES	Divider Present Indicates which system clock dividers are present in this instance of SCG. 1_xxxxb - System DIVCORE is present. x_xx1xb - System DIVBUS is present. x_xxx1b - System DIVSLOW is present.
26-16 —	Reserved
15-8 —	Reserved
7-0 CLKPRES	Clock Present Indicates which clock sources are present in this instance of SCG. Any bits not defined in this bit field are Reserved and always has the value 0 when read. 0000_0000b-0000_0001b - Reserved xxx1_xxxxb - RTC OSC (ROSC) is present. xxxx_1xxxb - Fast IRC (FIRC) is present. xxxx_x1xxb - Slow IRC (SIRC) is present. xxxx_xx1xb - System OSC (SOSC) is present.

26.4.1.4 Clock Status Register (CSR)

Offset

Register	Offset
CSR	10h

Function

The Clock Status Register (CSR) returns the currently configured system clock source and the system clock dividers for the core (DIVCORE) and peripheral interface clock (DIVSLOW). The SCG_CSR reports the configuration set by one of the clock control registers SCG_RCCR.

Note: Writing to the Clock Status Register (CSR) will result in a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				DIVBUS				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Returns the currently configured clock source generating the system clock. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - RTC OSC (ROSC_CLK) 0101b - Reserved 0110b - Reserved 0111b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 DIVBUS	Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

26.4.1.5 Run Clock Control Register (RCCR)

Offset

Register	Offset
RCCR	14h

Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in Run mode only. This register can only be written using a 32-bit write. Selecting a different clock source when

in RUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in RUN, new system clock divide ratios will not take affect until a new clock source is valid.

NOTE

Switching to new system clock source and system clock dividers must be done after previous RCCR changes have been completed and the Clock Status Register has updated to match the present RCCR setting.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				DIVBUS				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-27 —	Reserved
26-24 SCS	System Clock Source Selects the clock source generating the system clock in Run mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in Run mode requires that clock source to be enabled first and be valid before system clocks are allowed to switch to that clock source. **NOTE** Programming SCS to a different value should only be done after the previous SCS clock switch has finished. 000b - Reserved 001b - System OSC (SOSC_CLK) 010b - Slow IRC (SIRC_CLK) 011b - Fast IRC (FIRC_CLK) 100b - RTC OSC (ROSC_CLK) 101b - Reserved 110b - Reserved 111b - Reserved
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 DIVBUS	Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

26.4.1.6 SCG CLKOUT Configuration Register (CLKOUTCNFG)

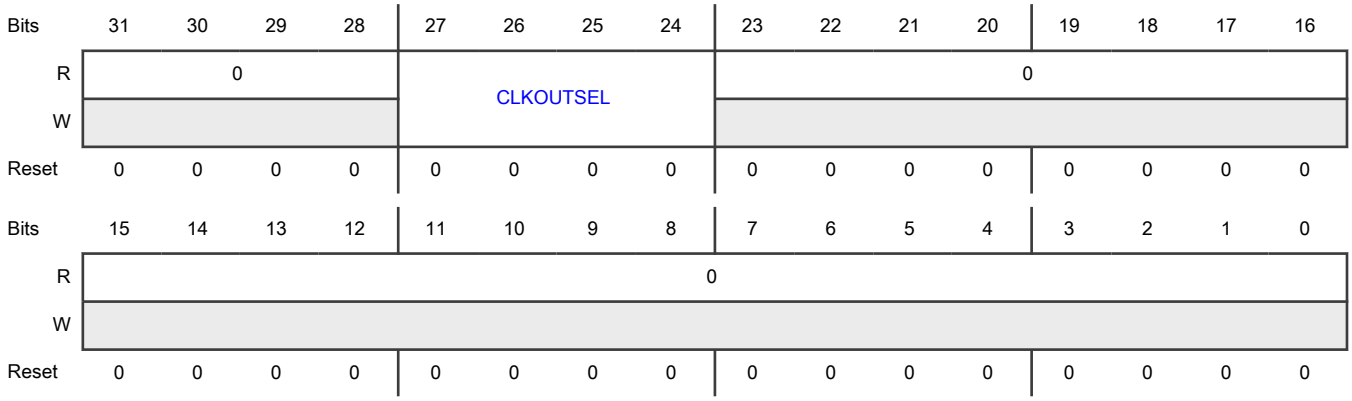
Offset

Register	Offset
CLKOUTCNFG	20h

Function

This register controls which SCG clock source is selected to be ported out to the CLKOUT pin.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 CLKOUTSEL	SCG Clkout Select Selects the SCG system clock. 0000b - SCG SLOW Clock 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - RTC OSC (ROSC_CLK) 0101b - Reserved 0110b - Reserved 0111b - Reserved 1111b - Reserved
23-0 —	Reserved

26.4.1.7 System OSC Control Status Register (SOSCCSR)

Offset

Register	Offset
SOSCCSR	100h

Function

Contains System OSC Clock Error, System OSC Selected, System OSC Valid, Lock Register, System OSC Clock Monitor Reset Enable, System OSC Clock Monitor Enable , System OSC Stop Enable , System OSC Enable.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					SOSCERR	SOSCSEL	SOSCVLD	LK	0					SOSCCMRE	SOSCCM
W						W1C										
Reset	0	0	0	0	0	u[1]	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													0	SOSCSTEN	SOSCEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. This flag is reset on Chip POR only

Fields

Field	Function
31-27 —	Reserved
26 SOSCERR	System OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one. 0b - System OSC Clock Monitor is disabled or has not detected an error 1b - System OSC Clock Monitor is enabled and detected an error
25 SOSCSEL	System OSC Selected 0b - System OSC is not the system clock source 1b - System OSC is the system clock source
24 SOSCVLD	System OSC Valid 0b - System OSC is not enabled or clock is not valid 1b - System OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - This Control Status Register can be written. 1b - This Control Status Register cannot be written.
22-18 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 SOSCCMRE	System OSC Clock Monitor Reset Enable 0b - Clock Monitor generates interrupt when error detected 1b - Clock Monitor generates reset when error detected
16 SOSCCM	System OSC Clock Monitor Enable Enables the clock monitor when SOSCVLD is set. If the clock is disabled in a low power mode then SOSCCM should be cleared to prevent unexpected SOSC loss of clock. 0b - System OSC Clock Monitor is disabled 1b - System OSC Clock Monitor is enabled
15-3 —	Reserved
2 —	Reserved
1 SOSCSTEN	System OSC Stop Enable 0b - System OSC is disabled in any of the sleep modes 1b - System OSC is enabled in SLEEP mode only if SOSCEN=1. SOSCSTEN must be cleared when its power domain is going to enter Deep Sleep or Power Down mode.
0 SOSCEN	System OSC Enable 0b - System OSC is disabled 1b - System OSC is enabled

26.4.1.8 Slow IRC Control Status Register (SIRCCSR)

Offset

Register	Offset
SIRCCSR	200h

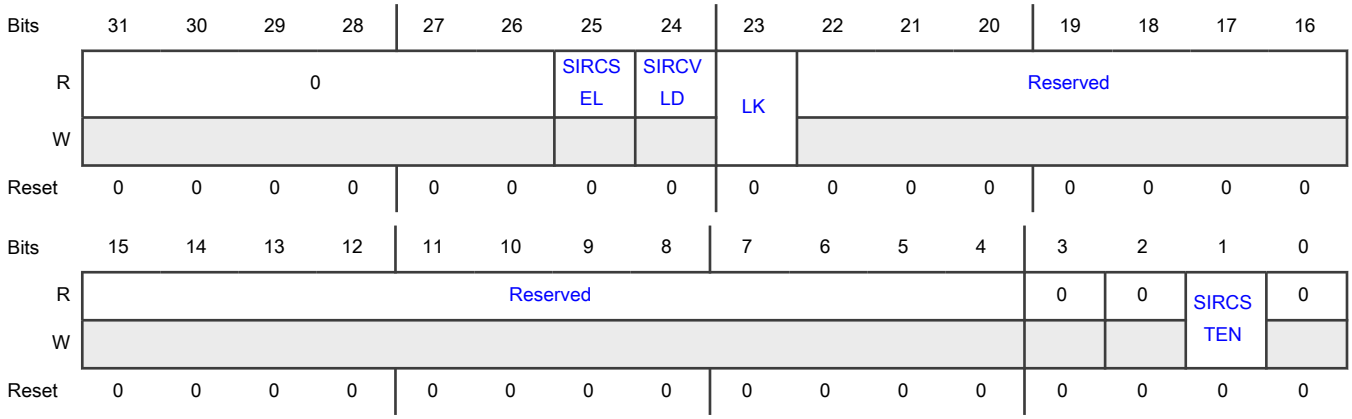
Function

Contains Slow IRC Selected, Slow IRC Valid, Lock Register , Slow IRC Stop Enable , Slow IRC Enable.

NOTE

SIRCVLD is reset only by POR.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 SIRCSEL	Slow IRC Selected 0b - Slow IRC is not the system clock source 1b - Slow IRC is the system clock source
24 SIRCVLD	Slow IRC Valid 0b - Slow IRC is not enabled or clock is not valid 1b - Slow IRC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. The purpose for this bitfield is to prevent runaway code from changing SIRC clock configurations. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-4 —	This field is reserved and is always has the value 0
3 —	This field is reserved and is always has the value 0
2 —	Reserved
1 SIRCSTEN	Slow IRC Stop Enable 0b - Slow IRC is disabled in sleep modes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Slow IRC is enabled in SLEEP mode
0 —	Reserved

26.4.1.9 Fast IRC Control Status Register (FIRCCSR)

Offset

Register	Offset
FIRCCSR	300h

Function
Contains Fast IRC status bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					FIRCERR	FIRCSEL	FIRCVLD	LK	0						
W						W1C										
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		COARSE_...	TRIM_LO...	FIRCTRUP	FIRCTREN	0				0	0	FIRCSTEN	FIRCEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-27 —	Reserved
26 FIRCERR	Fast IRC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - Error not detected with the Fast IRC trimming. 1b - Error detected with the Fast IRC trimming.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 FIRCSEL	Fast IRC Selected status 0b - Fast IRC is not the system clock source 1b - Fast IRC is the system clock source
24 FIRCVLD	Fast IRC Valid status 0b - Fast IRC is not enabled or clock is not valid. 1b - Fast IRC is enabled and output clock is valid. The clock is valid after there is an output clock from the FIRC analog.
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-14 —	Reserved
13-12 —	Reserved
11 COARSE_TRIM_BYPASS	Fast Coarse Auto Trim Bypass 0b - FIRC Coarse Auto Trim NOT Bypassed 1b - FIRC Coarse Auto Trim Bypassed
10 TRIM_LOCK	Fast IRC TRIM LOCK When FIRCTREN and FIRCTRUP are enabled, TRIM_LOCK will indicate when auto trimming is complete and output FIRC frequency has locked to target FIRC range. **NOTE** TRIM_LOCK will automatically get cleared if FIRCTREN and FIRCTRUP are not set. 0b - FIRC auto trim not locked to target frequency range. 1b - FIRC auto trim locked to target frequency range
9 FIRCTRUP	Fast IRC Trim Update 0b - Disable Fast IRC trimming updates 1b - Enable Fast IRC trimming updates
8 FIRCTREN	Fast IRC Trim Enable 0b - Disable trimming Fast IRC to an external clock source 1b - Enable trimming Fast IRC to an external clock source
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

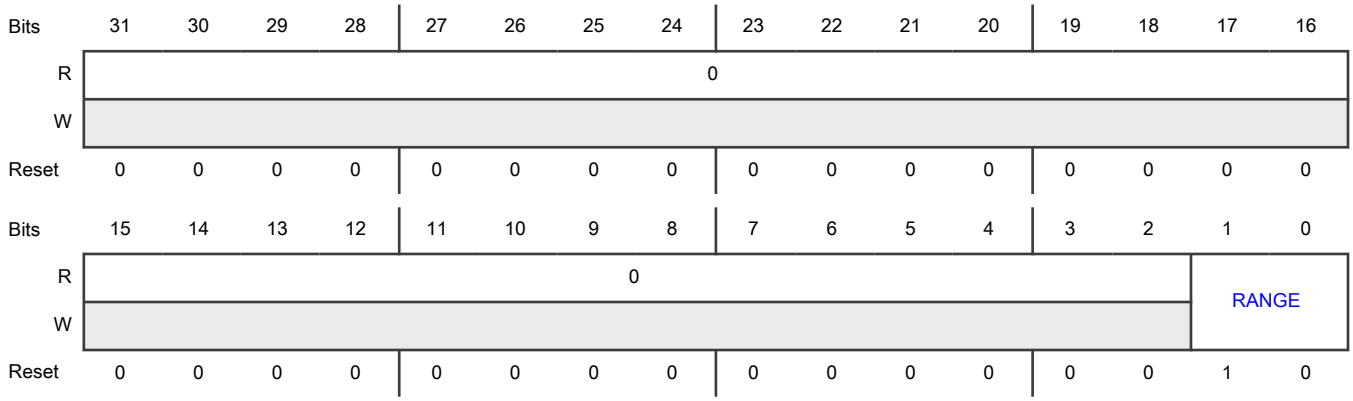
Field	Function
—	
3 —	Reserved
2 —	Reserved
1 FIRCSTEN	Fast IRC Stop Enable 0b - Fast IRC is disabled in sleep modes. 1b - Fast IRC is enabled in SLEEP modes
0 FIRCEN	Fast IRC Enable 0b - Fast IRC is disabled 1b - Fast IRC is enabled

26.4.1.10 Fast IRC Configuration Register (FIRCCFG)

Offset

Register	Offset
FIRCCFG	308h

Diagram



Fields

Field	Function
31-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 RANGE	Frequency Range RANGE can be changed while FIRC clock is enabled 00b - 48 MHz FIRC clock selected. 01b - 64 MHz FIRC clock selected. 10b - 96 MHz FIRC clock selected. 11b - 192 MHz FIRC clock selected.

26.4.1.11 Fast IRC Trim Configuration Register (FIRCTCFG)

Offset

Register	Offset
FIRCTCFG	30Ch

Function

The FIRCTCFG register cannot be changed when Fast IRC tuning is enabled. When the Fast IRC tuning is enabled, writes to this register are ignored, and there is no transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					TRIMDIV										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0						TRIMSRC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-16 TRIMDIV	Fast IRC Trim Predivide Divider of SOSC for FIRC trimming. **NOTE** When selecting SOSC as FIRC trimming source the TRIMDIV register must be set to correct div ratio to generate 1 MHz output reference trimming clock. **NOTE** TRIMDIV is a N-Divider supporting a div ratio of 1 (TRIMDIV=0x000) to a div ratio of 2048 (TRIMDIV=0x7ff).
15-11 —	Reserved
10-8 —	Reserved
7-2 —	Reserved
1-0 TRIMSRC	Trim Source Configures the external clock source to tune the Fast IRC. TRIMSRC must be configured before programming FIRCSTAT register for trim update 00b - Reserved 01b - Reserved 10b - System OSC. This option requires that SOSC be divided using the TRIMDIV field to get a frequency of 1 MHz. 11b - RTC OSC (32.768 kHz)

26.4.1.12 Fast IRC Status Register (FIRCSTAT)

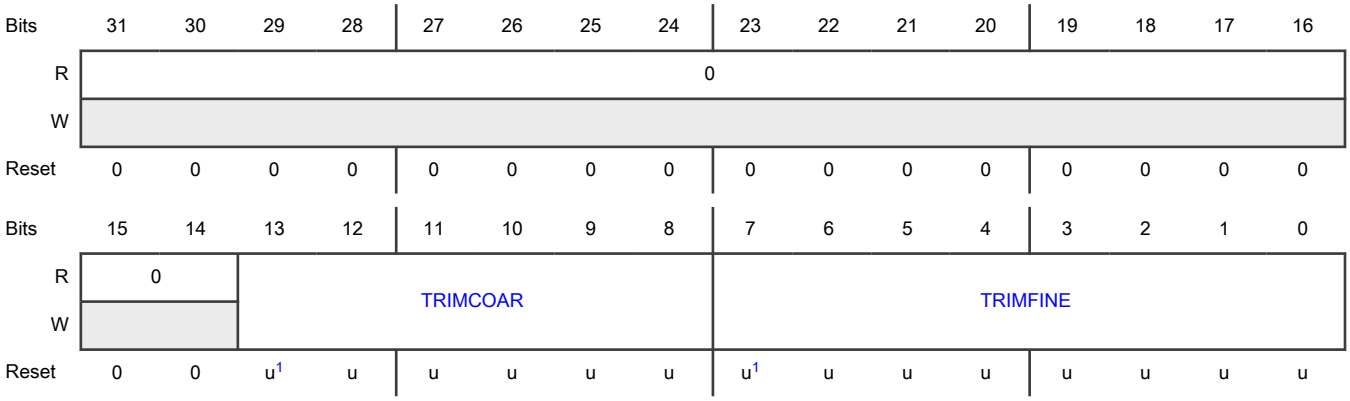
Offset

Register	Offset
FIRCSTAT	318h

Function

This register is loaded from IFR during reset. This register is uploaded with the trim values generated by FIRC auto-trimming which is enabled when FIRC is enabled and FIRCTREN=1 and FIRCTRUP=1. When FIRC auto-trimming is enabled and FIRCTRUP is off (Note: TRIMSRC needs to be programmed to TRIMSRC=10 or TRIMSRC=11), writes to this register are allowed and values written to this register are used to trim FIRC clock.

Diagram



1. Reset values are loaded out of IFR.

Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved
13-8 TRIMCOAR	Trim Coarse TRIMCOAR bits are used to coarsely trim the Fast IRC Clock to within approximately ±TBD% of the target frequency. When FIRC is enabled and auto trimming is enabled (FIRCTREN=1 and FIRCTRUP=1), then TRIMCOAR register gets uploaded with the trimmed coarse value. When FIRCTRUP=0, TRIMCOAR register is writable, to allow user programming of coarse trim values.
7-0 TRIMFINE	Trim Fine See the device's datasheet for the trim values.

26.4.1.13 RTC OSC Control Status Register (ROSCCSR)

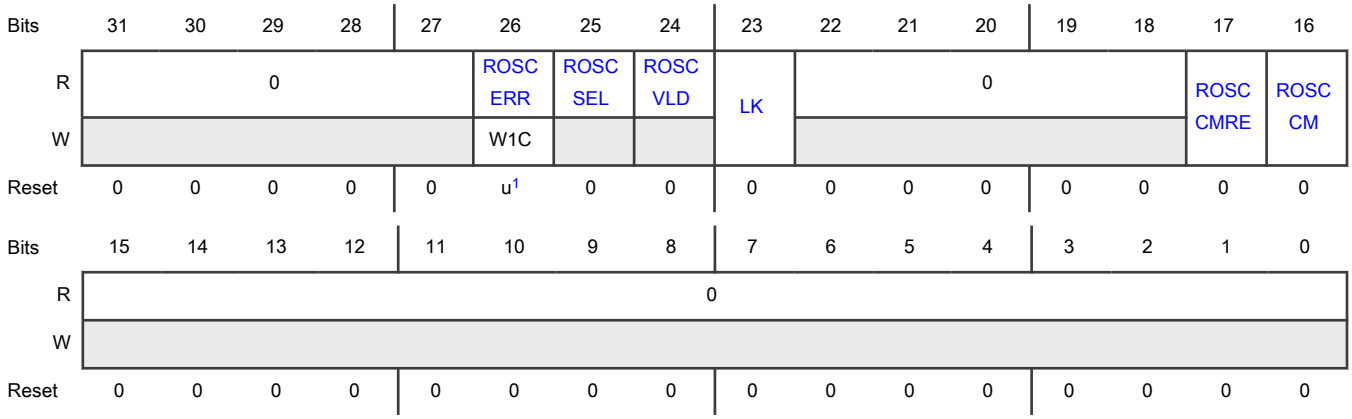
Offset

Register	Offset
ROSCCSR	400h

Function

Contains RTC OSC Clock Error, RTC OSC Selected, RTC OSC Valid, Lock Register, RTC OSC Clock Monitor Reset Enable, RTC OSC Clock Monitor.

Diagram



1. Reset values are loaded out of IFR.

Fields

Field	Function
31-27 —	Reserved
26 ROSCERR	RTC OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - RTC OSC Clock Monitor is disabled or has not detected an error 1b - RTC OSC Clock Monitor is enabled and detected an RTC loss of clock error
25 ROSCSEL	RTC OSC Selected 0b - RTC OSC is not the system clock source 1b - RTC OSC is the system clock source
24 ROSCVLD	RTC OSC Valid 0b - RTC OSC is not enabled or clock is not valid 1b - RTC OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-18 —	Reserved
17 ROSCCMRE	RTC OSC Clock Monitor Reset Enable 0b - Clock Monitor generates interrupt when error detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock Monitor generates reset when error detected
16 ROSCCM	RTC OSC Clock Monitor Enables the clock monitor when ROSCVLD is set. **NOTE** ROSC clock monitor remains enabled in SLEEP mode only if ROSCCM is enabled. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. **NOTE** The reference clock used to monitor the ROSC is the SIRC. SIRC clock must be programmed to be enabled in order to monitor the ROSC. SIRC is automatically disabled in DSLEEP, PDOWN, and DPDOWN power modes and clock monitor of ROSC will be disabled. 0b - RTC OSC Clock Monitor is disabled 1b - RTC OSC Clock Monitor is enabled
15-0 —	Reserved

Chapter 27
Module Reset and Clock Control (MRCC)

27.1 Chip-specific MRCC information

Table 196. Reference links to related information

Topic	Related module	Reference
Full description	MRCC	MRCC
System memory map		System memory map
Clocking		Clock distribution
Reset		Reset

27.1.1 Peripheral functional clock maximum frequency

The maximum frequency of the modules clock are not all the same in middle mode (MD, VDD_CORE = 1.0 V) and standard mode (SD, VDD_CORE = 1.1 V)).

Table 197. Peripheral clock maximum frequency

Peripherals	Max. frequency in MD (MHz)	Max. frequency in SD (MHz)
TPM0	24	24
TPM1	48	96
LPI2C0	48	48
LPI2C1	96	96
I3C0	96	96
LPSPI0	48	48
LPSPI1	96	96
LPUART0	24	24
LPUART1	96	96
FLEXIO0	48	96
CAN0[1]	48	96
LPIT0	48	96
ADC0	48	64

1. Refer to Ordering information for parts have this module.

27.2 Introduction

The Module Reset and Clock Control (MRCC) module provides reset control and clock configuration for on-chip modules. Each module has its own configuration register.

27.3 Features

Per individual module, the MRCC module enables software to:

- Gate all clocks to the module
- Select the source of the functional clock
- Divide the functional clock
- Hold the module in reset

Below is a block diagram of the MRCC module:

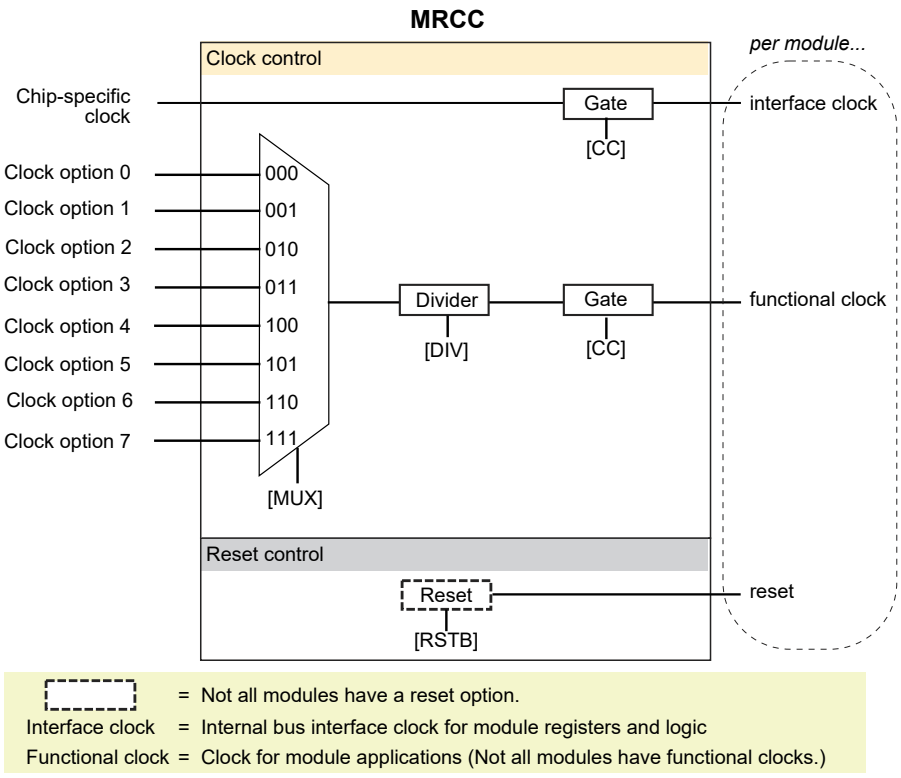


Figure 70. MRCC Block Diagram

27.4 Functional description

The MRCC module provides on-chip modules their own dedicated MRCC registers for clock gating, reset control and configuration options. Generally, each module's MRCC register contains a clock configuration (CC) field for the module's clocks.

NOTE

Before a module can be used, its clocks must be enabled (CC does not equal 00) and it must be released from reset (RSTB = 1). If a module is *not* released from reset (RSTB = 0), an attempt to access a register within that module is terminated with a bus error.

If a module has a functional clock, its MRCC register may provide options for the clock source, selected by programming the mux select (MUX) field. The selected functional clock source can then be divided by programming the divider (DIV) field. The divider factor is given by DIV+1.

NOTE

Before configuring a functional clock, the module's clocks must be disabled (CC = 00).

27.5 Memory map and register definition

Each module has its own dedicated MRCC register. See each module's MRCC register for details.

NOTE

To configure the clocking options available to a given module or to modify an existing configuration, first disable the module's clocks by writing 00 to its CC field.

27.5.1 MRCC register descriptions

27.5.1.1 MRCC memory map

MRCC base address: 4001_C000h

Offset	Register	Width (In bits)	Access	Reset value
4Ch	EWM0 Reset and Clock Control (MRCC_EWM0)	32	RW	8000_0000h
5Ch	SYSPM0 Reset and Clock Control (MRCC_SYSPM0)	32	RW	0000_0000h
68h	WDOG0 Reset and Clock Control (MRCC_WDOG0)	32	RW	0000_0002h
6Ch	WDOG1 Reset and Clock Control (MRCC_WDOG1)	32	RW	0000_0000h
74h	SFA0 Reset and Clock Control (MRCC_SFA0)	32	RW	8000_0000h
8Ch	CRC0 Reset and Clock Control (MRCC_CRC0)	32	RW	8000_0000h
90h	ELE Reset and Clock Control (MRCC_SECSUBSYS)	32	RW	8000_0000h
BCh	LPIT0 Reset and Clock Control (MRCC_LPIT0)	32	RW	8000_0000h
C0h	TSTMR0 Reset and Clock Control (MRCC_TSTMR0)	32	RW	0000_0002h
C4h	TPM0 Reset and Clock Control (MRCC_TPM0)	32	RW	8000_0000h
C8h	TPM1 Reset and Clock Control (MRCC_TPM1)	32	RW	8000_0000h
CCh	LPI2C0 Reset and Clock Control (MRCC_LPI2C0)	32	RW	8000_0000h
D0h	LPI2C1 Reset and Clock Control (MRCC_LPI2C1)	32	RW	8000_0000h
D4h	I3C0 Reset and Clock Control (MRCC_I3C0)	32	RW	8000_0000h
D8h	LPSPI0 Reset and Clock Control (MRCC_LPSPI0)	32	RW	8000_0000h
DCh	LPSPI1 Reset and Clock Control (MRCC_LPSPI1)	32	RW	8000_0000h
E0h	LPUART0 Reset and Clock Control (MRCC_LPUART0)	32	RW	8000_0000h
E4h	LPUART1 Reset and Clock Control (MRCC_LPUART1)	32	RW	8000_0000h
E8h	FLEXIO0 Reset and Clock Control (MRCC_FLEXIO0)	32	RW	8000_0000h
ECh	CAN0 Reset and Clock Control (MRCC_CAN0)	32	RW	8000_0000h
FCh	SEMA42 Reset and Clock Control (MRCC_SEMA0)	32	RW	8000_0000h
104h	DSB Reset and Clock Control (MRCC_DATA_STREAM_2P4)	32	RW	8000_0000h
108h	PORTA Reset and Clock Control (MRCC_PORTA)	32	RW	8000_0000h
10Ch	PORTB Reset and Clock Control (MRCC_PORTB)	32	RW	8000_0000h
110h	PORTC Reset and Clock Control (MRCC_PORTC)	32	RW	8000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
11Ch	ADC0 Reset and Clock Control (MRCC_LPADC0)	32	RW	8000_0000h
120h	LPCMP0 Reset and Clock Control (MRCC_LPCMP0)	32	RW	8000_0000h
124h	LPCMP1 Reset and Clock Control (MRCC_LPCMP1)	32	RW	8000_0000h
128h	VREF0 Reset and Clock Control (MRCC_VREF0)	32	RW	8000_0000h
404h	GPIOA Reset and Clock Control (MRCC_GPIOA)	32	RW	8000_0000h
408h	GPIOB Reset and Clock Control (MRCC_GPIOB)	32	RW	8000_0000h
40Ch	GPIOC Reset and Clock Control (MRCC_GPIOC)	32	RW	8000_0000h
410h	DMA0 Reset and Clock Control (MRCC_DMA0)	32	RW	8000_0000h
414h	FMC-NPX Reset and Clock Control (MRCC_PFLEXNVM)	32	RW	0000_0002h
41Ch	CTCM Reset and Clock Control (MRCC_SRAM0)	32	RW	0000_0002h
420h	STCM0 Reset and Clock Control (MRCC_SRAM1)	32	RW	0000_0002h
424h	STCM1 Reset and Clock Control (MRCC_SRAM2)	32	RW	0000_0002h
428h	STCM2 Reset and Clock Control (MRCC_SRAM3)	32	RW	0000_0002h

27.5.1.2 EWM0 Reset and Clock Control (MRCC_EWM0)

Offset

Register	Offset
MRCC_EWM0	4Ch

Function

This register configures reset and clock control to the EWM0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.3 SYSPM0 Reset and Clock Control (MRCC_SYSPM0)

Offset

Register	Offset
MRCC_SYSPM0	5Ch

Function

This register configures reset and clock control to the SYSPM module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														CC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.4 WDOG0 Reset and Clock Control (MRCC_WDOG0)

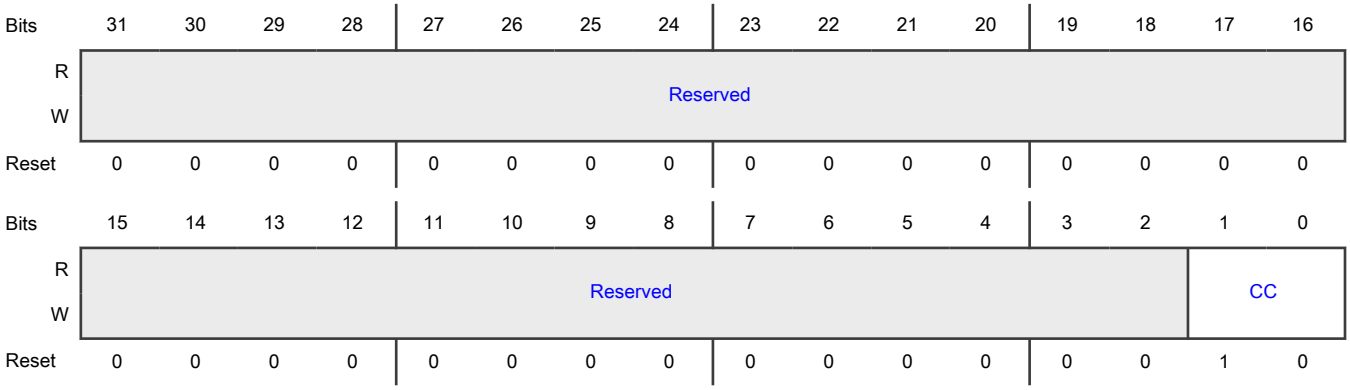
Offset

Register	Offset
MRCC_WDOG0	68h

Function

This register configures reset and clock control to the WDOG0 module.

Diagram



Fields

Field	Function
31-2 —	reserved reserved
1-0	Clock Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
CC	00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.5 WDOG1 Reset and Clock Control (MRCC_WDOG1)

Offset

Register	Offset
MRCC_WDOG1	6Ch

Function

This register configures reset and clock control to the WDOG1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														CC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.6 SFA0 Reset and Clock Control (MRCC_SFA0)

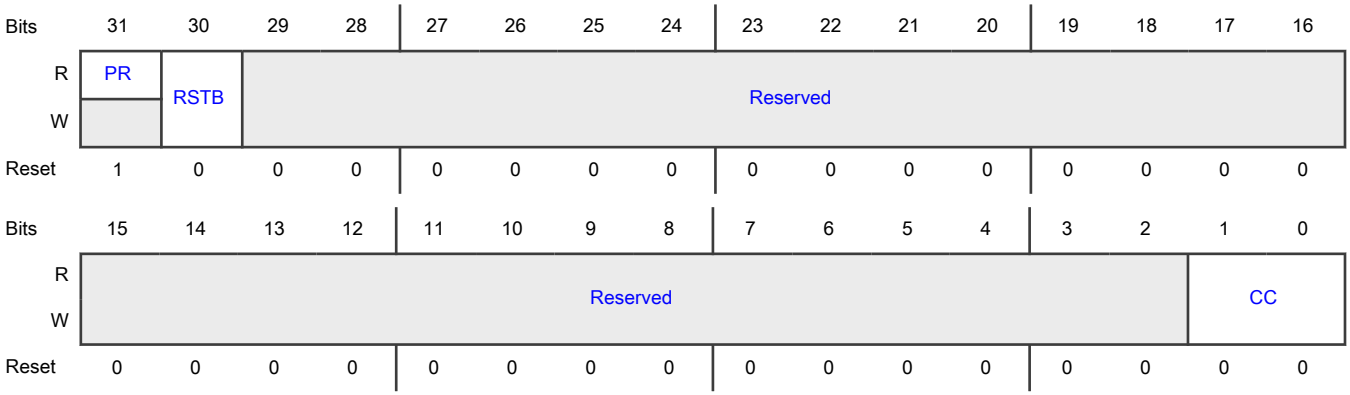
Offset

Register	Offset
MRCC_SFA0	74h

Function

This register configures reset and clock control to the SFA0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.7 CRC0 Reset and Clock Control (MRCC_CRC0)

Offset

Register	Offset
MRCC_CRC0	8Ch

Function

This register configures reset and clock control to the CRC0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.8 ELE Reset and Clock Control (MRCC_SECSUBSYS)

Offset

Register	Offset
MRCC_SECSUBSYS	90h

Function

This register configures reset and clock control to the ELE module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.9 LPIT0 Reset and Clock Control (MRCC_LPIT0)

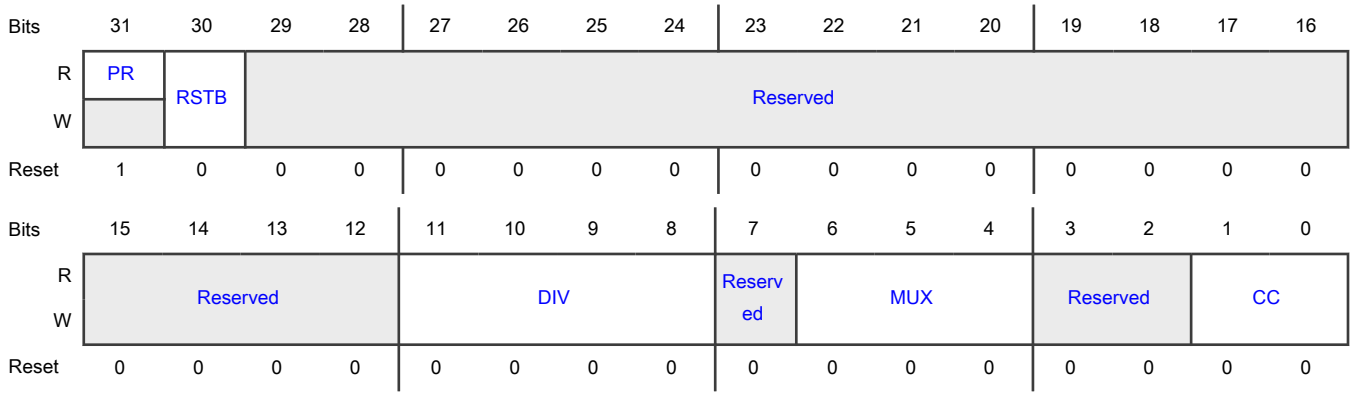
Offset

Register	Offset
MRCC_LPIT0	BCh

Function

This register configures reset and clock control to the LPIT0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.10 TSTMR0 Reset and Clock Control (MRCC_TSTMR0)

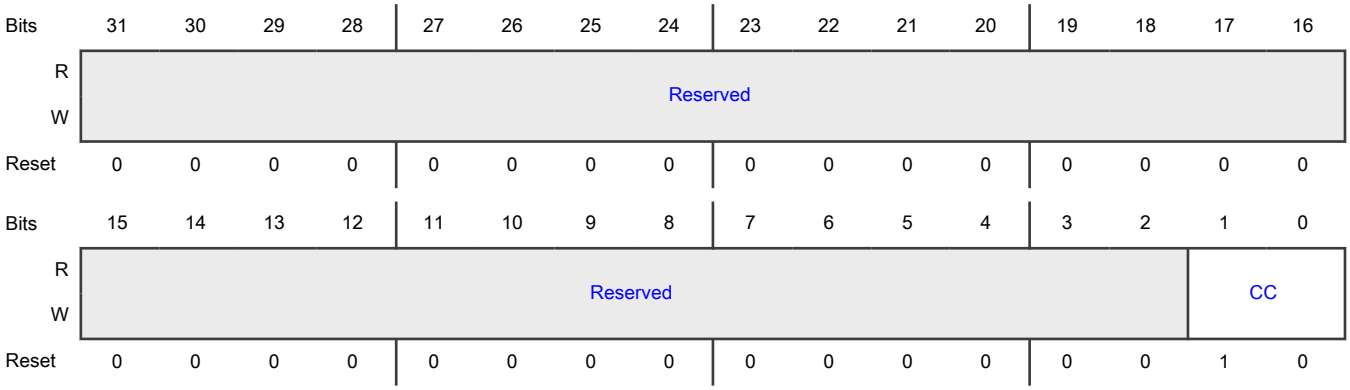
Offset

Register	Offset
MRCC_TSTMR0	C0h

Function

This register configures reset and clock control to the TSTMR0 module.

Diagram



Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.11 TPM0 Reset and Clock Control (MRCC_TPM0)

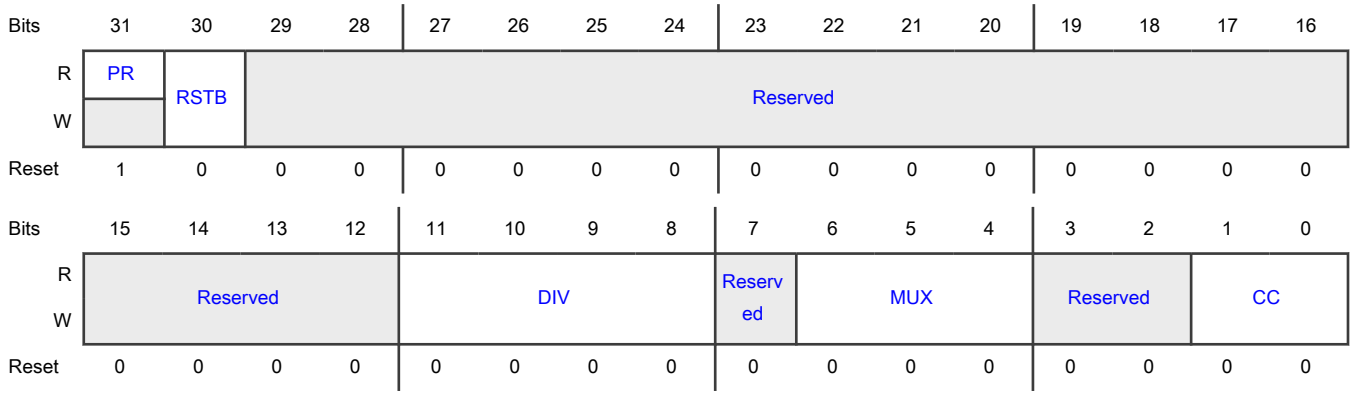
Offset

Register	Offset
MRCC_TPM0	C4h

Function

This register configures reset and clock control to the TPM0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK 101b - 32K-CLK
3-2 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.12 TPM1 Reset and Clock Control (MRCC_TPM1)

Offset

Register	Offset
MRCC_TPM1	C8h

Function

This register configures reset and clock control to the TPM1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				DIV				Reserved	MUX			Reserved		CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30	Reset Negation

Table continues on the next page...

Table continued from the previous page...

Field	Function
RSTB	0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK 101b - 32K-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.13 LPI2C0 Reset and Clock Control (MRCC_LPI2C0)

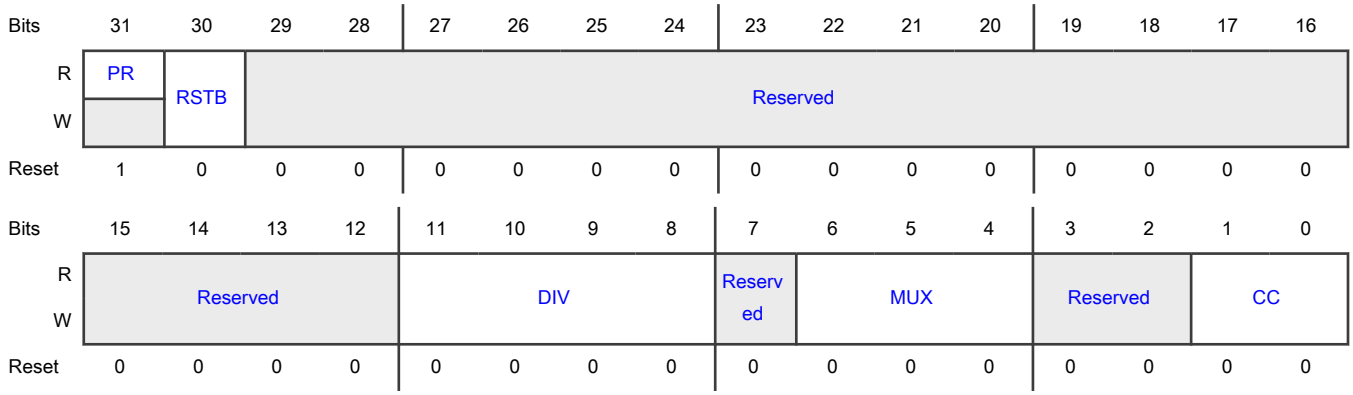
Offset

Register	Offset
MRCC_LPI2C0	CCh

Function

This register configures reset and clock control to the LPI2C0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0	Clock Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
CC	00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.14 LPI2C1 Reset and Clock Control (MRCC_LPI2C1)

Offset

Register	Offset
MRCC_LPI2C1	D0h

Function

This register configures reset and clock control to the LPI2C1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				DIV				Reserved	MUX			Reserved		CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.15 I3C0 Reset and Clock Control (MRCC_I3C0)

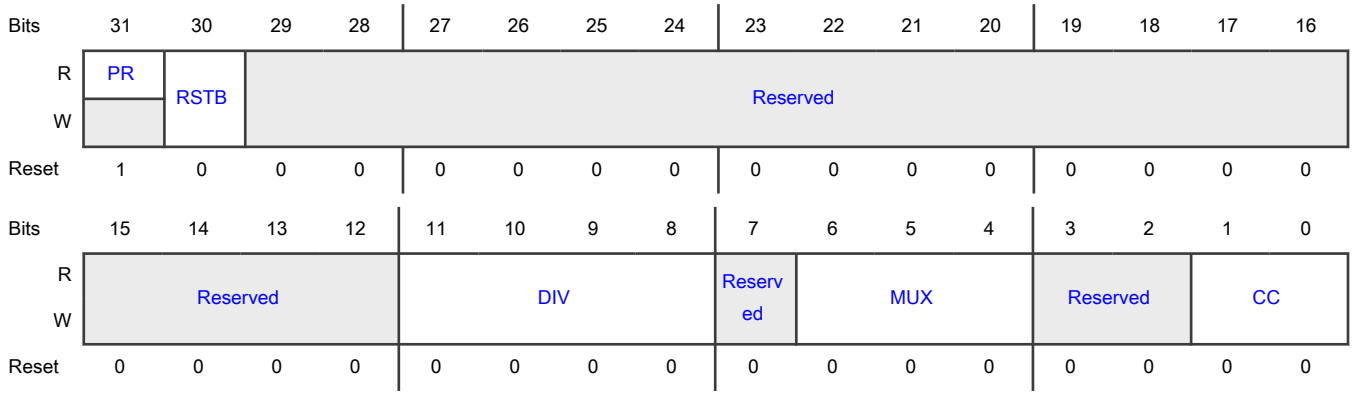
Offset

Register	Offset
MRCC_I3C0	D4h

Function

This register configures reset and clock control to the I3C0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0	Clock Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
CC	00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.16 LPSPI0 Reset and Clock Control (MRCC_LPSPI0)

Offset

Register	Offset
MRCC_LPSPI0	D8h

Function

This register configures reset and clock control to the LPSPI0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				DIV				Reserved	MUX			Reserved		CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

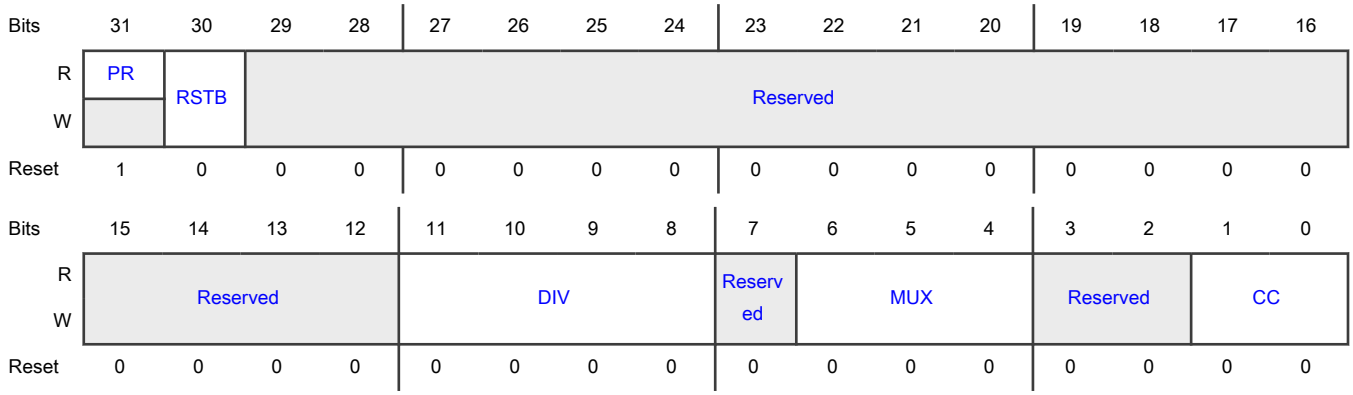
27.5.1.17 LPSPI1 Reset and Clock Control (MRCC_LPSPI1)

Offset

Register	Offset
MRCC_LPSPI1	DCh

Function
This register configures reset and clock control to the LPSPI1 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0	Clock Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
CC	00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.18 LPUART0 Reset and Clock Control (MRCC_LPUART0)

Offset

Register	Offset
MRCC_LPUART0	E0h

Function

This register configures reset and clock control to the LPUART0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				DIV				Reserved	MUX			Reserved		CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK 101b - 32K-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

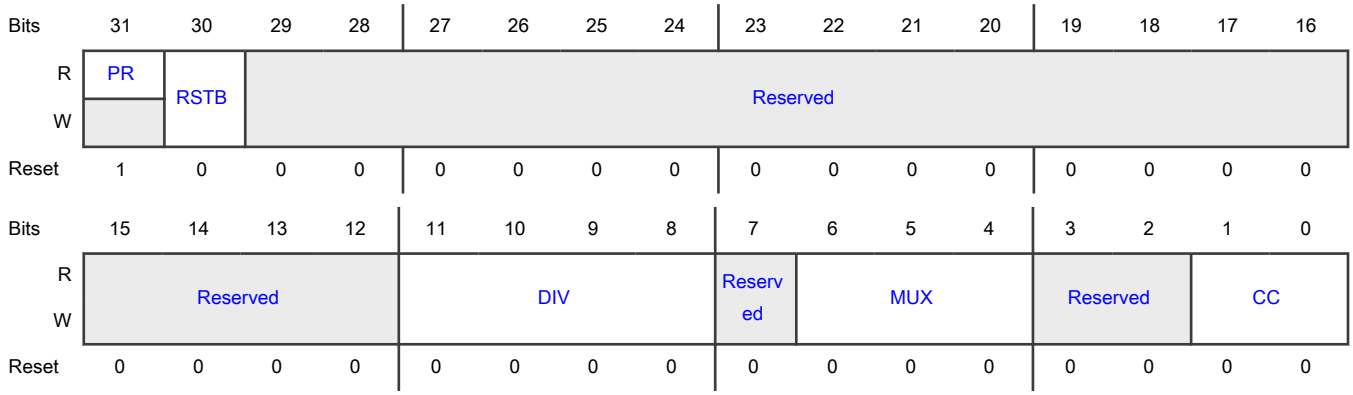
27.5.1.19 LPUART1 Reset and Clock Control (MRCC_LPUART1)

Offset

Register	Offset
MRCC_LPUART1	E4h

Function
This register configures reset and clock control to the LPUART1 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK 101b - 32K-CLK
3-2 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.20 FLEXIO0 Reset and Clock Control (MRCC_FLEXIO0)

Offset

Register	Offset
MRCC_FLEXIO0	E8h

Function

This register configures reset and clock control to the FlexIO0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				DIV				Reserved	MUX			Reserved		CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30	Reset Negation

Table continues on the next page...

Table continued from the previous page...

Field	Function
RSTB	0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.21 CAN0 Reset and Clock Control (MRCC_CAN0)

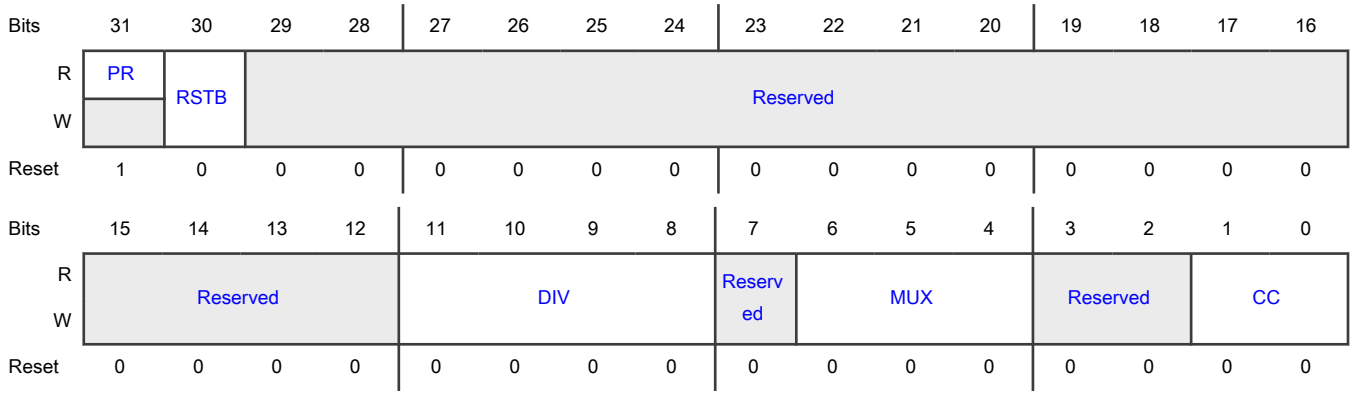
Offset

Register	Offset
MRCC_CAN0	ECh

Function

This register configures reset and clock control to the CAN0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.22 SEMA42 Reset and Clock Control (MRCC_SEMA0)

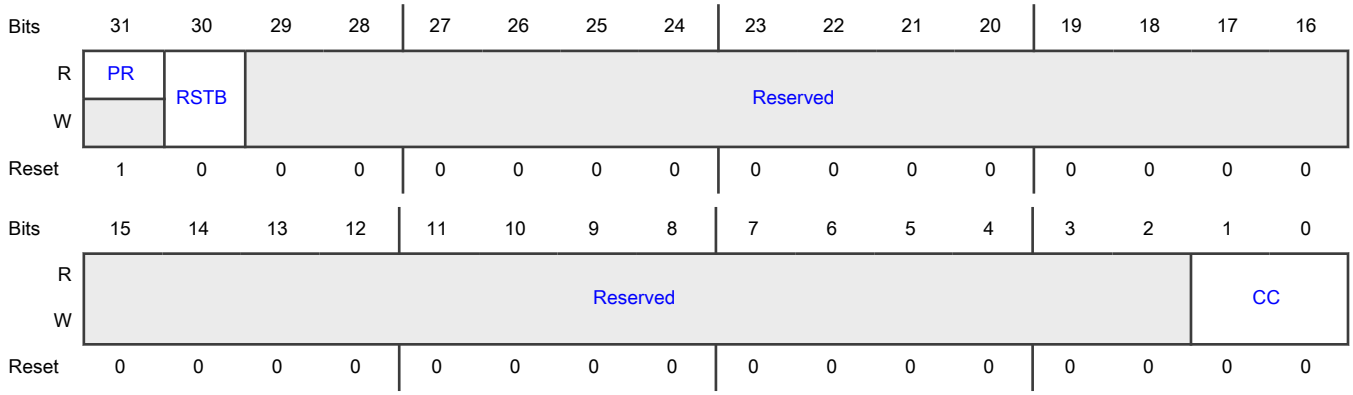
Offset

Register	Offset
MRCC_SEMA0	FCh

Function

This register configures reset and clock control to the SEMA42 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2	reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.23 DSB Reset and Clock Control (MRCC_DATA_STREAM_2P4)

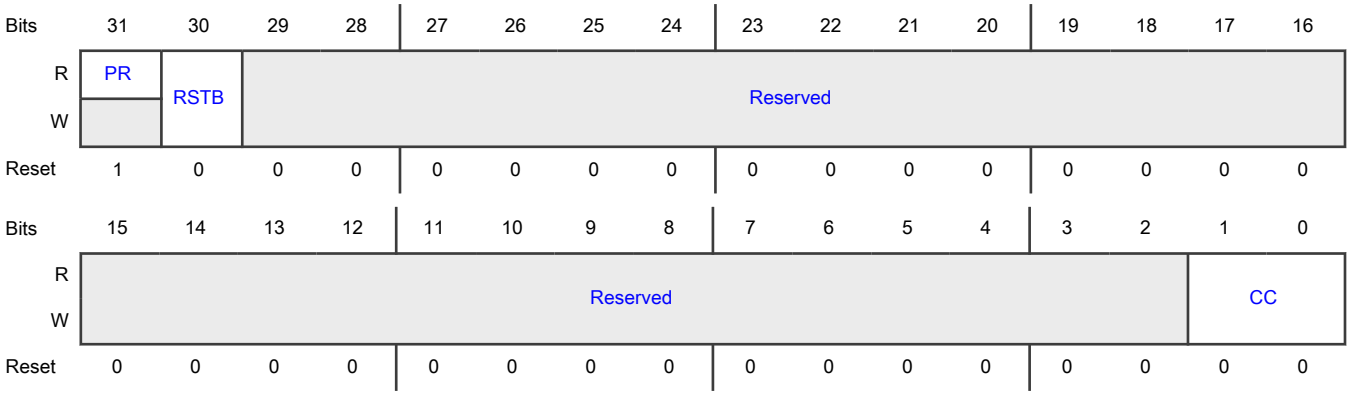
Offset

Register	Offset
MRCC_DATA_STREAM_2P4	104h

Function

This register configures reset and clock control to the DSB module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.24 PORTA Reset and Clock Control (MRCC_PORTA)

Offset

Register	Offset
MRCC_PORTA	108h

Function

This register configures reset and clock control to the PORTA module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	RSTB	Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														CC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.25 PORTB Reset and Clock Control (MRCC_PORTB)

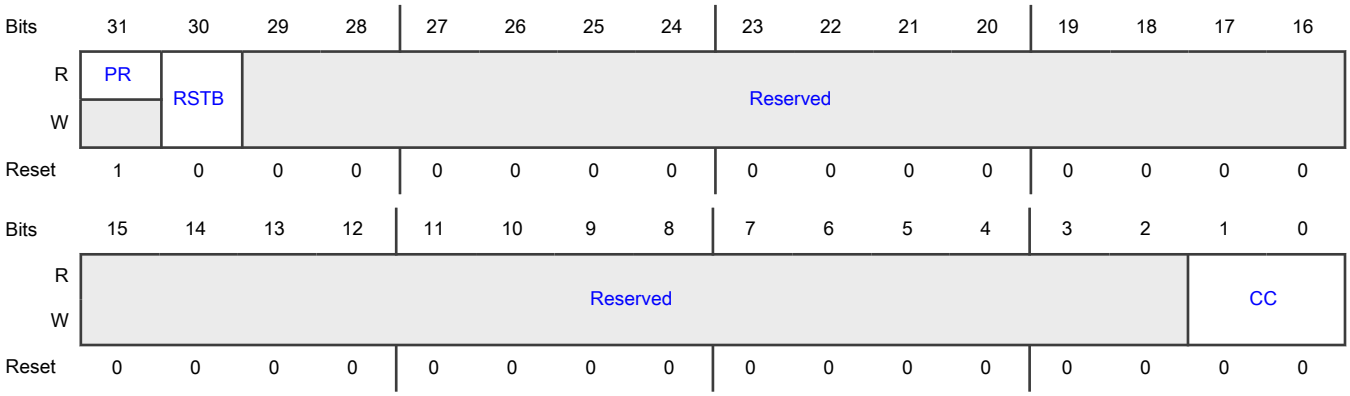
Offset

Register	Offset
MRCC_PORTB	10Ch

Function

This register configures reset and clock control to the PORTB module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.26 PORTC Reset and Clock Control (MRCC_PORTC)

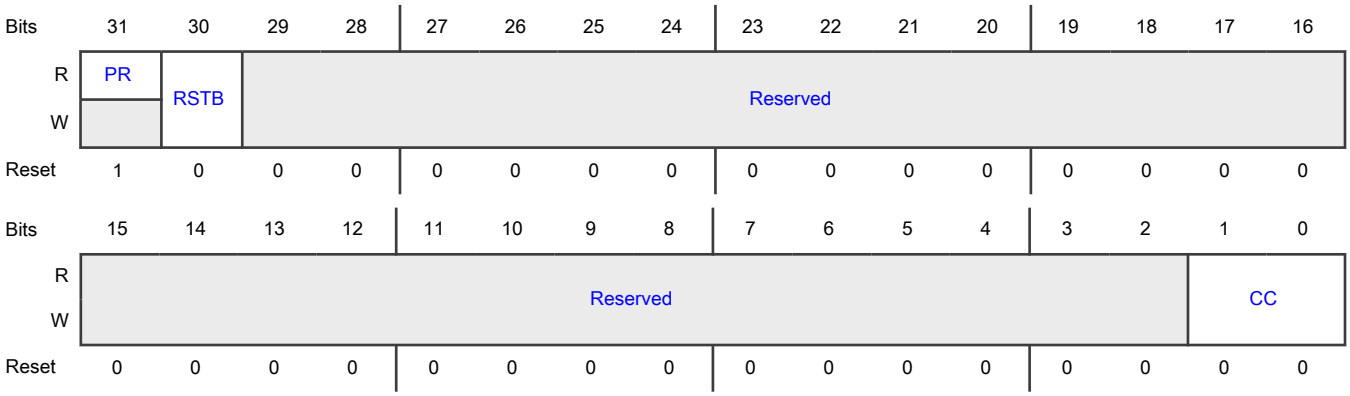
Offset

Register	Offset
MRCC_PORTC	110h

Function

This register configures reset and clock control to the PORTC module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.27 ADC0 Reset and Clock Control (MRCC_LPADC0)

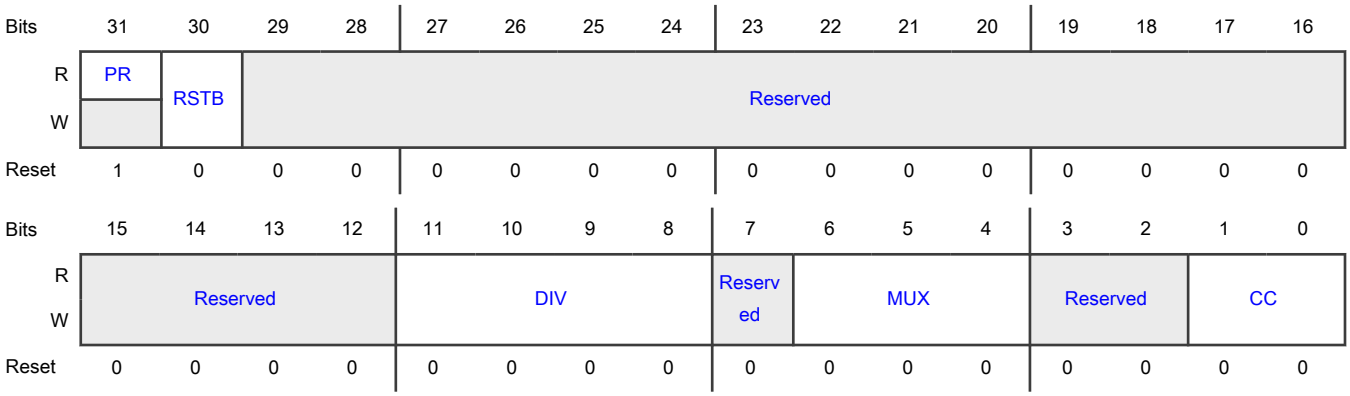
Offset

Register	Offset
MRCC_LPADC0	11Ch

Function

This register configures reset and clock control to the ADC0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-12 —	reserved reserved
11-8 DIV	Functional Clock Divider Configure functional clock integer divider when module clock is disabled (CC=00)
7 —	reserved reserved
6-4 MUX	Functional Clock Mux Select Configure functional clock source when module clock is disabled (CC=00) 000b - The clock is off 010b - FRO-6M 011b - FRO-192M 100b - SOSC-CLK
3-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.28 LPCMP0 Reset and Clock Control (MRCC_LPCMP0)

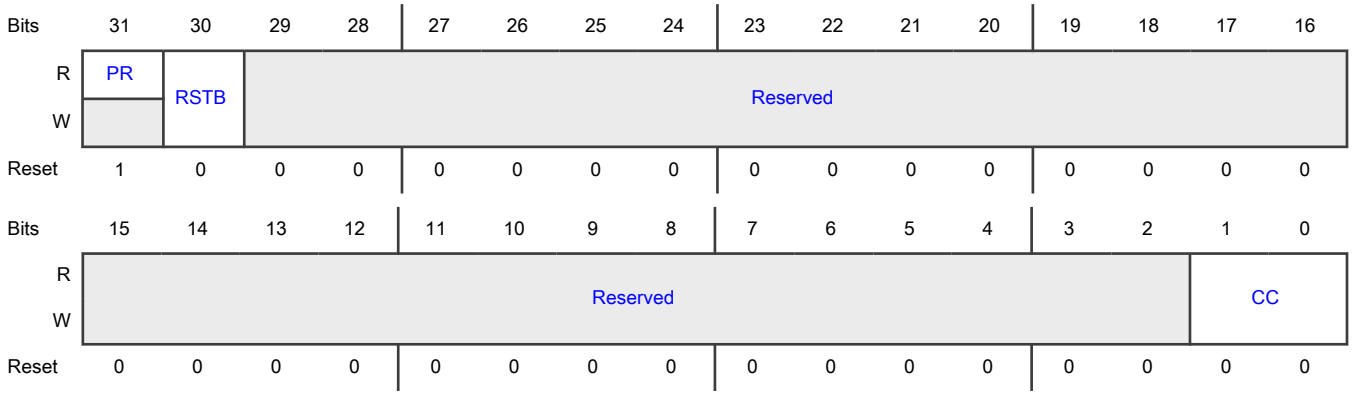
Offset

Register	Offset
MRCC_LPCMP0	120h

Function

This register configures reset and clock control to the LPCMP0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.29 LPCMP1 Reset and Clock Control (MRCC_LPCMP1)

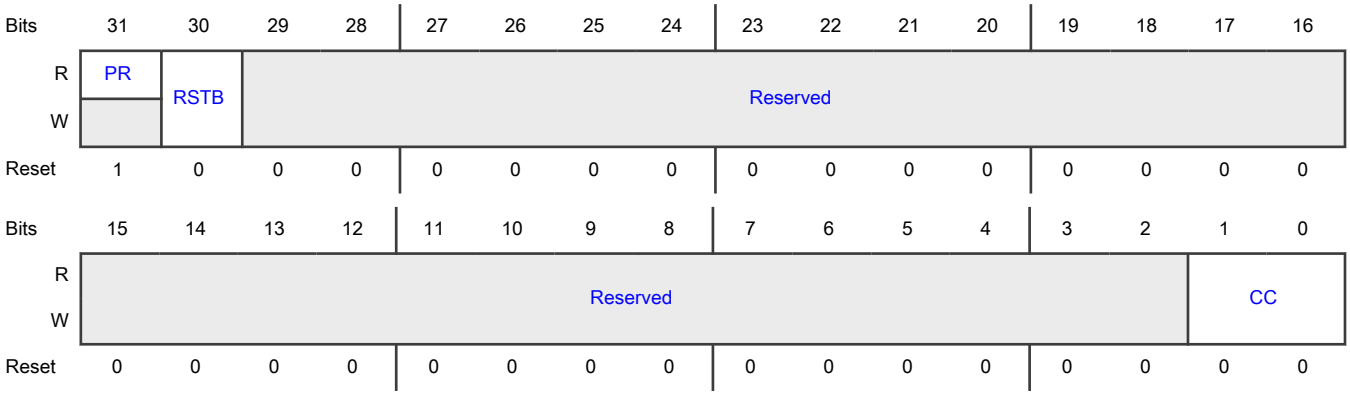
Offset

Register	Offset
MRCC_LPCMP1	124h

Function

This register configures reset and clock control to the LPCMP1 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.30 VREF0 Reset and Clock Control (MRCC_VREF0)

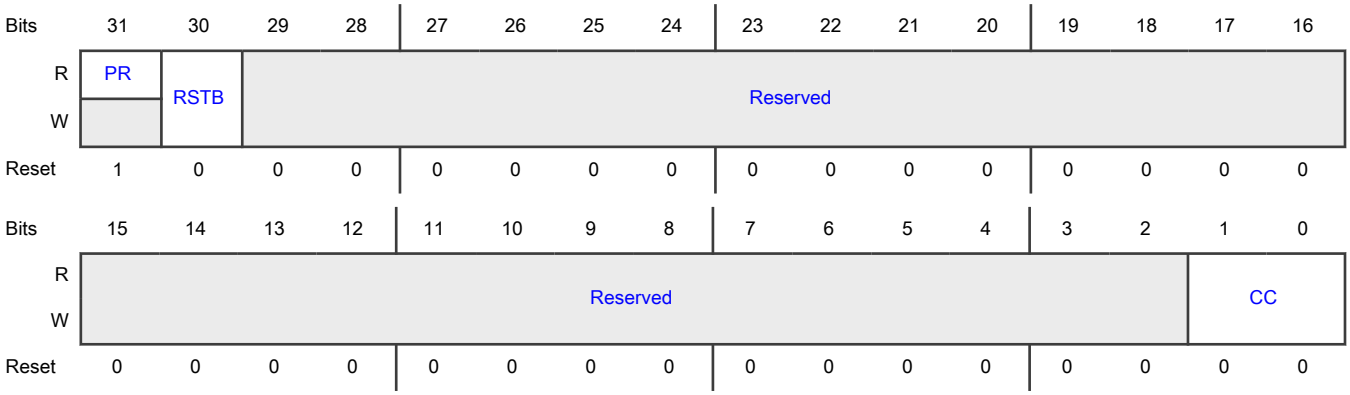
Offset

Register	Offset
MRCC_VREF0	128h

Function

This register configures reset and clock control to the VREF0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.31 GPIOA Reset and Clock Control (MRCC_GPIOA)

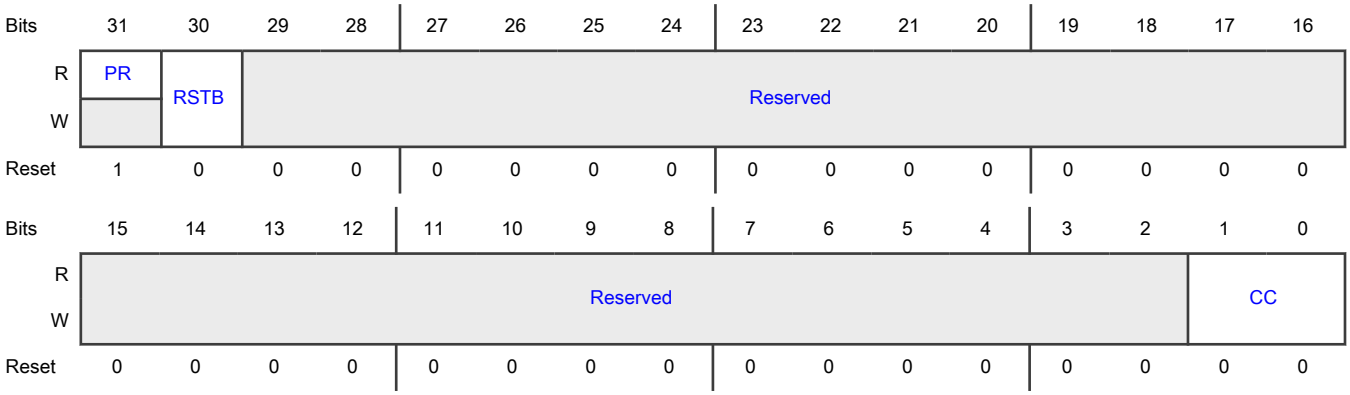
Offset

Register	Offset
MRCC_GPIOA	404h

Function

This register configures reset and clock control to the GPIOA module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.32 GPIOB Reset and Clock Control (MRCC_GPIOB)

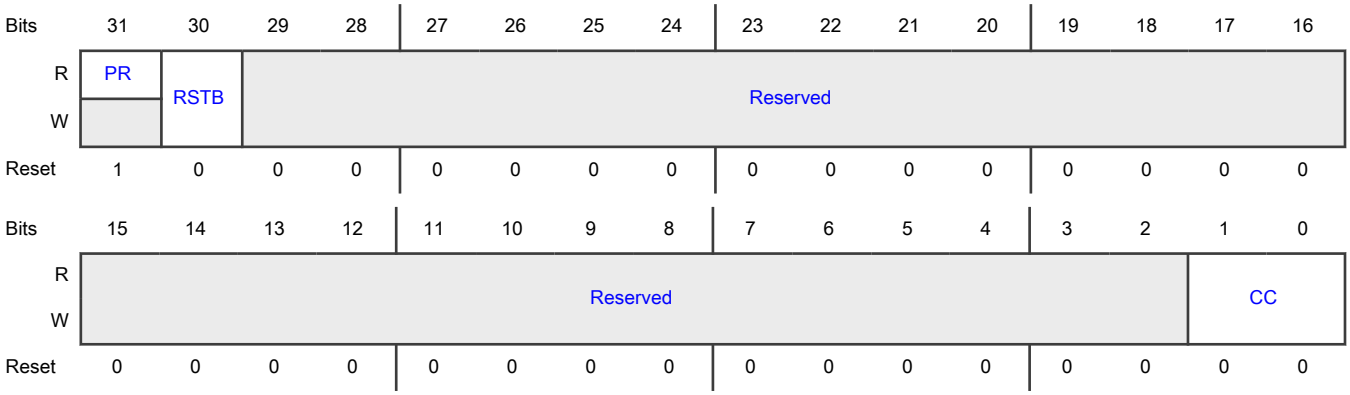
Offset

Register	Offset
MRCC_GPIOB	408h

Function

This register configures reset and clock control to the GPIOB module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.33 GPIOC Reset and Clock Control (MRCC_GPIOC)

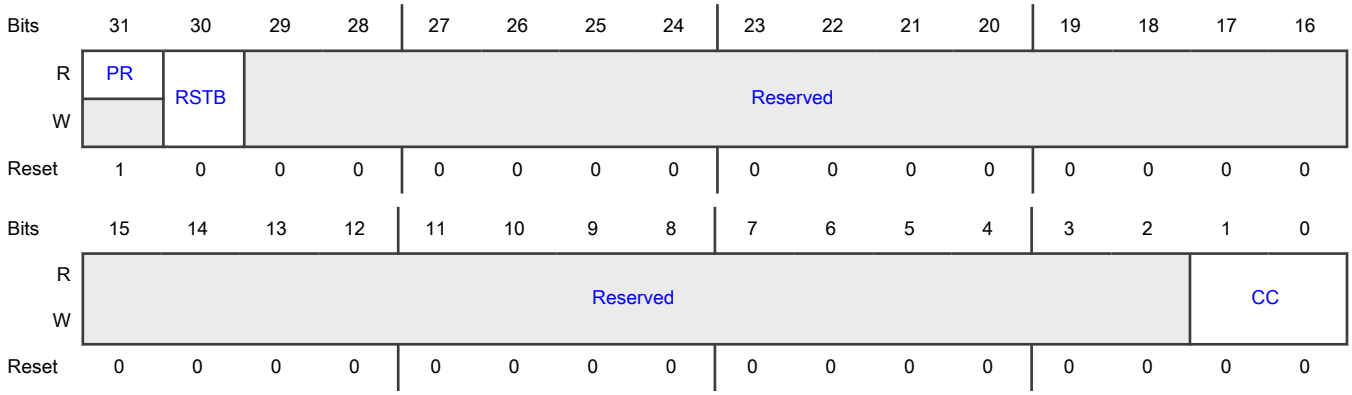
Offset

Register	Offset
MRCC_GPIOC	40Ch

Function

This register configures reset and clock control to the GPIOC module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.34 DMA0 Reset and Clock Control (MRCC_DMA0)

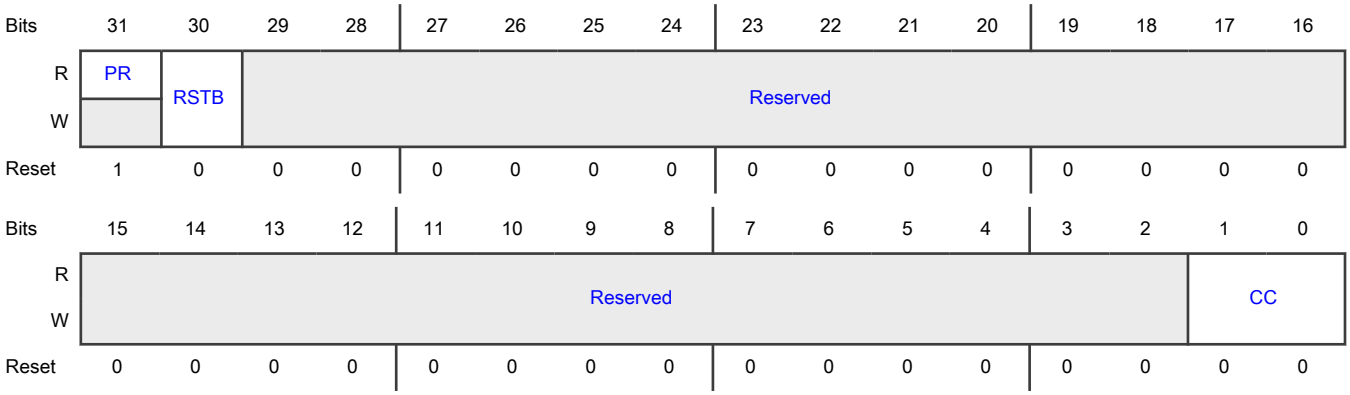
Offset

Register	Offset
MRCC_DMA0	410h

Function

This register configures reset and clock control to the DMA0 module.

Diagram



Fields

Field	Function
31 PR	Peripheral Present 0b - Module is not present; writes to this register are ignored 1b - Module is present
30 RSTB	Reset Negation 0b - Module is held in reset 1b - Module released from reset
29-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.35 FMC-NPX Reset and Clock Control (MRCC_PFLEXNVM)

Offset

Register	Offset
MRCC_PFLEXNVM	414h

Function

This register configures clock control to the FMC-NPX module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														CC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Reserved 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.36 CTCM Reset and Clock Control (MRCC_SRAM0)

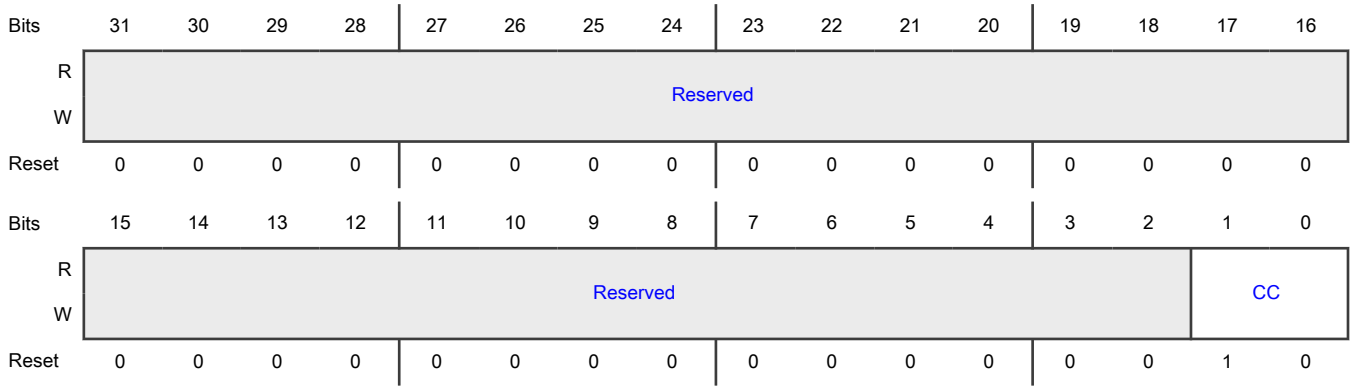
Offset

Register	Offset
MRCC_SRAM0	41Ch

Function

This register configures clock control to the CTCM0 and CTCM1 modules.

Diagram



Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.37 STCM0 Reset and Clock Control (MRCC_SRAM1)

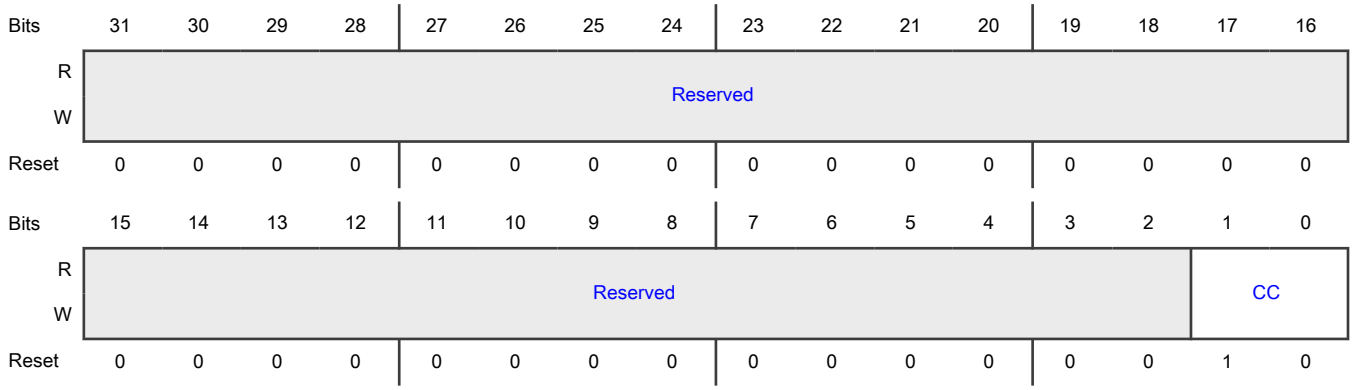
Offset

Register	Offset
MRCC_SRAM1	420h

Function

This register configures clock control to the STCM0 module.

Diagram



Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.38 STCM1 Reset and Clock Control (MRCC_SRAM2)

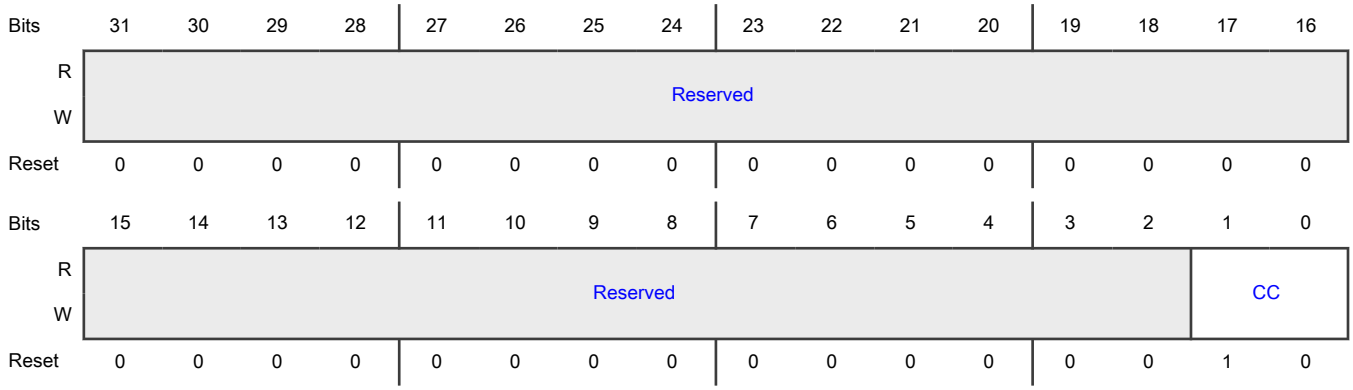
Offset

Register	Offset
MRCC_SRAM2	424h

Function

This register configures clock control to the STCM1 module.

Diagram



Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

27.5.1.39 STCM2 Reset and Clock Control (MRCC_SRAM3)

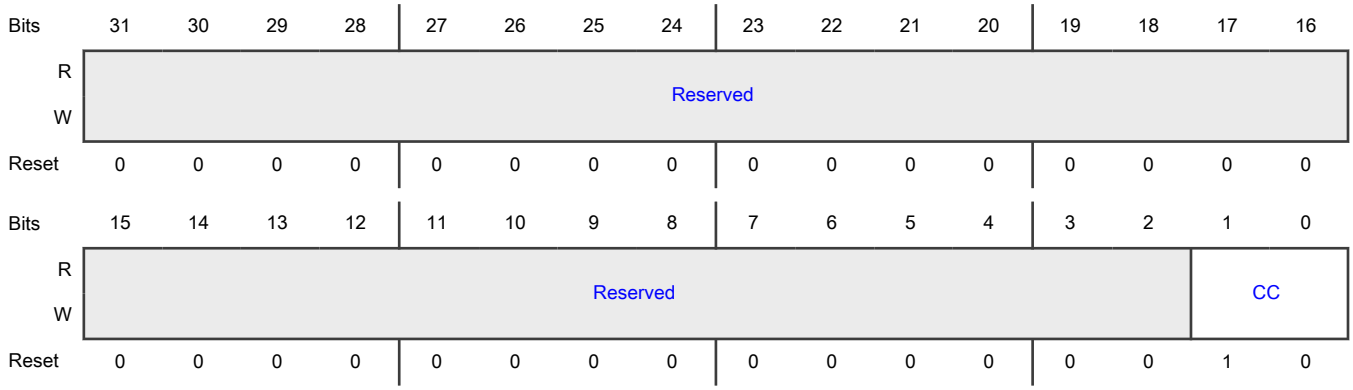
Offset

Register	Offset
MRCC_SRAM3	428h

Function

This register configures clock control to the STCM2 module.

Diagram



Fields

Field	Function
31-2 —	reserved reserved
1-0 CC	Clock Configuration 00b - Peripheral clocks are disabled; module does not stall low power mode entry 01b - Peripheral clocks are enabled; module does not stall low power mode entry 10b - Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle 11b - Peripheral clocks are enabled unless in SLEEP (or lower) mode; low power mode entry stalls until module is idle.

Chapter 28
Signal Frequency Analyser (SFA)

28.1 Chip-specific SFA information

Table 198. Reference links to related information

Topic	Related module	Reference
Full description	SFA	SFA
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

28.1.1 Module instances

This device has two instances of the SFA module, SFA0 and RF_SFA. The SFA0 does not support trigger based measurement function and all the related registers and bit fields are reserved.

28.1.2 SFA register configuration

The following tables list the instance specific configurations.

Table 199. Clock sources connected to the SFA0 clock under test counter

SFA0_CTRL[CUT_SEL]	Clock source
0001	CLKOUT
0110	32K_CLK
1000	OSC-RF
1010	Radio FRO-192M-DIV
1011	FRO-16K
Others	Reserved

Table 200. Clock sources connected to the RF_SFA clock under test counter

RF_SFA_CTRL[CUT_SEL]	Clock source
0/1	32K_CLK

Table 201. Clock sources connected to the SFA0 reference clock

SFA0_CTRL2[REF_CLK_SEL]	Reference clock source
00	OSC-RF
01	CPU_CLK
10	32K-CLK
11	Reserved

Table 202. Clock sources connected to the RF_SFA reference clock

RF_SFA_CTRL2[REF_CLK_SEL]	Reference clock source
0/1	OSC-RF

Table 203. Signals used in RF_SFA trigger based measurement

SFA0_CTRL[TRIG_START_SEL]/ SFA0_CTRL[TRIG_END_SEL]	Trigger source
0	Trigger configured by RFMC_RF2p4GHz_CTRL[SFA_TRIG_EN]
1	Reserved

28.2 Overview

The Signal Frequency Analyser (SFA) provides functionality to perform measurements of clock frequency, clock period, and time between triggers.

28.2.1 Block Diagram

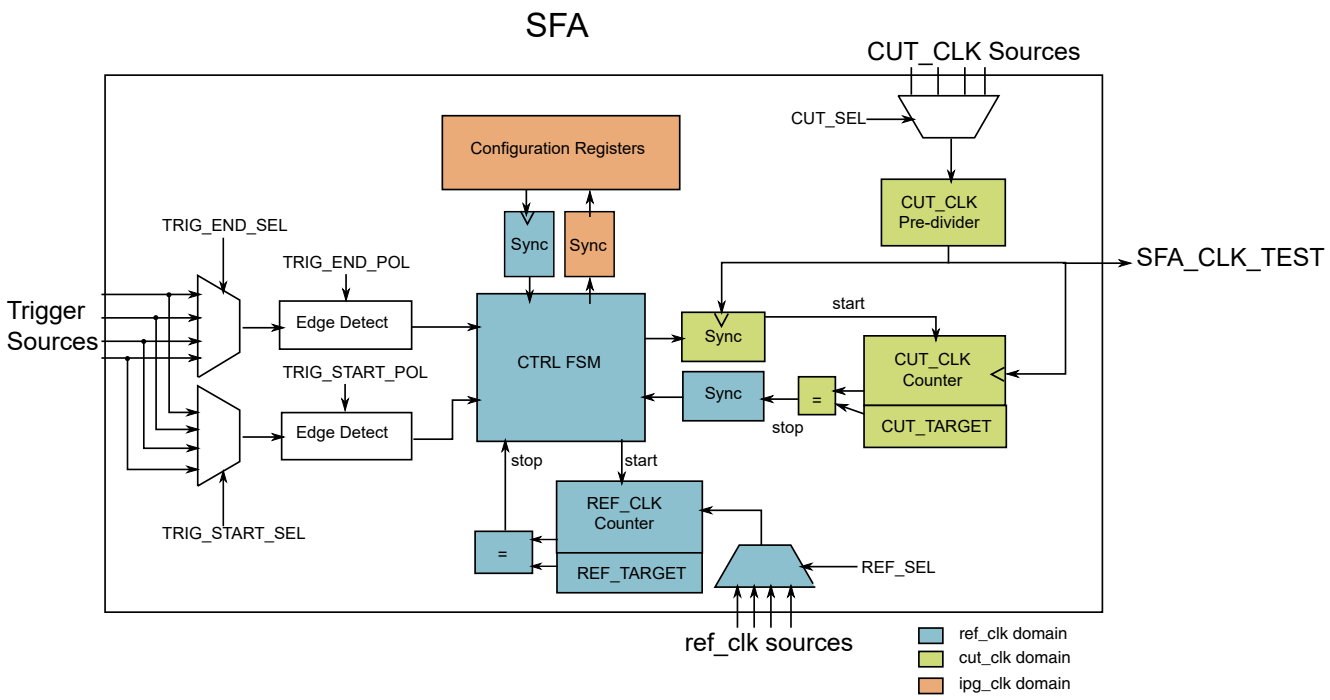


Figure 71. Signal Frequency Analyser Block Diagram

28.2.2 Features

The Signal Frequency Analyser includes the following features.

- Two 32-bit counters, one connected to the bus clock (REF clock), which serves as a reference, and the other connected to the clock under test (CUT).
- An 8-bit predivider is provided to divide down the CUT clock.
- A target count register for both the REF clock and CUT clock are provided to stop the counters.

- Selectable measurement modes to optimize test time by analysing either the frequency (for fast clocks), or the period (for slow clocks) of the CUT clock.
- Selectable sources for reference clock and CUT clock
- Support interrupt for CUT clk freq exceed setting freq limit

28.3 Functional Description

The Signal Frequency Analyser can be used to measure frequencies, periods, and time between events. The following use cases demonstrate how these capabilities can be used.

28.3.1 Frequency Measurements

The Signal Frequency Analyser can be used to measure the frequency of CUT CLK. The use cases below demonstrate how these capabilities can be used.

In general, either of the following methods may then be used to measure clock frequencies. To measure in frequency mode, register bit CTRL[MODE] must be set to 0x0 or 0x1.

Method 1 (when CUT frequency < REF frequency)(CTRL[MODE] == 0x0):

1. Set CUT and REF count targets
2. Perform a dummy write to REF counter (clears the *COUNT, *STARTED and *STOPPED bits)
3. Poll the CUT_STOPPED bit to make sure it cleared
4. Perform a dummy write to the CUT counter. This triggers:
    a. REF counter starts counting immediately
    b. CUT counter starts counting after a ~2 CUT cycle sync delay
5. When CUT counter reaches 1, MEAS_STARTED bit asserts after a ~2 REF cycle sync delay
6. Upon assertion of MEAS_STARTED, the REF_COUNT_ST_SAVED (a saved version of the current REF counter, which keeps running) will be written.
7. When CUT counter reaches target, CUT_STOPPED bit asserts after a ~2 REF cycle sync delay
8. Upon assertion of CUT_STOPPED, the REF_COUNT_END_SAVED (which is a saved version of the actual REF counter, which keeps running) will be written.
9. The measurement is considered complete once the REF_STOPPED and CUT_STOPPED bits have been asserted in the CUT_CNT register. The user can either poll the REF_STOPPED and CUT_STOPPED bits or enable SFA_IRQ_EN. If SFA_IRQ_EN is asserted than an interrupt will be issued when the measurement is complete.
10. Calculate CUT frequency by using the difference in the REF_COUNT_END_SAVED and REF_COUNT_ST_SAVED in comparison to the CUT target
11. If the value (REF_COUNT_END_SAVED-REF_COUNT_ST_SAVED) greater than the value setting in REG_HIGH_LIMIT_CNT , it means the measured cut clk frequency is less than minimum limit,if according enable bit is set, the FREQ_LT_MIN_IRQ will be generated
12. If the value (REF_COUNT_END_SAVED-REF_COUNT_ST_SAVED) less than the value setting in REG_LOW_LIMIT_CNT , it means the measured cut clk frequency is greater than maximum limit, if according enable bit is set, the FREQ_GT_MAX_IRQ will be generated

The following two figures represent two possible cases for frequency measurement with CTRL[MODE] = 0x0. The first image illustrates a complete frequency measurement. Note, it is assumed that the user has started the measurement some time before this waveform snapshot was taken. Therefore, the register writes to CUT counter and REF counter are omitted. In this example the CUT target of 0x0000_FFFF is reached before the REF target count is reached. The two significant measurement events occur when MEAS_STARTED is asserted and REF_CNT_STOPPED is asserted.

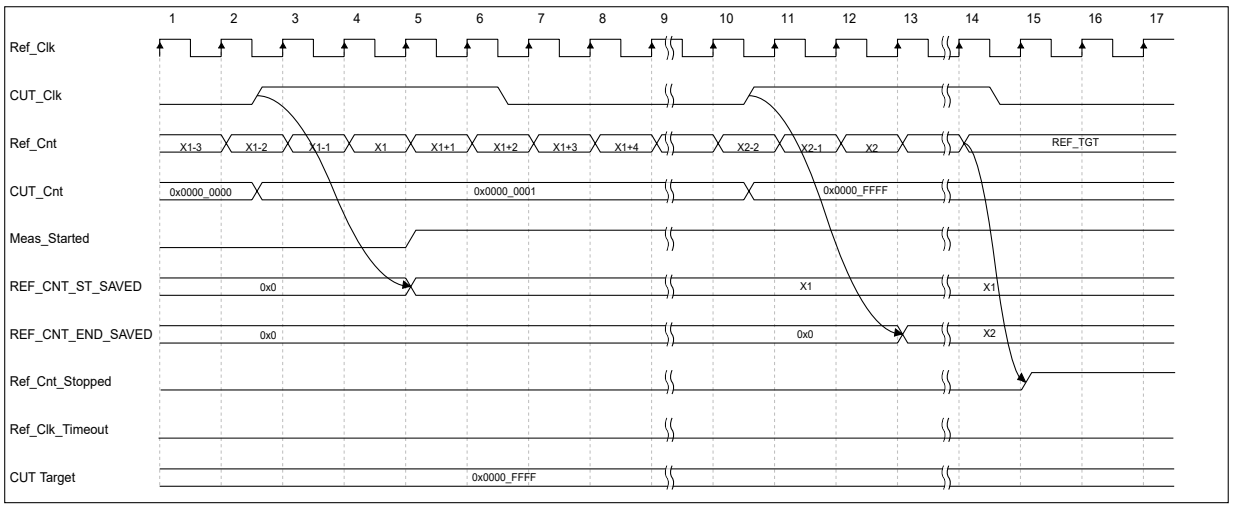


Figure 72. CUT Frequency Measurement with CUT Freq less than REF Freq

The next image illustrates the case when the frequency measurement is unable to be completed before REF_CNT_STOPPED is asserted. This is known as a reference clock timeout. In this example the CUT Target is set to 0x0000_FFFF; however, this value is never reached before REF_CNT = REF_CNT_STOPPED.

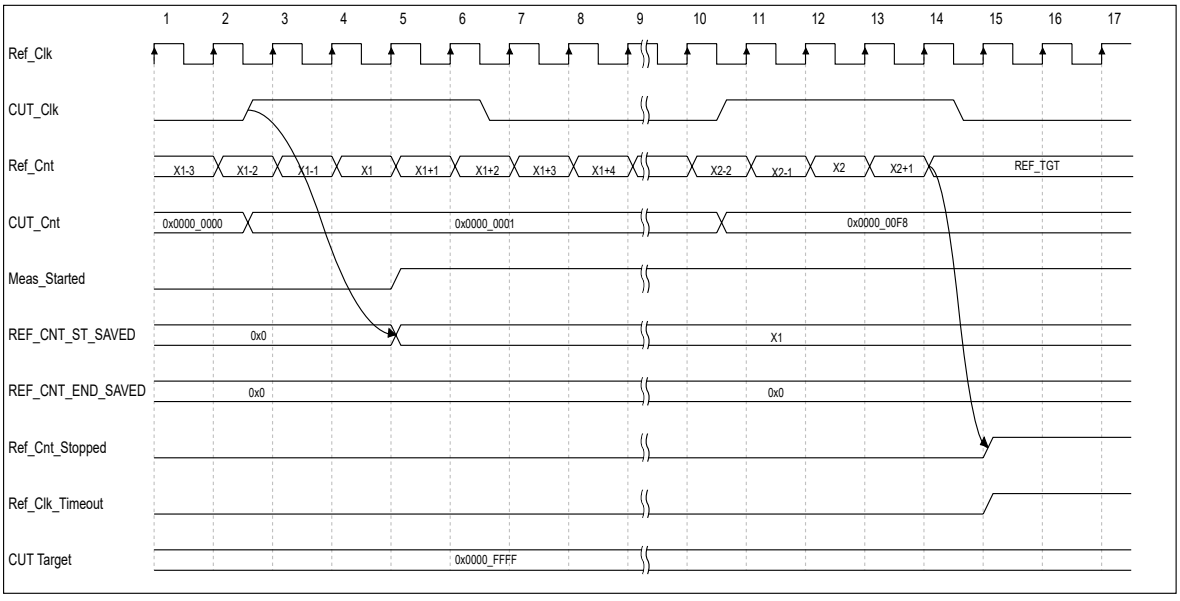


Figure 73. CUT Frequency Measurement Timeout with CUT Freq less than REF Freq

Method 2 (when CUT frequency > REF frequency)(CTRL[MODE] == 0x1):

1. Set CUT and REF count targets (Note: recommend CUT target be set to 0xFFFF_FFFF. There is no CUT_CNT_TIMEOUT flag.)
2. Perform a dummy write to REF counter (clears the *COUNT, *STARTED and *STOPPED bits)
3. Poll the CUT_STOPPED bit to make sure it is clear
4. Perform a dummy write to the CUT counter. This triggers:
    a. REF counter starts counting immediately
    b. CUT counter starts counting after a ~2 CUT cycle sync delay
5. When REF_COUNT reaches target, the CUT counter stops after a ~2 CUT cycle sync delay, and the CUT_STOPPED bit then sets after a ~2 REF cycle sync delay
6. Calculate CUT frequency by using the CUT_COUNT and REF target
7. To determine when the measurement has been completed, the user can either poll the CUT_STOPPED bit or enable SFA_IRQ_EN. An interrupt will be issued, if SFA_IRQ_EN is asserted and when both CUT_STOPPED and REF_STOPPED are asserted.
8. If the value CUT_CNT is greater than the value setting in CUT_HIGH_LIMIT_CNT , it means the measured cut clk frequency or trigger period is greater than maximum limit. If FREQ_GT_MAX interrupt flag is set, FREQ_GT_MAX IRQ will be generated.
9. If the value CUT_CNT less than the value setting in CUT_LOW_LIMIT_CNT , it means the measured cut clk frequency or trigger period is less than minimum limit, if according enable bit is set, the FREQ_LT_MIN_IRQ will be generated

The following image represents an example frequency measurement with CTRL[MODE] = 0x1. In this case, the CUT frequency should be greater than the REF frequency. This measurement is used to count the number of CUT_CLK cycles that occur in REF_TARGET REF clk cycles. Note that registers REF_CNT_*_SAVED should not be read during this mode. It can be implied that REF_CNT_END_SAVED - REF_CNT_ST_SAVED = REF_TARGET.

When performing a CTRL[MODE] = 0x1 measurement, the CUT_TARGET should be set to the max value to prevent a CUT_TARGET timeout. The measurement will end when REF_CNT reaches the value stored in REF_TARGET. If CUT_TARGET is reached before REF_TARGET is reached, this will indicate an error condition. There is no flag to indicate a CUT_TARGET timeout. This is why CUT_TARGET should be set to the maximum value. The CUT_CNT timeout occurs when CUT_CNT is equal to CUT_TARGET. To detect that the CUT_CNT timeout did not occur, the CUT_CNT register should be less than CUT_TARGET at the end of a CTRL[MODE] = 0x1 measurement. When CUT_CNT is less than CUT_TARGET, the measurement stops when REF_TARGET is met.

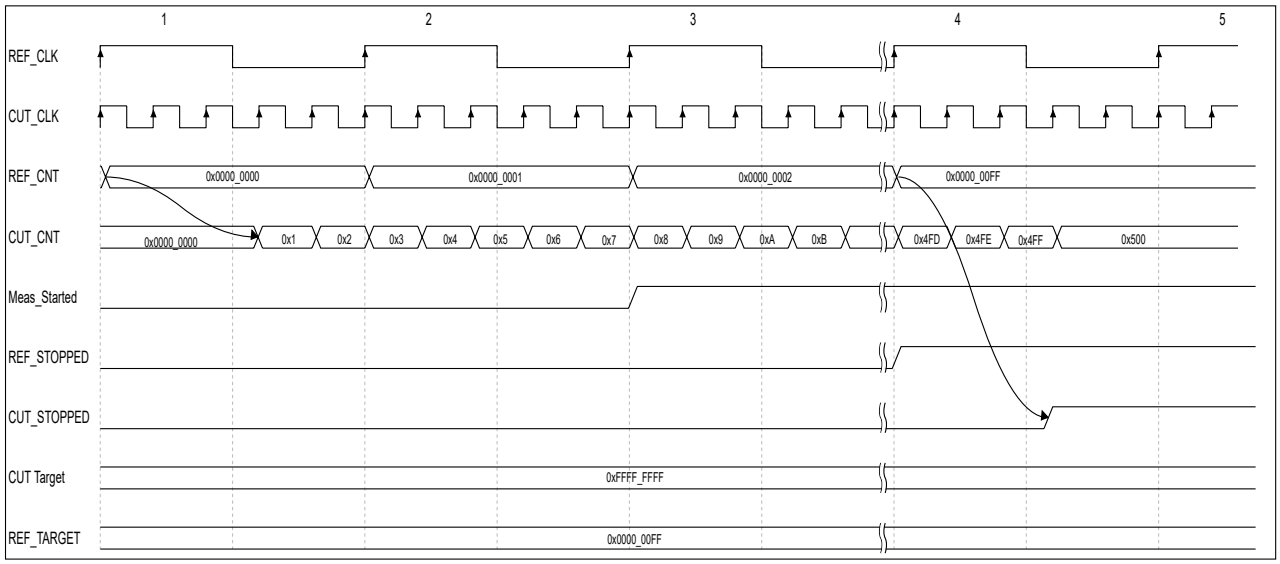


Figure 74. REF Frequency Measurement: CUT Freq greater than REF Freq

28.3.2 Clock Period Measurements

The signal frequency analyser has the capability to measure the period of CUT clock. To operate in clock period mode, the register CTRL[MODE] must be set to 0x2. This mode is used to count the number of REF CLK periods which occur during one period of CUT CLK.

Clock Period Measurement (CTRL[MODE] == 0x2). This mode is used to measure the period of CUT clock. It is assumed that CUT frequency < < REF frequency:

1. Select the source of CUT clock with CTRL[CUT_SEL], and configure any CUT pre-divide with CTRL[CUT_PREDIV].
2. Select a value for REF_TARGET. This value acts as a time-out function. The test will always end when REF_CNT == REF_TARGET.
3. Perform a dummy write to the REF counter. This triggers:
    a. REF counter starts counting immediately
    b. All previous counters will be cleared.
4. The MEAS_STARTED bit will indicate when the measurement sequence has begun.
5. The MEAS_STARTED bit is asserted on the first rising edge of the CUT after the dummy write to the REF counter. Upon assertion of MEAS_STARTED, the REF_COUNT_ST_SAVED (which is a saved version of the actual REF counter, which keeps running) will be written with the current REF_CNT value.
6. The REF_STOPPED bit will indicate when the measurement sequence has been completed.
7. Upon completion of one full period measurement (performed from one rising edge to the next rising edge), the REF_COUNT_END_SAVED (which is a saved version of the actual REF counter, which keeps running) will be written with the current value of REF_CNT.
8. Note that REF_STOPPED may be set before the measurement has been completed. This may indicate the REF_CNT reached REF_TARGET before receiving a full period of CUT CLK. In this case, the CNT_STAT[REF_CNT_TIMEOUT] will be set to 0b1 to indicate that a time-out occurred.

9. Calculate the period width by using the difference between the values of REF_CNT_END_SAVED and REF_COUNT_ST_SAVED.
10. To determine when the measurement has been completed, the user can either poll the REF_STOPPED bit or enable SFA_IRQ_EN. If SFA_IRQ_EN is asserted than an interrupt will be issued when REF_STOPPED is asserted.
11. If the value (REF_COUNT_END_SAVED-REF_COUNT_ST_SAVED) greater than the value setting in REG_HIGH_LIMIT_CNT , it means the measured cut clk frequency is less than minimum limit,if according enable bit is set, the FREQ_LT_MIN_IRQ will be generated
12. If the value (REF_COUNT_END_SAVED-REF_COUNT_ST_SAVED) less than the value setting in REG_LOW_LIMIT_CNT , it means the measured cut clk frequency is greater than maximum limit, if according enable bit is set, the FREQ_GT_MAX_IRQ will be generated

Figure 75 illustrates a CUT Clk period measurement cycle. It is assumed this timing diagram occurs sometime after beginning the measurement (dummy write to REF counter) with CTRL[MODE] == 0x2. In addition, it is assumed that SFA_IRQ_EN is asserted. In Figure 75 , the REF_TARGET register was set sufficiently high to receive a full period of CUT before REF_CNT timeout.

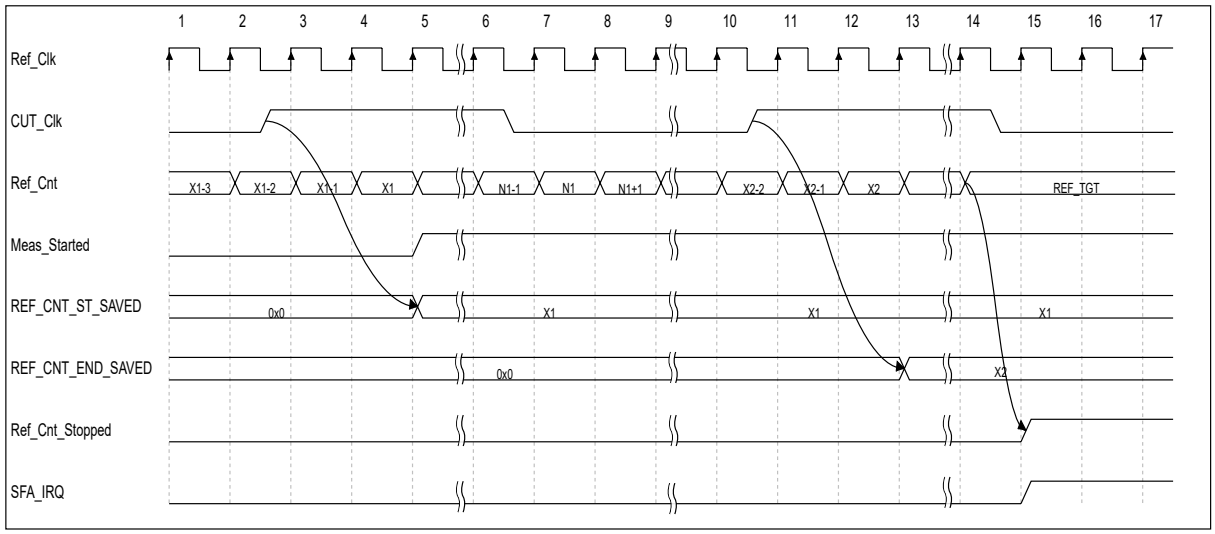


Figure 75. CUT Clk Period Measurement

Figure 76 illustrates what happens when REF_CNT reaches REF_TARGET before the SFA receives a full period of CUT Clk. The result will be an incomplete measurement, and REF_CNT_TIMEOUT will be asserted along with REF_STOPPED.

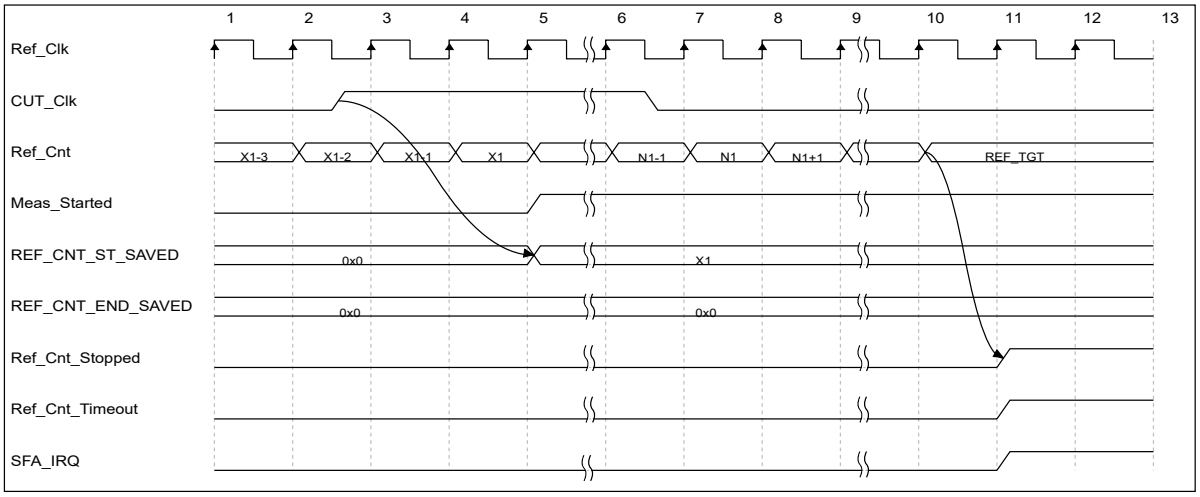


Figure 76. CUT Clk Period Timeout

28.3.3 Interrupts

This module has single interrupt: ipi_init. This interrupt is asserted when:

- CNT_STAT[4] is set and CTRL [5] is enabled
- CNT_STAT[5] is set and CTRL2[16] is enabled
- CNT_STAT[6] is set and CTRL2[17] is enabled

28.4 Memory Map/Register Definition

28.4.1 Signal Frequency Analyser register descriptions

28.4.1.1 Signal_Frequency_Analyser memory map

RF_SFA base address: 48A0_6300h

SFA0 base address: 4001_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Signal Frequency Analyser (SFA) Control (CTRL)	32	RW	0000_0000h
4h	Signal Frequency Analyser (SFA) Control Extended (CTRL_EXT)	32	RW	0000_0000h
8h	Signal Frequency Analyser Count Status Register (CNT_STAT)	32	RW	0000_0000h
Ch	Signal Frequency Analyser Clock Under Test Counter (CUT_CNT)	32	RW	0000_0000h
10h	Signal Frequency Analyser Reference Clock Counter (REF_CNT)	32	RW	0000_0000h
14h	Signal Frequency Analyser Clock Under Test Target Count (CUT_TARGET)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Signal Frequency Analyser Reference Clock Target Count (REF_TARGET)	32	RW	FFFF_FFFFh
1Ch	Signal Frequency Analyser Reference Clock Count Start Saved Register (REF_CNT_ST_SAVED)	32	R	0000_0000h
20h	Signal Frequency Analyser Reference Clock Count End Saved Register (REF_CNT_END_SAVED)	32	R	0000_0000h
24h	Extended control register for SFA (CTRL2)	32	RW	0000_0000h
28h	Record the low limit reference clock count (REF_LOW_LIMIT_CNT)	32	RW	0000_0000h
2Ch	This register record the low limit of ref clk counter (REF_HIGH_LIMIT_CNT)	32	RW	0000_0000h
30h	Record the CUT clock low limit counter (CUT_LOW_LIMIT_CNT)	32	RW	0000_0000h
34h	Record high limit count of cut clock (CUT_HIGH_LIMIT_CNT)	32	RW	0000_0000h

28.4.1.2 Signal Frequency Analyser (SFA) Control (CTRL)

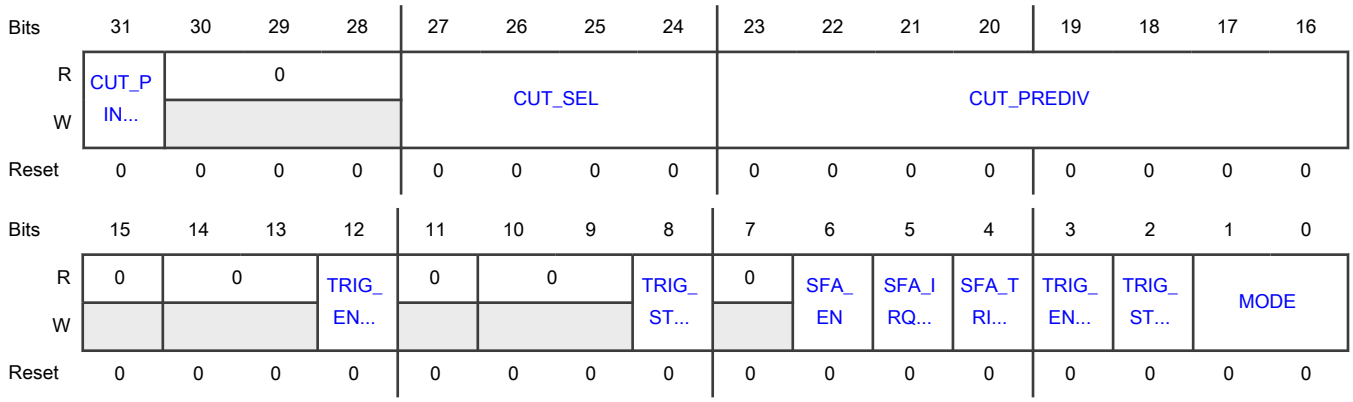
Offset

Register	Offset
CTRL	0h

Function

The SFA Control register contains fields to select both the type of measurement and the signal sources to be measured. The measurement mode will determine whether the SFA performs a frequency, period, or trigger based measurement.

Diagram



Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
CUT_PIN_EN</td><td>CUT_PIN_EN
This register field controls the connection of the clock under test to an external pin.</td></tr>
<tr><td>30-28
—</td><td>Reserved</td></tr>
<tr><td>27-24
CUT_SEL</td><td>CUT_SEL
These bits control which SoC clock is connected to the clock under test counter.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>RF_SFA</td><td>CTRL[24]</td><td>CTRL[27–25]</td></tr>
<tr><td>SFA0</td><td>CTRL</td><td>—</td></tr>
</table></td></tr>
<tr><td>23-16
CUT_PREDIV</td><td>CUT_PREDIV
The CUT_PREDIV 8-bit field provides an integer division of the input CUT signal prior to the CUT counter. Only integer values are implemented and the least significant bit is ignored.
0000_0000b - No Divide
0000_0001b - No Divide
0000_0010b - Divide by 2
0000_0011b - Divide by 2
0000_0100b - Divide by 4
0000_0101b - Divide by 4
0000_0110b - Divide by 6
0000_0111b - Divide by 6
0000_1000b - Divide by 8
0000_1001b - Divide by 8
0000_1010b-1111_1101b - Divide by CUT_PREDIV - CUT_PREDIV%2
1111_1110b - Divide by 254
1111_1111b - Divide by 254</td></tr>
<tr><td>15
—</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>14-13

—</td><td>Reserved</td></tr>
<tr><td>12

TRIG_END_SEL</td><td>Signal MUX For Trigger Based Measurement End

This bit selects which signal will be used to commence a trigger based measurement. An edge of this signal will act as a trigger to end the measurement sequence. Either rising edge or falling edge can be selected by programming TRIG_END_POL.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>RF_SFA</td><td>CTRL</td><td>—</td></tr>
<tr><td>SFA0</td><td>—</td><td>CTRL</td></tr>
</table>
</td></tr>
<tr><td>11

—</td><td>Reserved</td></tr>
<tr><td>10-9

—</td><td>Reserved</td></tr>
<tr><td>8

TRIG_START_SEL</td><td>Signal MUX For Trigger Based Measurement Start

This bit selects which signal will be used to start an edge to edge measurement. An edge of this signal will act as a trigger to begin the measurement sequence. Either rising edge or falling edge can be selected by programming EDGE_START_POL.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>RF_SFA</td><td>CTRL</td><td>—</td></tr>
<tr><td>SFA0</td><td>—</td><td>CTRL</td></tr>
</table>
</td></tr>
<tr><td>7

—</td><td>Reserved</td></tr>
<tr><td>6</td><td>SFA Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>SFA_EN</td><td>This bit is the general SFA Enable signal. To run any measurement, this signal must be set to 0b1. This signal must be set to 0b0 when modifying any clock inputs, clock divider values, or trigger selects. For example, prior to writing TRIG_END_POL, TRIG_START_POL, TRIG_START_SEL, TRIG_END_SEL, CUT_PREDIV, or CUT_SEL make sure that this bit is set to 0b0.
0b - The SFA is disabled.
1b - The SFA is enabled.</td></tr>
<tr><td>5
SFA_IRQ_EN</td><td>SFA Interrupt Enable
This signal determines if interrupts are enabled for the SFA. When enabled, an interrupt will be issued upon completion of a measurement.
0b - Interrupts are disabled.
1b - Interrupts are enabled.</td></tr>
<tr><td>4
SFA_TRIG_MEAS_EN</td><td>SFA Triggered Measurement Enable
The SFA can be triggered to start a new frequency measurement when operating with either MODE = 0x0 or MODE = 0x1. This register field will be inactive if MODE > 0x1. The start trigger will be selected with TRIG_START_SEL and the measurement will begin when the first edge is received with polarity TRIG_START_POL. Note, SFA_EN bit must be set prior to beginning any measurements. The startup process requires a dummy write to the REF_CNT register to enable trigger detection followed by a trigger from the selected source. Refer to section Frequency Measurements for more information.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>RF_SFA</td><td>CTRL</td><td>—</td></tr>
<tr><td>SFA0</td><td>—</td><td>CTRL</td></tr>
</table>
0b - The measurement will start by default with a dummy write to the REF and CUT counters.
1b - The measurement will start after receiging a dummy write to the REF_CNT followed by receiving the trigger edge selected by TRIG_START_SEL and TRIG_START_POL.</td></tr>
<tr><td>3
TRIG_END_POL</td><td>Trigger End Polarity
This signal determines the polarity of the trigger signal which will end a trigger based measurement.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>RF_SFA</td><td>CTRL</td><td>—</td></tr>
<tr><td>SFA0</td><td>—</td><td>CTRL</td></tr>
</table>
0b - Rising edge of TRIGER[TRIG_END_SEL] will end the measurement sequence.
1b - Falling edge of TRIGGER[TRIG_END_SEL] will end the measurement sequence.
</td></tr>
<tr><td>2
TRIG_START_POL</td><td>Trigger Start Polarity
This signal determines the polarity of the start signal when measuring in trigger mode.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>RF_SFA</td><td>CTRL</td><td>—</td></tr>
<tr><td>SFA0</td><td>—</td><td>CTRL</td></tr>
</table>
0b - Rising edge of TRIGGER[TRIG_START_SEL] will begin the measurement sequence.
1b - Falling edge of TRIGGER[TRIG_START_SEL] will begin the measurement sequence.
</td></tr>
<tr><td>1-0
MODE</td><td>MEASUREMENT MODE
Measurement mode determines which type of timing measurement is performed.
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Field value and description</th></tr>
<tr><td>RF_SFA</td><td>00b - Frequency measurement performed with REF frequency > CUT Frequency.
01b - Frequency measurement performed with REF frequency < CUT Frequency.
10b - CUT period measurement performed.
11b - Trigger based measurement performed. Note, each trigger pulse must be held for at least 2 ref_clk cycles.</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>SFA0</td><td>00b - Frequency measurement performed with REF frequency > CUT Frequency. 01b - Frequency measurement performed with REF frequency < CUT Frequency. 10b - CUT period measurement performed. 11b - Reserved.</td></tr></table>

28.4.1.3 Signal Frequency Analyser (SFA) Control Extended (CTRL_EXT)

Offset

Register	Offset
CTRL_EXT	4h

Function

The SFA CTRL_EXT register contains fields to select

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CUT_CLK_EN															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 CUT_CLK_EN	CUT_CLK_EN This register field provides an enable signal for each CUT_CLK source. Each bit corresponds to a single CUT_CLK. Bit 0 corresponds to CUT_CLK0, bit 1 corresponds to CUT_CLK1, and so on. Enabling each bit will enable corresponding CUT_CLK. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
RF_SFA	CTRL_EXT[0]	CTRL_EXT[15–1]
SFA0	CTRL_EXT	—

28.4.1.4 Signal Frequency Analyser Count Status Register (CNT_STAT)

Offset

Register	Offset
CNT_STAT	8h

Function

This register returns status flags to indicate the current state of the REF and CUT counters

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									FREQ_LT...	FREQ_GT...	SFA_IRQ	REF_CNT...	MEAS_ST...	CUT_STO...	REF_STO...
W										W1C	W1C	W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved
6 FREQ_LT_MIN_IRQ	FREQ_LT_MIN interrupt flag FREQ_LT_MIN interrupt flag, it means the cut clk frequency is lower than minimum setting,or trigger measurement period is greater than maximum setting
5 FREQ_GT_MAX_IRQ	FREQ_GT_MAX interrupt flag FREQ_GT_MAX interrupt flag, it means the cut clk frequency is greater than maximum setting,or trigger measurement period is less than minimum setting
4 SFA_IRQ	SFA Interrupt Request This flag indicates that REF_CNT has reached REF_TARGET. More generally, this flag indicates that a measurement has been completed and an interrupt has been issued. Writing a 0x1 to this bit position will clear the interrupt request.
3 REF_CNT_TIMEOUT	Reference Counter Time Out This flag indicates the REF counter reached REF_TARGET before the desired measurement was completed. Every test will commence when REF_CNT == REF_TARGET. For measurement accuracy, this bit should be equal to 0b0 when REF_STOPPED == 0b1. This bit will be cleared on any write to REF_CNT when not performing a measurement.
2 MEAS_STARTED	Measurement Started Flag In frequency measurement mode, this flag indicates when the CUT counter has started counting. Set when CUT counter increments to 1. In period measurement mode, this flag indicates that the first edge of the CUT has been received for period measurement. In edge to edge signal measurement mode, this flag will indicate when TRIGGER[TRIG_START_SEL] has been received for measurement. This event will also push the REF_CNT to the REF_CNT_ST_SAVED register. This flag is only active during CTRL[MODE] == 0x0, 0x2 or 0x3. Cleared on any write to REF_CNT when not performing a measurement.
1 CUT_STOPPED	CUT_STOPPED This flag indicates the CUT counter has stopped counting. Set when either REF or CUT counter has reached its target. Cleared on any write to REF_CNT when not performing a measurement.
0 REF_STOPPED	REF_STOPPED This flag indicates the REF counter has stopped counting. Set when REF counter has reached its target. Cleared on any write to REF_CNT when not performing a measurement. This bit can be polled to indicate when a measurement is complete.

28.4.1.5 Signal Frequency Analyser Clock Under Test Counter (CUT_CNT)

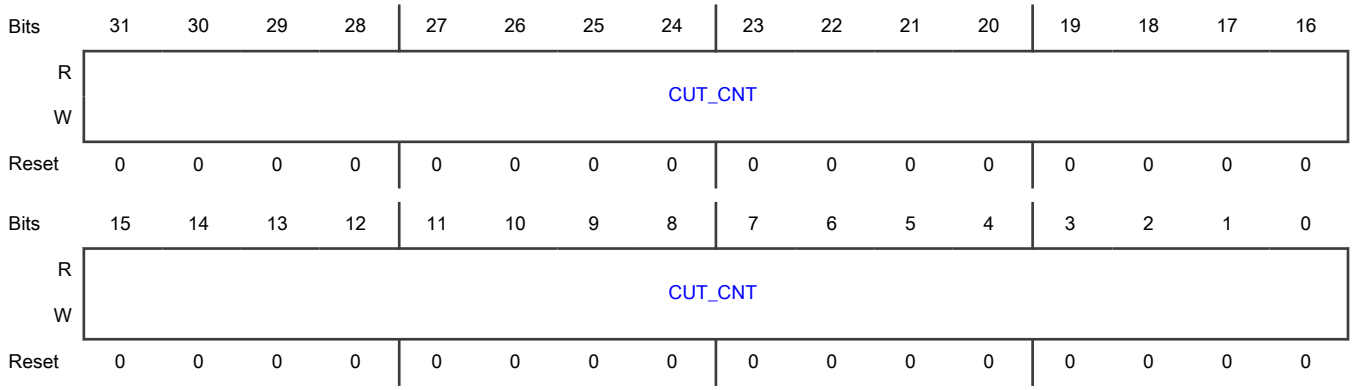
Offset

Register	Offset
CUT_CNT	Ch

Function

This register contains the current count of the clock under test. Writes to this register will start the clock under test and reference counters. Counting will stop when either the count matches the contents of the clock under test target count register or when reference counter matches the reference target count register.Note, attempts to read this register before the CUT_STOPPED flag has set will return 0x0000_0000

Diagram



Fields

Field	Function
31-0 CUT_CNT	CUT_CNT Current clock under test count. Writes to this register will start the clock under test and reference counters.

28.4.1.6 Signal Frequency Analyser Reference Clock Counter (REF_CNT)

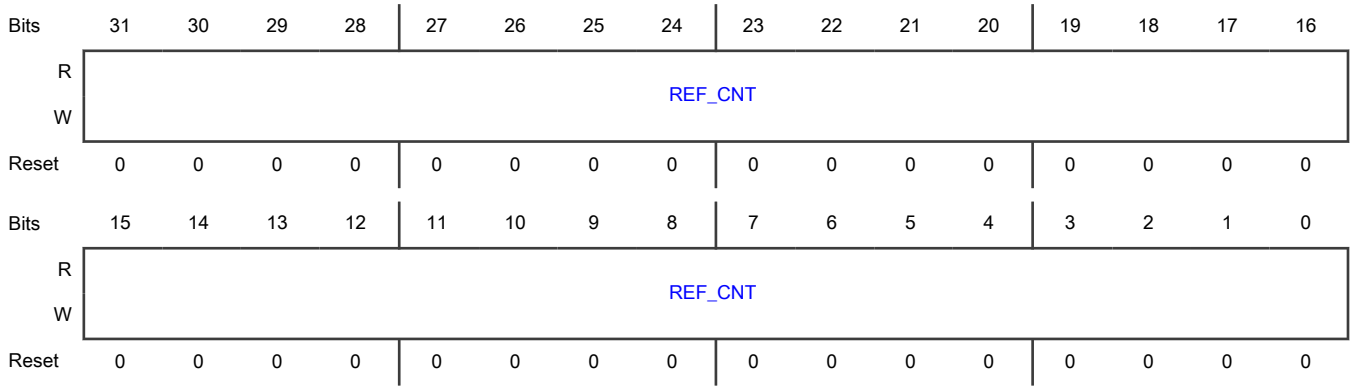
Offset

Register	Offset
REF_CNT	10h

Function

This 32-bit register contains the current count of the reference clock. Writes to this register will clear the clock under test and reference counters. Counting will stop if the count matches the contents of the reference clock target count register.

Diagram



Fields

Field	Function
31-0 REF_CNT	REF_CNT Current reference clock count. Writing to this register with any value will clear the clock under test and reference clock count and status flags registers.

28.4.1.7 Signal Frequency Analyser Clock Under Test Target Count (CUT_TARGET)

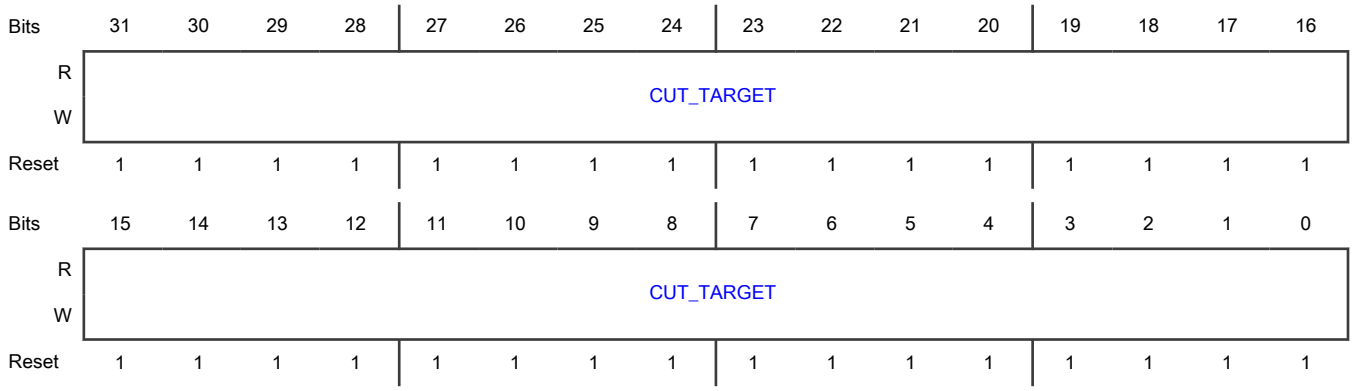
Offset

Register	Offset
CUT_TARGET	14h

Function

This register contains the target count for the clock under test. Both the reference and clock under test count will stop if the clock under test count matches the contents of this register.

Diagram



Fields

Field	Function
31-0 CUT_TARGET	CUT_TARGET Target count for the clock under test. Note, a measurement of type MODE = 0x0 or 0x1 is not valid if CUT_TARGET = 0x00000000. This value must be set to an integer > 0x0 to perform a valid measurement.

28.4.1.8 Signal Frequency Analyser Reference Clock Target Count (REF_TARGET)

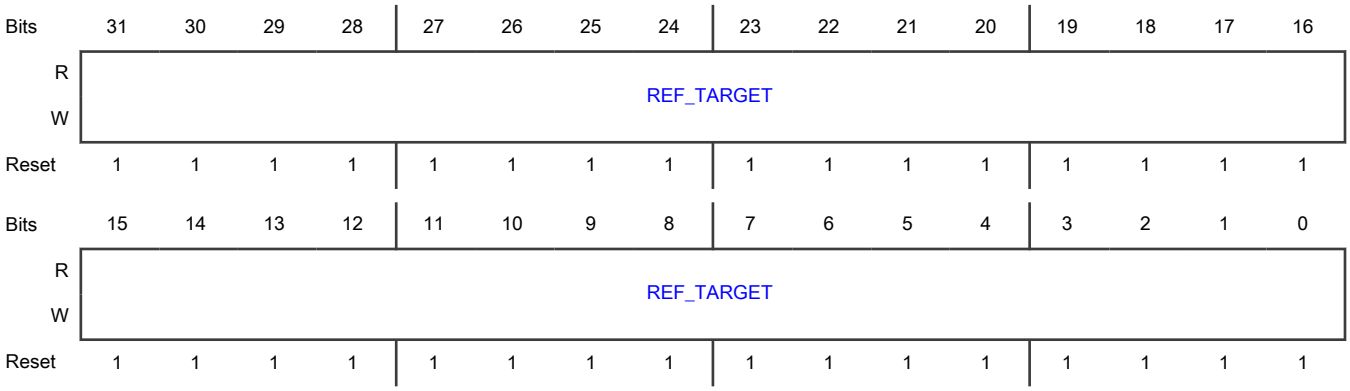
Offset

Register	Offset
REF_TARGET	18h

Function

This register contains the target count for the reference clock. Both the reference and clock under test count will stop if the reference clock count matches the contents of this register.

Diagram



Fields

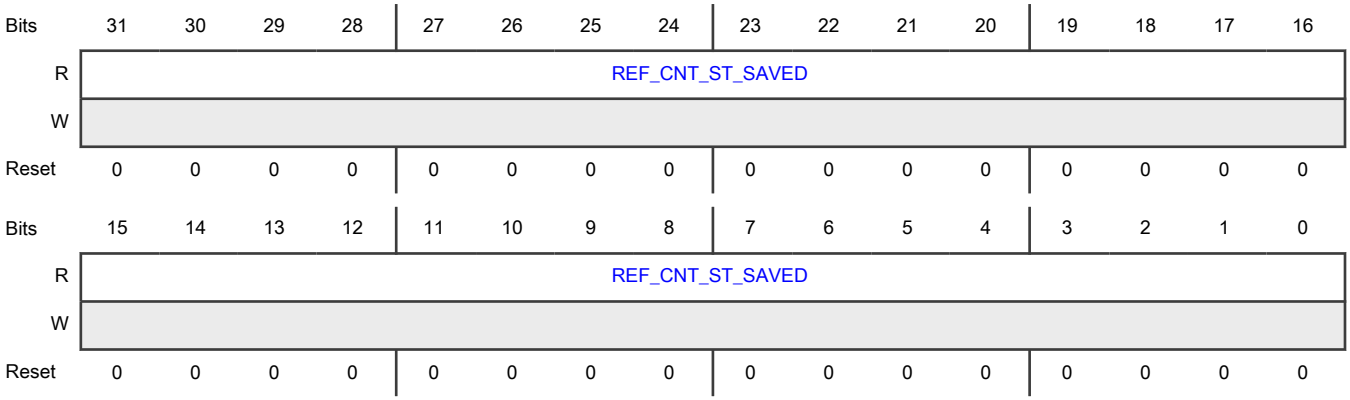
Field	Function
31-0 REF_TARGET	REF_TARGET Target count for the reference clock. Note, a measurement is not valid if REF_TARGET = 0x00000000. This value must be set to an integer > 0x0 to perform a valid measurement.

28.4.1.9 Signal Frequency Analyser Reference Clock Count Start Saved Register (REF_CNT_ST_SAVED)

Offset

Register	Offset
REF_CNT_ST_SAVED	1Ch

Diagram



Fields

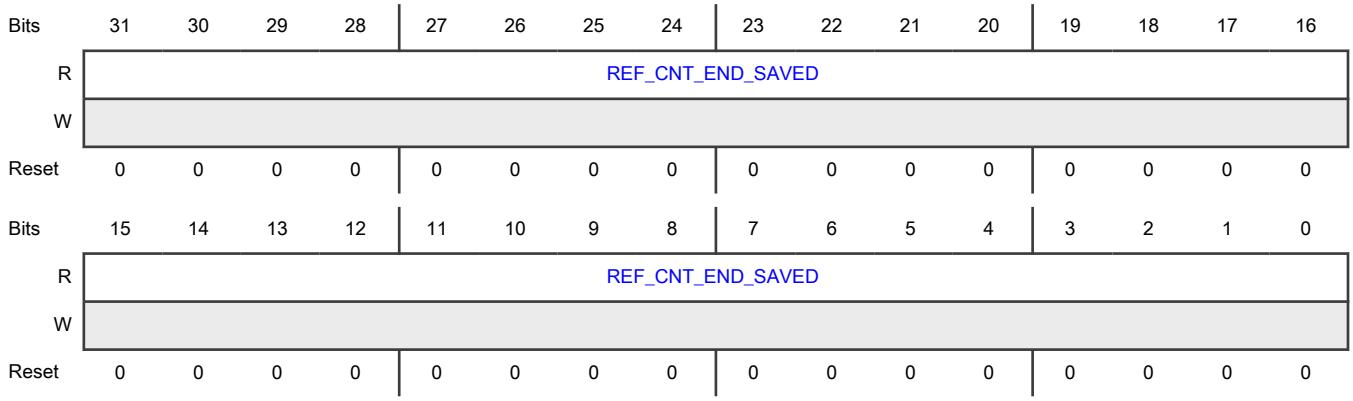
Field	Function
31-0 REF_CNT_ST_SAVED	REF_CNT_ST_SAVED Saved reference clock counter. This register is loaded with the current value of REF_CNT on assertion of the MEAS_STARTED flag. Note, during frequency measurements with CTRL[MODE] = 0x1, this register will be unused. It is assumed that the REF CNT start will be 0x0. This register is cleared on any write to REF_CNT when not performing a measurement.

28.4.1.10 Signal Frequency Analyser Reference Clock Count End Saved Register (REF_CNT_END_SAVED)

Offset

Register	Offset
REF_CNT_END_SAVED	20h

Diagram



Fields

Field	Function
31-0 REF_CNT_END_SAVED	REF_CNT_END_SAVED Saved reference clock counter. This register is loaded with the current value of REF_CNT on assertion of either the CUT_STOPPED or REF_STOPPED flags. Note, during frequency measurements with CTRL[MODE] = 0x1, this register will be unused. It is assumed that the REF_CNT_END will be equal to REF_TARGET. Cleared on any write to REF_CNT when not performing a measurement.

28.4.1.11 Extended control register for SFA (CTRL2)

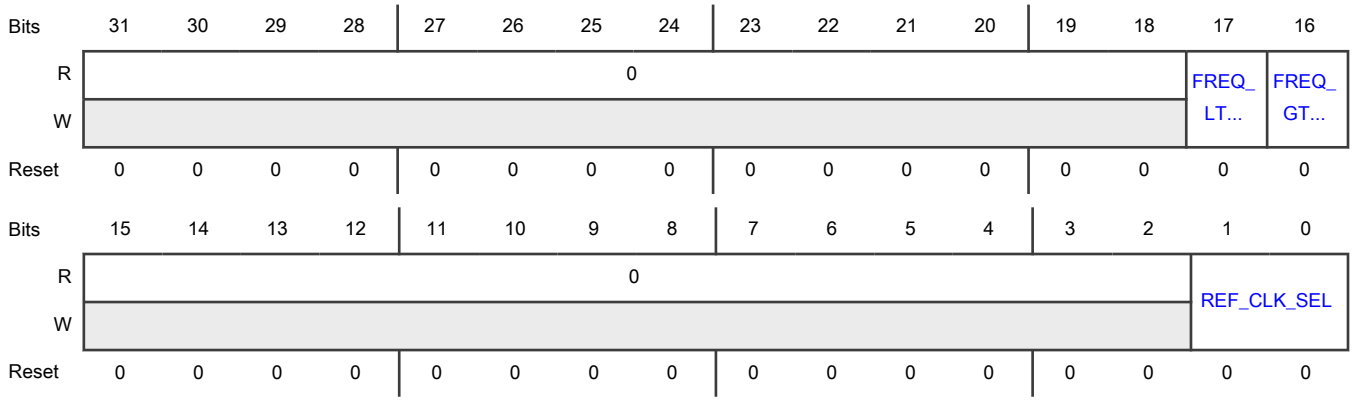
Offset

Register	Offset
CTRL2	24h

Function

This register an extended control register for SFA

Diagram



Fields

Field	Function
31-18 —	Reserved
17 FREQ_LT_MIN_IRQ_EN	FREQ_LT_MIN interrupt enable This bit used to enable FREQ_LT_MIN interrupt.
16 FREQ_GT_MAX_IRQ_EN	FREQ_GT_MAX interrupt enable This bit used to enable FREQ_GT_MAX interrupt.
15-2 —	Reserved
1-0 REF_CLK_SEL	Reference clock select These bits control which SoC clock is connected to the reference clock. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
RF_SFA	CTRL2[0]	CTRL2[1]
SFA0	CTRL2	—

28.4.1.12 Record the low limit reference clock count (REF_LOW_LIMIT_CNT)

Offset

Register	Offset
REF_LOW_LIMIT_CNT	28h

Function
Record the low limit reference clock count.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REF_LOW_LIMIT_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	REF_LOW_LIMIT_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 REF_LOW_LIMIT_CNT	Low limit reference clock count value Low limit reference clock count value.If reference clock counter number in measure period is less than this value.Then FREQ_GT_MAX interrupt will generate if according enable bit is set.

28.4.1.13 This register record the low limit of ref clk counter (REF_HIGH_LIMIT_CNT)

Offset

Register	Offset
REF_HIGH_LIMIT_CNT	2Ch

Function

This register record the low limit of ref clk counter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REF_HIGH_LIMIT_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	REF_HIGH_LIMIT_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 REF_HIGH_LIMIT_CNT	High limit reference clock count value High limit reference clock count value.If reference clock counter number in measure period is greater than this value.Then FREQ_LT_MIN interrupt will generate if according enable bit is set.

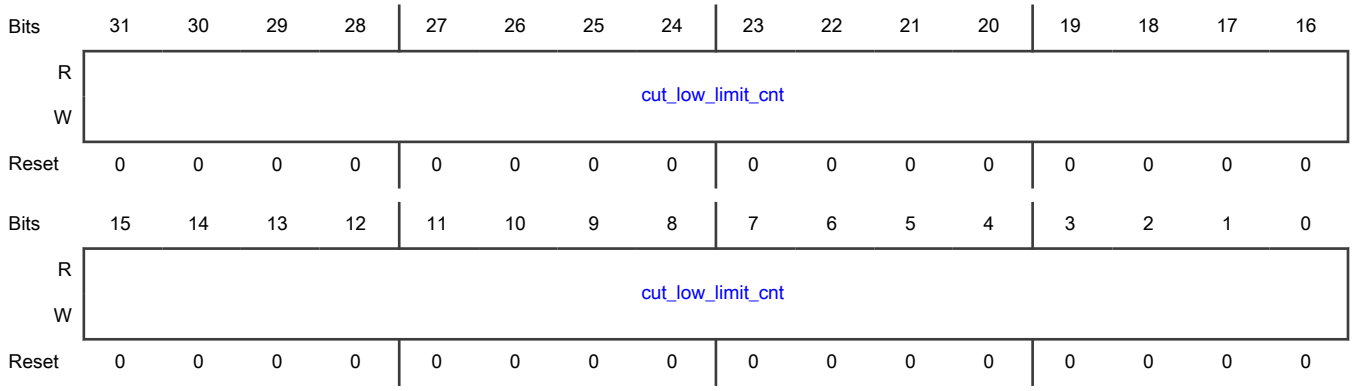
28.4.1.14 Record the CUT clock low limit counter (CUT_LOW_LIMIT_CNT)

Offset

Register	Offset
CUT_LOW_LIMIT_CNT	30h

Function
Record the CUT clock low limit counter.This register is only used when CTRL.MODE=0x01

Diagram



Fields

Field	Function
31-0 cut_low_limit_cnt	Low limit cut clock count value Low limit cut clock count value.If cut clock counter number in measure period is less than this value.Then FREQ_LT_MIN interrupt will generate if according enable bit is set.

28.4.1.15 Record high limit count of cut clock (CUT_HIGH_LIMIT_CNT)

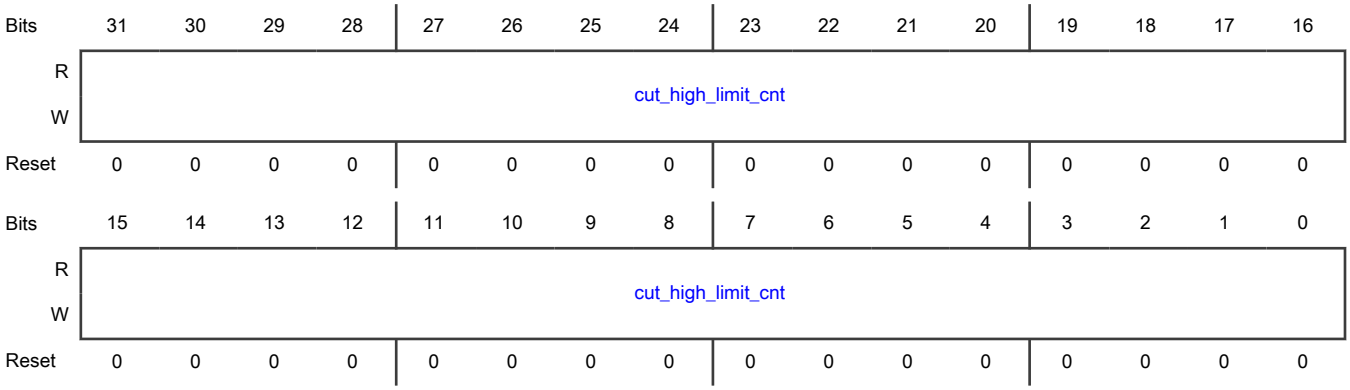
Offset

Register	Offset
CUT_HIGH_LIMIT_CNT	34h

Function

Record high limit count of cut clock. This register is only used when CTRL.MODE=0x01

Diagram



Fields

Field	Function
31-0 cut_high_limit_cnt	High limit cut clock count value High limit cut clock count value.If cut clock counter number in measure period is greater than this value.Then FREQ_GT_MAX interrupt will generate if according enable bit is set.

Chapter 29
32 kHz Clock Control Module (CCM32K)

29.1 Chip-specific CCM32K information

Table 204. Reference links to related information

Topic	Related module	Reference
Full description	CCM32K	CCM32K
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

29.1.1 Module instances

This device has one instance of the CCM32K module.

29.1.2 Reset

The CCM32K module supports only the POR.

29.2 Overview

The 32 kHz Clock Control Module (CCM32K) provides the 32 kHz clock to the device. This module includes the free running oscillator (FRO) and Crystal Oscillator (OSC) which serve as the clock generators of 32 kHz clock. The module has an internal clock validation mechanism to ensure the clock from OSC is stable; and switch from the FRO to OSC when clock from OSC is stable or switch back to FRO clock if software disables the OSC.

29.2.1 Block diagram

The following figure shows a block diagram for the CCM32K module.

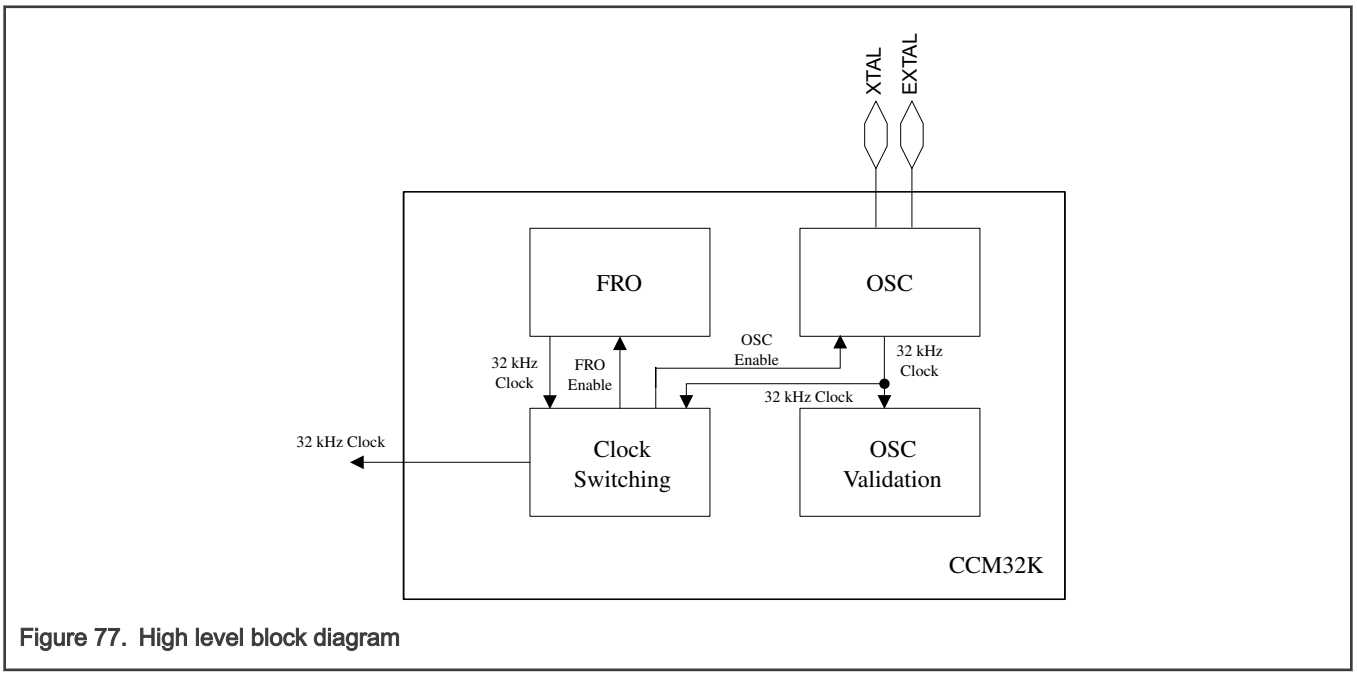


Figure 77. High level block diagram

29.2.2 Features

The CCM32K module has the following features:

- 32-kHz Free Running Oscillator (FRO) that is enabled after POR to provide clock to the device. The FRO can be disabled by software if needed.
- FRO trim values are loaded from IFR which can be overriden by software if needed.
- 32-kHz Crystal Oscillator (OSC) that is kept disabled after POR. The OSC can be enabled by software when needed.
- Clock validation logic that checks for the quality of clock from OSC and indicates if the clock from the OSC is stable.
- Clock switching logic that switches to OSC clock when it is stable and switches back to FRO clock when software disables the OSC.
- Locking mechanism for all registers for OSC and FRO. Once locked, the registers cannot be unlocked until the next POR. This prevents unwanted modifications to the registers.
- The loss of clock (LOC) monitoring for the external 32-khz crystal oscillator with internal 32-khz FRO as reference clock.

29.3 Functional description

The following sections describe functional details of the CCM32K module.

29.3.1 OSC clock validation

The clock from the external crystal oscillator is not stable immediately after enabling it and takes certain time before it becomes stable and can be used. The clock validation mechanism ensures that sufficient amount of clock cycles have passed before enabling the clock to other modules on the device that require 32 kHz clock. When STATUS[OSC32K_ACTIVE] and STATUS[OSC32K_RDY] are asserted, 32 kHz oscillator clock can be used by other modules.

29.3.2 Clock switching

Clock switching between OSC32K and FRO32K

- Switch FRO32K to OSC32K by configuring CGC32K[CLK_SEL_32K] to 1b if OSC32K clock is available.
- Switch OSC32K to FRO32K by configuring CGC32K[CLK_SEL_32K] to 0b if FRO32K clock is available.
- No clock source active if CGC32K[CLK_SEL_32K] is high and OSC32K is disabled.
- No clock source active if CGC32K[CLK_SEL_32K] is low and FRO32K is disabled.

29.3.3 Clock monitor

CCM32K has two clock monitors. One uses FRO_ 32K as its clock source. The other uses OSC_32K as its clock source. Each monitor checks a divided version of the other clock at the same frequency but divided by 16. The monitor expects a clock edge every 8 cycles, and you can trim the monitor to assert after 10, 12, 14, or 16 cycles with no edge.

Changing the trim makes the clock monitor more sensitive and reduces the latency, but also makes it more susceptible to variation in the duty cycle. The accuracy and latency of the monitor can vary by a cycle due to synchronization differences, so for each trim point:

- At 2 cycles after the expected edge the monitor asserts between 12.5% and 37.5% frequency variation with a maximum latency of 11 cycles
- At 4 cycles after the expected edge the monitor asserts between 37.5% and 62.5% frequency variation with a maximum latency of 13 cycles
- At 6 cycles after the expected edge the monitor asserts between 62.5% and 87.5% frequency variation with a maximum latency of 15 cycles
- At 8 cycles after the expected edge the monitor asserts between 87.5% and 112.5% frequency variation with a maximum latency of 17 cycles

The frequency of the clock monitor can also be trimmed to either 1 kHz or 64 Hz. Configuring to a slower clock increases the detector's latency but reduces power consumption.

NOTE

You must enable both the FR032k and OSC32k and OSC_RDY sets before enabling the clock monitor.

29.4 External signals

This table describes the CCM32K signals that are SoC level pins.

Signal	I/O	Description
EXTAL32K	I/O	32-kHz Crystal pin
XTAL32K	I/O	32-kHz Crystal pin

29.5 Initialization

This section describes the initialization of the CCM32K module.

The CCM32K module powers up, loads the default trims from flash and starts up the FRO after Power-On-Reset (POR) is complete. The trim values get loaded after POR into their respective registers. Software must follow the steps below to further configure the module.

- FRO32K as clock source
  1. Enable the FRO32K by setting FRO32K_CTRL[FRO_EN] as 1b. The FRO32K_CTRL[FRO_EN] is reset to 1b by default.
  2. Wait until the STATUS[FRO32K_ACTIVE] is 1b to identify FRO32K available.
  3. Clear CGC32K[CLK_SEL_32K] to be 0b to select FRO32K as clock source. The CGC32K[CLK_SEL_32K] is reset to 0b by default.
- OSC32K as clock source
  1. Set OSC32K_CTRL[OSC_EN] and OSC32K_CTRL[SOX_EN] as 1b,
  2. Select internal capacitance bank value through OSC32K_CTRL[XTAL_CAP_SEL] and OSC32K_CTRL[EXTAL_CAP_SEL] if internal capacitance back is enabled (OSC32K_CTRL[CAP_SEL_EN]).
  3. Wait until theSTATUS[OSC32K_RDY] is 1b to identify OSC32K available.
  4. Set CGC32K[CLK_SEL_32K] as 1b to select OSC32K as clock source.
- OSC32K bypass clock as clock source
  1. Set OSC32K_CTRL[OSC_EN] and OSC32K_CTRL[OSC_BYP_EN] high.
  2. Wait until theSTATUS[OSC32K_RDY] is 1b to identify OSC32K available.
  3. Set CGC32K[CLK_SEL_32K] high , then OSC32K bypass clock is selected as clock source. This clock is treated as same clock on EXTAL32 pad.

NOTE

OSC32K_ACTIVE or FRO32_ACTIVE will be asserted only when related oscillator is selected as clock source and CGC32K[CLK_OE_32K] is not all 0.

29.6 Memory Map and register definition

This section includes the CCM32K module memory map and detailed descriptions of all registers.

29.6.1 CCM32K register descriptions

29.6.1.1 CCM32K memory map

CCM32K base address: 4001_F000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Free Running 32 kHz Oscillator Control Register (FRO32K_CTRL)	32	RW	0088_0001h
4h	Free Running 32 kHz Oscillator Trim Register (FRO32K_TRIM)	32	RW	0000_0400h
8h	32 kHz OSC Control Register (OSC32K_CTRL)	32	RW	See section
Ch	Status Register (STATUS)	32	R	0000_0010h
14h	Clock Monitor Control Register (CLKMON_CTRL)	32	RW	0000_0000h
18h	Clock Monitor Test Register (CLKMON_TST)	32	RW	0000_0000h
1Ch	32 kHz Clock Gate Control Register (CGC32K)	32	RW	0000_001Fh

29.6.1.2 Free Running 32 kHz Oscillator Control Register (FRO32K_CTRL)

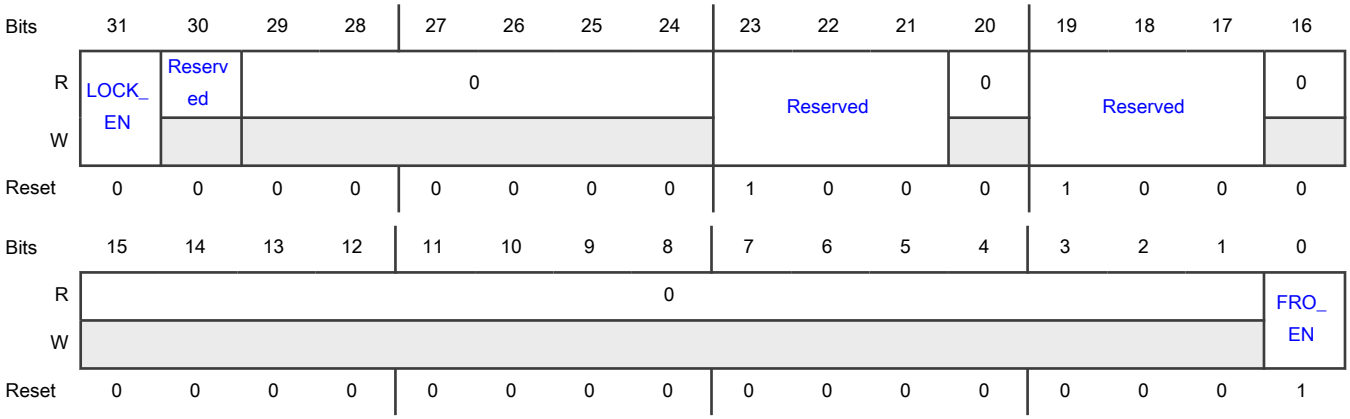
Offset

Register	Offset
FRO32K_CTRL	0h

Function

The Free Running 32 kHz Oscillator Control Register (FRO32K_CTRL) provides control fields for the 32 kHz Free Running Oscillator.

Diagram



Fields

Field	Function
31 LOCK_EN	Write Access Lock When set to '1', this field locks all further write accesses to this register until a POR occurs. Writing '0' after setting this field to '1' has no effect. 0b - Register write access is unlocked 1b - Register write access is locked
30 —	Reserved bits. Reads only zeros.
29-24 —	Reserved bits. Reads only zeros.
23-21 —	Reserved bits. Reads only zeros. Writes have no effect but it is preferred that these bits are not written to.
20 —	Reserved bits. Reads only zeros.
19-17 —	Reserved bits. Reads only zeros. Writes have no effect but it is preferred that these bits are not written to.
16-1 —	Reserved bits. Reads only zeros.
0 FRO_EN	FRO Enable This field enables the FRO analog oscillator. There is a 120 µs typical start-up time before clocks are output from the FRO. 0b - FRO is disabled 1b - FRO is enabled

29.6.1.3 Free Running 32 kHz Oscillator Trim Register (FRO32K_TRIM)

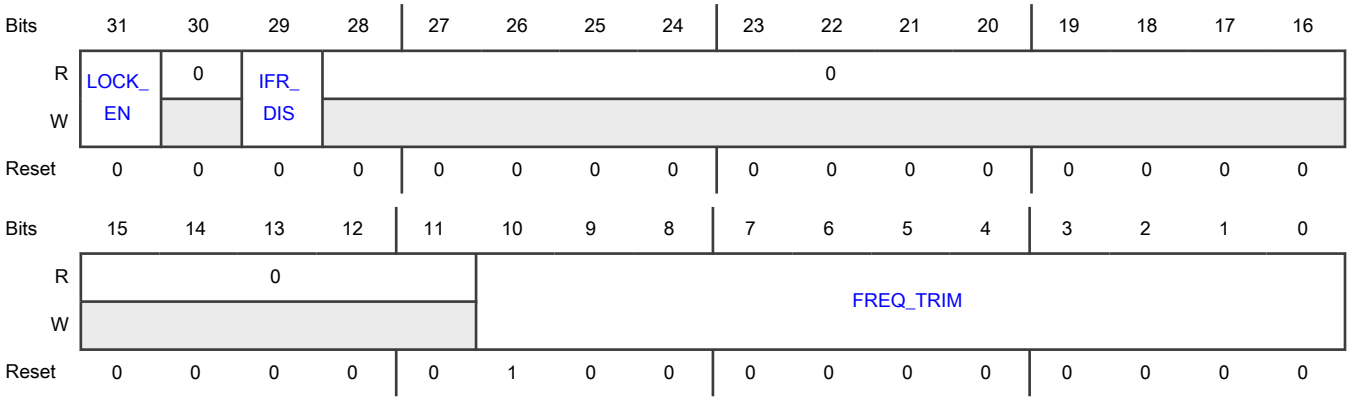
Offset

Register	Offset
FRO32K_TRIM	4h

Function

The Free Running 32 kHz Oscillator Trim Register (FRO32K_TRIM) provides frequency trim configuration for the 32 kHz Free Running Oscillator.

Diagram



Fields

Field	Function
31 LOCK_EN	Write Access Lock When set to '1', this field locks all further write accesses to this register until a POR occurs. Writing '0' after setting this field to '1' has no effect. 0b - Register write access is unlocked 1b - Register write access is locked
30 —	Reserved
29 IFR_DIS	IFR Loading Disable Control This field controls whether the FREQ_TRIM field gets overwritten when IFR value gets loaded in the SoC. Setting this field to '1' blocks the FREQ_TRIM field from being loaded again. If LOCK_EN bit is set to '1', this field is ignored and no IFR values get loaded. 0b - IFR loading is enabled 1b - IFR loading is disabled
28-11 —	Reserved
10-0 FREQ_TRIM	Frequency Trim These field controls the frequncy of the FRO. The frequency is decreased monotonically when this field is changed progressively from 00000000000b to 11111111111b. This field is set from the IFR (Flash) but can be overridden by software. This field can be changed only when the FRO32K_CTRL[FRO_EN] field is '0'. **NOTE** To ensure monotonic change in frequency in a closed loop mode, the bit 0 of this field should always be kept '0'. The IFR_DIS and LOCK_EN fields control how this field gets loaded. If IFR_DIS is '1', this field will not be loaded with IFR value. If LOCK_EN bit is '1', this field will be locked for both IFR load and software writes. 100_0000_0000b - Default trim value

29.6.1.4 32 kHz OSC Control Register (OSC32K_CTRL)

Offset

Register	Offset
OSC32K_CTRL	8h

Function

The 32 kHz OSC Control Register (OSC32K_CTRL) controls the 32 kHz Crystal Oscillator.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCK_EN	Reserved			0			SOX_EN	0		COARSE_AMP_GAIN		0	Reserved		
W																
Reset	0	u	u	u	u	0	0	0	0	0	0	0	0	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	XTAL_CAP_SEL				EXTAL_CAP_SEL				CAP_SEL...	Reserved		Reserved		0	OSC_BYP...	OSC_EN
W																
Reset	0	0	0	0	0	0	0	0	0	u	u	0	0	0	0	0

Fields

Field	Function
31 LOCK_EN	Write Access Lock bit When set to '1', this bit will lock all further write accesses to this register until a POR occurs. Writing '0' after setting this bit to '1' has no effect. 0b - Register write access is unlocked 1b - Register write access is locked
30-27 —	Reserved bits. Do not write to these bits.
26-25 —	Reserved bits. Reads only zeros.
24 SOX_EN	Crystal mode enable Enables the crystal oscillator SOX mode. 0b - Not supported for crystal mode operation 1b - Required for crystal mode operation

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-22 —	Reserved
21-20 COARSE_AMP_GAIN	Amplifier gain adjustment bits to allow the use of a wide range of external crystal ESR values. **NOTE** See the device data sheet for the ranges supported by this device. 00b - ESR_Range0 01b - ESR_Range1 10b - ESR_Range2 11b - ESR_Range3
19 —	Reserved
18-16 —	Reserved bits. Do not write to these bits.
15-12 XTAL_CAP_SEL	Crystal load capacitance selection bits Selects the internal capacitance for the XTAL pin from the capacitor bank. The final load capacitance can be calculated as $C_L = ((C_{extal} \cdot C_{xtal})/(C_{extal} + C_{xtal})) + C_{shunt}$ where: • C_{extal} is the selected capacitance on EXTAL pin • C_{xtal} is the selected capacitance on XTAL pin • C_{shunt} is the internal shunt capacitance whose value can be determined from the device's datasheet The selected capacitance value is shown below. 0000b - 0 pF 0001b - 2 pF 0010b - 4 pF 0011b - 6 pF 0100b - 8 pF 0101b - 10 pF 0110b - 12 pF 0111b - 14 pF 1000b - 16 pF 1001b - 18 pF

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1010b - 20 pF 1011b - 22 pF 1100b - 24 pF 1101b - 26 pF 1110b - 28 pF 1111b - 30 pF
11-8 EXTAL_CAP_SEL	Crystal load capacitance selection bits Selects the internal capacitance for the EXTAL pin from the capacitor bank. The final load capacitance can be calculated as $C_L = ((C_{extal} . C_{xtal})/(C_{extal} + C_{xtal})) + C_{shunt}$ where: • C_{extal} is the selected capacitance on EXTAL pin • C_{xtal} is the selected capacitance on XTAL pin • C_{shunt} is the internal shunt capacitance whose value can be determined from the device's datasheet The selected capacitance value is shown below. 0000b - Invalid configuration 0001b - 2 pF 0010b - 4 pF 0011b - 6 pF 0100b - 8 pF 0101b - 10 pF 0110b - 12 pF 0111b - 14 pF 1000b - 16 pF 1001b - 18 pF 1010b - 20 pF 1011b - 22 pF 1100b - 24 pF 1101b - 26 pF 1110b - 28 pF 1111b - 30 pF
7 CAP_SEL_EN	Crystal Load Capacitance Selection Enable Enables the internal capacitor banks on the EXTAL and XTAL pins that vary the load capacitance on these pins.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Internal capacitance bank is not enabled 1b - Internal capacitance bank is enabled
6-5 —	Reserved bits. Do not write to these bits.
4-3 —	Reserved bits. Reads only zeros. Writes have no effect but it is preferred that these bits are not written to.
2 —	Reserved bits. Reads only zeros.
1 OSC_BYP_EN	Crystal Oscillator Bypass Enable When this field is set to 1b, the crystal oscillator will be bypassed, and the output of the crystal oscillator is the same as that of the external oscillator from the EXTAL pin. Both OSC_EN and this bit must be set together to 1b to enable the bypass mode of the oscillator. To exit bypass mode, both OSC_EN and OSC_BYP_EN bits must be set to '0'. User must not move the OSC from bypass mode directly to normal mode. 0b - Crystal oscillator is not bypassed 1b - Crystal oscillator is bypassed
0 OSC_EN	Crystal Oscillator Enable This field enables the crystal oscillator when set to '1'. The oscillator will start giving the clock output but the clock is not available to the device until the STATUS[OSC_RDY] and STATUS[OSC_ACTIVE] fields are asserted. 0b - Oscillator is disabled 1b - Oscillator is enabled

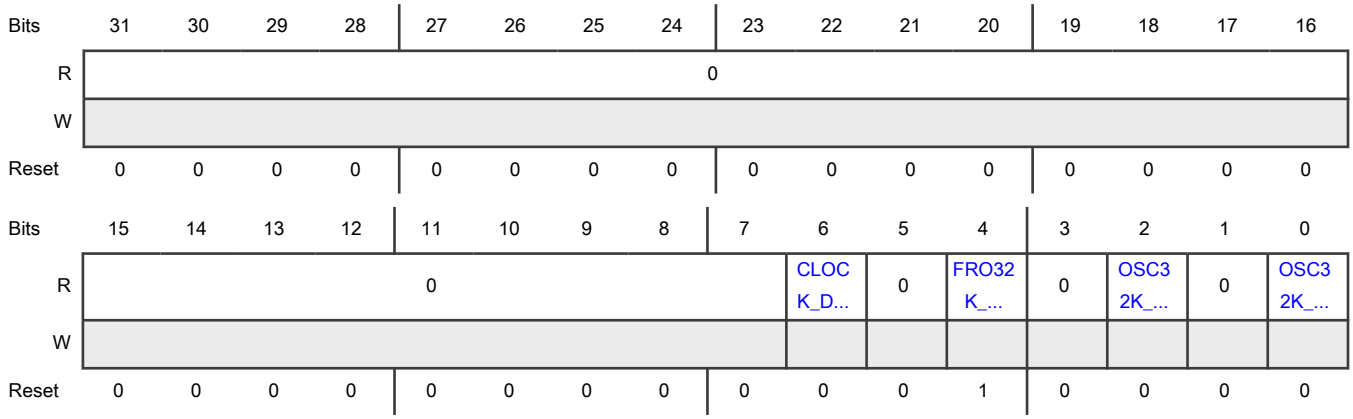
29.6.1.5 Status Register (STATUS)

Offset

Register	Offset
STATUS	Ch

Function
This register provides status bits for the CCM32K module.

Diagram



Fields

Field	Function
31-7 —	Reserved bits. Reads only zeros.
6 CLOCK_DET	Clock Detect The Clock monitor has detected an error. 0b - Clock error is not detected 1b - Clock error is detected
5 —	Reserved bits. Reads only zeros.
4 FRO32K_ACTIVE	32 kHz FRO active bit Indicates whether the 32 kHz free running oscillator is active and in use as the clock source or not. 0b - FRO32K is not the active clock source 1b - FRO32K is the active clock source
3 —	Reserved bits. Reads only zeros.
2 OSC32K_ACTIVE	32 kHz Oscillator active bit Indicates whether the 32 kHz crystal oscillator is active and in use as the clock source or not. 0b - OSC32K is not the active clock source 1b - OSC32K is the active clock source
1 —	Reserved bits. Reads only zeros.
0	32 kHz Oscillator ready bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
OSC32K_RDY	A '1' on this bit indicates that clock output from the crystal oscillator is stable. The software can decide to switch to the OSC32K clock when this bit is set. However, this bit does not clear when due to any reason the OSC32K stops providing a clock. Other on-chip tamper detection methods might indicate if clock has stopped due to removal of crystal or other reasons. This bit de-asserts during a POR event OR if the OSC32K_CTRL[OSC_EN] bit set to 0. 0b - Clock output from crystal oscillator is not stable. 1b - Clock output from crystal oscillator is stable.

29.6.1.6 Clock Monitor Control Register (CLKMON_CTRL)

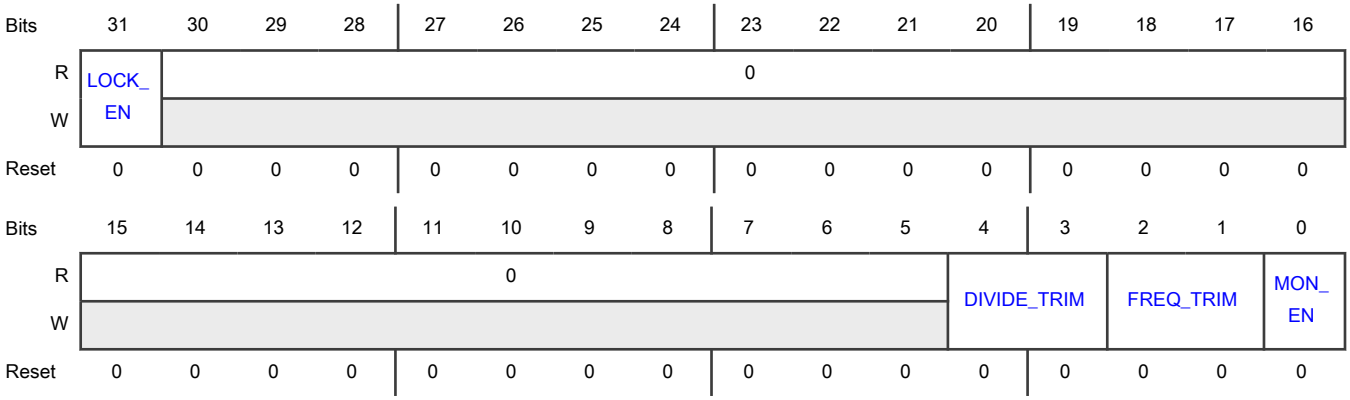
Offset

Register	Offset
CLKMON_CTRL	14h

Function

Provides control bits for clock monitor.

Diagram



Fields

Field	Function
31 LOCK_EN	Write Access Lock bit When set to '1', this bit will lock all further write accesses to this register until a POR occurs. Writing '0' after setting this bit to '1' has no effect. 0b - Register write access is unlocked 1b - Register write access is locked

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-5 —	Reserved bits. Reads only zeros.
4-3 DIVIDE_TRIM	Divide Trim Clock monitor divide trim 00b - Clock monitor operates at 1 kHz for both FRO32K and OSC32K 01b - Clock monitor operates at 64 Hz for FRO32K and clock monitor operates at 1 kHz for OSC32K (Reserved) 10b - Clock monitor operates at 1 kHz for FRO32K and clock monitor operates at 64 Hz for OSC32K (Reserved) 11b - Clock monitor operates at 64 Hz for both FRO32K and OSC32K
2-1 FREQ_TRIM	Frequency trim bits Clock monitor frequency trim 00b - Clock monitor asserts 2 cycle after expected edge (assert after 10 cycles with no edge) 01b - Clock monitor asserts 4 cycles after expected edge (assert after 12 cycles with no edge) 10b - Clock monitor asserts 6 cycles after expected edge (assert after 14 cycles with no edge) 11b - Clock monitor asserts 8 cycles after expected edge (assert after 16 cycles with no edge)
0 MON_EN	CLKMON Enable Enables the CLKMON when set to '1'. 0b - CLKMON is disabled 1b - CLKMON is enabled

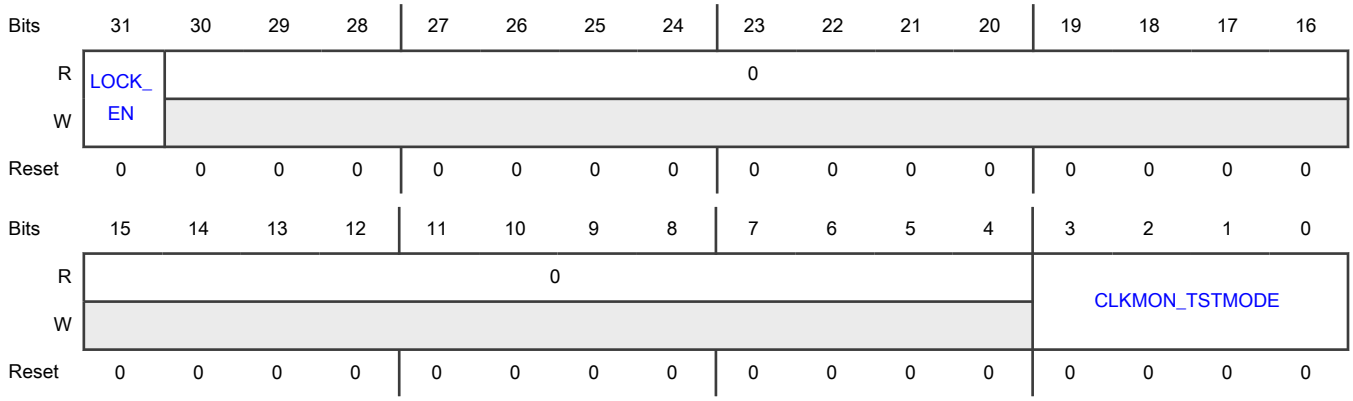
29.6.1.7 Clock Monitor Test Register (CLKMON_TST)

Offset

Register	Offset
CLKMON_TST	18h

Function
Provides control bits for clock monitor test.

Diagram



Fields

Field	Function
31 LOCK_EN	Write Access Lock bit When set to '1', this bit will lock all further write accesses to this register until a POR occurs. Writing '0' after setting this bit to '1' has no effect. 0b - Register write access is unlocked 1b - Register write access is locked
30-4 —	Reserved bits. Reads only zeros.
3-0 CLKMON_TSTMODE	Test Mode Configure the CLKMON test mode. Fields '0' and '1' indicate the reference clock and enable signal respectively after entering test mode. Moreover, fields '2' and '3' are for monitored clock part. When set to '0000' following '0101' after resetting CLKMON_CTRL [MON_EN], these fields can completely turn off clock monitor.

29.6.1.8 32 kHz Clock Gate Control Register (CGC32K)

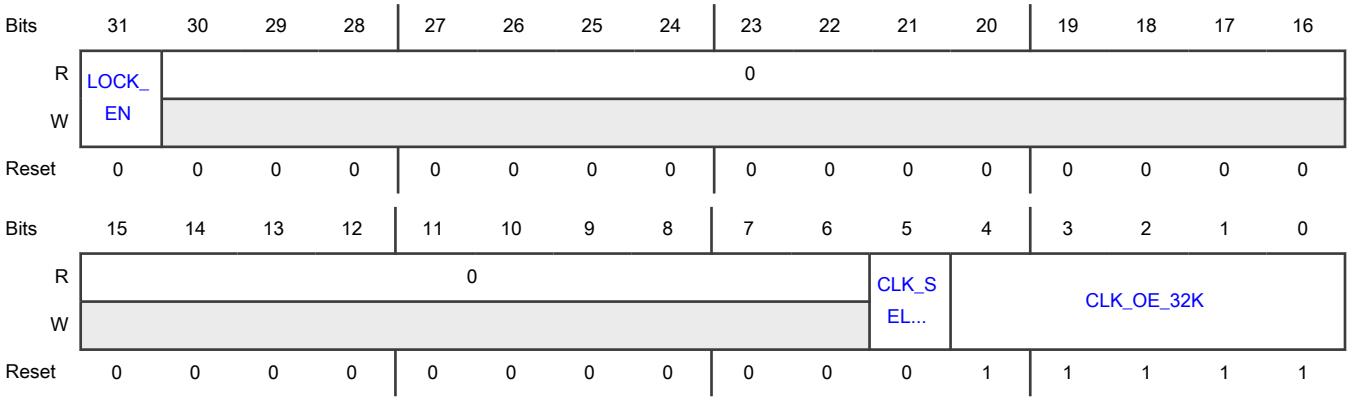
Offset

Register	Offset
CGC32K	1Ch

Function

Provides 32 kHz clock output gate control bits for peripherals.

Diagram



Fields

Field	Function
31 LOCK_EN	Write Access Lock bit When set to '1', this bit will lock all further write accesses to this register until a POR occurs. Writing '0' after setting this bit to '1' has no effect. 0b - Register write access is unlocked 1b - Register write access is locked
30-6 —	Reserved bits. Reads only zeros.
5 CLK_SEL_32K	32 kHz clock source selection bit 0b - FRO32K clock output is selected as clock source 1b - OSC32K clock output is selected as clock source
4-0 CLK_OE_32K	32 kHz clock output enable bits Each bit is used by different peripherals. • CLK_OE_32K [0]: RTC • CLK_OE_32K [1]: RFMC • CLK_OE_32K [2]: NBU • CLK_OE_32K [3]: WUU/RMC/PORTD • CLK_OE_32K [4]: Other modules 0_0000b - Clock output is disabled 0_0001b - Clock output is enabled

Chapter 30
Power Management

30.1 Introduction

This chapter describes the power modes supported with this device. It also describes the power domains and the voltage supply options connected to these power domains.

NOTE

Refer to Radio low power mode for power management of radio sub-system.

30.2 Power domains

A power domain is a collection of circuits that can be powered off even when a voltage source is still supplied. This section shows the power domains within the device and the modules belong to these domains.

NOTE

If a module belongs to several power domains, the module will be powered off when any of the power domain is powered off.

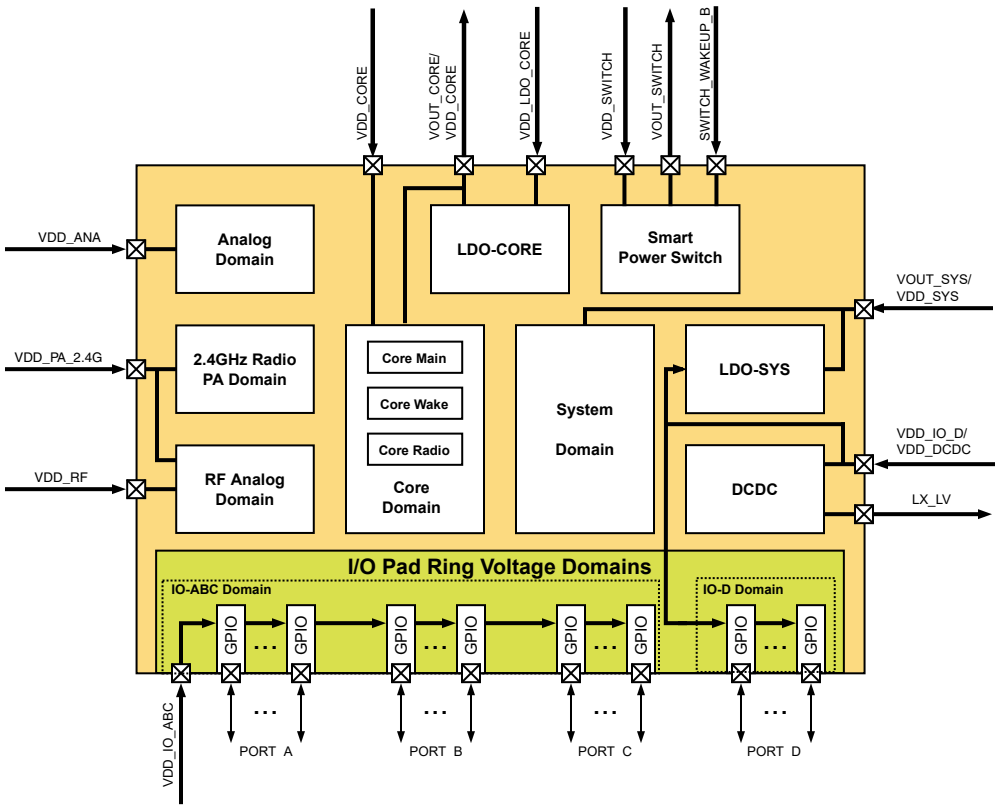


Figure 78. Power domains and voltage supplies

Table 205 lists all the power domains in the figure, and Table 206 lists all the regulators in the figure. Refer to Smart power switch for details of the smart power switch.

Table 205. Peripheral power domain assignments

Power domain	Voltage supply	Module
Analog Domain	VDD_ANA VSS_ANA	ADC0
		VREF0
Core Main	VDD_CORE_MAIN	CM33
		DSP
		FPU
		MPU
		NVIC
		SYSTICK
		TZM
		DAP
		DWT
		ITM
		TPIU
		AXBS0
		eDMA0
		MSCM
		SMSCM
		PBRIDGE2
		TRGMUX
		MRCC
		SFA0
		CACHE-CODE
		FLASH
		ROM-BOOT
		CRC0
		EdgeLock Secure Enclave
		TRDC0
		LPIT0
		TPM1
		FlexIO0
		LPI2C 1
		I3C0

Table continues on the next page...

Table 205. Peripheral power domain assignments (continued)

Power domain	Voltage supply	Module
		LPSPI 1
		LPUART 1
		GPIOB/C
		SEMA42
		PORTB/C
		ADC0
		VREF0
		TCM-SYS
CORE Wake	VDD_CORE_WAKE	SWD
		CMC0
		EWM0
		WDOG0/1
		FRO-6M
		WRCC
		SCG0
		TSTMR0
		TPM0
		LPI2C0
		LPSPI0
		LPUART0
		GPIOA
		PORTA
		LPCMP0, LPCMP1
Core Radio	VDD_CORE_2.4G	RF-2.4G
		NBU
		RF-FMU
		RF-FRO192M
IO-ABC	VDD_IO_ABC	PORTA
		PORTB
		PORTC
		LPCMP0, LPCMP1
IO-D	VDD_IO_D	LDO-SYS

Table continues on the next page...

Table 205. Peripheral power domain assignments (continued)

Power domain	Voltage supply	Module
		PORTD
Smart Power Switch	VDD_SWITCH	Power switch
		RAM LDO
		FRO16K
		Power Switch Controller
System	VDD_SYSmorerows="38"	RFMC0
		Bluetooth LE LL(from NBU)
		WUU0
		FRO192M
		LPTMR0, LPTMR1
		SPC0
		OSC-RTC
		FRO32K
		REGFILE0/1 -RTC
		TAMPER
		RTC0
		GPIOD
		PORTD
RF Analog	VDD_RF	OSC-RF
		RF-2.4G
2.4GHz Radio PA	VPA_2P4GHZ	RF-PA-2.4G

NOTE

VDD_CORE_MAIN, VDD_CORE_WAKE and VDD_CORE_2.4G connect to the VDD_CORE.

Table 206. On-chip regulator

On chip regulator	Voltage supply	Descriptions
DCDC	VDD_DCDC	DCDC provides the best power efficiency.
LDO-CORE	VDD_LDO_CORE	LDO-CORE provides the lower cost option.
LDO-SYS	VDD_IO_D	—

30.3 Power modes

The device supports Active, Sleep, Deep Sleep, Power Down, and Deep Power Down modes. Any reset brings the chip back to the Active mode.

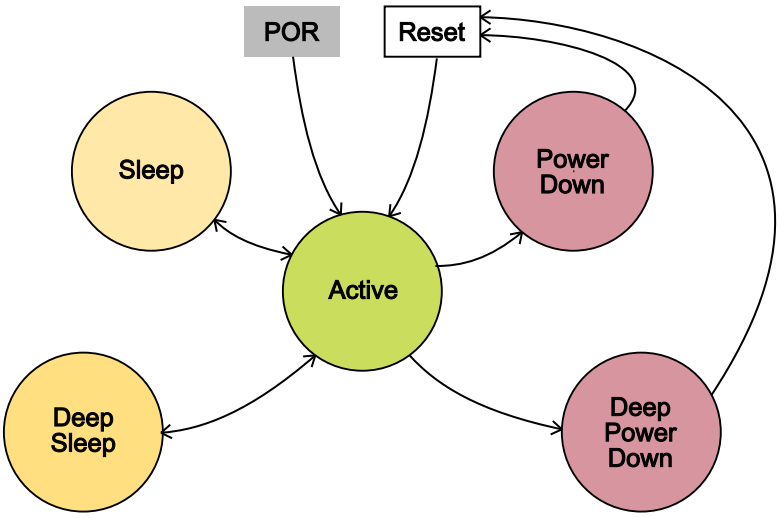


Figure 79. Power modes

30.3.1 Active mode

In Active mode, CPU execution is possible.

To achieve the performance requirements of a given system application while minimizing power consumption, Active mode permits the following power-saving options when possible:

- Configure the VDD_CORE subdomains voltage level to balance power and performance.
- Individually configure SRAM as active/shutdown by software.
- Gate the clocks to unused modules.
- Configure the unused VDD_CORE subdomains to state retention mode.
- Power-gate the unused VDD_CORE subdomains.

30.3.2 Sleep mode

In Sleep mode, CPU execution is halted. The core clock is gated off. The system clock as well as the bus clocks, if enabled, continue to operate.

Sleep mode supports the following behaviors based on different clock configurations:

- CPU clock is OFF. The system clock and bus clock remain ON. Most modules can remain operational.
- CPU clock, system clock and bus clock are all OFF. Some modules can remain operational with low power asynchronous clock sources and serve as wake-up sources.

Wake from Sleep mode is triggered by an interrupt or a wakeup event.

Sleep mode also supports a partial wake-up where a bus master other than the CPU (eg. DMA) is recovered by a wakeup event. The device automatically re-enters sleep mode after this module finishes its task.

To achieve the performance requirements of a given system application while minimizing power consumption, Sleep mode permits the following power-saving options when possible:

- Configure the VDD_CORE subdomains voltage level to balance power and performance.
- Individually configure SRAM as active / deep sleep / shutdown by software.
- Gate the clocks to unused modules.

- Configure the unused VDD_CORE subdomains to state retention mode.
- Power-gate the unused VDD_CORE subdomains.

30.3.3 Deep Sleep mode

Deep Sleep mode places most of the chip into a static state. It is the lowest power mode that can retain all registers.

The device wakes from Deep Sleep mode through the interrupt routine or wake-up event.

The core clock, system clock and the bus clock are gated off.

In Deep Sleep mode, the whole VDD_CORE_MAIN subdomain is put into state retention mode.

In Deep Sleep mode, the peripherals inside the VDD_CORE_WAKE subdomain can remain operational using an asynchronous clock and can wake the device.

Refer to Low Power Mode Controllers for peripherals in the VDD_CORE_2.4G.

External signals via GPIO or VDD_SYS domain peripherals can wake the device.

The VDD_CORE subdomains voltage level can be independently selected from either the VDD_CORE voltage or the reduced voltage scaling (IVS) voltage to balance different module clock frequencies and the power leakage.

SRAM can also be individually configured as deep sleep or shutdown by software.

30.3.4 Power Down mode

Power Down mode places the chip in power off state.

The device wakes from Power Down mode through the reset routine.

In Power Down mode, the whole VDD_CORE_MAIN subdomain is power gated.

In Power Down mode, the peripherals inside the VDD_CORE_WAKE subdomain can optionally remain operational using an asynchronous clock and can wake the device.

Refer to Low Power Mode Controllers for peripherals in the VDD_CORE_2.4G.

External signals via GPIO, VDD_SYS domain peripherals, or the RTC power domain peripherals can wake the device.

SRAM can also be individually configured as retention or off by software.

30.3.5 Deep Power Down mode

Deep Power Down mode permits the lowest power level.

The device wakes from Deep Power Down mode through the Reset routine.

In Deep Power Down mode, the whole VDD_CORE domain (including all subdomains) is power gated.

In Deep Power Down mode, the on-chip regulator for VDD_CORE is powered off. The VDD_SYS on-chip regulator is powered on , and the VDD_SYS power domain is enabled. The external power supply to any of the on-chip regulators that are disabled should be turned off to avoid extra leakage.

External reset or the VDD_SYS domain peripherals can wake the device.

SRAMs without RAM LDO supplied cannot be retained in Deep Power Down mode.

30.4 Module operation in low power modes

The following table shows module functionality in low power modes.

Table 207. Cortex M33 core module operation in low power modes

Module	Sleep	Deep Sleep	Power Down	Deep Power Down
Core modules				
Cortex M33	Static	Static	OFF	OFF
NVIC	Optional	Static	OFF	OFF
WIC	ON	Static	OFF	OFF
Cache	Static	Static	OFF	OFF
System modules				
AXBSx	Optional	Static	OFF	OFF
eDMAx	Optional	Static	OFF	OFF
WDOGx	ON	Optional	Optional	OFF
EWM	Optional	Static	Static	OFF
TRDCx	ON	Static	OFF	OFF
CMCx	ON	Optional	Optional	OFF
SPCx	ON	ON	ON	ON
WUUx	ON	ON	ON	ON
Clock modules				
FRO-192M	Optional	OFF	OFF	OFF
FRO-6M	Optional	Optional	Optional	OFF
FRO-32K	ON	ON	Optional	Optional
OSC-RTC	ON	ON	Optional	Optional
OSC-RF	ON	Optional	Optional	OFF
SCGx	ON	Static	Static	OFF
MRCC	ON	Static	OFF	OFF
Memory modules				
Flash	Optional LP	OFF/LP	OFF	OFF
SRAM	Optional	Optional	Optional	OFF
RTC Reg File	ON	ON	ON	ON
ROM	ON	OFF	OFF	OFF
Security modules				
EdgeLock Secure Enclave	ON	Static	OFF	OFF
PRINCE	ON	Static	OFF	OFF
Digital Tamper	ON	ON	ON	ON
CRCx	ON	Static	OFF	OFF

Table continues on the next page...

Table 207. Cortex M33 core module operation in low power modes (continued)

Module	Sleep	Deep Sleep	Power Down	Deep Power Down
Timer modules				
TPM0	ON	Optional	Optional	OFF
TPM1	ON	Static	OFF	OFF
LPITx	ON	Static	OFF	OFF
LPTMRx	ON	ON	ON	ON
TSTMR	ON	Optional	Optional	OFF
RTC	ON	ON	ON	ON
Communication modules				
LPUART0	ON	Optional	Optional	OFF
LPUART1	ON	Static	OFF	OFF
SEMA(or SEMA42)	ON	Static	OFF	OFF
LPSPI0	ON	Optional	Optional	OFF
LPSPI1	ON	Static	OFF	OFF
LPI2C0	ON	Optional	Optional	OFF
LPI2C1	ON	Static	OFF	OFF
I3C0	ON	Static	OFF	OFF
CAN0	ON	Static	OFF	OFF
Human interface modules				
GPIOA	ON	Optional	Optional	Static
GPIOB/C	ON	Static	Static	Static
GPIOD	ON	ON	ON	ON
Analog modules				
ADC-GPx	Optional	Static	OFF	OFF
CMP-GPx	Optional	Optional	Optional	OFF
VREF	Optional	Static	Static	OFF
Radio modules				
RF-2.4G[1]	Optional	Optional	Optional	Optional

1. See more details about 2.4 GHz radio modules in Low Power Mode Controllers.

Table 208. Power domain module operation in low power modes

Module	Sleep	Deep Sleep	Power Down	Deep Power Down
POWER Modules				
LDO-CORE	Optional LP	ON, LP or OFF	ON, LP or OFF	OFF

Table continues on the next page...

Table 208. Power domain module operation in low power modes (continued)

Module	Sleep	Deep Sleep	Power Down	Deep Power Down
LDO-SYS	Optional LP	ON, LP	ON, LP	ON, LP
DCDC	Optional LP	ON, LP or OFF	ON, LP or OFF	OFF
POR	ON	ON	ON	ON
VDD_CORE HVD / LVD	Optional	Optional	Optional	OFF
VDD_IO_ABC HVD / LVD	Optional	Optional	Optional	OFF
VDD_SYS HVD / LVD	Optional	Optional	Optional	Optional

Table 209. Power domain operation in low power modes

Domain	Active	Sleep	Deep Sleep	Power Down	Deep Power Down
VDD_CORE_MAIN	ACT	ACT	RET	OFF	OFF
VDD_CORE_WAKE	ACT	ACT	ACT/RET	ACT/RET/OFF	OFF
VDD_CORE_SRAM [1]	ACT/RET/OFF	ACT/RET/OFF	RET/OFF	RET/OFF	OFF
VDD_CORE_2P4G	ACT/RET/OFF	ACT/RET/OFF	ACT/RET/OFF	ACT/RET/OFF	OFF
VDD_CORE_2P4G_SRAM [2]	ACT/RET/OFF	ACT/RET/OFF	ACT/RET/OFF	ACT/RET/OFF	OFF
VDD_SYS	ACT	ACT	ACT	ACT	ACT

1. System SRAM is partitioned and each part is individually configurable by CMC.
2. Radio SRAM is partitioned into Tx/Rx RAM and each part is individually configurable by Radio.

Since VDD_CORE_MAIN, VDD_CORE_WAKE and VDD_CORE_2P4G are independent power domain, there are various low-power mode combinations among them. The SPC will do the arbitration on their low-power requests and select the highest low-power mode as system power mode.

From system point of view, it has several scenarios:

1. MCU in low-power mode, Radio switch between run and low-power mode (typical case for Bluetooth LE event without CPU involved)
2. Radio wakeup MCU to deal with connectivity upper layer software like Bluetooth LE Host or applications
3. MCU switch between run and low-power mode for general application, Radio in low-power mode (Radio is not enabled)

30.5 Power supply configurations

This section shows some of the most common power supply configurations.

30.5.1 Recommended configuration for tradeoff between cost and power efficiency

The following figure shows a power efficient configuration of the device power supplies.

In this configuration:

- External 3.3 V for DCDC, VDD_IO_D, VDD_IO_ABC, LDO-SYS

- LDO-CORE, VDD_RF come from DCDC output
- Generate VPA_2P4GHZ internally from VDD_RF

NOTE

VPA_2P4GHZ voltage depends on Tx power level. It can be powered internally or externally. When internal supply is used, VDD_RF voltage must be equal to or exceed VPA + 275 mV. For this specific configuration, it can support up to +10 dBm.

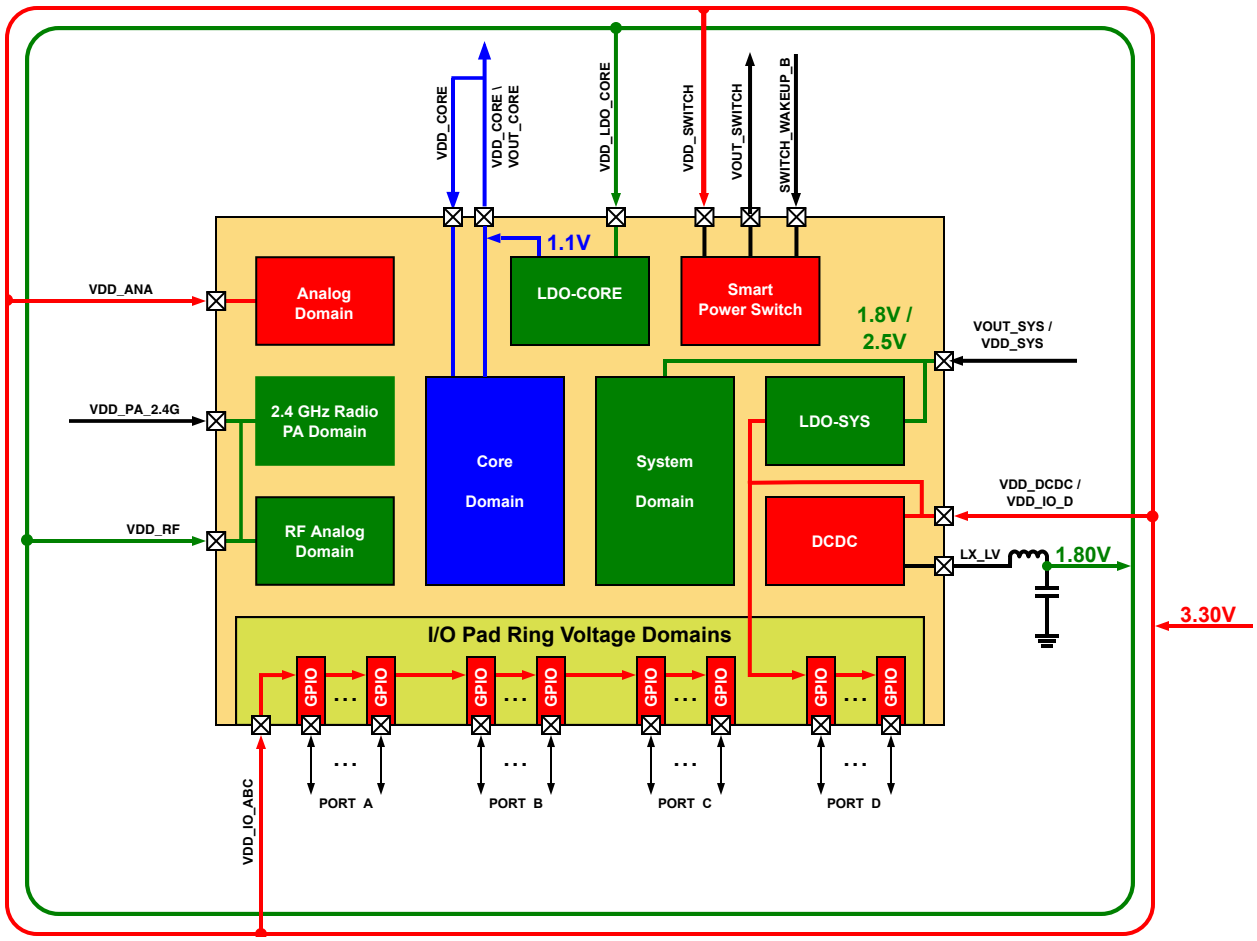


Figure 80. Power efficient schematic

30.5.2 Low cost supply configuration

The following figure shows a low cost configuration of the device power supplies.

In this configuration:

- Bypass DCDC
- External 3.3 V (range: 1.71 – 3.6 V) for LDO-CORE, LDO-SYS, VDD_IO_D, VDD_IO_ABC and VDD_RF
- Generate VPA_2PGHZ internally from VDD_RF

NOTE

VPA_2P4GHZ voltage depends on Tx power level. It can be powered internally or externally. When internal supply is used, VDD_RF voltage must be equal to or exceed VPA + 275 mV.

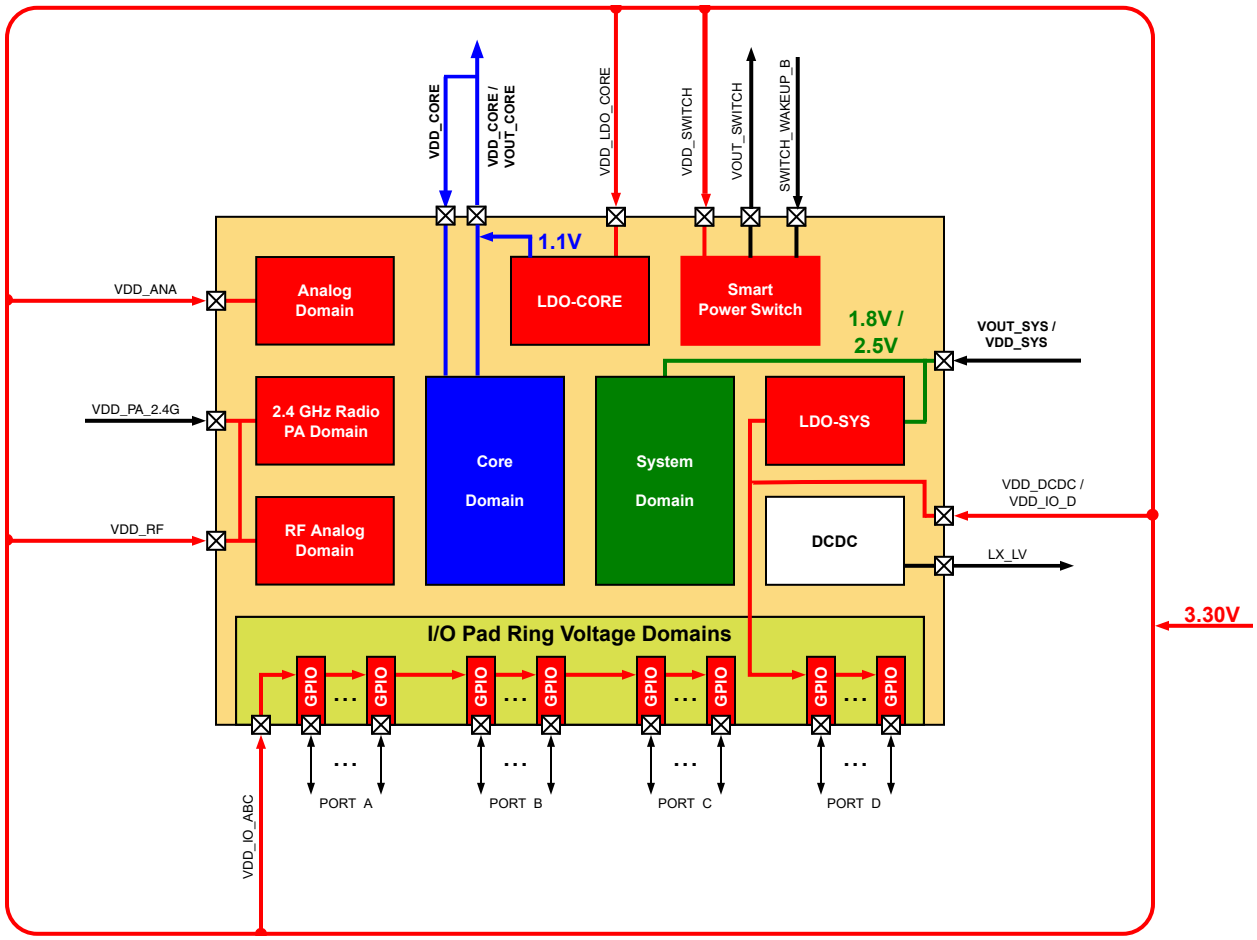


Figure 81. Low cost schematic

30.5.3 PMIC supply configuration

The following figure shows a PMIC configuration of the device power supplies.

In this configuration:

- Bypass LDO_CORE/DCDC
- External 1.2 V for VDD_CORE
- External 3.3 V for VDD_IO_D, VDD_IO_ABC
- External 1.2V for VDD_RF
- External 2.4 V for VPA_2P4GHZ

NOTE

VPA_2P4GHZ voltage depends on Tx power level. It can be powered internally or externally. When internal supply is used, VDD_RF voltage must be equal to or exceed VPA + 275 mV.

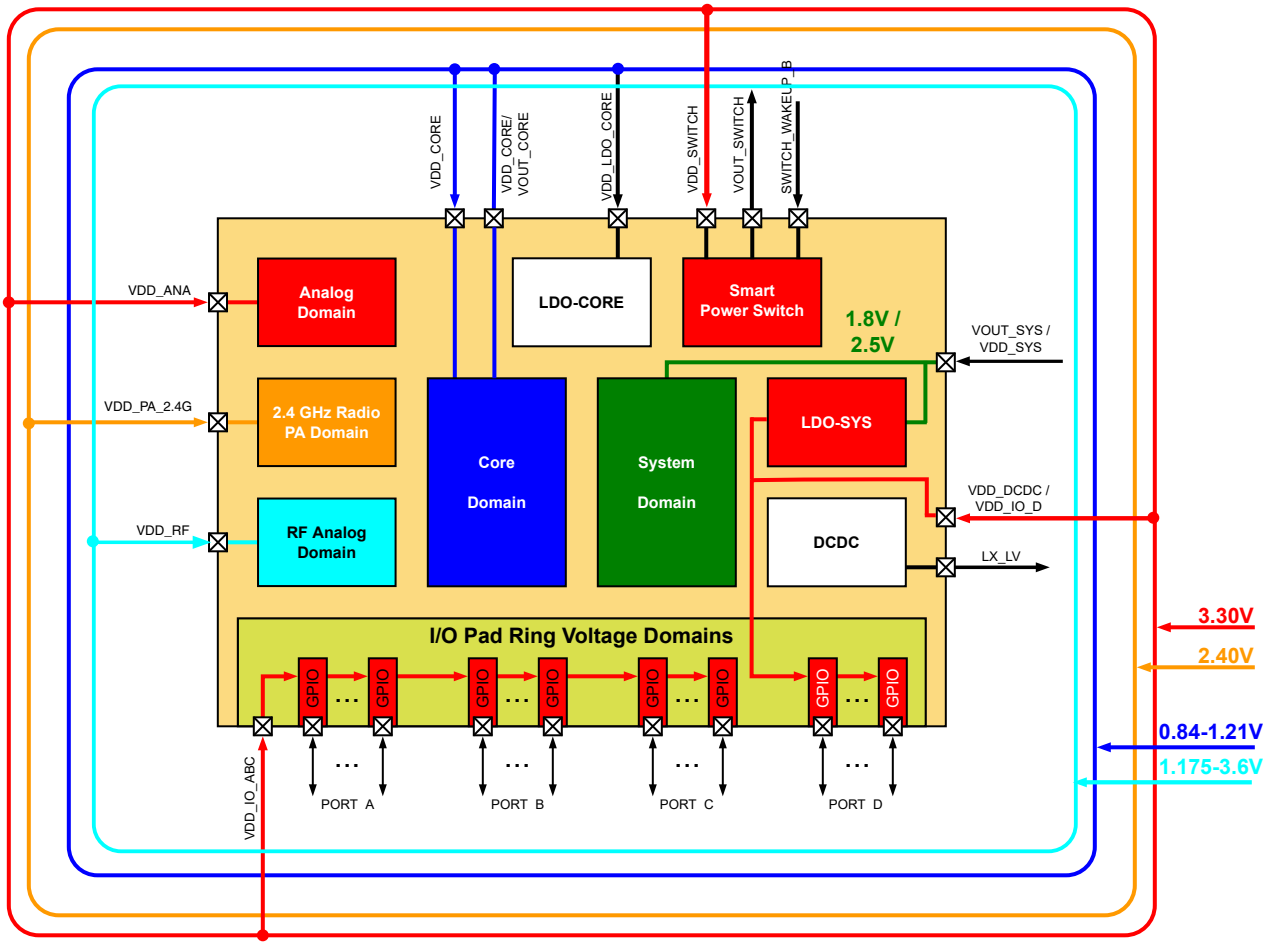


Figure 82. PMIC schematic

Additionally, LDO_SYS can also be bypassed. It requires connecting VOUT_SYS/VDD_SYS and VDD_LDO_SYS to 1.8 V and disabling the LDO_SYS at SPC_CNTRL[SYSLDO_EN]. Consider that fuse programming is not allowed while LDO_SYS is bypassed.

30.6 Power optimization

30.6.1 VDD_CORE voltage scaling

This device supports VDD_CORE voltage scaling for different performance and power consumption.

This device supports the following VDD_CORE voltage scaling methods:

- By adjusting VDD_CORE power supply input level directly with internal IVS disabled. This method is available for all power modes. The user must configure VDD_CORE voltage level through SPC registers when VDD_CORE is supplied by on-chip regulators (LDO-CORE). The device power consumption can benefit from the regulator high efficiency when DCDC is used as VDD_CORE source. However, the VDD_CORE voltage needs time to reach new levels for charging or discharging the VDD_CORE pin capacitor.
- By internal IVS (internal voltage scaling) module. This method will apply only to Deep Sleep and Power Down modes. The user should configure the target IVS output voltage level before entering low power modes. The VDD_CORE input voltage will remain unchanged. The IVS manipulates the internal supply voltage directly. The voltage will reach the new target value much faster.

Each VDD_CORE subdomain can independently enable or disable their own IVS in low power modes. For example, the VDD_CORE_MAIN subdomain can enable its IVS for a lower supply voltage. At the same time, the VDD_CORE_WAKE subdomain can continue operating with the higher supply level from VDD_CORE.

30.6.2 SRAM Configuration

The SRAM is partitioned, allowing the mode of each SRAM partition to be independently configured:

SRAM power mode	Description
Active	SRAM can be accessed normally.
Deep Sleep	SRAM cannot be accessed, but the data is retained.
Shutdown	SRAM cannot be accessed, and the data is not retained.

Depending on the requirements of the system application, software can optionally configure an individual SRAM partition to be in a lower, power-saving mode relative to the overall device power mode:

When the overall device power mode is...	An SRAM partition may be configured into this mode...
Active	Active or Shutdown
Sleep	Active or Deep Sleep or Shutdown
Deep Sleep or Power Down	Deep Sleep or Shutdown

NOTE

All SRAM partitions are forced into Shutdown mode after the device enters Deep Power Down mode.

30.6.3 Flash low power options

To balance power usage versus performance requirements, the flash module can be configured independent of the overall device power mode:

Flash power mode	Description
Full power	Supports full speed read, program and erase.
Low speed active	Supports reduced frequency read, program, and erase.

The flash module supports low-power feature by configuring CMC_FLASHCR registers. It can configure Flash in low-power mode when MCU is in Active mode or CM33 CORE is sleeping. In parallel, it provide option to automatic wakeup Flash from low-power mode when any flash access.

NOTE

To avoid unexpected access latencies, take care when configuring the flash module to a low power mode when the overall device is in Active mode. The flash module needs time to recover full-speed functionality after entering a low power mode.

30.6.4 Peripheral clock gating

To conserve power, the clocks to modules can be turned off or divided by configuring the MRCC module registers.

30.6.5 Voltage monitor optimization

Depending on the requirements of the system application, software can disable individual voltage monitors (LVD and/or HVD) within the different power domains to save power.

30.6.6 Wake-up time consideration

This device supports different low power modes for power consumption level and wake-up time balance.

The wake-up time from Sleep/Deep Sleep mode is the accumulation of following time:

- VDD_CORE power source recovery time
- The longer of the following:
  - Clock recovery time
  - Flash recovery time
- Interrupt Latency

The wake-up time from Power Down mode is the accumulation of following time:

- VDD_CORE power source recovery time
- The longer of the following:
  - Clock recovery time
  - Flash recovery time
- System Reset Latency

The wake-up time from Deep Power Down mode is:

- The boot-up time required after device released by POR. This includes everything included in a cold power-up, including the Boot ROM latency.

The clock recovery time can be affected by following factors:

- The system clock source status in low power modes. There will be no recovery time needed if the clock remains ON in low power mode.
- The system clock source before entering low power mode.

The flash recovery time depends on the flash status in low power mode.

The VDD_CORE power source recovery time can be affected by following factors:

- The regulator type used. On-chip linear regulator starts up faster than on-chip DCDC.
- The regulator status in low power modes. There will be no recovery time needed if the regulator remains in full-power regulation in low power mode.
- Whether VDD_CORE voltage scaling is enabled before and after entering low power modes
- The VDD_CORE domain voltage scaling method used. Voltage scaling by internal IVS recovers faster.
- The external capacitor on VDD_CORE pin will affect recovery time if scaling is achieved by adjusting VDD_CORE voltage directly.

30.7 Smart power switch

The smart power switch supports two major use-cases:

- It can be used to switch off all or part of MCU power supply to get lower power consumption than deep power down mode. See Figure 83 as an example.

- It can be used to power external device. For example, sensor device will be power off most of time and can keep power when measurement. Figure 80 is the example with the VOUT_SWITCH powering an external sensor, and Figure 84 is another example to power an external sensor and an I/O rail.

It has several ways to turn on power switch

- The POR of power switch input supply
- external SWITCH_WAKEUP_B pin. Once driven low, it can turn on power switch;
- internal wakeup logic from SPC if VSYS is powered ON;
- internal timer using internal FRO16K

NOTE

To support standby RAM LDO and RAM power switch, smart power switch input supply (VDD_SWITCH) is required to be powered permanently.

In Figure 83:

- External 3.3 V supply to Smart Power Switch
- Power switch output connected to DCDC, LDO-SYS, VDD_ANA, VDD_IO_D and VDD_IO_ABC
- DCDC output connected to LDO-CORE, VDD_RF

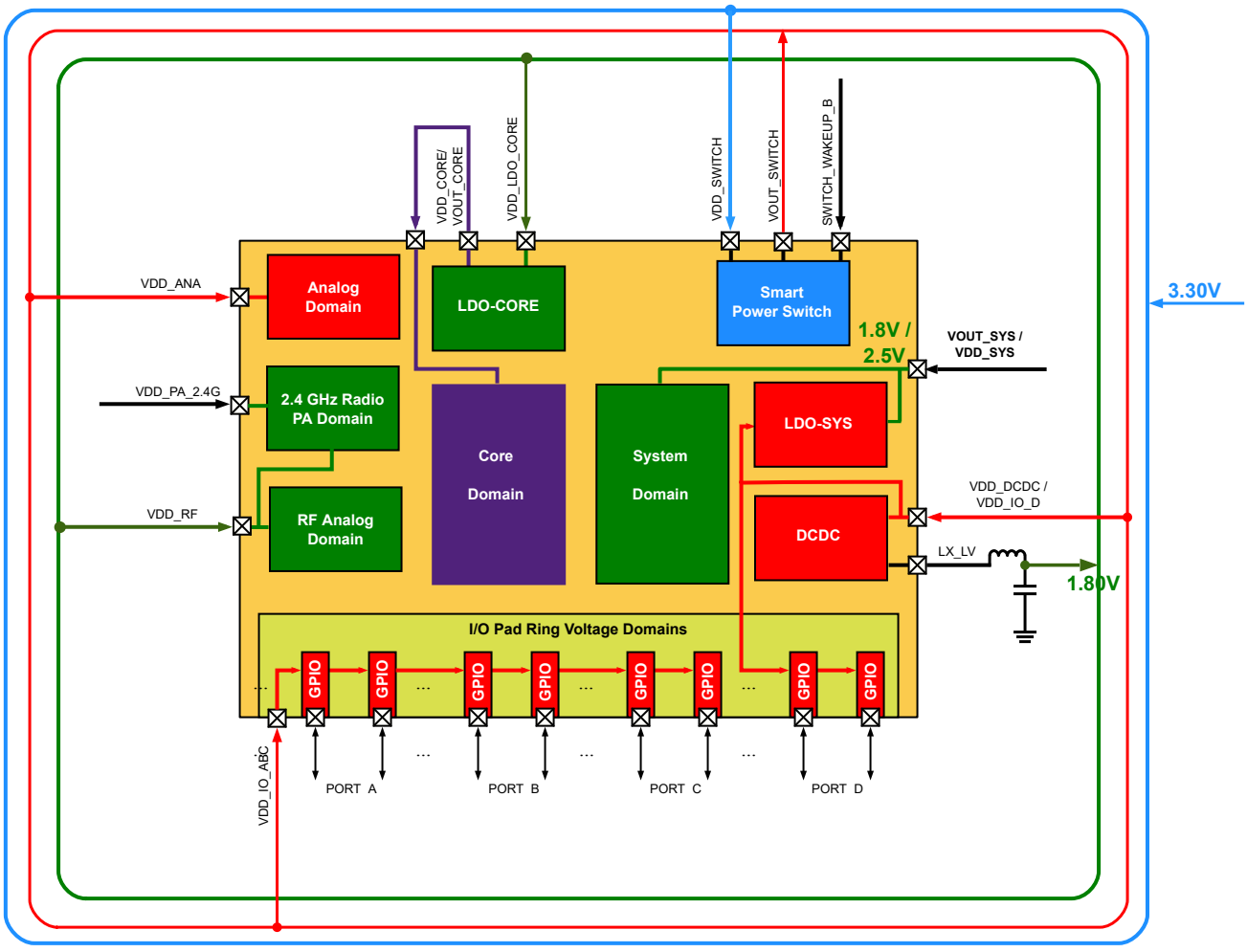


Figure 83. Power all supply

In Figure 84:

- External 3.3 V supply to Smart Power Switch and VDD_DCDC/VDD_IO_D
- Power switch output connected to VDD_ANA and VDD_IO_ABC
- DCDC output connected to LDO-CORE, VDD_RF

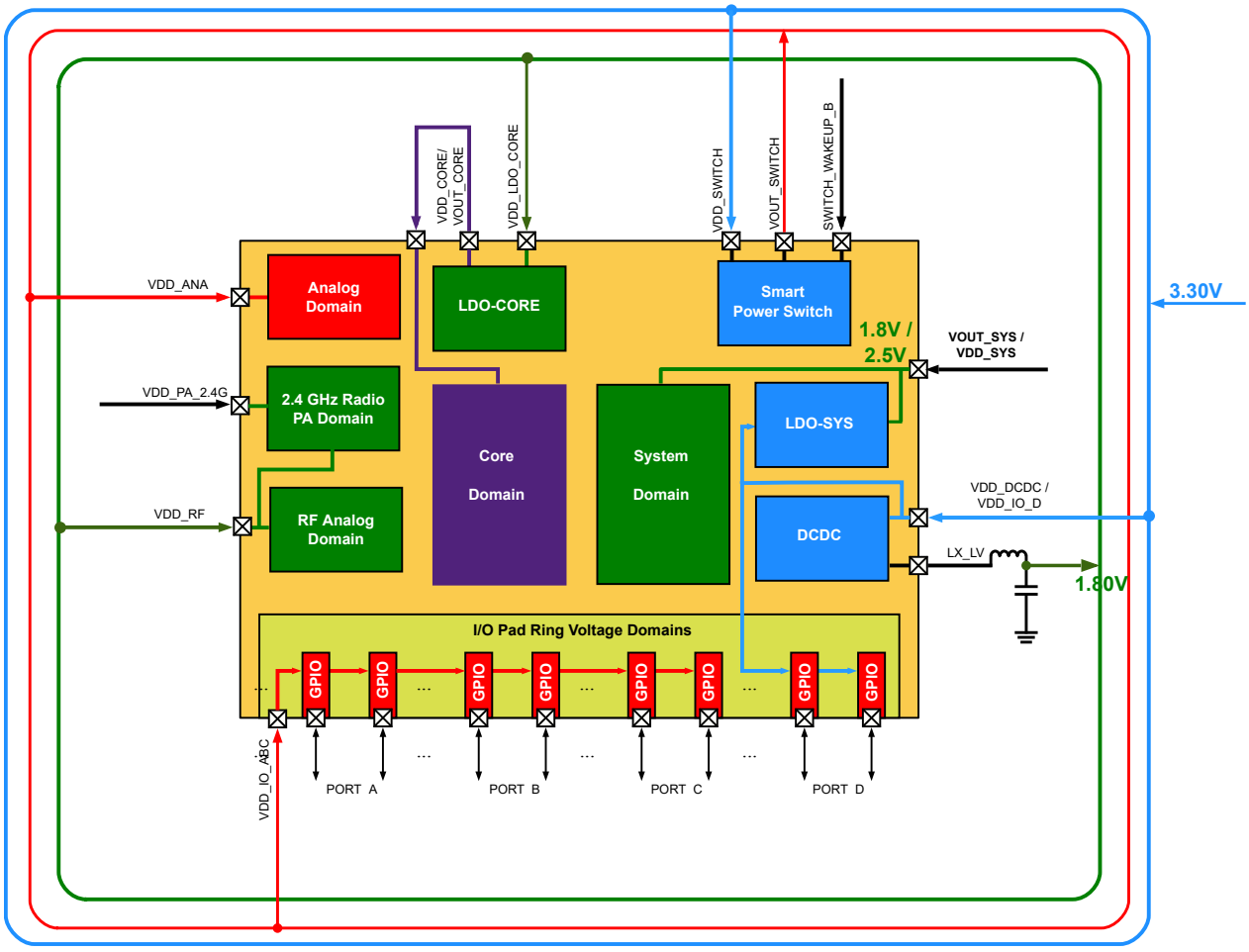


Figure 84. Power all supply except VDD_IO_D/VDD_DCDC

Chapter 31
Core Mode Controller (CMC)

31.1 Chip-specific CMC information

Table 210. Reference links to related information

Topic	Related module	Reference
Full description	CMC	CMC
System memory map		System memory map
Power Management		Power Management
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

31.1.1 Module instances

This device has one instance of the CMC module, CMC0.

31.1.2 TCM arrays

Table 211. TCM SRAM assignment

TCM	SRAM instance	AXBS slave port assignment	Size	Base Address	Controlled by	ECC
Code TCM	CTCM0[1]	S1	8 KB	0x0400_0000	CMC_SRAMSDR0[SDE0] CMC_SRAMDSR0[DSE0]	Yes
	CTCM1[1]	S1	8 KB	0x0400_2000	CMC_SRAMSDR0[SDE1] CMC_SRAMDSR0[DSE1]	Yes
System TCM	STCM0[1]	S2	16 KB	0x2000_0000	CMC_SRAMSDR0[SDE2] CMC_SRAMDSR0[DSE2]	Yes
	STCM1[1]	S2	16 KB	0x2000_4000	CMC_SRAMSDR0[SDE3] CMC_SRAMDSR0[DSE3]	Yes
	STCM2[1]	S2	32 KB	0x2000_8000	CMC_SRAMSDR0[SDE4] CMC_SRAMDSR0[DSE4]	Yes
	STCM3[1]	S3	32 KB	0x2001_0000	CMC_SRAMSDR0[SDE5] CMC_SRAMDSR0[DSE5]	No
	STCM4[1]	S3	8 KB	0x2001_8000	CMC_SRAMSDR0[SDE6] CMC_SRAMDSR0[DSE6]	No
	STCM5[2]	S4	8 KB	0x2001_A000	VBAT_LDORAMC[ISO]	Yes

1. If the corresponding SRAMSDR0[SDE] bit is set, any access to the TCM will trig bus fault.
2. If VBAT_LDORAMC[ISO] bit is set and ECC is enabled, any access to the STCM5 will trig ECC fault.

31.2 Overview

CMC provides the sequencing of the CPU and associated logic through the different operating modes.

This chapter describes the available CPU operating modes, including reset, active mode, and low-power modes.

31.2.1 Block diagram

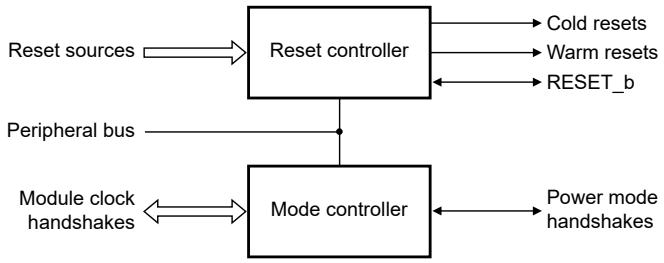


Figure 85. Block diagram

31.2.2 Features

- Configures the low-power mode options
- Provides reset controller including reset status register and warm reset configuration
- Provides boot mode register and status bits

31.3 Functional description

This section provides the functional description of the block.

31.3.1 Modes of operation

The Arm Cortex-M CPU enters a low-power mode with the execution of a WFI or WFE instruction or via the SLEEPONEXIT mechanism. CMC configures the low-power mode that is entered.

The following table describes the low-power modes available to each core.

Table 212. Operating modes

Mode	Entry	Exit	Description
Reset	POR Warm reset WUU via Power-Down	Active	Core and peripherals are reset. SRAM optionally retained on power-down wake-up.
Active	Reset NVIC WUU	WFI WFE SLEEPONEXIT	Core executing instructions, peripherals optionally active.

Table continues on the next page...

Table 212. Operating modes (continued)

Mode	Entry	Exit	Description
Sleep	WFI WFE SLEEPONEXIT	NVIC Reset	Core waiting for interrupt or event, peripherals optionally active. Core and peripherals retain state, SRAM optionally retained.
Deep Sleep	WFI WFE SLEEPONEXIT	WUU via Active Reset	Core and peripherals waiting for wake-up. Core and peripherals retain state, SRAM optionally retained.
Power-Down	WFI WFE SLEEPONEXIT	WUU via Reset Reset	Core and peripherals not retained, SRAM optionally retained.
Deep-Power Down	WFI WFE SLEEPONEXIT	WUU via Reset Reset	Core, peripherals, and SRAM not retained.

31.3.2 Active mode

This section describes power options in Active mode.

31.3.2.1 Flash memory

This section describes the Flash Memory Low-Power mode.

You can place the internal flash memory in Low-Power mode under the following conditions:

- Under software control (FLASHCR[FLASHDIS])
- When CPU is sleeping (FLASHCR[FLASHDOZE])
- When the chip enters a low-power mode

When FLASHCR[FLASHDIS] or FLASHCR[FLASHDOZE] becomes 1, you can configure the internal flash memory to automatically exit Low-Power mode during the access on any attempted access to flash memory space. FLASHCR[FLASHWAKE] controls this.

When the internal flash memory enters a low-power mode, the wake-up time of the flash memory automatically delays the first access upon exit from Low-Power mode. If there is no access between the internal flash memory exiting Low-Power and subsequent re-entry of Low-Power mode, the wake-up time of the flash memory delays the re-entry.

31.3.2.2 System SRAM

The system SRAM is divided into 8 different power partitions. You can individually power gate these partitions either all the time or only in low-power modes. System SRAM contents in a power gated partition are not retained. You must not access system SRAM partitions when they are power gated.

- Write 1 to SRAMSDR[i] to always power gate system SRAM power partition "i".
- Write 1 to SRAMDSR[i] to power gate system SRAM power partition "i" during a low-power mode only.

31.3.3 Low-power modes

This section describes the low-power modes.

31.3.3.1 Low-power entry

The following steps occur during the Low-Power mode entry sequence.

Table 213. Entry sequence

Step	CKMODE (min)	LPMODE (min)	Description
1	0h	0h	Core enters Low-Power mode by WFI/WFE instruction or via SLEEPONEXIT.
2	1h	0h	• Wake-up interrupt controller is enabled. • Core is clock gated.
3	3h	0h	• Request AHB initiators to enter Low-Power mode. • Wait for acknowledge. • AHB initiators are clock gated.
4	3h	0h	• Request AHB peripherals to enter Low-Power mode. • Wait for acknowledge. • AHB peripherals are clock gated.
5	7h	0h	• Request peripherals to enter Low-Power mode. • Wait for acknowledge. • Peripherals are clock gated.
6	Fh	0h	Request power management (regulators, bandgap) to enter Low-Power mode.
7	Fh	1h	• SRAMs enter Low-Power mode. • System clocks are gated.
8	Fh	3h	Power domain is placed in Low-Power retention state (Deep Sleep).
9	Fh	7h	Power domain is switched off (Power-Down).
10	Fh	Fh	Regulators are powered off (Deep Power-Down).

31.3.3.2 DMA wakeup

You can configure the DMA to generate a wake-up when a DMA request for configured channel asserts. This is supported when you request the DMA to enter Low-Power mode and retain the DMA state (not Power Down or Deep Power Down).

The DMA wakeup triggers an exit from Low-Power mode for everything except the core, which remains clock gated. Re-enter Low-Power mode after the DMA request negates, and the normal low-power mode entry sequence is followed.

Because a DMA wake-up results in everything except the core from exiting Low-Power mode, you must ensure that other modules not involved in the wakeup remain in a known state. To accomplish this, disable the modules before the initial entry in Low-Power mode or configure the Doze field in selected modules.

If the flash memory is not required during the DMA wakeup, NXP recommends that FLASHCR[FLASHDOZE] = 1.

NOTE

If the DMA request does not negate due to the DMA transfer, the chip remains in a higher power state until an interrupt or other wake-up sources can wakeup the core.

An interrupt that occurs during a DMA wakeup causes an immediate exit from Low-Power mode without affecting the DMA transfer.

31.3.3.3 Power domains

CMC configures Low-Power mode of 2 different power domains when the core enters a low-power mode with CKMODE = Fh. You can configure the different power domains to enter different low-power modes , but you must not configure the WAKE domain to a lower-power mode than any of the other domains.

NOTE

The WAKE domain LPMODE register setting must always be less than or equal to the LPMODE register setting for all other power domains.

CMC controls the following power domains:

- MAIN
- WAKE

The table below lists allowed combinations for MAIN and WAKE.

Table 214. Power Mode Combinations for MAIN and WAKE

CMC PMCTRLMAIN	CMC PMCTRLWAKE
0x0	0x0
0x1	0x1
0x3	0x3
0x3	0x1
0xF	0xF

Configure all power domains to the same Low-Power mode using Global Power Mode Control (GPMCTRL). Alternatively, configure each PMCTRL register to assign an individual low-power mode to each domain.

31.3.3.4 Debug in low-power modes

When the debugger asserts CDBGPWRUPREQ, it requests the debug logic to power up and enable the functionality depending on DBGCTL[SOD].

- When DBGCTL[SOD] = 0, the core clock remains enabled even when the core is sleeping (CKMODE = 0h). CDBGPWRUPACK remains asserted.
- When DBGCTL[SOD] = 1, ignore the debug request when the core sleeps. CDBPWRUPQACK negates when the core is sleeping.

NOTE

Do not attach a debugger when the JTAG/SWD logic is powered down.

31.3.4 Clocks

- The slow bus clock clocks CMC registers and logic.
- FRO 1 MHz clock clocks the timeout counters.

31.3.5 Reset

This section describes the sources of reset and the different resets that you can generate.

31.3.5.1 Reset sources

This section describes the different reset sources.

Power-on reset (POR)

When you initially apply power to the MCU or the supply voltage is below the POR falling threshold, the POR circuit triggers the POR condition.

The POR condition asserts a cold reset in all power domains. SRS[POR] and SRS[LVD] become 1 on a POR condition.

Low voltage detect (LVD)

The LVD circuit enables by default and keeps the MCU in reset until the supply voltage rises above the LVD rising threshold. When the LVD circuit is enabled, it triggers an LVD reset condition if the supply voltage is below the LVD falling threshold.

SRS[LVD] becomes 1 on an LVD reset condition. The LVD reset condition asserts a cold reset in all power domains. Only the LVD or HVD detection logic is unaffected by LVD or HVD resets.

High voltage detect (HVD)

The HVD circuit disables by default, but when enabled, it triggers an HVD reset condition if the supply voltage exceeds the HVD threshold.

SRS[HVD] becomes 1 on an HVD reset condition. The HVD reset condition asserts a cold reset in all power domains. Only the LVD or HVD detection logic is unaffected by LVD or HVD resets.

Wakeup (WAKEUP)

On a wakeup from Power Down or Deep-Power Down mode, the power management logic triggers a Wake-up reset in the power domains that had powered off.

The wake-up reset condition asserts a cold reset in all power domains that were powered down. The system power domain, including the RESET_b pin, is unaffected by a wake-up from Power Down or Deep-Power Down mode (unless another reset source triggers the wake-up). SRS[WAKEUP] becomes 1 on a wake-up reset condition.

External pin reset (RESET_b)

The RESET_b pin is a bidirectional open-drain pin with an internal pull-up resistor. The RESET_b pin function depends on the mode:

- During reset, the RESET_b drives low until the MCU completes initialization, at which point the RESET_b pin is released. If the RESET_b pin asserts externally, then the MCU remains in reset until the RESET_b input is pulled high.
- During active and low-power modes, the RESET_b pin can assert externally to force the MCU into the PIN reset condition.

The RESET_b pin implements a digital filter that you can configure to filter out glitches on the RESET_b pin that are less than 1–32 CMC clock cycles. The filter bypasses in low-power modes when you disable the CMC clock.

The PIN reset condition asserts a warm reset in all power domains. SRS[PIN] and SRS[WARM] become 1 on a PIN reset condition.

Debug access port reset (DAP)

Any debug access port that can initiate a reset request from a connected debugger triggers the DAP reset condition.

The DAP reset condition asserts a warm reset in all power domains. SRS[DAP] and SRS[WARM] become 1 on a DAP reset condition.

Reset timeout (RSTACK)

The reset state machine includes a timeout counter that is triggered by the reset state machine not progressing within 8192 cycles of the FRO 1 MHz clock. This triggers the RSTACK reset condition.

The RSTACK reset condition asserts a warm reset in all power domains. SRS[RSTACK], SRS[FATAL], and SRS[WARM] become 1 on an RSTACK reset condition.

Low-power timeout (LPACK)

The low-power entry state machine includes a timeout counter that triggers if a module does not acknowledge entry into Low-Power mode after 8192 cycles of the FRO 1 MHz clock. This triggers the LPACK reset condition.

The LPACK reset condition asserts a warm reset in all power domains. SRS[LPACK] and SRS[WARM] become 1 on an LPACK reset condition.

System clock generation (SCG)

The system clock generation includes loss of clock and loss of lock monitors that you can configure to generate a reset, this triggers the SCG reset condition.

The SCG reset condition asserts a warm reset in all power domains. SRS[SCG], SRS[FATAL], and SRS[WARM] become 1 on an SCG reset condition.

Watchdog 0 (WDOG0)

The watchdog timer monitors the software by expecting periodic refreshing of the watchdog counter. When this does not occur, it triggers the WDOG0 reset condition.

The WDOG0 reset condition asserts a warm reset in all power domains. SRS[WDOGO] and SRS[WARM] become 1 on a WDOG0 reset condition.

Software (SW)

The software can request a system reset by configuring the system reset request in the Cortex-M33 core, this triggers the software reset condition.

The software reset condition asserts a warm reset in all power domains. SRS[SW] and SRS[WARM] become 1 on a software reset condition.

Lockup (LOCKUP)

The Cortex-M33 core enters Lockup state as a result of certain illegal operations, this triggers the LOCKUP reset condition.

The LOCKUP reset condition asserts an MCU reset in all power domains. SRS[LOCKUP] and SRS[WARM] become 1 on a LOCKUP reset condition.

Watchdog 1 (WDOG1)

The watchdog timer monitors the software by expecting periodic refreshing of the watchdog counter. When this does not occur, it triggers the WDOG1 reset condition.

The WDOG1 reset condition asserts a warm reset in all power domains. SRS[WDOG1] and SRS[WARM] become 1 on a WDOG1 reset condition.

Security violation (SECVIO)

When a security module detects a security violation, it triggers the SECVIO reset condition.

The SECVIO reset condition asserts a warm reset in all power domains. SRS[SECVIO], SRS[FATAL], and SRS[WARM] become 1 on a SECVIO reset condition.

31.3.5.2 Reset types

This section describes the different resets that you can generate.

Cold reset

Each power domain generates a cold reset to reset the debug logic and mode control logic. The cold reset for each power domain can assert as a result of the following events.

- Initial POR of the chip
- Voltage detect monitor (LVD, for example)
- Power domain wake-up from Power Down or Deep-Power Down (varies per domain)

A cold reset does not guarantee the contents of on-chip SRAM, except for a Power Down wake-up and then only for the SRAM that is configured to retain state.

Warm reset

- Cold Reset or any of the remaining warm reset sources generates a warm reset.
- A warm reset resets most of the logic in each power domain.
- Warm resets are divided into fatal reset sources and non-fatal reset sources.

You can configure the non-fatal reset sources to generate an interrupt instead of the warm reset. The warm reset averts if you can clear the non-fatal reset source (including status flag) within 8192 cycles of the FRO 1 MHz clock. Non-fatal resets retain the contents of on-chip SRAM.

You cannot configure the fatal reset sources to generate an interrupt and do not guarantee the contents of on-chip SRAM.

31.3.5.3 Reset sequence

Power-on reset

Perform the following steps as part of the POR (or voltage detect monitor) sequence:

1. RESET_b pin drives low and internal reset signals asserted.
2. POR or LVD signals negate and clocks are enabled in their default configuration.
3. Internal reset remains asserted for 8 CMC clock cycles.
4. Internal reset to flash memory and fuse controllers negates and their initialization sequences commence.
5. Initialization sequences for flash memory and fuse complete.
6. Internal trim registers are loaded and RESET_b pin is tri-stated.
7. Reset state machine waits for RESET_b pin input to pull high or drive high externally.
8. Internal reset signals negate.
9. Core exits reset and fetches the initial program counter and stack pointer from ROM.

Deep-power down wake-up

Perform the following steps as part of the deep-power down wake-up sequence:

1. Internal reset signals asserted.
2. Wake-up signal negates and clocks are enabled in their default configuration.
3. Internal reset remains asserted for 8 CMC clock cycles.
4. Internal reset to flash memory and fuse controllers negates and commence their initialization sequences.
5. Initialization sequences for flash memory and fuse complete.
6. Internal trim registers are loaded.
7. Internal reset signals negate.
8. Core exits reset and fetches the initial program counter and stack pointer from ROM.

A wake-up from deep-power down via the RESET_b pin follows the warm reset sequence.

Power down wake-up

Perform the following steps as part of the power down wake-up sequence:

1. Internal reset signals asserted.
2. Clocks are enabled in their default configuration.
3. Internal reset remains asserted for 8 CMC clock cycles.
4. Internal trim registers are loaded.
5. Internal reset signals negate.
6. Core exits reset and fetches the initial program counter and stack pointer from ROM.

A wake-up from power down via the RESET_b pin follows the warm reset sequence.

Warm reset

Perform the following steps as part of the warm reset sequence:

1. RESET_b pin drives low and internal reset signals asserted.
2. Clocks are enabled in their default configuration.
3. Internal reset remains asserted for 8 CMC clock cycles.
4. Internal reset to flash memory and fuse controllers negates and commence their initialization sequences.
5. Initialization sequences for flash memory and fuse complete.
6. Internal trim registers are loaded and RESET_b pin is tri-stated.
7. Reset state machine waits for RESET_b pin input to pull high or drive high externally.
8. Internal reset signals negate.
9. Core exits reset and fetches the initial program counter and stack pointer from ROM.

31.3.6 Interrupts

CMC generates a single interrupt, which System Reset Interrupt Enable (SRIE) configures.

31.4 External signals

Table 215. External signals

Signal	Description	Direction
RESET_B	Bidirectional open-drain reset pin with pullup that asserts low during warm reset except when waking up from a low-power mode.	Input or output
BOOT_CONFIGn	The input pin is sampled at the end of the RESET_B assertion and stored in Mode (MR0).	Input

31.5 Initialization

By default, the digital glitch filter is disabled on RESET_b pin. To enable the filter, configure:

- RPC[LPFEN], RPC[FILTEN], or both.
- RPC[FILTCFG].

31.6 Application information

To configure for Deep-Power Down Low-Power mode entry:

1. Write Fh to Clock Control (CKCTRL)
2. Write 8h to Power Mode Protection (PMPROT)
3. Write Fh to Global Power Mode Control (GPMCTRL)
4. Execute WFI instruction

To configure for Power-Down Low-Power mode entry:

1. Write Fh to Clock Control (CKCTRL)
2. Write 4h to Power Mode Protection (PMPROT)
3. Write 7h to Global Power Mode Control (GPMCTRL)
4. Execute WFI instruction

To configure for Deep Sleep Low-Power mode entry:

1. Write Fh to Clock Control (CKCTRL)
2. Write 2h to Power Mode Protection (PMPROT)
3. Write 3h to Global Power Mode Control (GPMCTRL)
4. Execute WFI instruction

To configure for MAIN in power-down and WAKE in Deep Sleep Low-Power mode entry:

1. Write Fh to Clock Control (CKCTRL)
2. Write 3h to Power Mode Protection (PMPROT)
3. Write 3h to PMCTRLMAIN
4. Write 1h to PMCTRLWAKE
5. Execute WFI instruction

31.7 Memory map and register descriptions

This section describes the registers in the CMC module.

31.7.1 CMC register descriptions

NOTE

Different CMC registers reset on different reset types.

NOTE

You must read back the last register written to before executing the WFI instruction. This ensures that before the MCU enters the low power mode, all register writes associated with setting up the low power mode are complete, failure to do this may result in the low power mode not being entered correctly.

31.7.1.1 CMC memory map

CMC0 base address: 4000_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0301_0000h
10h	Clock Control (CKCTRL)	32	RW	0000_0000h
14h	Clock Status (CKSTAT)	32	RW	0000_0000h
18h	Power Mode Protection (PMPROT)	32	RW	0000_0000h
1Ch	Global Power Mode Control (GPMCTRL)	32	RW	0000_0000h
20h	Power Mode Control (PMCTRLMAIN)	32	RW	0000_0000h
24h	Power Mode Control (PMCTRLWAKE)	32	RW	0000_0000h
80h	System Reset Status (SRS)	32	R	See section
84h	Reset Pin Control (RPC)	32	RW	0000_0000h
88h	Sticky System Reset Status (SSRS)	32	RW	0000_0006h
8Ch	System Reset Interrupt Enable (SRIE)	32	RW	0000_8800h
90h	System Reset Interrupt Flag (SRIF)	32	RW	0000_0000h
9Ch	Reset Count Register (RSTCNT)	32	R	0000_0000h
A0h	Mode (MR0)	32	RW	0000_0000h
B0h	Force Mode (FM0)	32	RW	0000_0000h
C0h	SRAM Shut Down Register (SRAMDIS0)	32	RW	0000_0000h
D0h	SRAM Deep Sleep Register (SRAMRET0)	32	RW	0000_0000h
E0h	Flash Control (FLASHCR)	32	RW	0000_0000h
100h	BootROM Status Register (BSR)	32	RW	0000_0000h
10Ch	BootROM Lock Register (BLR)	32	RW	0000_0002h
110h	Core Control (CORECTL)	32	RW	0000_0000h
120h	Debug Control (DBGCTL)	32	RW	0000_0000h

31.7.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains version numbers for module design and feature set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number.

31.7.1.3 Clock Control (CKCTRL)

Offset

Register	Offset
CKCTRL	10h

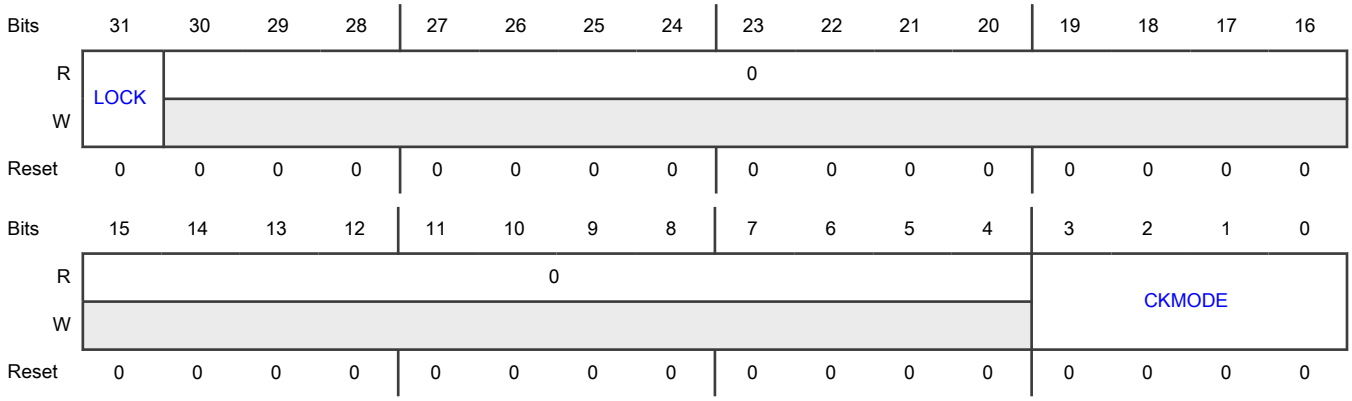
Function

Configures the amount of clock gating when the core asserts sleeping due to WFI, WFE, or SLEEPONEXIT.

NOTE

This register resets on WAKE warm reset.

Diagram



Fields

Field	Function
31 LOCK	Lock Locks the register and blocks writes until the next reset. 0b - Allowed 1b - Blocked
30-4 —	Reserved
3-0 CKMODE	Clocking Mode Configures the amount of clock gating when the core enters a low-power mode because of WFI, WFE or SLEEPONEXIT. Configuring CKMODE > 0 requires the SLEEPDEEP field in the Arm core to become 1. Configuring PMCTRLx[LPMODE] > 0 requires writing Fh to CKMODE. 0000b - Core clock is on 0001b - Core clock is off 0011b - Core and platform clocks are gated 0111b - Core, platform, and peripheral clocks are gated, but no change in Low-Power mode 1111b - Core, platform, and peripheral clocks are gated, and core enters Low-Power mode.

31.7.1.4 Clock Status (CKSTAT)

Offset

Register	Offset
CKSTAT	14h

Function

Returns the clock gating status and wake-up source from the previous Low-Power mode entry, provided the core was clock gated. This requires configuring CKCTRL[CKMODE] > 0 and the wake-up event must occur after the core is clock gated. The register contents are only valid when CKSTAT[VALID] = 1.

NOTE

This register resets on WAKE warm reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALID								0							
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WAKEUP							0				CKMODE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VALID	Clock Status Valid Indicates that the core clock was gated since you last wrote 0 to this field. 0b - Core clock not gated 1b - Core clock was gated due to Low-Power mode entry
30-15 —	Reserved
14-8 WAKEUP	Wake-up Source Returns any wake-up sources from the previous Low-Power mode entry. [0] - Wake-up source is reset interrupt or wakeup from Deep Power-Down [1] - Wake-up source is debug request [2] - Wake-up source is interrupt [3] - Wake-up source is DMA wakeup [4] - Wake-up source is WUU request [5] - Wake-up source is bus master wakeup [6] - Wake-up source is bus master wakeup A wakeup from Power-Down or Deep Power-Down writes 1 to WAKEUP[0]. Depending on Low-Power mode, other wake-up bits may not become 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 —	Reserved
3-0 CKMODE	Low Power Status Returns the result of the previous Low-Power mode entry. 0000b - Core clock not gated 0001b - Core clock was gated 0011b - Core and platform clocks were gated 0111b - Core, platform, and peripheral clocks were gated 1111b - Core, platform, and peripheral clocks were gated, and power domain entered Low-Power mode All other values are reserved.

31.7.1.5 Power Mode Protection (PMPROT)

Offset

Register	Offset
PMPROT	18h

Function

Provides protection for entry in low-power modes, you must allow a low-power mode before configuring any Power Mode Control register (PMCTRLx) to that mode.

NOTE

This register resets on WAKE warm reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCK	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												LPMODE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LOCK	Lock Register Locks the register and blocks writes until the next reset. 0b - Allowed 1b - Blocked
30-4 —	Reserved
3-0 LPMODE	Low-Power Mode Bit 0: When set, allows the PMCTRLx[LPM] field to be configured for Sleep. Bit 1: When set, allows the PMCTRLx[LPM] field to be configured for Deep Sleep. Bit 2: When set, allows the PMCTRLx[LPM] field to be configured for Power Down. Bit 3: When set, allows the PMCTRLx[LPM] field to be configured for Deep Power Down. 0000b - Not allowed 0001b - Allowed 0010b - Allowed 0011b - Allowed 0100b - Allowed 0101b - Allowed 0110b - Allowed 0111b - Allowed 1000b - Allowed 1001b - Allowed 1010b - Allowed 1011b - Allowed 1100b - Allowed 1101b - Allowed 1110b - Allowed 1111b - Allowed

31.7.1.6 Global Power Mode Control (GPMCTRL)

Offset

Register	Offset
GPMCTRL	1Ch

Function

Updates all PMCTRLx registers with the value written.

NOTE

This register resets on WAKE cold reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0			
W													LPMODE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-4 —	Reserved
3-0 LPMODE	Low-Power Mode Specifies that all PMCTRLx[LPMODE] fields update with the value written.

31.7.1.7 Power Mode Control (PMCTRLMAIN - PMCTRLWAKE)

Offset

Register	Offset
PMCTRLMAIN	20h
PMCTRLWAKE	24h

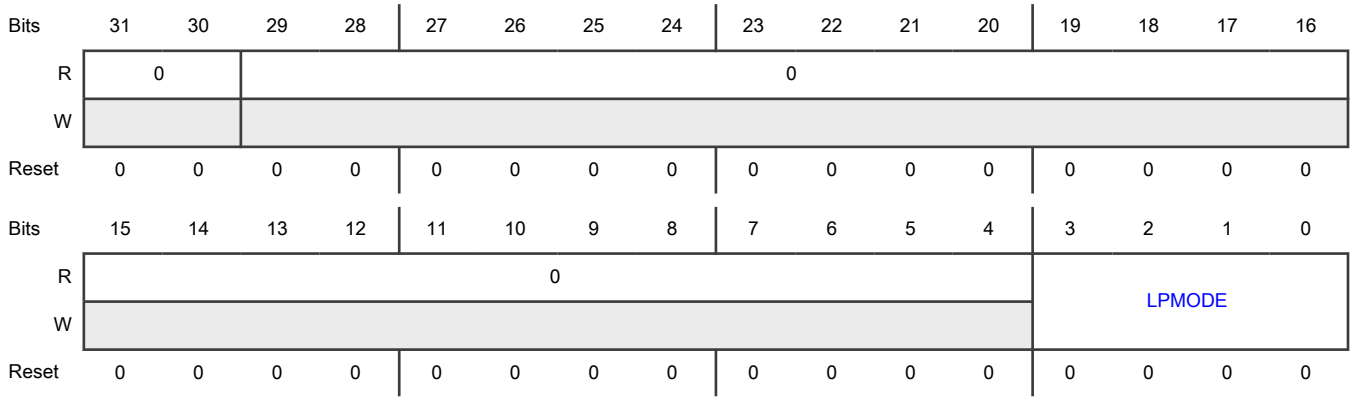
Function

Configures entry into low-power modes for each power domain, provided that the selected power mode is allowed via an appropriate setting of Power Mode Protection (PMPROT), and CKMODE = Fh.

NOTE

This register resets by WAKE cold reset.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-4 —	Reserved
3-0 LPMODE	Low-Power Mode Selects the desired Low-Power mode when a core executes WFI or WFE instruction. Writes to this field are blocked if you have not enabled the protection level using Power Mode Protection (PMPROT). You must not configure this field for the WAKE domain to a lower power mode than any other power domain. **NOTE** See Power domains section for details on allowed combinations. 0000b - Active 0001b - Sleep 0011b - Deep Sleep 0111b - Power Down 1111b - Deep-Power Down

31.7.1.8 System Reset Status (SRS)

Offset

Register	Offset
SRS	80h

Function

Updates on every MAIN warm reset to indicate the type/source of the most recent reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SECVIO	0	0	0	0	WDOG1	0	0	0					0	0
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCKUP	SW	WDOG0	SCG	LPACK	RSTACK	DAP	PIN	0		FATAL	WARM	HVD	LVD	POR	WAKEUP
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 —	Reserved
30 SECVIO	Security Violation Reset Indicates a security violation logic that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated 1b - Reset generated
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 WDOG1	Watchdog 1 Reset Indicates the WatchDog 1 timeout that generates a reset. 0b - Reset is not generated 1b - Reset is generated
24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23 —	Reserved
22-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 LOCKUP	Lockup Reset Indicates the Arm core indication of a LOCKUP event that causes a reset. 0b - Reset not generated 1b - Reset generated
14 SW	Software Reset Indicates a software reset request from the Arm core (SYSRESETREQ) that causes a reset. 0b - Reset not generated 1b - Reset generated
13 WDOG0	Watchdog 0 Reset Indicates the WatchDog 0 timeout that causes a reset. 0b - Reset is not generated 1b - Reset is generated
12 SCG	System Clock Generation Reset Indicates a loss-of-clock or loss-of-lock event in the SCG that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset is not generated 1b - Reset is generated
11 LPACK	Low Power Acknowledge Timeout Reset Indicates a timeout in the Low Power Mode entry logic that causes a reset. This timeout is generated if a peripheral does not acknowledge entry into Low-Power mode within 8192 cycles of FRO 1 MHz clock. 0b - Reset not generated 1b - Reset generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 RSTACK	Reset Timeout Indicates a timeout or other error condition in the system reset generation logic that causes a reset. This is a fatal reset source, and SRAM contents are not guaranteed. 0b - Reset not generated 1b - Reset generated
9 DAP	Debug Access Port Reset Indicates a reset request from a DAP that causes a reset. 0b - Reset was not generated 1b - Reset was generated
8 PIN	Pin Reset Indicates an external assertion of the RESET_b pin that causes a reset. 0b - Reset was not generated 1b - Reset was generated
7-6 —	Reserved
5 FATAL	Fatal Reset Asserts if the last reset source was a fatal reset source. You cannot guarantee SRAM contents following a fatal reset source. 0b - Reset was not generated 1b - Reset was generated
4 WARM	Warm Reset Asserts if the last reset source was a warm reset source. 0b - Reset not generated 1b - Reset generated
3 HVD	High Voltage Detect Reset Indicates an HVD that causes a reset. You cannot guarantee SRAM contents following an HVD reset source. 0b - Reset not generated 1b - Reset generated
2 LVD	Low Voltage Detect Reset Indicates an LVD that causes a reset. You cannot guarantee SRAM contents following an LVD reset source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Reset not generated 1b - Reset generated
1 POR	Power-on Reset Indicates the POR detection logic that causes a reset. You cannot guarantee SRAM contents following a POR source. 0b - Reset not generated 1b - Reset generated
0 WAKEUP	Wake-up Reset Indicates a wake-up from Power Down or Deep-Power Down mode that causes a reset. 0b - Reset not generated 1b - Reset generated

31.7.1.9 Reset Pin Control (RPC)

Offset

Register	Offset
RPC	84h

Function

NOTE

WAKE cold reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						LPFEN	FILTEN	0				FILTCFG			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 LPFEN	Low-Power Filter Enable Enables the low-power reset pin filter in both Active and Low-Power modes when this field = 1. The RESET_b pin must assert for more than three FRO 1 MHz clock cycles to always propagate through the filter. A glitch less than two FRO 1 MHz clock cycles never propagates through the filter. 0b - Disables 1b - Enables
8 FILTEN	Filter Enable Enables the slow clock reset pin filter in Active modes. The RESET_b pin must assert for more than FILTCFG + 1 slow system clock cycles to always propagate through the filter. A glitch less than FILTCFG slow system clock cycles never propagates through the filter. If RPC[LPFEN] also becomes 1, then the filters operate in series. 0b - Disables 1b - Enables
7-5 —	Reserved
4-0 FILTCFG	Reset Filter Configuration Configures the reset pin filter's width from 1 to 32 slow system clock cycles.

31.7.1.10 Sticky System Reset Status (SSRS)

Offset

Register	Offset
SSRS	88h

Function

Stores all sources of system reset that has generated a system reset since the last WAKE cold reset and that you have not cleared. SSRS does not update following a core software reset.

NOTE

This register resets on WAKE cold reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SECVIO	0	0	0	0	WDOG1	0	0	0					0	0
W		W1C					W1C									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCKUP	SW	WDOG0	SCG	LPACK	RSTACK	DAP	PIN	0		FATAL	WARM	HVD	LVD	POR	WAKEUP
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31 —	Reserved
30 SECVIO	Security Violation Reset Indicates a security violation that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated 1b - Reset generated
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 WDOG1	Watchdog 1 Reset Indicates the WatchDog 1 timeout that causes a reset. 0b - Reset is not generated 1b - Reset is generated
24	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
23 —	Reserved
22-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 LOCKUP	Lockup Reset Indicates the core lockup that causes a reset. 0b - Reset not generated 1b - Reset generated
14 SW	Software Reset Indicates the software request from the Arm core (SYSRESETREQ) that causes a reset. 0b - Reset not generated 1b - Reset generated
13 WDOG0	Watchdog 0 Reset Indicates the WatchDog 0 timeout that causes a reset. 0b - Reset is not generated 1b - Reset is generated
12 SCG	System Clock Generation Reset Indicates an SCG loss of lock or loss of clock that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset is not generated 1b - Reset is generated
11 LPACK	Low Power Acknowledge Timeout Reset Indicates a low power acknowledge timeout that causes a reset. 0b - Reset not generated 1b - Reset generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 RSTACK	Reset Timeout Indicates a reset controller timeout that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated 1b - Reset generated
9 DAP	DAP Reset Indicates a DAP reset request that causes a reset. 0b - Reset not generated 1b - Reset generated
8 PIN	Pin Reset Indicates a RESET_B pin that causes a reset. 0b - Reset not generated 1b - Reset generated
7-6 —	Reserved
5 FATAL	Fatal Reset Indicates a fatal reset source that causes a reset. 0b - Reset was not generated 1b - Reset was generated
4 WARM	Warm Reset Indicates a warm reset source that causes a reset. 0b - Reset not generated 1b - Reset generated
3 HVD	High Voltage Detect Reset Indicates the HVD that causes a reset. 0b - Reset not generated 1b - Reset generated
2 LVD	Low Voltage Detect Reset Indicates the LVD that causes a reset. 0b - Reset not generated 1b - Reset generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 POR	Power-on Reset Indicates the POR that causes a reset. 0b - Reset not generated 1b - Reset generated
0 WAKEUP	Wake-up Reset Indicates the wake-up from VLLS mode that causes a reset. 0b - Reset not generated 1b - Reset generated

31.7.1.11 System Reset Interrupt Enable (SRIE)

Offset

Register	Offset
SRIE	8Ch

Function

Delays the assertion of a system reset for 8194 cycles of the FRO 1 MHz clock when an interrupt is generated. This allows you to perform a graceful shutdown or to abort the warm reset provided you can clear the pending reset source by resetting the source and then clearing the pending flag. This feature cannot delay a cold or fatal warm reset source. System Reset Status (SRS) updates after the warm reset occurs.

The reset interrupt is not supported in Power Down or Deep-Power Down mode.

NOTE

This register is reset on WAKE Cold Reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0	0	WDOG	0	0	0					0	0
W							1									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCK	SW	WDOG	0	LPAC	0	DAP	PIN	0							
W	UP		0		K											
Reset	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 WDOG1	Watchdog 1 Reset 0b - Interrupt disabled 1b - Interrupt enabled
24 —	Reserved
23 —	Reserved
22-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 LOCKUP	Lockup Reset **NOTE** A core in the lockup state cannot service interrupts. 0b - Interrupt disabled 1b - Interrupt enabled
14 SW	Software Reset 0b - Interrupt disabled 1b - Interrupt enabled
13 WDOG0	Watchdog 0 Reset 0b - Interrupt disabled 1b - Interrupt enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 —	Reserved
11 LPACK	Low Power Acknowledge Timeout Reset 0b - Interrupt disabled 1b - Interrupt enabled
10 —	Reserved
9 DAP	DAP Reset 0b - Interrupt disabled 1b - Interrupt enabled
8 PIN	Pin Reset 0b - Interrupt disabled 1b - Interrupt enabled
7-0 —	Reserved

31.7.1.12 System Reset Interrupt Flag (SRIF)

Offset

Register	Offset
SRIF	90h

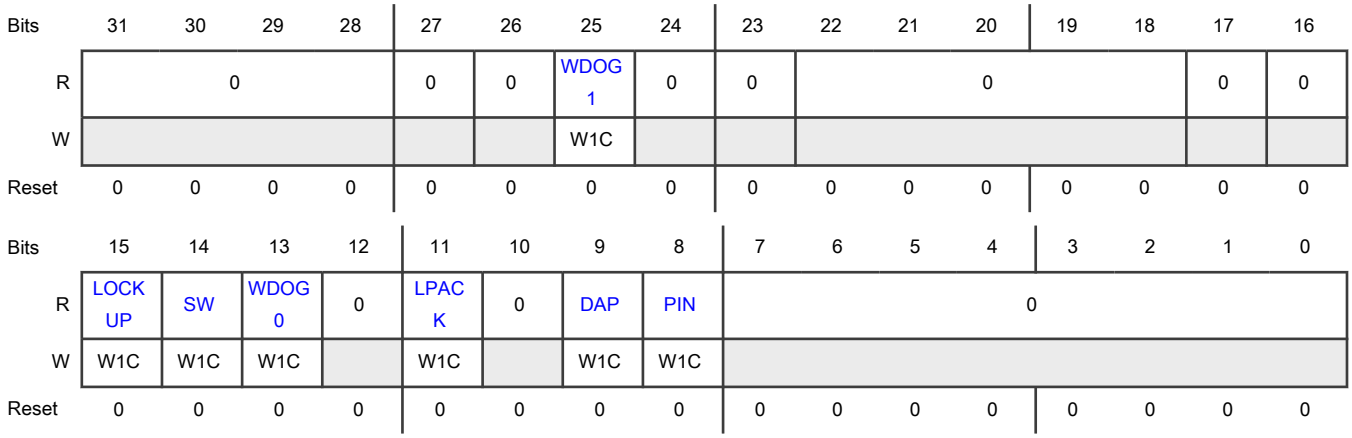
Function

Returns the source of the reset interrupt. You can clear the pending reset source by resetting the source and then clearing the pending flag.

NOTE

This register resets on WAKE warm reset.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26 —	Reserved
25 WDOG1	Watchdog 1 Reset 0b - Reset source not pending 1b - Reset source pending
24 —	Reserved
23 —	Reserved
22-18 —	Reserved
17 —	Reserved
16 —	Reserved
15	Lockup Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
LOCKUP	0b - Reset source not pending 1b - Reset source pending
14 SW	Software Reset 0b - Reset source not pending 1b - Reset source pending
13 WDOG0	Watchdog 0 Reset 0b - Reset source not pending 1b - Reset source pending
12 —	Reserved
11 LPACK	Low Power Acknowledge Timeout Reset 0b - Reset source not pending 1b - Reset source pending
10 —	Reserved
9 DAP	DAP Reset 0b - Reset source not pending 1b - Reset source pending
8 PIN	Pin Reset 0b - Reset source not pending 1b - Reset source pending
7-0 —	Reserved

31.7.1.13 Reset Count Register (RSTCNT)

Offset

Register	Offset
RSTCNT	9Ch

Function

The Reset Count Register returns the number of reset sequences completed since the last WAKE Cold Reset.

NOTE

This register is reset on WAKE Cold Reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Count Number of reset sequences completed since the last WAKE Cold Reset. When the count reaches the maximum value, it will saturate and no longer increment.

31.7.1.14 Mode (MR0)

Offset

Register	Offset
MR0	A0h

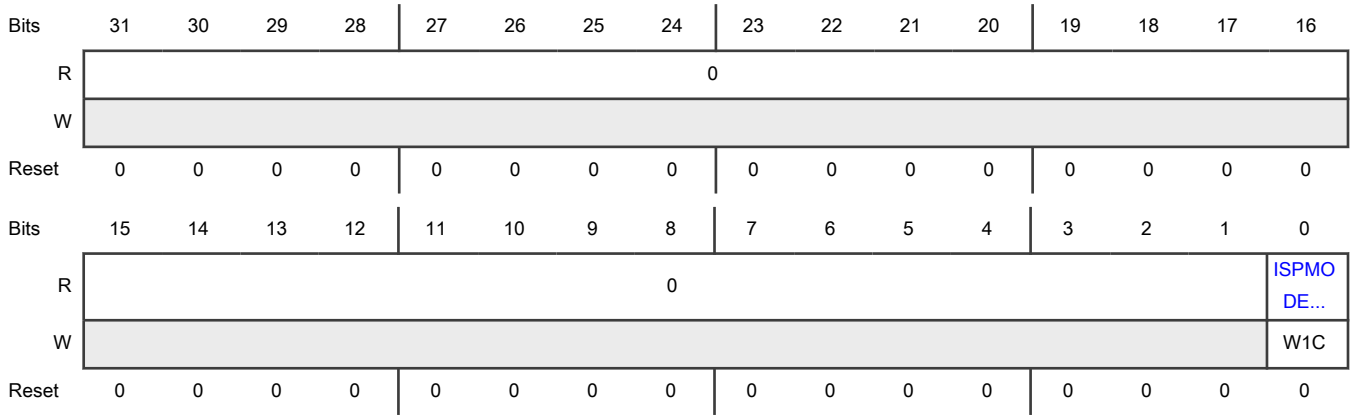
Function

Contains the state of the boot mode pins sampled at the end of the reset.

NOTE

This register resets on WAKE warm reset, and the reset value may vary depending on the pin state.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ISPMODE_n	Boot Configuration Returns the logic state of the BOOT_CONFIGn pin on the last negation of RESET_b pin.

31.7.1.15 Force Mode (FM0)

Offset

Register	Offset
FM0	B0h

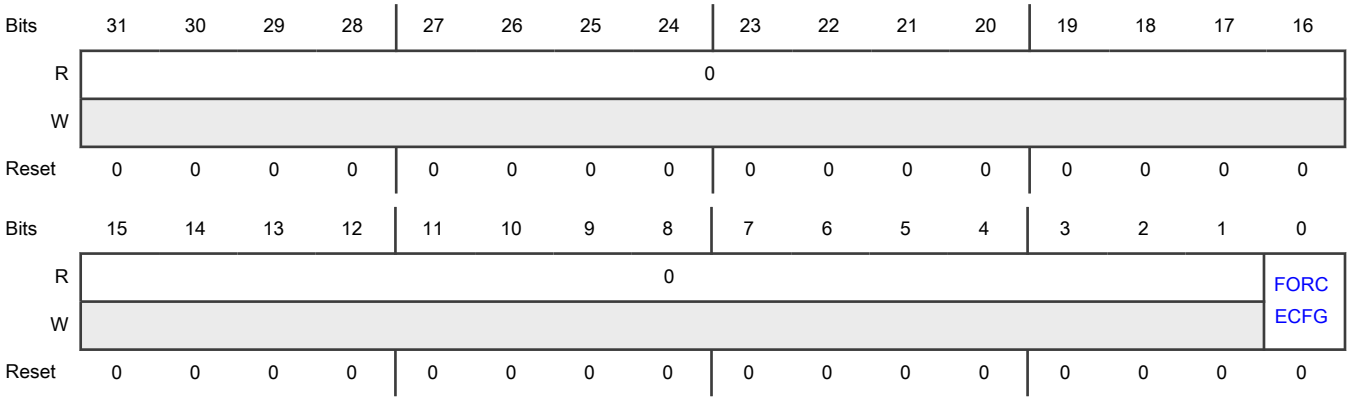
Function

Contains registers that override the state of the boot mode pins.

NOTE

This register resets on WAKE cold reset.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FORCECFG	Boot Configuration Forces the corresponding field in Mode (MR0) to assert on next system reset. 0b - No effect 1b - Asserts

31.7.1.16 SRAM Shut Down Register (SRAMDIS0)

Offset

Register	Offset
SRAMDIS0	C0h

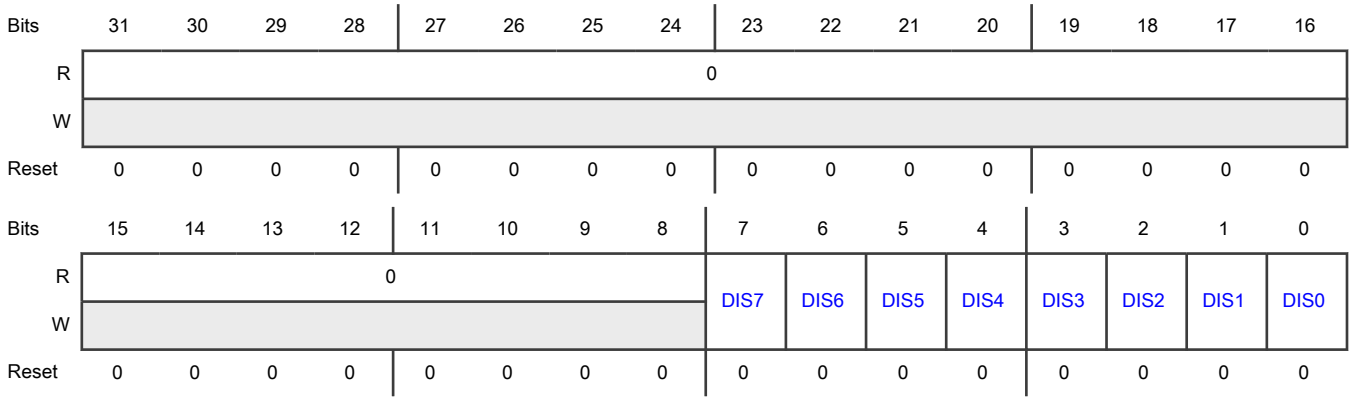
Function

Powers off the specific on-chip System SRAM arrays whenever the corresponding register field becomes 1. You must not access the SRAM array and also do not retain the SRAM array contents.

NOTE

This register resets on MAIN warm reset.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DISn	Shut Down Enable **NOTE** See the chip-specific CMC section for register bit details.

31.7.1.17 SRAM Deep Sleep Register (SRAMRET0)

Offset

Register	Offset
SRAMRET0	D0h

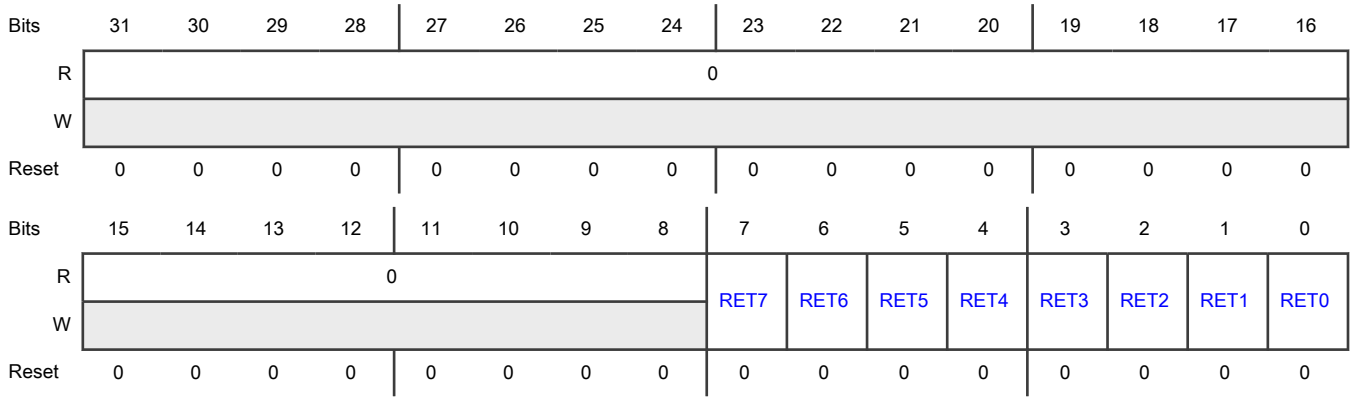
Function

Configures if the on-chip System SRAM arrays are retained or powered off in low-power modes.

NOTE

This register resets on WAKE warm reset.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RETn	Deep Sleep Enable Controls whether the SRAM array is retained or powered off in Low-Power modes. **NOTE** You cannot retain SRAM arrays in Low-Power modes if their power domain is switched off. See the chip-specific CMC section for register bit details.

31.7.1.18 Flash Control (FLASHCR)

Offset

Register	Offset
FLASHCR	E0h

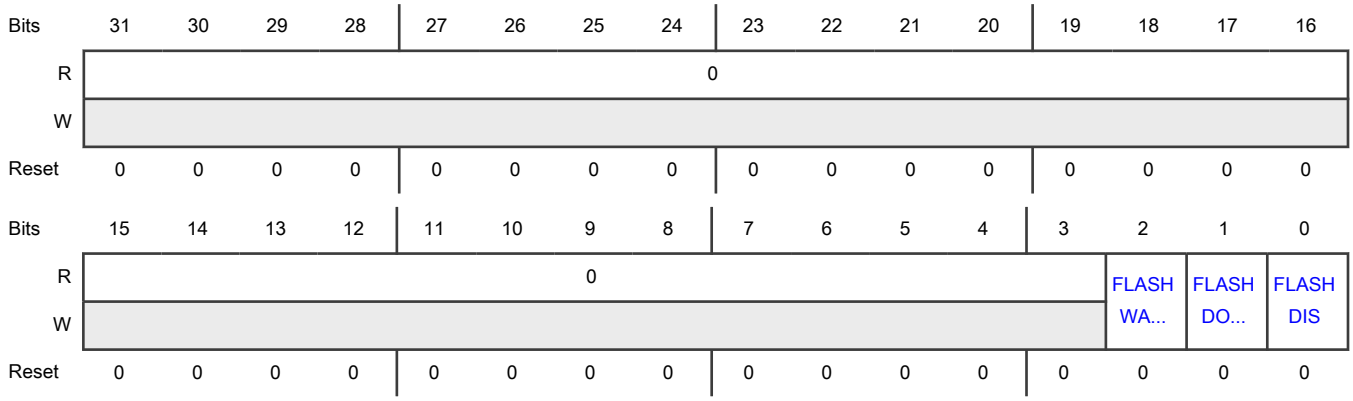
Function

Controls Low-Power mode of the on-chip flash memory.

NOTE

This register resets on WAKE warm reset.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 FLASHWAKE	Flash Wake Specifies that when this field becomes 1, an attempt to access the flash memory when it is in Low-Power state because of FLASHCR[FLASHDIS] or FLASHCR[FLASHDOZE], causes the flash memory to exit Low-Power state for the duration of the flash memory access. 0b - No effect 1b - Flash memory is not disabled during flash memory accesses
1 FLASHDOZE	Flash Doze Disables flash memory accesses and places flash memory in Low-Power state whenever the core clock is gated (CKMODE > 0) because of execution of WFI, WFE, or SLEEPONEXIT. Other bus masters that attempt to access the flash memory stalls until the core is no longer sleeping. 0b - No effect 1b - Flash memory is disabled when core is sleeping (CKMODE > 0)
0 FLASHDIS	Flash Disable Places flash memory in Low-Power state when this field becomes 1. Relocate the interrupts out of flash memory before disabling it. 0b - No effect 1b - Flash memory is disabled

31.7.1.19 BootROM Status Register (BSR)

Offset

Register	Offset
BSR	100h

Function

The BootROM Status Register and BootROM Lock Register can only be written when the Lock field equals 010. All other values are reversed and will lock the BootROM registers.

NOTE

This register resets on WAKE cold reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STAT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STAT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT	Provides status information written by the BootROM.

31.7.1.20 BootROM Lock Register (BLR)

Offset

Register	Offset
BLR	10Ch

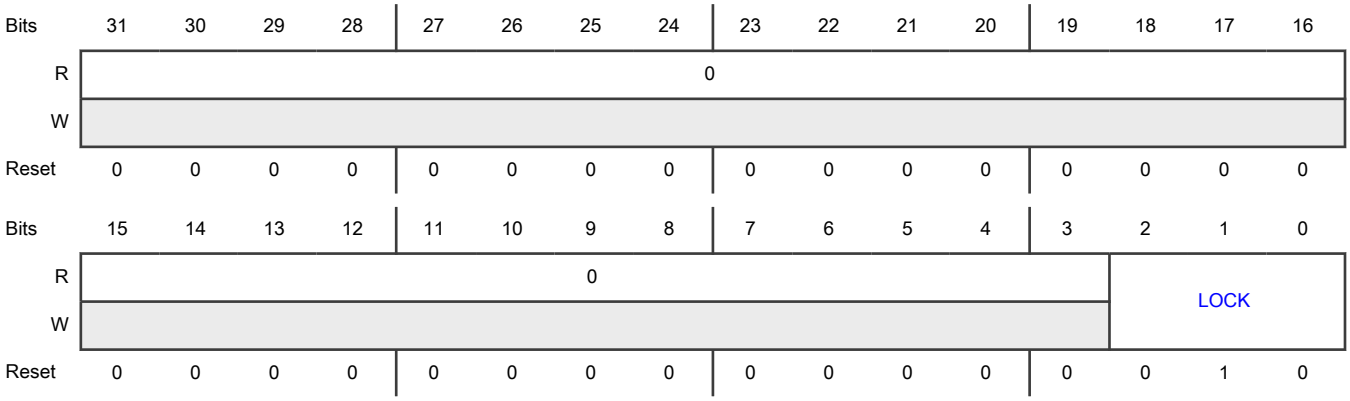
Function

The BootROM Status Register and BootROM Lock Register can only be written when the Lock field equals 010. All other values are reversed and will lock the BootROM registers.

NOTE

This register is reset on MAIN Warm Reset.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 LOCK	Lock The BootROM Status Register and BootROM Lock Register can only be written when the Lock field equals 010. All other values are reversed and will lock the BootROM registers. 010b - BootROM Status and Lock Registers can be written 101b - BootROM Status and Lock Registers cannot be written

31.7.1.21 Core Control (CORECTL)

Offset

Register	Offset
CORECTL	110h

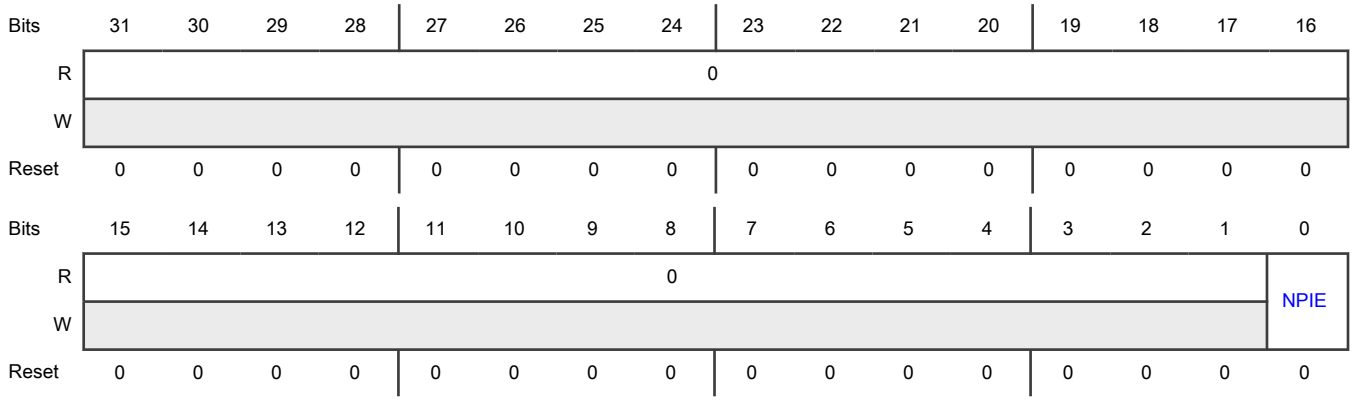
Function

Configures options for the core.

NOTE

This register resets on MAIN warm reset.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 NPIE	Non-maskable Pin Interrupt Enable Enables or disables the pin interrupt. You can write to this field only when NPIE = 0. 0b - Disables 1b - Enables

31.7.1.22 Debug Control (DBGCTL)

Offset

Register	Offset
DBGCTL	120h

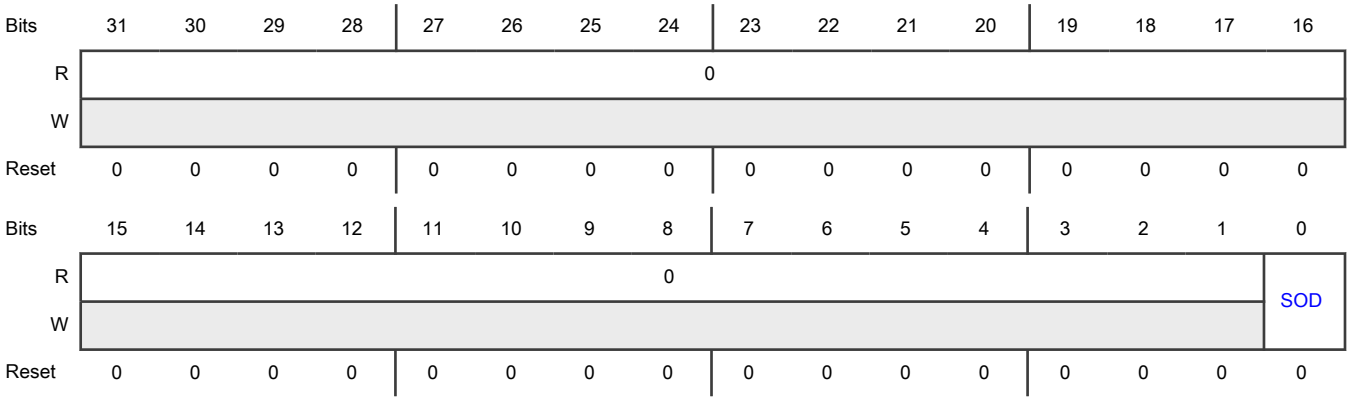
Function

Configures options for debug.

NOTE

This register resets on WAKE cold reset.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SOD	Sleep Or Debug Configures whether the debug remains enabled when core sleeps. 0b - Remains enabled 1b - Disabled

Chapter 32
System Power Control (SPC)

32.1 Chip-specific SPC information

Table 216. Reference links to related information

Topic	Related module	Reference
Full description	SPC	SPC
System memory map		System memory map
Power Management		Power Management
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing
Reset		Reset

NOTE

The "switch" in this chapter refers to the "smart power switch" in the Power Management.

32.1.1 Module instances

- This device has one instance of the SPC module, SPC0.
- If the LPCMP is being used in DPDOWN then the low power IREF must remain enabled.

32.1.2 Voltage regulators

This section describes the various on-chip regulators.

Table 217. On-chip regulators

Regulator name	Description
DCDC_LV	The DCDC_LV can be used to power VDD_SYS domain. It can power the IO ring, other on-chip modules, and external components as well. However, user should take care that the total load current doesn't exceed the maximum capacity.
LDO_CORE	The LDO_CORE can be used to power VDD_CORE domain.
LDO_SYS	The LDO_SYS can be used to power VDD_SYS domain. It can power the IO ring, other on-chip modules, and external components as well. However, user should take care that the total load current doesn't exceed the maximum capacity.

32.1.3 Voltage monitors

32.1.3.1 VDD_SYS voltage monitors

The following sections describe the voltage monitors inspecting the VDD_SYS supply.

32.1.3.1.1 Power-On-Reset (POR)

During power-on, the POR keeps the device under reset until the VDD_SYS supply voltage reaches the specified threshold. When VDD_SYS is above the VPOR limit, the device POR reset is released. However, the device stays in reset until the LVD on VDD_SYS is released.

32.1.3.1.2 Low Voltage Detect

This device integrates a Low Voltage Detector (LVD) circuit on VDD_SYS input voltage. The LVD monitors the VDD_SYS power supply voltage by comparing it to a programmable threshold. When VDD_SYS drops below the threshold, the user has the option of generating an LVD reset or an LVW interrupt. The VDD_SYS LVD circuit is ON by default. The user can disable the VDD_SYS LVD through software.

32.1.3.1.3 High Voltage Detect

This device integrates a High Voltage Detector (HVD) circuit on VDD_SYS input voltage. The HVD monitors the VDD_SYS power supply voltage by comparing it to a programmable threshold. When VDD_SYS rises above the threshold, the user has the option of generating an HVD reset or an HVW interrupt. The VDD_SYS HVD circuit is OFF by default. The user can enable the VDD_SYS HVD through software.

32.1.3.2 VDD_CORE voltage monitors

The following sections describe the voltage monitors for the VDD_CORE supply.

32.1.3.2.1 Low Voltage Detect

This device integrates a Low Voltage Detector (LVD) circuit on VDD_CORE input voltage. The LVD monitors the VDD_CORE power supply voltage by comparing it to a programmable threshold. When VDD_CORE drops below the threshold, the user has the option of generating an LVD reset or an LVW interrupt. The VDD_CORE LVD circuit is ON by default. The user can disable the VDD_CORE LVD through software.

32.1.3.2.2 High Voltage Detect

This device integrates a High Voltage Detector (HVD) circuit on VDD_CORE input voltage. The HVD monitors the VDD_CORE power supply voltage by comparing it to a programmable threshold. When VDD_CORE rises above the threshold, the user has the option of generating an HVD reset or an HVW interrupt. The VDD_CORE HVD circuit is OFF by default. The user can enable the VDD_CORE HVD through software.

32.1.3.3 VDD_IO_ABC voltage supply monitors

The following sections describe the voltage monitors for the power supply that powers PORTA, PORTB and PORTC.

32.1.3.3.1 Low Voltage Detect

This device integrates a Low Voltage Detector (LVD) circuit on the input voltage that powers PORTA. The LVD monitors that power supply voltage by comparing it to a programmable threshold. When input voltage powering PORTA drops below the threshold, the user has the option of generating an LVD reset or an LVW interrupt. The LVD circuit for this power supply is ON by default. The user can disable the LVD through software.

32.1.3.3.2 High Voltage Detect

This device integrates a High Voltage Detector (HVD) circuit on the input voltage that powers PORTA. The HVD monitors that power supply voltage by comparing it to a programmable threshold. When the input voltage powering PORTA rises above the threshold, the user has the option of generating an HVD reset or an HVW interrupt. The HVD circuit for this power supply is OFF by default. The user can enable the HVD through software.

32.1.4 SC[ISO_CLR] bits

Each bit in the SC[ISO_CLR] field controls the ISO for one of the chip LV power domains:

- Bit 16 - Core Main domain
- Bit 17 - Core Wake domain
- Bit 18 - Core Radio domain

Refer to Power domains for more details.

32.1.5 EVD_CFG[EVDISO], EVD_CFG[EVDLPISO], and EVD_CFG[EVDSTAT] fields

Each bit of the EVDISO, EVDLPISO, and EVSTAT fields corresponds to one of the IO or analog supplies.

- [0] - VDD_IO_ABC
- [1] - VDD_IO_D
- [2] - VDD_ANA

32.1.6 SRAMCTL[VSM] field

For this device, the VSM has only two options.

- [01b] - SRAM configured for 1.0 V operation and the values in SRAM_TMTR1 is loaded for the read/write timing margin
- [10b] - SRAM configured for 1.1 V operation and the values in SRAM_TMTR2 is loaded for the read/write timing margin

Follow the steps below to change the SRAM settings, refer to #unique_937 for more details.

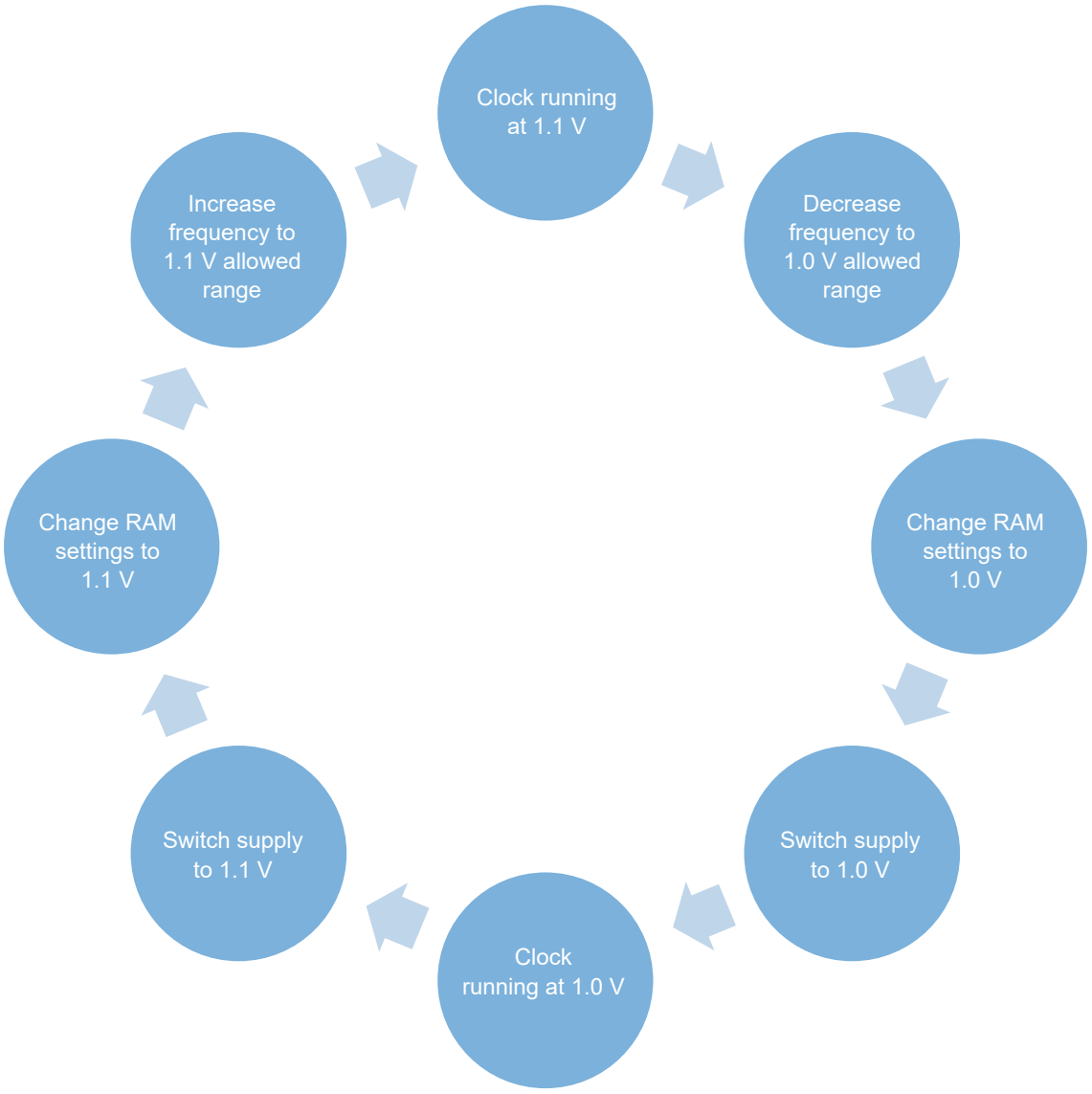


Figure 86. SRAM settings change steps

32.1.7 LP_CFG[IO_HVDE] field

If the LP_CFG[IO_HVDE] is set to 1, it monitors VDD_IO_ABC on PORTA, PORTB and PORTC.

32.1.8 DCDC pulsed mode clocking

DCDC can optionally enter in DCDC pulsed mode. In DCDC pulsed mode, the DCDC refresh period is based on a clock counter which is feed by the 32K_CLK clock in this device.

32.2 Overview

SPC contains and controls the following components:

- Two low-drop-out (LDO) voltage regulators (1.1 V LDO_CORE and 1.8 V LDO_SYS)
- One DCDC used for regulating system, core, and radio power

- Circuits that monitor and assure correct voltage levels:
  - Power-on reset (POR)
  - Low-voltage detect (LVD)
  - Analog glitch detector
  - High-voltage detect (HVD)

SPC turns these components on and off, and switches them to different power mode levels based on power-mode change requests from each of the chip power domains.

32.2.1 Block diagram

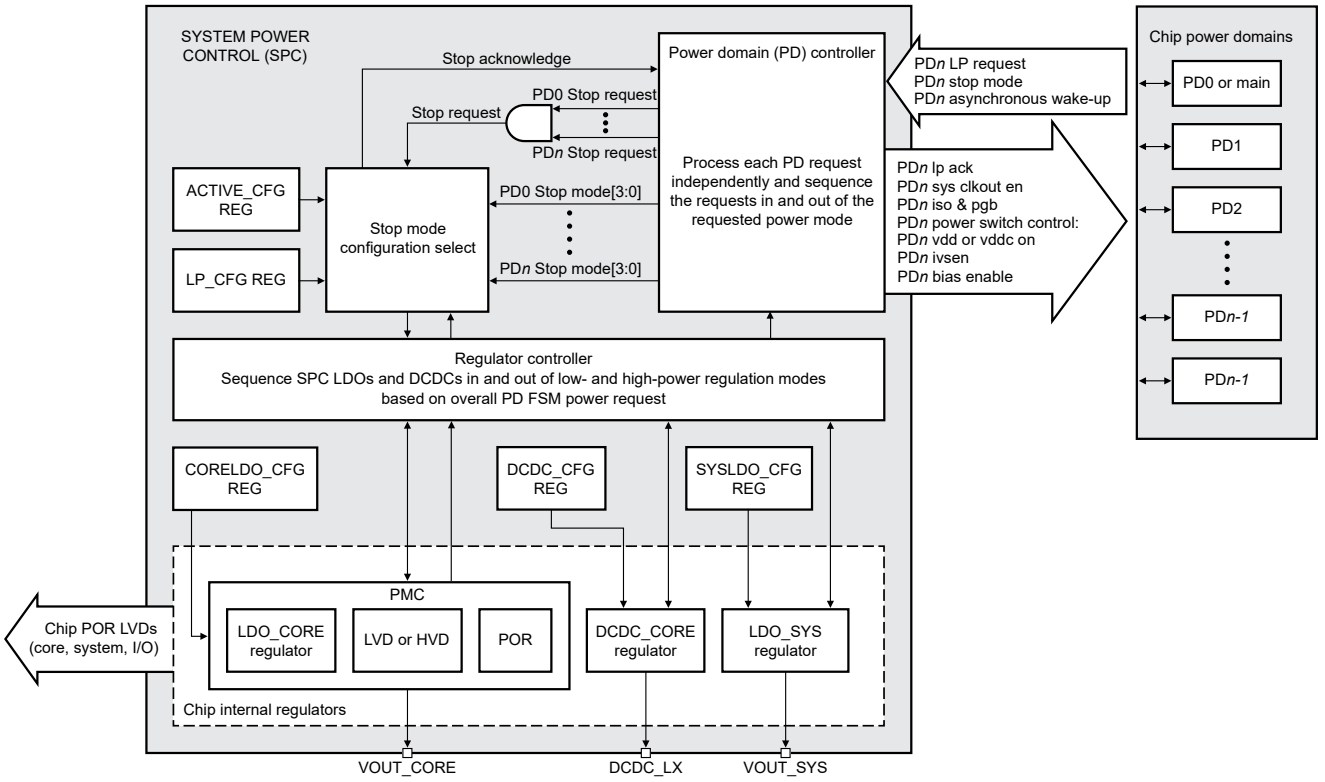


Figure 87. Block diagram

32.2.2 Features

- Selectable core internal voltage regulator (1.2 V LDO_CORE or 1.8 V DCDC)
- Selectable system internal voltage regulator (1.8 V LDO_SYS)
- Radio voltage regulator (1.8 V LDO_SYS or 1.8 V DCDC)
- Active POR providing brown-out detect
- Low-voltage detect
- High-voltage detect
- Core power gate control
- Analog glitch detector

32.3 Functional description

32.3.1 Power mode transitions

Figure 88 shows the power mode state transitions available for each power-down domain. A POR or LVD reset initially returns the power-down domains to ACTIVE state.

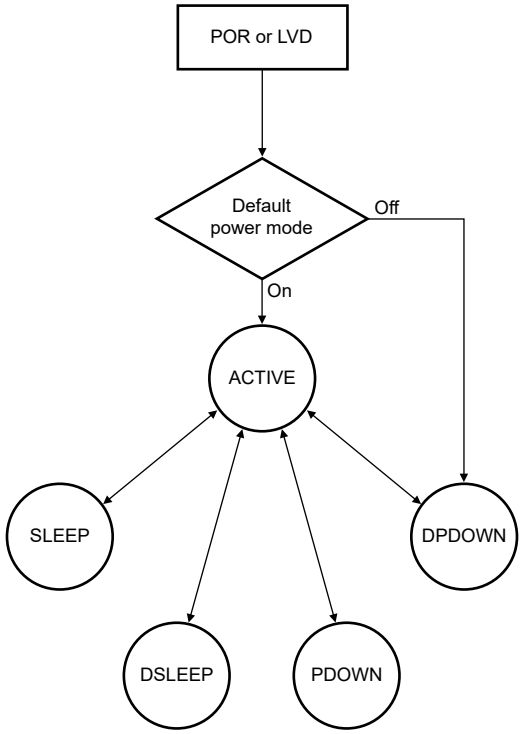


Figure 88. Power mode state diagram

SPC allows each power domain to enter or exit low-power modes independently of the other. You must ensure that any common system resources (for example, clock sources) are always available whenever required for a power-domain current power mode. For example, if a clock source in power domain 0 drives power domain 1, you must ensure that power domain 0 does not enter a low-power mode, because that would disable the clock source.

Figure 89 defines the low-power entry and exit flow for each of the SPC power-down domains.

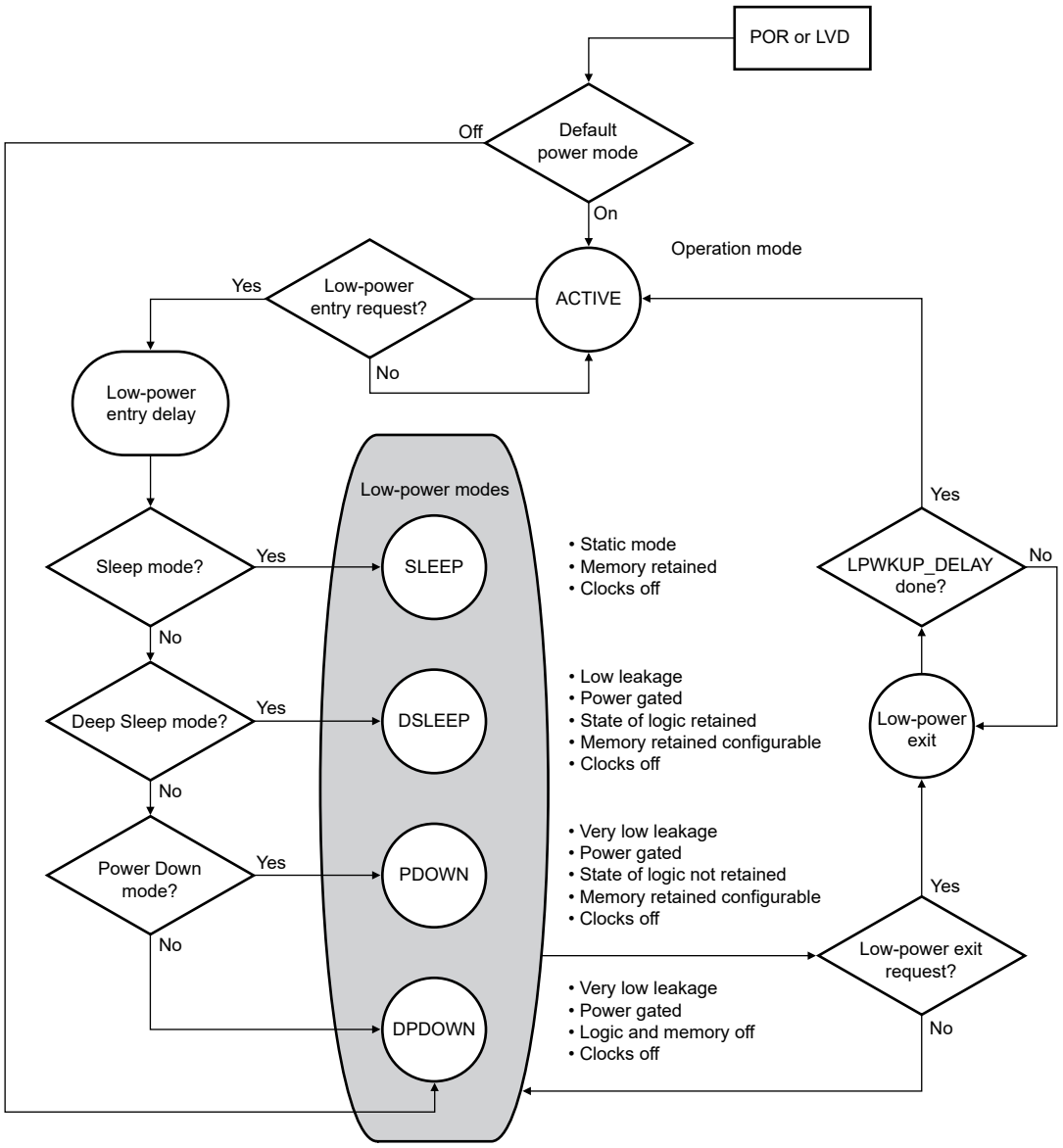


Figure 89. SPC regulator power mode state diagram

NOTE

This section mentions the power modes supported by the module. For the power modes supported on your device, see the chip-specific Power management chapter.

32.3.2 Active mode voltage updates

This section describes voltage and frequency changes in Active mode.

You can alter the operating voltage and frequency during Active mode to optimize power consumption depending on the application's performance requirements.

When increasing voltage and frequency in Active mode, you must perform the following steps:

1. Increase voltage to a new level (ACTIVE_CFG[CORELDO_VDD_LVL] and/or ACTIVE_CFG[DCDC_VDD_LVL]).
2. Wait for voltage change to complete (SC[BUSY] = 0).

3. Configure flash memory to support higher voltage level and frequency (FMU_FCTRL[RWSC].
4. Configure SRAM to support higher voltage levels (SRAMCTL[VSM]).
5. Request SRAM voltage update (write 1 to SRAMCTL[REQ]).
6. Wait for SRAM voltage change to complete (SRAMCTL[ACK] = 1).
7. Clear request for SRAM voltage change (write 0 to SRAMCTL[REQ]).
8. Increase frequency to a new level (for example, SCG_RCCR).
9. You can continue execution.

When decreasing voltage and frequency in Active mode perform the following steps:

1. Decrease frequency to a new level (for example, SCG_RCCR).
2. Configure flash memory to support lower voltage level and frequency (FMU_FCTRL[RWSC].
3. Configure SRAM to support lower voltage levels (SRAMCTL[VSM]).
4. Request SRAM voltage update (write 1 to SRAMCTL[REQ]).
5. Wait for SRAM voltage change to complete (SRAMCTL[ACK] = 1).
6. Clear request for SRAM voltage change (write 0 to SRAMCTL[REQ]).
7. Decrease voltage to a new level (ACTIVE_CFG[CORELDO_VDD_LVL] and/or ACTIVE_CFG[DCDC_VDD_LVL]).
8. Wait for voltage change to complete (SC[BUSY] = 0).
9. You can continue execution.

32.3.3 Low-Power Request (LPREQ) pin

The LPREQ pin asserts after low-power entry and negates after low-power wakeup.

The pin is intended to communicate with an external PMIC (for example, to enter low-power mode) or to control external switches (for example, if certain power supply rails are switched off in the low-power mode). The pin can eliminate the latency when communicating with PMIC—for example, through I^2C—for quick power-state transitions. The PMIC can be configured to use the LPREQ pin to switch between two preconfigured power states.

SPC controls the state of the LPREQ pin based on how you configure Low-Power Request Configuration (LPREQ_CFG). You control the LPREQ pin in Active mode. SPC controls the pin when the chip transitions from Active to a low-power mode, and after wake-up from these power modes.

NOTE

If the power supply rails are switched off externally, software must configure the internal isolation of these power domains using External Voltage Domain Configuration (EVD_CFG).

To use the LPREQ pin:

1. Specify the pin polarity (LPREQ_CFG[LPREQPOL]).
2. Enable the pin output (LPREQ_CFG[LPREQOE]).
3. Configure the pin mux for the desired pin using the PORT PCR registers.
4. If the external PMIC or switch needs additional time after wake-up, use Low Power Wake-Up Delay (LPWKUP_DELAY) to extend the wake-up time.

32.3.4 DCDC

DCDC is a hysteretic converter. It is active during a burst period, when it switches the output to charge to a specific voltage. Then DCDC stops switching for a quiet period until the output voltage drops to a lower level, and it does another active burst.

32.3.4.1 DCDC drive strength

DCDC supports multiple drive strength options that you can choose:

- Normal
- Low
- Pulse refresh

The normal and low drive strengths differ in these DCDC characteristics:

- Maximum load current
- Quiescent current
- Transient response

With low drive strength, you can benefit from the lower quiescent current if:

- The load current is below the I_{LOAD} specification in the data sheet.
- Your application can tolerate the slower transient response.

Be careful when choosing low drive strength. If you exceed I_{LOAD} or require faster transient response, you must use normal drive strength. Furthermore, most other DCDC features are only available with normal drive strength.

Pulse-refresh drive strength offers even lower quiescent current in low-power modes, by deactivating DCDC when not active. This strength has an active pulsing period where DCDC charges the output voltage. Then DCDC deactivates for a configurable amount of time. No analog comparator is powered to monitor the DCDC output voltage. Instead, this drive strength is time-based using an SPC timer where DCDC_BURST_CFG[PULSE_REFRESH_CNT] specifies the timer period. Every period, SPC enables DCDC to charge the voltage again. When DCDC is off, no voltage regulation occurs. You must specify DCDC_BURST_CFG[PULSE_REFRESH_CNT] to manage the voltage drop on the DCDC capacitor and avoid a brownout condition.

32.3.4.2 DCDC burst

You can control and monitor the DCDC burst. Synchronizing with the quiet period between bursts can be valuable for noise-sensitive applications—for example, during a high-resolution ADC measurement or a radio transmission.

Before requesting a burst to start, clear the DCDC_BURST_CFG[BURST_ACK] flag. Then, your firmware can request a burst by writing 1 to DCDC_BURST_CFG[BURST_REQ]. SPC sets the DCDC_BURST_CFG[BURST_ACK] flag when the burst has completed and DCDC enters a quiet period.

SPC also provides signals to synchronize the burst activity with other peripherals in the chip. If DCDC_BURST_CFG[EXT_BURST_EN] = 1 before a burst starts, the output DCDC_BURST_TRIG_PULSE signal asserts when a burst completes, and can trigger other actions such as an ADC conversion. Another SPC output signal is DCDC_BURST_ACTIVE, which EXT_BURST_EN does not gate. DCDC_BURST_ACTIVE can also trigger other peripherals such as a timer. You can use this signal to estimate the power that DCDC delivers.

When bursting, you can control the DCDC switching center frequency with the frequency-stabilization feature. DCDC_CFG[FREQ_CNTRL_ON] enables this feature, and when the feature is enabled, DCDC_CFG[FREQ_CNTRL] controls it. The main parameters that control the period between active bursts and quiet period are:

- Load current
- External capacitor
- DCDC hysteresis

NOTE

Writing 1 to DCDC_CFG[FREQ_CNTRL_ON] stabilizes the frequency by limiting DCDC current. See the DCDC I_{LOAD} maximum limit in the data sheet for this mode.

32.3.4.3 DCDC voltage changes

To support power optimization of applications, DCDC can provide multiple output voltages.

Two separate registers, which specify Active mode output and Low-Power mode output, independently control the DCDC output voltage. Whenever an SoC mode change occurs, by using two independent registers, there is no need to reconfigure the output voltage. ACTIVE_CFG[DCDC_VDD_LVL] controls the DCDC output voltage when the SoC mode is Active or in Sleep mode. The DCDC drive strength is normal (ACTIVE_CFG[DCDC_VDD_DS] = 10b). LP_CFG[DCDC_VDD_LVL] controls the DCDC output voltage when the SoC mode is in a Low-Power mode (i.e., not Active or Sleep mode) or the DCDC drive strength is low (ACTIVE_CFG[DCDC_VDD_DS] = 01b) and the SoC is in an Active or Sleep mode.

When Active Power Mode Configuration (ACTIVE_CFG) controls the output, DCDC output options present:

- 1.25 V
- 1.35 V
- 1.8 V
- 2.5 V

See Active mode voltage updates to change the DCDC output level in Active mode.

When Low-Power Mode Configuration (LP_CFG) controls the output, DCDC output options present:

- 1.25 V
- 1.35 V
- 1.8 V

You have to configure LPWKUP_DELAY to the correct value. In Deep Sleep mode, you must also align the output voltage with the clock frequencies of any clocks kept alive in the Low-Power mode. If you keep some clock sources active, ensure to leave the bandgap enabled.

To change the DCDC output level in Low-Power mode, follow this procedure:

1. Configure LP_CFG[DCDC_VDD_LVL] to desired level.
2. Configure LP_CFG[DCDC_VDD_DS] = 0x1.
3. Configure ACTIVE_CFG[DCDC_VDD_LVL] to same level as in configure LP_CFG[DCDC_VDD_LVL].

32.3.5 Clocking

Table 218. Clocks

SPC component:	Clock source
Registers and logic	Slow bus clock
Delay counters	10 MHz internal clock

SPC uses the internal clock that PMC analog block generates to sequence the system regulator and chip power domains in and out of different power modes. This clock has a frequency of 10 MHz. SPC automatically enables the clock when it detects a power mode change, including entry into low-power modes and exit from low-power modes of any of the system power domains, or configuration changes to either Active Power Mode Configuration (ACTIVE_CFG) or Low-Power Mode Configuration (LP_CFG). This clock cannot run unless the PMC bandgap is up and enabled. Once SPC completes all transactions and is not busy, that is, SC[BUSY] = 0, this clock will turn off synchronously.

32.3.6 Reset

32.3.6.1 Reset sources

32.3.6.1.1 Power-on reset (POR)

When you initially apply power to the chip, or the supply voltage is below the POR falling threshold, the POR circuit triggers the POR condition. The POR condition asserts a cold reset in all power domains.

32.3.6.1.2 Low-voltage detect (LVD)

SPC supports the following LVD circuits:

- Core VDD
- System VDD
- I/O VDD

Each of these circuits is enabled by default and keeps the chip in reset until each of these supply voltages rises above the LVD rising threshold. Enabling any of the LVD circuits triggers an LVD reset condition if the corresponding supply voltage is below the LVD falling threshold.

The LVD reset condition asserts a cold reset in all power domains. The LVD- and HVD-detection logic is the only logic that HVD and LVD resets do not affect. The corresponding status fields in Voltage Detect Status (VD_STAT) become 1 after the corresponding LVD reset condition occurs.

32.3.6.1.3 High-voltage detect (HVD)

SPC supports the following HVD circuits:

- Core VDD
- System VDD
- I/O VDD

Each of these circuits is enabled by default and triggers an HVD reset condition if the corresponding supply voltage is above the HVD threshold.

Any of these reset condition asserts a cold reset in all power domains. The LVD- and HVD-detection logic is the only logic that HVD and LVD resets do not affect. The corresponding status fields in Voltage Detect Status (VD_STAT) become 1 after the corresponding HVD reset condition occurs.

32.3.6.1.4 Glitch detect

This chip guards against short pulses on the core VDD and VSS. This security feature can detect glitches due to a hacking attack and alerts the core to reset the chip or generate an interruptn operation.

After power-on reset, the core VDD glitch-detect system is enabled. When the system is in Active mode, use ACTIVE_CFG[GLITCH_DETECT_DISABLE] to enable or disable the glitch-detect system. When the system is in Low-Power mode, use LP_CFG[GLITCH_DETECT_DISABLE] to enable or disable the glitch-detect system.

The glitch-detect system uses a 4-bit ripple counter to monitor the quantity of detected glitches. The VDD_CORE_GLITCH_DETECT_SC[GLITCH_DETECT_FLAG] status flags capture the state of each of these counters.

NOTE

The chip has analog voltage glitch detector (aGDET) that enhances protection against voltage manipulation. After the chip is out of reset, the NXP FMU logic configures aGDET to reset the chip when voltage manipulation is detected. The aGDET is validated on silicon across multiple process, voltage, and temperature corners by activating a large number of possible data paths. To avoid false detections due to unforeseen environment or application behavior, NXP recommends disabling aGDET in the early start-up code. You can also configure hardware interrupt for aGDET and handle action in the interrupt handler by disabling default reset actions. aGDET can be used by the application when security-critical operations, for example, cryptographic operation using private or symmetric keys (digital signature, encryption, decryption, and so on) or sensitive decision-making process. However, you must ensure that the system remains free from false detections in those circumstances. See the SDK example in this document for details on sensor usage.

32.3.6.2 Reset sequence

The following steps occur as part of the POR, LVD, or HVD sequence:

1. The RESET_b pin deasserts.
2. The internal reset signals assert.
3. The POR or LVD signals negate.
4. System clocks are enabled.

32.3.7 Interrupts

SPC generates a single interrupt. The trigger for this interrupt comes from:

- Any of the LVD, HVD, or glitch-detect circuits when they are not configured for reset and their corresponding HVDIE or LVDIE register.
- The Interrupt Enable (IE) register (for the glitch-detect case).

32.4 External signals

Signal	Description	Direction
LPREQ	Low-power request pin used to signal external power-management circuits for a change in supply voltage	Output
VOUT_CORE	Output of the LDO_CORE regulator	Output
VOUT_SYS	Output of the LDO_SYS regulator	Output
DCDC_LX	Output of the DCDC_CORE regulator	Output

32.5 Initialization

To initialize SPC with LDO_CORE off, write 0 to CNTRL[CORELDO_EN].

To initialize SPC with LDO_SYS off, write 0 to CNTRL[SYSLDO_EN].

To initialize SPC with DCDC_CORE off, write 0 to CNTRL[DCDC_EN].

Except for the above, SPC does not require any special initialization.

32.6 Application information

To change DCDC level in Low-Power mode:

1. Configure LP_CFG[DCDC_VDD_LVL] to desired level.

2. Configure LP_CFG[DCDC_VDD_DS] = 0x1
3. Configure ACTIVE_CFG[DCDC_VDD_LVL] to same level programmed in Configure LP_CFG[DCDC_VDD_LVL]

To disable Bandgap in Active mode:

1. Disable all LVD's and HVD's in ACTIVE_CFG[29:24]= 0x00
2. Disable Glitch Detect, ACTIVE_CFG[GLITCH_DETECT_DISABLE] = 1
3. Configure LDO's and DCDC to Low Drive Strength in ACTIVE_CFG register
4. Configure ACTIVE_CFG[BGMODE] = 0x0

To disable Bandgap in Low-Power mode:

1. Disable all LVD's and HVD's in LP_CFG[29:24]= 0x00
2. Disable Glitch Detect, LP_CFG[GLITCH_DETECT_DISABLE] = 1
3. Configure LDO's and DCDC to Low Drive Strength in LP_CFG register
4. Configure LP_CFG[BGMODE] = 0x0

32.7 SPC register descriptions

Different chip reset types affect the reset of different SPC registers. Each register description provides details. See the Reset chapter for more information about the types of reset on this chip.

You may use only 32-bit writes for any writable SPC registers. 8-bit or 16-bit writes cause a transfer error.

32.7.1 SPC memory map

SPC0 base address: 4001_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0000_0000h
10h	Status Control (SC)	32	RW	0000_0000h
14h	SPC Regulator Control (CNTRL)	32	RW	0000_0007h
1Ch	Low-Power Request Configuration (LPREQ_CFG)	32	RW	0000_0000h
20h	SPC Configuration (CFG)	32	RW	0000_0000h
30h - 38h	SPC Power Domain Mode Status (PD_STATUS0 - PD_STATUS2)	32	RW	0000_0000h
40h	SRAM Control (SRAMCTL)	32	RW	0000_0001h
E0h	General Purpose Wake-up (WAKEUP)	32	RW	0000_0000h
100h	Active Power Mode Configuration (ACTIVE_CFG)	32	RW	3F10_0E15h
104h	Low-Power Mode Configuration (LP_CFG)	32	RW	0002_1D04h
120h	Low Power Wake-Up Delay (LPWKUP_DELAY)	32	RW	0000_0000h
124h	Active Voltage Trim Delay (ACTIVE_VDELAY)	32	RW	0000_00C8h
130h	Voltage Detect Status (VD_STAT)	32	RW	0000_0000h
134h	Core Voltage Detect Configuration (VD_CORE_CFG)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138h	System Voltage Detect Configuration (VD_SYS_CFG)	32	RW	0000_0001h
13Ch	IO Voltage Detect Configuration (VD_IO_CFG)	32	RW	0000_0101h
140h	External Voltage Domain Configuration (EVD_CFG)	32	RW	0000_0000h
144h	Glitch Detect Status Control (GLITCH_DETECT_SC)	32	RW	0000_003Fh
300h	LDO_CORE Configuration (CORELDO_CFG)	32	RW	0000_0000h
400h	LDO_SYS Configuration (SYSLDO_CFG)	32	RW	0000_0101h
500h	DCDC Configuration (DCDC_CFG)	32	RW	0000_0000h
504h	DCDC Burst Configuration (DCDC_BURST_CFG)	32	RW	0140_0000h

32.7.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Indicates:

- The version integrated for this instance on the chip.
- Inclusion or exclusion of several optional features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Specification Number Indicates the feature set number. 0000_0000_0000_0000b - Standard features All other values are reserved.

32.7.3 Status Control (SC)

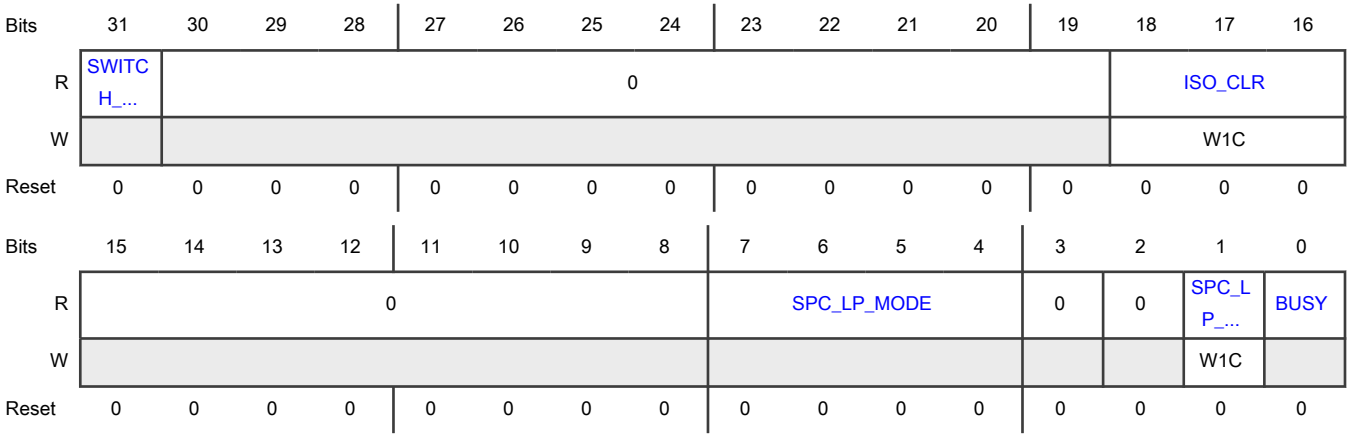
Offset

Register	Offset
SC	10h

Function

Indicates the current SPC status.

Diagram



Fields

Field	Function
31	Power Switch State

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWITCH_STATE	Indicates the power switch state. 0b - Off 1b - On
30-19 —	Reserved
18-16 ISO_CLR	Isolation Clear Flags Indicates whether certain peripherals and I/O pads are in a latched state as a result of having been in PDOWN mode. Each bit of this field is a flag associated with a module on this chip. See the chip-specific SPC information for the module-bit association. For each flag: • A value of 0 means that peripherals and I/O pads are in the normal run state. • A value of 1 means that peripherals and I/O pads can retain their state in their power domain. If a flag in this field is 1, writing 1 to that flag clears the flag and releases the I/O pads and peripherals to their normal run-mode state. After recovering from a power-down mode, you must restore the chip configuration before clearing a flag in this field. In particular, you must restore the pin configuration for enabled WUU wake-up to avoid any WUU flag from being falsely set when the associated ISO_CLR flag is cleared. This field resets after a system reset.
15-8 —	Reserved
7-4 SPC_LP_MODE	Power Domain Low-Power Mode Request Indicates the last low-power mode that the power domain requested. 0000b - Sleep mode with system clock running 0001b - SLEEP with system clock off 0010b - DSLEEP with system clock off 0100b - PDOWN with system clock off 1000b - DPDOWN with system clock off
3 —	Reserved
2 —	Reserved
1 SPC_LP_REQ	SPC Power Mode Configuration Status Flag Indicates when all power-down domains requested low-power mode and SPC entered a low-power state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field behaves differently for register reads and writes. When reading 0b - SPC is in Active mode; the ACTIVE_CFG register has control 1b - All power domains requested low-power mode; SPC entered a low-power state; power-mode configuration based on the LP_CFG register When writing 0b - No effect 1b - Clear the flag
0 BUSY	SPC Busy Status Flag Indicates whether SPC is busy. SPC sets this flag: • When SPC executes any type of power mode transition in Active mode or any of the chip low-power modes. • Due to LP_CFG[CORELDO_VDD_LVL] and LP_CFG[DCDC_VDD_LVL] changes while in Active mode. **NOTE** Wait until this flag is clear before changing power-mode configuration registers. 0b - Not busy 1b - Busy

32.7.4 SPC Regulator Control (CNTRL)

Offset

Register	Offset
CNTRL	14h

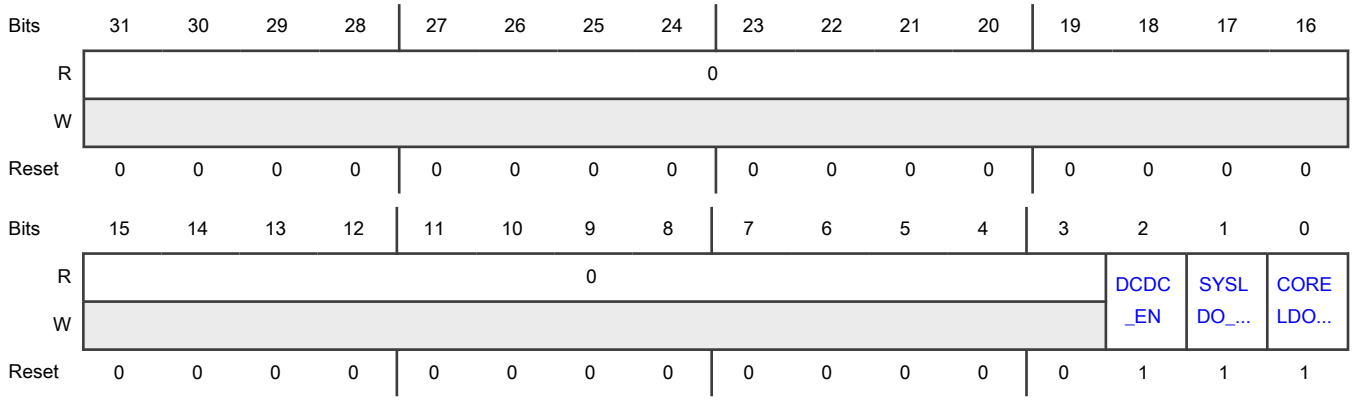
Function

Enables the SPC regulators.

You can write to the fields in this register only one time.

The register resets only after a POR, LVD, or HVD event.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 DCDC_EN	DCDC_CORE Regulator Enable 0b - Disable 1b - Enable
1 SYSLDO_EN	LDO_SYS Regulator Enable 0b - Disable 1b - Enable
0 CORELDO_EN	LDO_CORE Regulator Enable 0b - Disable 1b - Enable

32.7.5 Low-Power Request Configuration (LPREQ_CFG)

Offset

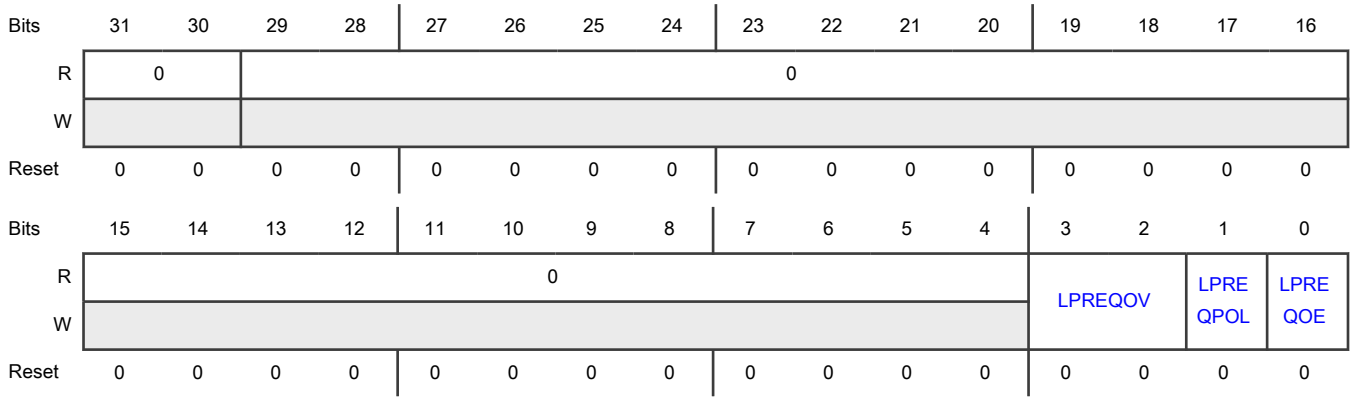
Register	Offset
LPREQ_CFG	1Ch

Function

Configures the low-power output request pin.

The register resets only after a POR, LVD, or HVD event.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-4 —	Reserved
3-2 LPREQOV	Low-Power Request Output Override Forces the low-power request pin high. 00b - Not forced 01b - Reserved 10b - Forced low (ignore LPREQPOL settings) 11b - Forced high (ignore LPREQPOL settings)
1 LPREQPOL	Low-Power Request Output Pin Polarity Control Controls the true polarity of the low-power request output pin. 0b - High 1b - Low
0 LPREQOE	Low-Power Request Output Enable Enables the low-power request output pin. 0b - Disable 1b - Enable

32.7.6 SPC Configuration (CFG)

Offset

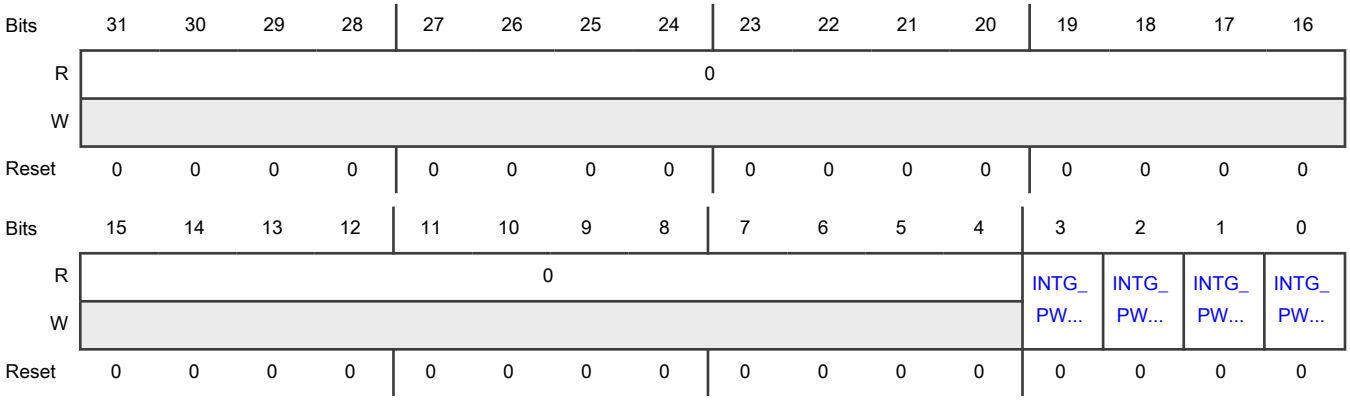
Register	Offset
CFG	20h

Function

Controls the features of the integrated power switch that you can use to gate board-level supplies.

The register resets only after a POR, LVD, or HVD event.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 INTG_PWSWTCH_WKUP_ACTIVE_EN	Integrated Power Switch Wake-up Enable Enables the switch in active mode. When this field is 1, SPC drives a pin in active mode to close the switch. 0b - Disable 1b - Enable
2 INTG_PWSWTCH_SLEEP_ACTIVE_EN	Integrated Power Switch Active Enable Disables the switch in active mode. When this field is 1, SPC drives a pin in active mode to open the switch. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 INTG_PWSWTCH_WKUP_EN	Integrated Power Switch Wake-up Enable Enables the switch after waking up from any low-power mode. When this field is 1, SPC drives a pin during low-power wake-up sequence to close the switch. 0b - Disable 1b - Enable
0 INTG_PWSWTCH_SLEEP_EN	Integrated Power Switch Sleep Enable Disables the switch entering in any low-power mode. When this field is 1, SPC drives a pin during low-power entry sequence to open the switch. 0b - Disable 1b - Enable

32.7.7 SPC Power Domain Mode Status (PD_STATUS0 - PD_STATUS2)

Offset

Register	Offset
PD_STATUS0	30h
PD_STATUS1	34h
PD_STATUS2	38h

Function

Indicates power mode status for each of the SPC power domains. See the chip-specific SPC information to determine which power domains are associated with this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				LP_MODE				0			PD_LP_R...	0			PWR_REQ...
W												W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-8 LP_MODE	Power Domain Low Power Mode Request Indicates the last low-power mode that the power domain requested. 0000b - SLEEP with system clock running 0001b - SLEEP with system clock off 0010b - DSLEEP with system clock off 0100b - PDOWN with system clock off 1000b - DPDOWN with system clock off
7-5 —	Reserved
4 PD_LP_REQ	Power Domain Low Power Request Flag Set when low power mode was requested by Power Domain since last cleared by software. 0b - Did not request 1b - Requested
3-1 —	Reserved
0 PWR_REQ_STATUS	Power Request Status Flag Indicates whether the power domain requested low-power mode. 0b - Did not request 1b - Requested

32.7.8 SRAM Control (SRAMCTL)

Offset

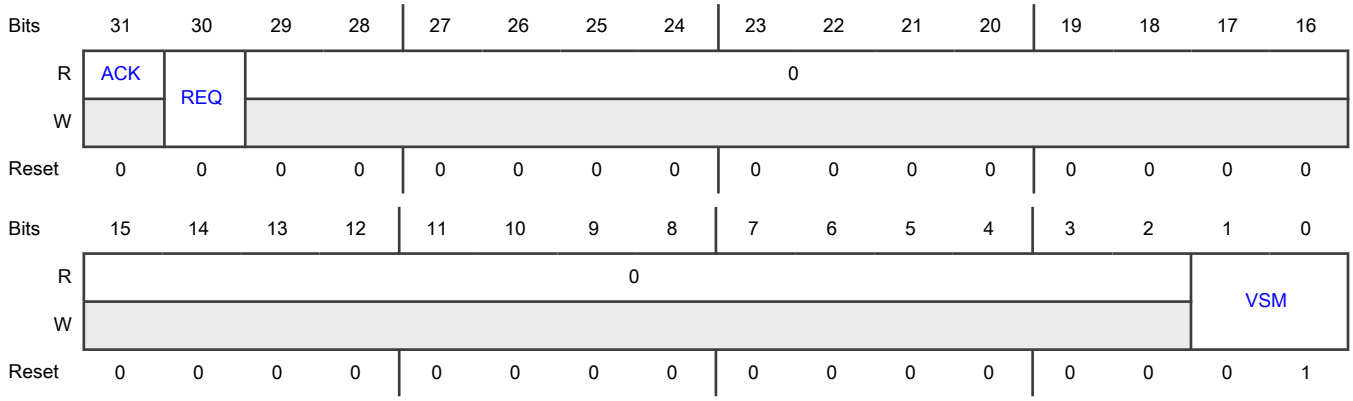
Register	Offset
SRAMCTL	40h

Function

Configures the SRAM timing for different voltage levels.

When transitioning between two voltage levels, you must keep the SRAM timing at the lowest voltage level until the voltage change is complete.

Diagram



Fields

Field	Function
31 ACK	SRAM Voltage Update Request Acknowledge Indicates whether SPC acknowledged the request for an SRAM trim-value change. 0b - Not acknowledged 1b - Acknowledged
30 REQ	SRAM Voltage Update Request Allows you to request an SRAM trim value change. After requesting the change, you must wait for the ACK field to become 1, then write 0 to REQ. 0b - Do not request 1b - Request
29-2 —	Reserved
1-0 VSM	Voltage Select Margin Specifies the operating voltage for the SRAM's read/write timing margin. 00b - Reserved 01b - 1.0 V 10b - 1.1 V 11b - SRAM configured for 1.1 V operation

32.7.9 General Purpose Wake-up (WAKEUP)

Offset

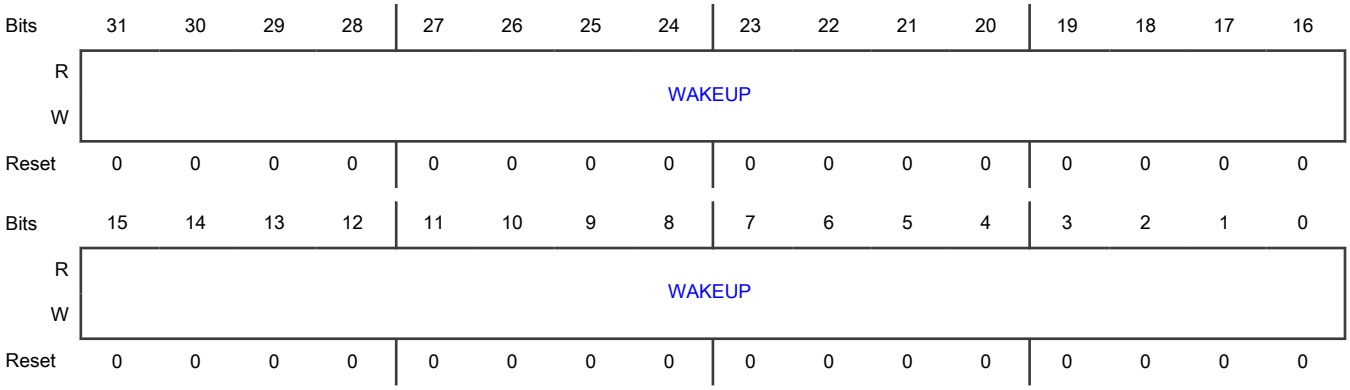
Register	Offset
WAKEUP	E0h

Function

Specifies the wake-up value that the boot ROM uses.

This register resets after POR, LVD ,or any system reset except wake-up from Power Down or Deep Power Down modes.

Diagram



Fields

Field	Function
31-0 WAKEUP	Wake-up Specifies the value that the boot ROM uses to quickly restore Cortex-M33 core context or to switch execution to a defined address in the flash memory or SRAM after wake-up.

32.7.10 Active Power Mode Configuration (ACTIVE_CFG)

Offset

Register	Offset
ACTIVE_CFG	100h

Function

Controls the SPC regulators settings in the Active power mode.

Changes to the drive strength or voltage level for any of the LDOs or DCDC sets the SC[BUSY] flag until SPC changes its state to the new value.

NOTE

The LVDE and HVDE fields reset only with a POR. All other fields reset only with a system reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		IO_HVDE	SYS_HVDE	CORE_HV...	IO_LVDE	SYS_LVDE	CORE_LV...	0	0	BGMODE		0	LPBUFF_...	0	
W																
Reset	0	0	1	1	1	1	1	1	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			GLITCH_...	DCDC_VDD_LVL		DCDC_VDD_DS		0	SYSLDO_...	0	SYSLDO_...	CORELDO_VDD_LVL		0	CORELDO...
W																
Reset	0	0	0	0	1	1	1	0	0	0	0	1	0	1	0	1

Fields

Field	Function
31-30 —	Reserved
29 IO_HVDE	IO High-Voltage Detection Enable Enables the IO high-voltage detection. When IO_HVDE = 1, you must write a value to BGMODE that enables the bandgap. Only a POR resets this field. This field monitors VDD IO on PORT. See the chip-specific SPC information for port details. 0b - Disable 1b - Enable
28 SYS_HVDE	System High-Voltage Detection Enable Enables the system high-voltage detection. When SYS_HVDE = 1, you must write a value to BGMODE that enables the bandgap. Only a POR resets this field. 0b - Disable 1b - Enable
27 CORE_HVDE	Core High-Voltage Detection Enable Enables the core high-voltage detection. When CORE_HVDE = 1, you must write a value to BGMODE that enables the bandgap. Only a POR resets this field. 0b - Disable 1b - Enable
26	IO Low-Voltage Detection Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
IO_LVDE	Enables the IO low-voltage detection. When IO_LVDE = 1, you must write a value to BGMODE that enables the bandgap. Only a POR resets this field. This field monitors VDD IO on PORT. See the chip-specific SPC information for port details. 0b - Disable 1b - Enable
25 SYS_LVDE	System Low-Voltage Detection Enable Enables the system low-voltage detection. When SYS_LVDE = 1, you must write a value to BGMODE that enables the bandgap. Only a POR resets this field. 0b - Disable 1b - Enable
24 CORE_LVDE	Core Low-Voltage Detection Enable Enables the core low-voltage detection. When CORE_LVDE = 1, you must write a value to BGMODE that enables the bandgap. Only a POR resets this field. 0b - Disable 1b - Enable
23 —	Reserved
22 —	Reserved
21-20 BGMODE	Bandgap Mode Specifies the bandgap mode configuration. SPC forces BGMODE to 01b to enable the Bandgap, if: • A write to disable the Bandgap is detected. • Any of the regulators are in normal drive strength. • Any of the LVD or HVDs are enabled. • VDD CORE glitch detection is enabled. 00b - Bandgap disabled 01b - Bandgap enabled, buffer disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Bandgap enabled, buffer enabled 11b - Reserved
19 —	Reserved
18 LPBUFF_EN	CMP Bandgap Buffer Enable Enables the buffer-stored reference voltage to CMP. 0b - Disable 1b - Enable
17-13 —	Reserved
12 GLITCH_DETECT_DISABLE	Glitch Detect Disable Reset on POR only State of GLITCH_DETECT_DISABLE will be ignored if Bandgap is disabled and Glitch Detect hardware will be forced to OFF state. 0b - Low Voltage Glitch Detect enabled 1b - Low Voltage Glitch Detect disabled
11-10 DCDC_VDD_LVL	DCDC VDD Regulator Voltage Level Specifies the DCDC VDD regulator level. When switching the DCDC drive strength from low to normal, ensure that the DCDC high VDD LVL setting is set to the same level that was set prior to switching the DCDC to low drive strength. Otherwise, if the LVDs are enabled, an unexpected LVD can occur. 00b - Low undervoltage (1.25 V) 01b - Midvoltage (1.35 V) 10b - Normal voltage (2.5 V) 11b - Safe-mode voltage (1.8 V)
9-8 DCDC_VDD_DS	DCDC VDD Drive Strength Specifies the DCDC VDD regulator drive strength. If you specify normal drive strength, you must write a value to ACTIVE_CFG[BGMODE] that enables the bandgap. 01b - Low 10b - Normal All other values are reserved.
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 SYSLDO_VDD_LVL	LDO_SYS VDD Regulator Voltage Level Specifies the LDO_SYS VDD regulator level. You must write 0 to SYS_HVDE before writing 1 to SYSLDO_VDD_LVL. The VDD_SYS voltage can only operate at the overdrive voltage for a limited amount of time for the life of the chip. See the chip data sheet for the maximum voltage and time specifications. This is intended only for programming eFuses. 0b - Normal voltage (1.8 V) 1b - Overdrive voltage (2.5 V)
5 —	Reserved
4 SYSLDO_VDD_DS	LDO_SYS VDD Drive Strength Specifies the LDO_SYS VDD regulator drive strength If you specify normal drive strength, you must write a value to ACTIVE_CFG[BGMODE] that enables the bandgap. If you have enabled LVDs or HVDs, and attempt to specify low drive strength, SPC ignores the attempt. SPC forces a normal drive strength when SYSLDO_VDD_LVL = 1. 0b - Low 1b - Normal
3-2 CORELDO_VDD_LVL	LDO_CORE VDD Regulator Voltage Level Selects the LDO_CORE VDD regulator level. If the CORELDO_VDD_DS fields are set to the same value in both the ACTIVE_CFG and LP_CFG registers, the CORELDO_VDD_LVL's in the ACTIVE_CFG and LP_CFG register must be set to the same voltage level settings. You can change the core VDD levels for the LDO_CORE low power regulator only when CORELDO_VDD_DS=1. When switching CORELDO_VDD_DS from low to normal drive strength, ensure the LDO_CORE high VDD LVL setting is set to the same level that was set prior to switching to the LDO_CORE drive strength (CORELDO_VDD_DS). Otherwise, if the LVDs are enabled, an unexpected LVD can occur. You must specify the same level for both CORELDO_VDD_LVL and DCDC_VDD_LVL, even if LDO_CORE is off. 00b - Reserved 01b - Regulate to mid voltage (1.05 V) 10b - Regulate to normal voltage (1.1 V) 11b - Regulate to safe-mode voltage (1.15 V)

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 —	Reserved
0 CORELDO_VDD_DS	LDO_CORE VDD Drive Strength Selects the LDO_CORE VDD regulator drive strength **NOTE** When setting CORELDO_VDD_DS to Normal, BGMODE must be programmed to a value that enables the Bandgap. Writes to set drive strength to Low will be ignored if LVD/HVDs are kept enabled. 0b - Low 1b - Normal

32.7.11 Low-Power Mode Configuration (LP_CFG)

Offset

Register	Offset
LP_CFG	104h

Function

Controls the SPC regulator settings in low-power stop modes.

This register resets only after a POR or LVD event.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		IO_HVDE	SYS_HVDE	CORE_HV...	IO_LVDE	SYS_LVDE	CORE_LV...	LP_IREF...	0	BGMODE		0	LPBUFF_...	CORE VDD...	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			GLITCH_...	DCDC_VDD_LVL		DCDC_VDD_DS		0	0		SYSLDO_...	CORELDO_VDD_LVL		0	CORE LDO...
W																
Reset	0	0	0	1	1	1	0	1	0	0	0	0	0	1	0	0

Fields

Field	Function
31-30 —	Reserved
29 IO_HVDE	IO High Voltage Detect Enable This field monitors VDD IO on PORT. For port details, see the chip-specific SPC information. This field resets only after a POR, LVD, or HVD event. When IO_HVDE = 1, you must use LP_CFG[BGMODE] to enable the bandgap. If you enable the bandgap to support voltage detection, the low-power mode Idd increases. 0b - Disable 1b - Enable
28 SYS_HVDE	System High Voltage Detect Enable This field resets only after a POR, LVD, or HVD event. When SYS_HVDE = 1, you must use LP_CFG[BGMODE] to enable the bandgap. If you enable the bandgap to support voltage detection, the low-power mode Idd increases. 0b - Disable 1b - Enable
27 CORE_HVDE	Core High Voltage Detect Enable This field resets only after a POR, LVD, or HVD event. When CORE_HVDE = 1, you must use LP_CFG[BGMODE] to enable the bandgap. If you enable the bandgap to support voltage detection, the low-power mode Idd increases. 0b - Disable 1b - Enable
26 IO_LVDE	IO Low Voltage Detect Enable This field monitors VDD IO on PORT. For port details, see the chip-specific SPC information. This field resets only after a POR, LVD, or HVD event. When IO_LVDE = 1, you must use LP_CFG[BGMODE] to enable the bandgap. If you enable the bandgap to support voltage detection, the low-power mode Idd increases. 0b - Disable 1b - Enable
25 SYS_LVDE	System Low Voltage Detect Enable This field resets only after a POR, LVD, or HVD event. When SYS_LVDE = 1, you must use LP_CFG[BGMODE] to enable the bandgap. If you enable the bandgap to support voltage detection, the low-power mode Idd increases. 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
24 CORE_LVDE	Core Low Voltage Detect Enable This field resets only after a POR, LVD, or HVD event. When CORE_LVDE = 1, you must use LP_CFG[BGMODE] to enable the bandgap. If you enable the bandgap to support voltage detection, the low-power mode Idd increases. 0b - Disable 1b - Enable
23 LP_IREFEN	Low-Power IREF Enable You can disable the low-power IREF only in DPDOWN mode. In all other low-power modes, writing 1 to this field has no effect and the low-power IREF is always enabled. See the chip-specific SPC information for the modules that require the low-power IREF to be enabled in DPDOWN mode when you use the module in DPDOWN mode. 0b - Disable for power saving in Deep Power Down mode 1b - Enable
22 —	Reserved
21-20 BGMODE	Bandgap Mode Specifies the bandgap mode configuration. BGMODE will be set to 01b to keep the Bandgap enabled if a write to disable Bandgap is detected while keeping any of the regulators in normal drive strength or if any of the LVD/HVDs are kept enabled or if VDD CORE glitch detect are kept enabled. 00b - Bandgap disabled 01b - Bandgap enabled, buffer disabled 10b - Bandgap enabled, buffer enabled 11b - Reserved
19 —	Reserved
18 LPBUFF_EN	CMP Bandgap Buffer Enable Enables the buffer-stored reference voltage to CMP. The CMP bandgap buffer is automatically disabled and turned off in DPDOWN mode. 0b - Disable 1b - Enable
17	CORE VDD Internal Voltage Scaling (IVS) Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
COREVDD_IVS_EN	Enables the IVS regulator. When enabled, the IVS regulator scales the external input CORE VDD to a lower voltage level to reduce internal leakage during Deep Sleep and Power Down modes. COREVDD_IVS_EN is automatically 0 in SLEEP or DPDOWN modes. CORE VDD can only get IVS in DSLEEP and PDOWN modes. 0b - Disable 1b - Enable
16 —	Reserved
15-13 —	Reserved
12 GLITCH_DETECT_DISABLE	Glitch Detect Disable This field resets only after a POR, LVD, or HVD event. If you have disabled the bandgap, SPC ignores the value of GLITCH_DETECT_DISABLE and deactivates the glitch-detect hardware. 0b - Enable 1b - Disable
11-10 DCDC_VDD_LVL	DCDC VDD Regulator Voltage Level Specifies the DCDC VDD regulator level. 00b - Low under voltage (1.25 V) 01b - Mid voltage (1.35 V) 10b - Reserved 11b - Safe-mode voltage (1.8 V)
9-8 DCDC_VDD_DS	DCDC VDD Drive Strength Specifies the drive strength of the DCDC VDD regulator. If you specify normal drive strength, you must write a value to LP_CFG[BGMODE] that enables the bandgap. The DCDC regulator is always turned off and disabled in DPDOWN mode. Pulse Refresh mode is invalid in SLEEP mode. If you have enabled LVDs or HVDs, and attempt to specify low or pulse drive strength, SPC ignores the attempt and forces the normal drive strength. SPC ignores writes to reserved values. 00b - Pulse refresh

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Low 10b - Normal 11b - Reserved
7 —	Reserved
6-5 —	Reserved
4 SYSLDO_VDD_DS	LDO_SYS VDD Drive Strength Specifies the drive strength of the LDO_SYS VDD regulator. If you specify normal drive strength, you must write a value to LP_CFG[BGMODE] that enables the bandgap. If you have enabled LVDs or HVDs, and attempt to specify low drive strength, SPC ignores the attempt. 0b - Low 1b - Normal
3-2 CORELDO_VDD_LVL	LDO_CORE VDD Regulator Voltage Level Specifies the LDO_CORE VDD regulator level for low-power sleep modes. If the CORELDO_VDD_DS fields are set to the same value in both the ACTIVE_CFG and LP_CFG registers, the CORELDO_VDD_LVL's in the ACTIVE_CFG and LP_CFG register must be set to the same voltage level settings. You can change the core VDD levels for the LDO_CORE low power regulator only when ACTIVE_CFG[CORELDO_VDD_DS] = 1. So, before entering any of the low-power states (DSLEEP, PDOWN, DPDOWN) with LDO_CORE low power regulator selected (LP_CFG[CORELDO_VDD_DS] = 0), you must use CORELDO_VDD_LVL to select the correct regulation level during ACTIVE run mode. Updating CORELDO_VDD_LVL sets the SC[BUSY] flag. That flag remains set for at least the total time delay that Active Voltage Trim Delay (ACTIVE_VDELAY) specifies. Before changing CORELDO_VDD_LVL, you must wait until the SC[BUSY] flag clears before entering the selected low-power sleep mode. This ensures the correct core VDD voltage levels are set after the chip is in the chosen mode. When LP_CFG[DCDC_VDD_LVL] is not 00b, you must program LP_CFG[CORELDO_VDD_LVL] to have same value as LP_CFG[DCDC_VDD_LVL], even if LDO_CORE is off. When LP_CFG[DCDC_VDD_LVL] is 00b, you must disable VDD_CORE LVD in Low-Power mode, and LP_CFG[CORELDO_VDD_LVL] can be different with LP_CFG[DCDC_VDD_LVL]. 00b - Retention voltage 01b - Mid voltage (1.05 V) 10b - Normal voltage (1.1 V) 11b - Safe-mode voltage (1.15 V)

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 —	Reserved
0 CORELDO_VDD_DS	LDO_CORE VDD Drive Strength Specifies the drive strength of the LDO_CORE VDD regulator. The LDO_CORE regulator is forced to disabled in DPDOWN Sleep mode. If you specify normal drive strength, you must write a value to LP[BGMODE] that enables the bandgap. If you have enabled LVDs or HVDs, and attempt to specify low drive strength, SPC ignores the attempt. 0b - Low 1b - Normal

32.7.12 Low Power Wake-Up Delay (LPWKUP_DELAY)

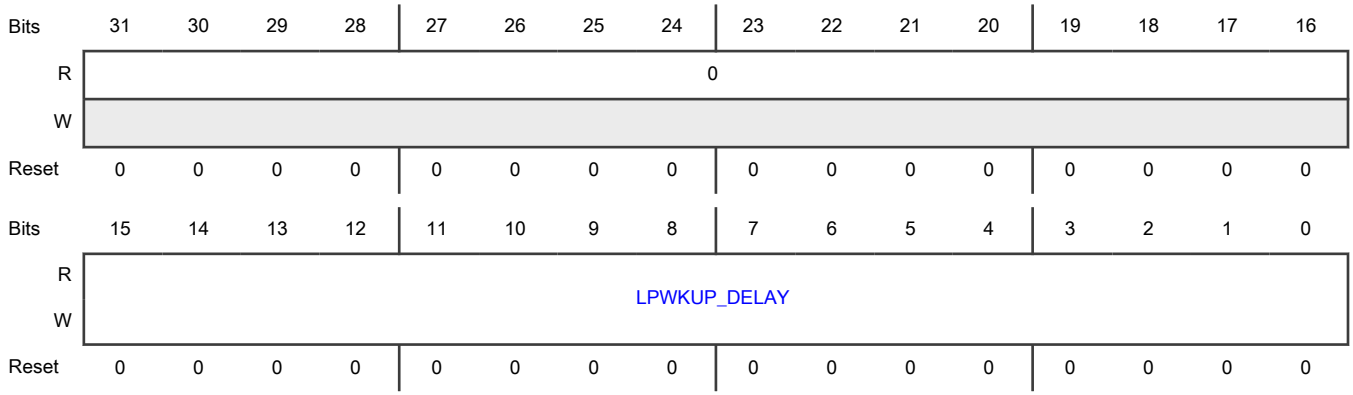
Offset

Register	Offset
LPWKUP_DELAY	120h

Function

Works with the LPREQ pin to extend the wake-up time if the external PMIC or switch needs additional time after wake-up.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 LPWKUP_DELAY	Low-Power Wake-Up Delay Specifies the number of SPC timer clock cycles that SPC waits after exiting low-power modes. This field resets only after a POR, LVD, or HVD event. When voltage levels are not the same between ACTIVE mode and Low Power mode, you must write a nonzero value to this field.

32.7.13 Active Voltage Trim Delay (ACTIVE_VDELAY)

Offset

Register	Offset
ACTIVE_VDELAY	124h

Function

Specifies the delay due to voltage level changes to the regulators in Active mode.

This register resets only after a POR event.

SPC loads the ACTIVE_VDELAY values are loaded from IFR or FUSE after any reset. NXP trims this reset value for the recommended wait time.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ACTIVE_VDELAY															
Reset	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 ACTIVE_VDELAY	Active Voltage Delay Specifies the number of SPC timer clock cycles that SPC waits when changing the core-regulator voltage level in Active mode.

32.7.14 Voltage Detect Status (VD_STAT)

Offset

Register	Offset
VD_STAT	130h

Function

Displays the LVD or HVD captured for:

- Core VDD.
- System VDD.
- IO VDD.

This register resets only after a POR event.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									IOVDD_H...	SYSVDD_...	COREVDD...	0	IOVDD_L...	SYSVDD_...	COREVDD...
W										W1C	W1C	W1C		W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 IOVDD_HVDF	IO VDD HVD Flag Indicates an IO VDD HVD event. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event not detected 1b - Event detected When writing 0b - No effect 1b - Clear the flag
5 SYSVDD_HVDF	System HVD Flag Indicates a system HVD event. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event not detected 1b - Event detected When writing 0b - No effect 1b - Clear the flag
4 COREVDD_HVDF	Core VDD HVD Flag Indicates a core VDD HVD event. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event not detected 1b - Event detected When writing 0b - No effect 1b - Clear the flag
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 IOVDD_LVDF	IO VDD LVD Flag Indicates an IO VDD LVD event. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event not detected 1b - Event detected When writing 0b - No effect 1b - Clear the flag
1 SYSVDD_LVDF	System Low-Voltage Detect Flag Indicates a system LVD event. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event not detected 1b - Event detected When writing 0b - No effect 1b - Clear the flag
0 COREVDD_LVDF	Core Low-Voltage Detect Flag Indicates a core LVD event. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Event not detected 1b - Event detected When writing 0b - No effect 1b - Clear the flag

32.7.15 Core Voltage Detect Configuration (VD_CORE_CFG)

Offset

Register	Offset
VD_CORE_CFG	134h

Function

Provides these functions:

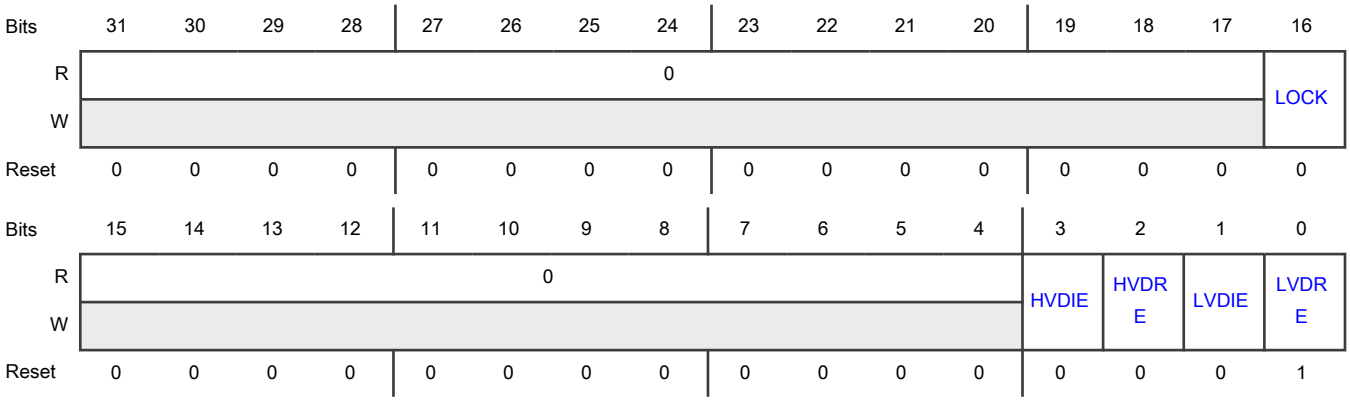
- Configures the low-voltage trip point.
- Enables the core LVD reset and interrupts.

This register resets only after a POR event.

NOTE

If you write 1 to both LVDRE and LVDIE, SPC generates an interrupt after exiting from LVDRE reset. If you don't want this to occur, configure the LVD or HVD so only one is enabled.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 LOCK	Core Voltage Detect Reset Enable Lock Allows writing to the LVDRE and HVDRE fields. 0b - Allow 1b - Deny
15-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 HVDIE	Core VDD HVD Interrupt Enable Enables the Core VDD HVD (COREVDD_HVDF) event to generate a hardware interrupt. If you write 1 to HVDIE with HVD set, SPC automatically generates an HVD interrupt. 0b - Disable 1b - Enable
2 HVDRE	Core VDD HVD Reset Enable Enables the core VDD HVD (COREVDD_HVDF) event to generate a hardware reset. Before writing to this field, you must write 0 to LOCK. 0b - Disable 1b - Enable
1 LVDIE	Core LVD Interrupt Enable Enables the Core LVD (COREVDD_LVDF) event to generate a hardware interrupt. If you write 1 to LVDIE with LVDF set, SPC automatically generates an LVD interrupt. 0b - Disable 1b - Enable
0 LVDRE	Core LVD Reset Enable Enables the core LVD (COREVDD_LVDF) event to generate a hardware reset. Before writing to this field, you must write 0 to LOCK. 0b - Disable 1b - Enable

32.7.16 System Voltage Detect Configuration (VD_SYS_CFG)

Offset

Register	Offset
VD_SYS_CFG	138h

Function

Provides these functions:

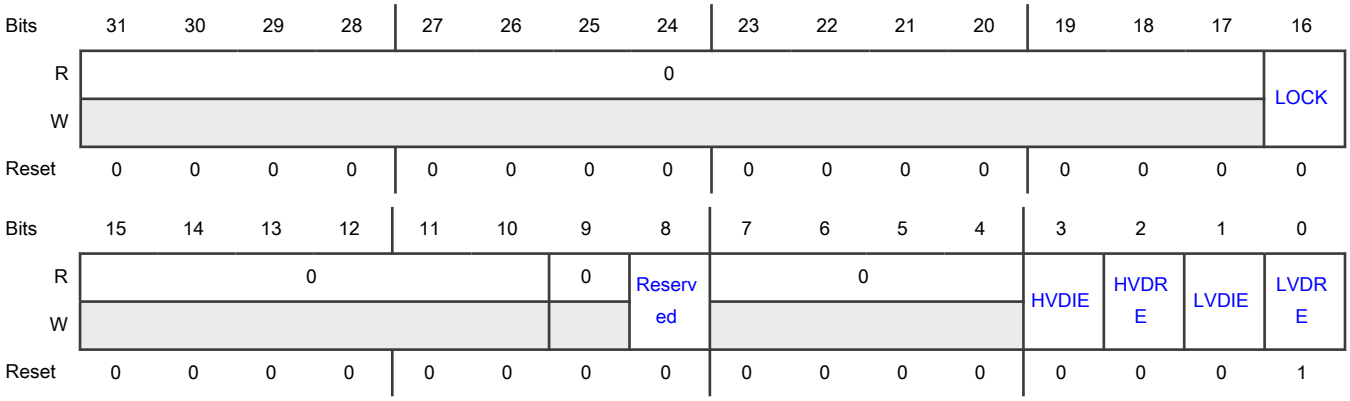
- Configures the low-voltage trip point.
- Enables the system LVD reset and interrupts.

This register resets only after a POR event.

NOTE

If you write 1 to both LVDRE and LVDIE, SPC generates an interrupt after exiting from LVDRE reset. If you don't want this to occur, configure the LVD or HVD so only one is enabled.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 LOCK	System Voltage Detect Reset Enable Lock Allows writing to the LVDRE, HVDRE, and LVSEL fields. 0b - Allow 1b - Deny
15-10 —	Reserved
9 —	Reserved
8 —	Reserved
7-4 —	Reserved
3 HVDIE	System HVD Interrupt Enable Enables the system HVD (SYSVDD_HVDF) event to generate a hardware interrupt. If you write 1 to HVDIE with HVDF set, SPC automatically generates an HVD interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
2 HVDRE	System HVD Reset Enable Enables the system HVD (SYSVDD_HVDF) event to generate a hardware reset. Before writing to this field, you must write 0 to LOCK. 0b - Disable 1b - Enable
1 LVDIE	System LVD Interrupt Enable Enables the system LVD (SYSVDD_LVDF) event to generate a hardware interrupt. If you write 1 to LVDIE with LVDF set, SPC automatically generates an LVD interrupt. 0b - Disable 1b - Enable
0 LVDRE	System LVD Reset Enable Enables the system LVD (SYSVDD_LVDF) event to generate a hardware reset. Before writing to this field, you must write 0 to LOCK. 0b - Disable 1b - Enable

32.7.17 IO Voltage Detect Configuration (VD_IO_CFG)

Offset

Register	Offset
VD_IO_CFG	13Ch

Function

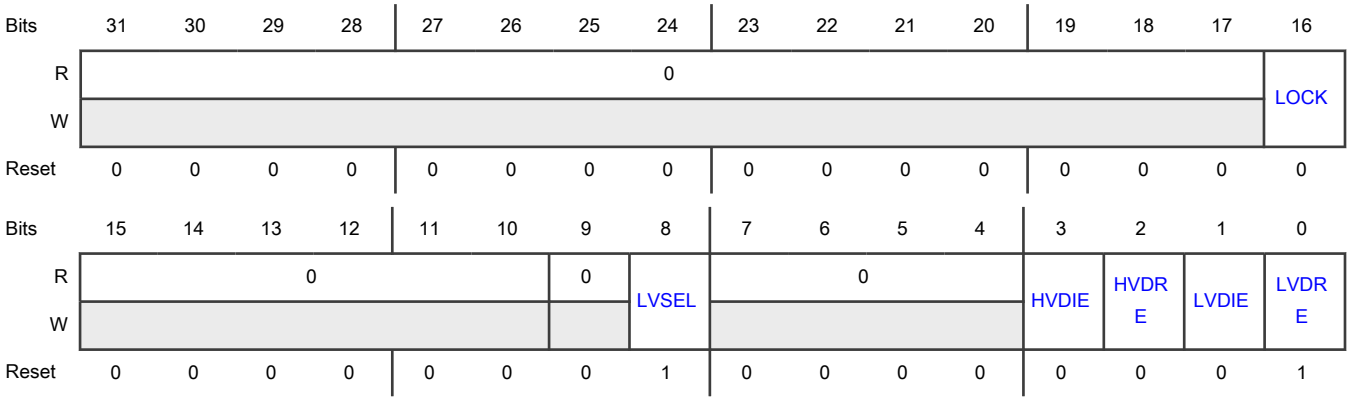
Enables the IO VDD LVD and HVD reset and interrupts.

This register resets only after a POR event.

NOTE

If you write 1 to both LVDRE and LVDIE, SPC generates an interrupt after exiting from LVDRE reset. If you don't want this to occur, configure the LVD or HVD so only one is enabled.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 LOCK	IO Voltage Detect Reset Enable Lock Allows writing to the LVDRE, HVDRE, and LVSEL fields. 0b - Allow 1b - Deny
15-10 —	Reserved
9 —	Reserved
8 LVSEL	IO VDD Low-Voltage Level Select Specifies the LVD trip point for the IO VDD monitor. See the chip data sheet for the high-range and low-range values. Before writing to this field, you must write 0 to LOCK. If you want to change LVSEL, you must do this after disabling the LVD reset and interrupt. Otherwise, SPC could generate an LVD due to the LVSEL change. 0b - High range 1b - Low range
7-4 —	Reserved
3 HVDIE	IO VDD HVD Interrupt Enable Enables the IO VDD HVD (IOVDD_HVDF) event to generate a hardware interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If you write 1 to HVDIE with HVDF set, SPC automatically generates an HVD interrupt. 0b - Disable 1b - Enable
2 HVDRE	IO VDD HVD Reset Enable Enables the IO VDD HVD (IOVDD_HVDF) event to generate a hardware reset. Before writing to this field, you must write 0 to LOCK. 0b - Disable 1b - Enable
1 LVDIE	IO VDD LVD Interrupt Enable Enables the IO VDD LVD (IOVDD_LVDF) event to generate a hardware interrupt. If you write 1 to LVDIE with LVDF set, SPC automatically generates an LVD interrupt. 0b - Disable 1b - Enable
0 LVDRE	IO VDD LVD Reset Enable Enables the IO VDD LVD (IOVDD_LVDF) event to generate a hardware reset. Before writing to this field, you must write 0 to LOCK. 0b - Disable 1b - Enable

32.7.18 External Voltage Domain Configuration (EVD_CFG)

Offset

Register	Offset
EVD_CFG	140h

Function

Isolates signals from external voltage domains under certain conditions.

Chip pins supply the external voltage domains. These domains are controlled at the board level. This register allows you to control internal isolations for signals from these domains if either of the following conditions is true:

- They are not powered on the board.
- They are powered off externally in low-power modes (using the SPC_LPREQ pin).

Any logic in the isolated voltage domain also resets, because the voltage domain is assumed to be powered off. This reset does not impact any VDD_CORE internal power domains or registers—only I/O pads, analog components, or both.

This register resets only after a POR event.

Each bit of the EVDISO, EVDLPISO, and EVDSTAT fields corresponds to one of the I/O or analog supplies according to Table 219.

Table 219. Mapping of bits to supplies

Bit position in field	Description
0	VDD_IO_ABC
1	VDD_IO_D
2	VDD_ANA

NOTE

Bit 0 VDD_IO_ABC should not be set in EVD_CFG[EVDISO] and it should only be set in EVD_CFG[EVDLPISO] when moving to Deep Power Down mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													EVDSTAT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0[1]	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				EVDLPISO				0				EVDISO			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. The reset value of this field can change based on external voltage conditions.

Fields

Field	Function
31-19 —	Reserved
18-16 EVDSTAT	External Voltage Domain Status Indicates the status of the external voltage domains. See Table 219 for the mapping of each bit of this field. Each bit's value has the following meaning: 0b - Isolated or not powered 1b - Not isolated
15-11 —	Reserved
10-8 EVDLPISO	External Voltage Domain Low-Power Isolation Isolates the external voltage domain in low-power modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Use this field if you use the SPC_LPREQ pin to power off any voltage domain in low-power modes. See Table 219 for the mapping of each bit of this field. Each bit's value has the following meaning: 0b - Not isolated 1b - Isolated
7-3 —	Reserved
2-0 EVDISO	External Voltage Domain Isolation Isolates the external voltage domain. Use this field if any voltage domain is always powered off on the board, or whenever a voltage domain is powered off under software control. See Table 219 for the mapping of each bit of this field. Each bit's value has the following meaning: 0b - Not isolated 1b - Isolated

32.7.19 Glitch Detect Status Control (GLITCH_DETECT_SC)

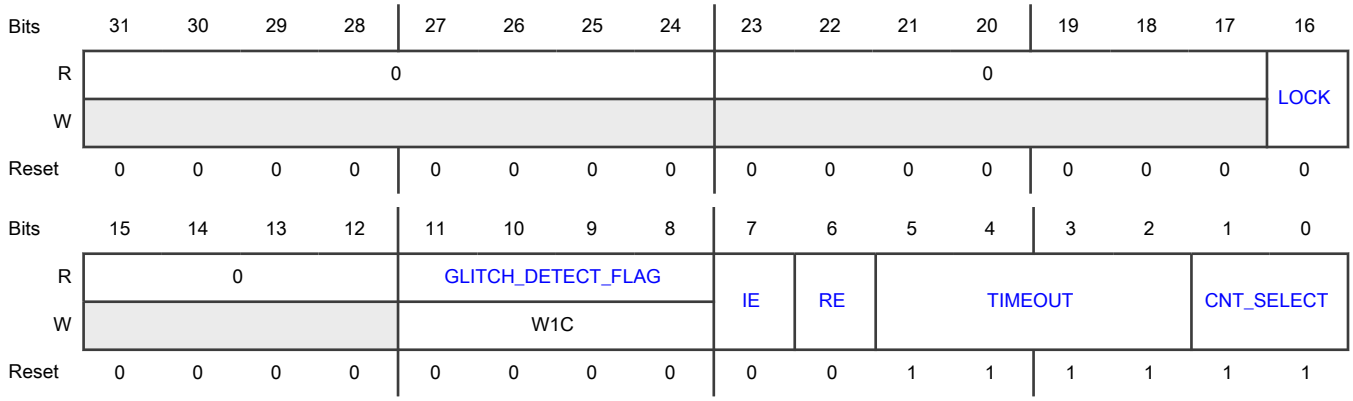
Offset

Register	Offset
GLITCH_DETECT_SC	144h

Function

Controls the GLITCH_DETECT operation.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-17 —	Reserved
16 LOCK	Glitch Detect Reset Enable Lock Bit Used to prevent writes to the GLITCH_RE register bit. 0b - Writes to RE are allowed. 1b - Writes to RE are ignored.
15-12 —	Reserved
11-8 GLITCH_DETECT_FLAG	GLITCH_DETECT_FLAG State of 4-bit Glitch Ripple Counter output. Based on CNT_SELECT program value, the correct 4-bit output state is monitored to generate interrupt or reset based on the settings of IE and RE.
7 IE	Glitch Detect Interrupt Enable Enables GLITCH_DETECT_FLAG[CNT_SELECT] to generate a hardware interrupt. **NOTE** Setting IE with GLITCH_DETECT_FLAG[CNT_SELECT] set will cause an automatic glitch detect interrupt to get generated. 0b - GLITCH_DETECT_FLAG[CNT_SELECT] does not generate hardware interrupt (user polling) 1b - GLITCH_DETECT_FLAG[CNT_SELECT] does generate hardware interrupt
6 RE	Glitch Detect Reset Enable Enables GLITCH_DETECT_FLAG[CNT_SELECT] to generate a POR/LVD reset. **NOTE** Writes to this bit require the VDD_CORE_GLITCH_DETECT_SC[LOCK] bit to be cleared. 0b - GLITCH_DETECT_FLAG[CNT_SELECT] does not generate POR/LVD reset 1b - GLITCH_DETECT_FLAG[CNT_SELECT] does generate POR/LVD reset
5-2 TIMEOUT	Timeout Specifies the timeout value used to reset glitch detect and compare logic after an initial glitch is detected. The timeout is based on the internal 10 MHz reference clock. After initial glitch is detected, the total timeout used to reset the glitch-detect logic is (TIMEOUT + initialization of internal timer clock).

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 CNT_SELECT	Counter Select Specifies the ripple-counter bit to use for detecting a glitch in the VDD core. 00b - 0 01b - 1 10b - 2 11b - 3

32.7.20 LDO_CORE Configuration (CORELDO_CFG)

Offset

Register	Offset
CORELDO_CFG	300h

Function

Configures LDO_CORE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				0				0			DPDOWN_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							0	0				0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-21 —	Reserved
20-17 —	Reserved
16 DPDOWN_PULLDOWN_DISABLE	LDO_CORE Deep Power Down Pulldown Disable Forces the LDO_CORE to discharge in Deep Power Down modes if CNTRL[CORELDO_EN] = 1. 0b - LDO_CORE pulldown in Deep Power Down not disabled 1b - LDO_CORE pulldown in Deep Power Down disabled
15-9 —	Reserved
8 —	Reserved
7-4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

32.7.21 LDO_SYS Configuration (SYSLDO_CFG)

Offset

Register	Offset
SYSLDO_CFG	400h

Function

Configures LDO_SYS.

This register resets only after a POR or LVD event.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							0	0			0	0			ISINKEN
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1[1]

1. This value gets loaded with IFR or FUSE values during reset.

Fields

Field	Function
31-20 —	Reserved
19 —	Reserved
18-16 —	Reserved
15-9 —	Reserved
8 —	Reserved
7-5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 ISINKEN	Current Sink Enable Enables the current-sink feature of the system's low-power regulator. Discharges the output voltage when LDO_SYS is disabled. 0b - Disable 1b - Enable

32.7.22 DCDC Configuration (DCDC_CFG)

Offset

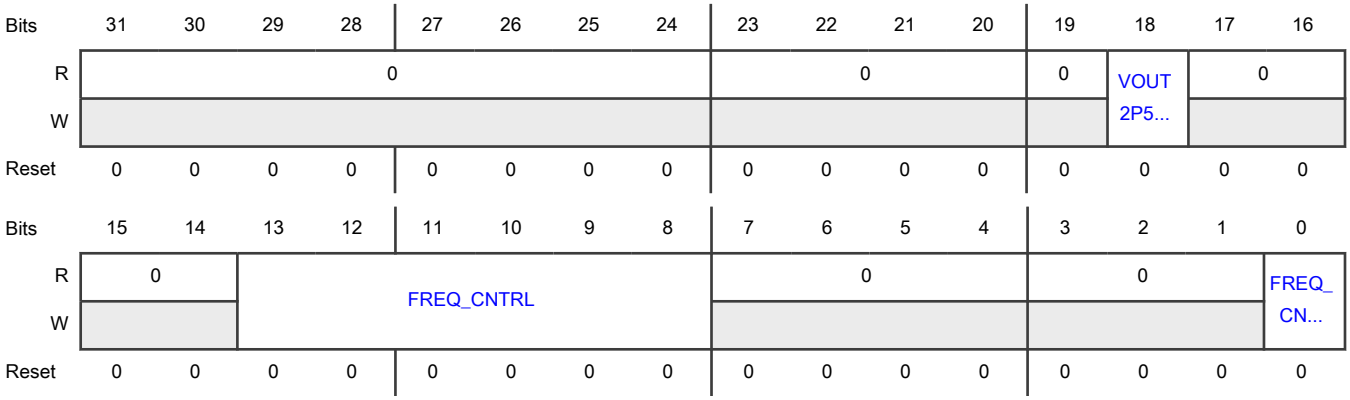
Register	Offset
DCDC_CFG	500h

Function

Configures DCDC.

This register resets only after a POR, LVD, or HVD event.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 —	Reserved
19 —	Reserved
18 VOUT2P5_SEL	2.5 V Output Select Specifies the value of DCDC Vout. This field is only meaningful in Active Mode when DCDC drive strength is set to Normal. When VOUT2P5_SEL = 1, the DCDC_VTRIM[LVL10] trim values adjust the DCDC 2.5 V output. 0b - From DCDC_VDD_LVL register 1b - 2.5 V

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-16 —	Reserved
15-14 —	Reserved
13-8 FREQ_CNTRL	DCDC Burst Frequency Control Specifies the frequency of the current burst.
7-4 —	Reserved
3-1 —	Reserved
0 FREQ_CNTRL_ON	DCDC Burst Frequency Control Enable Enables DCDC frequency stabilization. 0b - Disable 1b - Enable

32.7.23 DCDC Burst Configuration (DCDC_BURST_CFG)

Offset

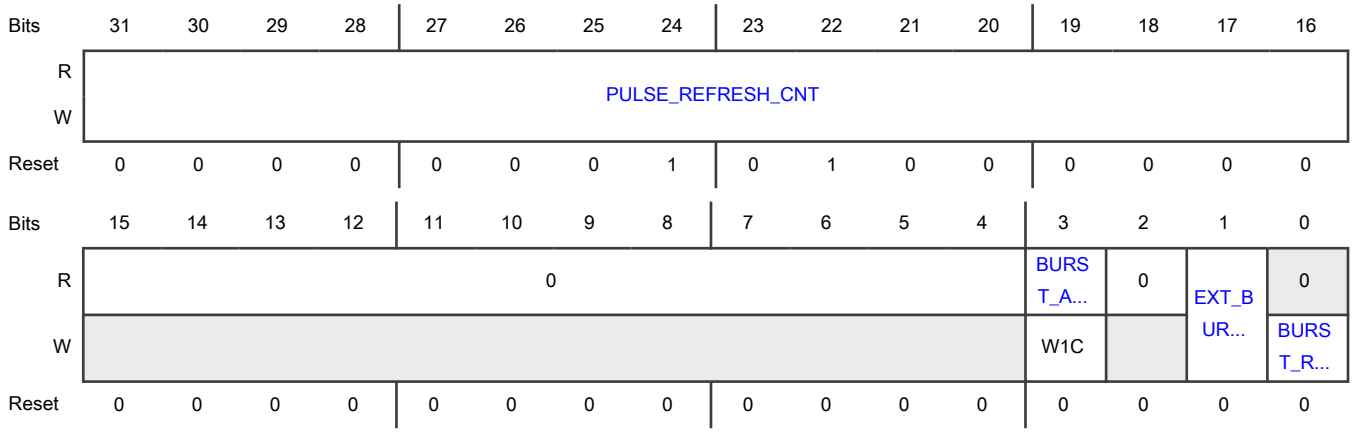
Register	Offset
DCDC_BURST_CFG	504h

Function

Controls the DCDC burst operation.

This register resets only after a POR, LVD, or HVD event.

Diagram



Fields

Field	Function
31-16 PULSE_REFRESH_CNT	Refresh Count Value Controls the DCDC refresh frequency when DCDC is in Pulse Refresh mode. When DCDC is enabled and running in Pulse Refresh mode, PULSE_REFRESH_CNT controls how often the DCDC is pulsed on and off based on a low-power reference clock. You must specify a count value that is long enough to allow a DCDC burst to occur. The pulse duration (time between on and off) is: (reference clock period) × (PULSE_REFRESH_CNT + 2) **NOTE** See the chip-specific information for more on DCDC pulsed mode clocking.
15-4 —	Reserved
3 BURST_ACK	Burst Acknowledge Flag Indicates that the DCDC burst completed, and therefore DCDC entered Quiet mode. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not complete 1b - Completed When writing 0b - No effect 1b - Clear the flag
2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 EXT_BURST_EN	External Burst Request Enable Enables external burst requests. When this field is 1, SPC asserts the output burst trigger DCDC_BURST_TRIG_PULSE after the DCDC burst request has been acknowledged. **NOTE** Enable external bursts before writing 1 to BURST_REQ and after BURST_ACK is 0. 0b - Disable 1b - Enable
0 BURST_REQ	Software Burst Request Generates a software burst request to DCDC. **NOTE** Do not generate a new burst request until the previous burst request has completed and you acknowledged this by clearing the BURST_ACK flag. 0b - Do not generate 1b - Generate

Chapter 33
Smart Power Switch (VBAT)

33.1 Chip-specific VBAT information

Table 220. Reference links to related information

Topic	Related module	Reference
Full description	VBAT	VBAT
System memory map		System memory map
Power management		Power management map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

33.1.1 Signal naming

The table below lists the VBAT signal in this device.

Table 221. Device signal name - VBAT signal name

Device signal name	VBAT signal name
SWITCH_WAKEUP_B	WAKEUP_b

33.1.2 VBAT supply

VBAT is supplied from the chip power supply VDD_SWITCH.

33.1.3 VBAT backup SRAM LDO for memory retention

VBAT module in the device can optionally retain the contents of the 8KB SRAM STCM8 array that supports ECC.

33.2 Overview

VBAT works with the power management system to implement power-saving mechanisms. It provides bandgap timers and a voltage regulator to supply retention SRAM in low-power modes.

VBAT implements the 16 kHz internal clock source via FRO 16kHz.

33.2.1 Block diagram

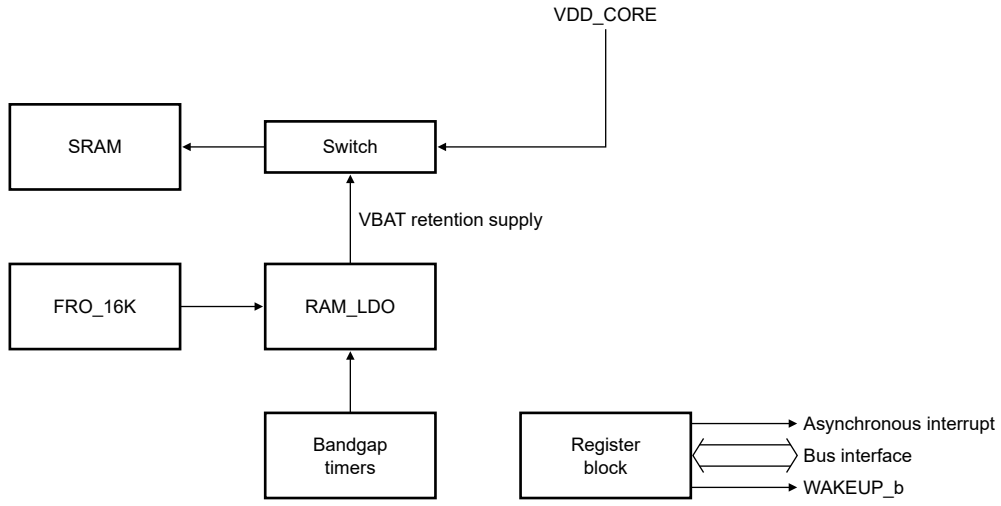


Figure 90. Block diagram

33.2.2 Features

- Supports internal 16 kHz free-running oscillator (FRO16K)
- Supports backup retention SRAM regulator

33.3 Functional description

The following sections describe the functional details of VBAT.

33.3.1 Operations

This section describes the operations of VBAT.

33.3.1.1 FRO 16.384 kHz clock

FRO16K is enabled by default on POR. You can disable it or lock it to prevent any change to the enable field. FRO 16k should be enabled prior to enabling the backup SRAM LDO regulator or VBAT bandgap timers.

33.3.1.2 SRAM LDO

SRAM LDO is a retention regulator that retains VBAT retention SRAM in low-power modes, including when VBAT is the only available supply. When IRQENA[LDO_RDY] becomes 1, the chip can enter into a low-power mode where the VBAT supply retains SRAM. Enable the bandgap and LDO before entering a low-power mode where the VBAT supply powers the retention SRAM. Enable Refresh mode for the lowest power consumption.

It takes approximately 8ms for the LDO_RDY flag to set after software enables the bandgap and LDO. When using the SRAM LDO it is recommended to leave it enabled in active modes to avoid this delay before entering a low power mode.

Although SRAM is automatically retained in all low-power modes, you can manually switch SRAM to the VBAT retention supply anytime. Do not access the SRAM when it receives power from the VBAT retention supply. You must manually reverse these steps before accessing the SRAM again.

To manually switch VBAT to power the SRAM:

1. Isolate the SRAM array (write 1 to LDORAMC[ISO]).
2. Switch the supply from VDD_CORE to the VBAT retention LDO (write 1 to LDORAMC[SWI]).

When SRAM LDO is disabled, LDORAMC[RETn] controls whether the retention of VBAT SRAM happens using VDD_CORE in low-power modes.

In general, using the SRAM LDO to retain the VBAT retention SRAM is a lower power option than using VDD_CORE directly.

NOTE

Enable FRO16K before enabling SRAM LDO or the bandgap.

33.3.1.3 Bandgap timer

Two software-configurable bandgap timers are available when the SRAM LDO bandgap is enabled. These timers share logic with the bandgap timer refresh logic to minimize power, so they use the FRO16K clock source and are only available when the bandgap is enabled. When you change the configured timeout, NXP recommends that you:

1. Disable the timer.
2. Clear the flag.
3. Write the new timeout value.
4. Enable the timeout again.

33.3.2 Low-power modes

VBAT remains functional in all low-power modes.

33.3.3 Debug mode

Debug modes do not affect VBAT.

33.3.4 Clocks

VBAT implements the following clock domains:

- Bus interface clock - access the registers
- FRO16 kHz internal clock

33.3.5 Reset

Only the POR resets VBAT.

33.3.6 Interrupts

VBAT generates one interrupt combining the following sources:

- VBAT POR
- Wake-up pin assertion
- Bandgap timer 0
- Bandgap timer 1

33.4 External signals

Table 222. External signals

Signal	Description	Direction
WAKEUP_b	Active low wake-up pin, falling edge sets the WAKEUP_FLAG. When WAKEUP_b pin is configured for open drain operation, an external wakeup source can drive this pin low, and the falling edge will assert the WAKEUP_FLAG	Input/Output

33.5 Initialization

To enable and lock the FRO16K:

1. Write 1h to FROCTLA[FRO_EN].
2. Write 1h to FROLCKA[LOCK].
3. Alter FROCLKE[CLKE] to clock gate different FRO16K outputs to different peripherals to reduce power consumption.

To enable and lock the LDO and bandgap:

1. Enable the FRO16K.
2. Write 7h to LDO_RAM Control A (LDOCTLA).
3. Wait for STATUSA[LDO_RDY] to become 1.
4. Write 1h to LDOLCKA[LOCK].

33.6 Application information

Configure the VBAT LDO to retain the state of any required SRAM to guard against an unexpected loss of power to the system:

1. Enable the LDO and bandgap.
2. Configure LDORAMC[RETn] to retain the appropriate arrays.

For a software-controlled power down, follow these steps to ensure robust retention of memory before switching off external power:

1. Write 1b to LDORAMC[ISO].
2. Write 1b to LDORAMC[SWI].

You must reverse the above steps to access the SRAM arrays, which the VBAT LDO retains. That is, follow these steps after external power is again switched on:

1. Write 0b to LDORAMC[SWI].
2. Write 0b to LDORAMC[ISO].

33.7 Memory map and register definition

This section includes VBAT memory map and detailed descriptions of all registers.

NOTE

The VBAT registers are reset on VBAT Cold Reset only (VBAT POR).

33.7.1 VBAT register descriptions

33.7.1.1 VBAT memory map

VBAT0 base address: 4002_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0003h
10h	Status A (STATUSA)	32	RW	0000_0001h
18h	Interrupt Enable A (IRQENA)	32	RW	0000_0001h
20h	Wake-up Enable A (WAKENA)	32	RW	0000_0001h
200h	FRO16K Control A (FROCTLA)	32	RW	0000_0001h
218h	FRO16K Lock A (FROLCKA)	32	RW	0000_0000h
220h	FRO16K Clock Enable (FROCLKE)	32	RW	0000_0000h
300h	LDO_RAM Control A (LDOCTLA)	32	RW	0000_0000h
318h	LDO_RAM Lock A (LDOLCKA)	32	RW	0000_0000h
320h	RAM Control (LDORAMC)	32	RW	0000_0000h
330h	Bandgap Timer 0 (LDOTIMER0)	32	RW	0000_0000h
338h	Bandgap Timer 1 (LDOTIMER1)	32	RW	0000_0000h

33.7.1.2 Version ID (VERID)

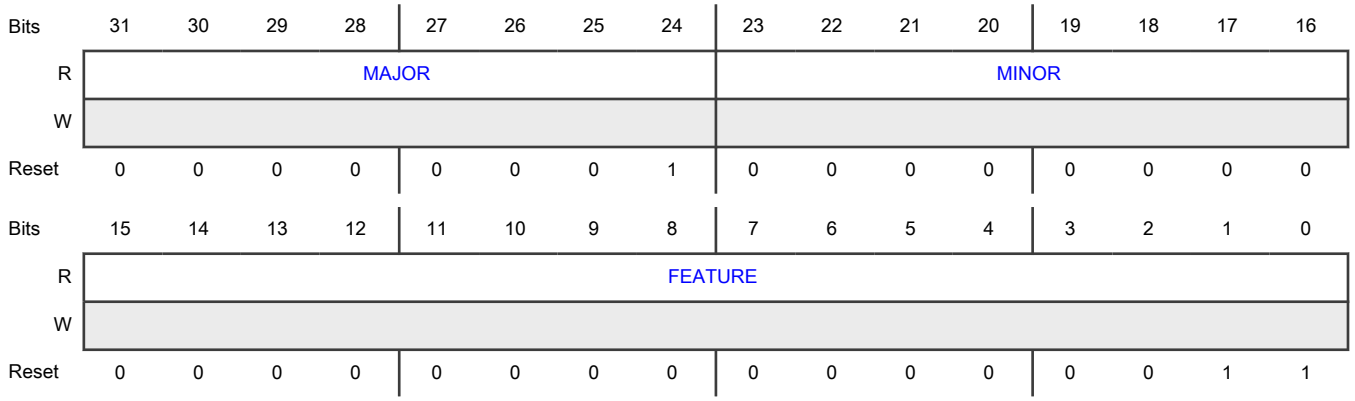
Offset

Register	Offset
VERID	0h

Function

Records the specific version of VBAT in the chip.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number, indicating the feature set present in this instance of VBAT.

33.7.1.3 Status A (STATUSA)

Offset

Register	Offset
STATUSA	10h

Function

Contains all status flags for VBAT.

See the chip's security reference manual for security-related field information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	LDO_RDY	TIMER1_...	TIMER0_...	WAKEUP_...	POR_DET
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 LDO_RDY	LDO Ready Indicates whether LDO is enabled and ready to enter Low-Power mode. 0b - Disabled (not ready) 1b - Enabled (ready)
3 TIMER1_FLAG	Bandgap Timer 1 Flag Asserts periodically according to the bandgap timer 1 period. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not reached 1b - Reached When writing 0b - No effect 1b - Clear the flag
2 TIMER0_FLAG	Bandgap Timer 0 Flag Asserts periodically according to the bandgap timer 0 period. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not reached 1b - Reached When writing 0b - No effect 1b - Clear the flag
1 WAKEUP_FLAG	Wakeup Pin Flag Asserts whenever VBAT detects a falling edge on the wake-up pin. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not asserted

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Asserted When writing 0b - No effect 1b - Clear the flag
0 POR_DET	POR Detect Flag Indicates if the VBAT POR has asserted. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not reset 1b - Reset When writing 0b - No effect 1b - Clear the flag

33.7.1.4 Interrupt Enable A (IRQENA)

Offset

Register	Offset
IRQENA	18h

Function
Contains all interrupt enables for VBAT.

NOTE
See the chip's security reference manual for security-related field information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	LDO_RDY	TIMER1_...	TIMER0_...	WAKEUP_...	POR_DET
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 LDO_RDY	LDO Ready Enables the corresponding interrupt in Status A (STATUSA). 0b - Disable 1b - Enable
3 TIMER1_FLAG	Bandgap Timer 2 Enables the corresponding interrupt in Status A (STATUSA). 0b - Disable 1b - Enable
2 TIMER0_FLAG	Bandgap Timer 0 Enables the corresponding interrupt in Status A (STATUSA). 0b - Disable 1b - Enable
1 WAKEUP_FLAG	Wakeup Pin Flag Enables the corresponding interrupt in Status A (STATUSA). 0b - Disable 1b - Enable
0 POR_DET	POR Detect Enables the corresponding interrupt in Status A (STATUSA).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

33.7.1.5 Wake-up Enable A (WAKENA)

Offset

Register	Offset
WAKENA	20h

Function

Contains all wake-up enables for VBAT.

NOTE

See the chip's security reference manual for security-related field information

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	LDO_RDY	TIMER1_...	TIMER0_...	WAKEUP_...	POR_DET
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-20 —	Reserved
19 —	Reserved
18 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 LDO_RDY	LDO Ready Enables the corresponding wake-up in Status A (STATUSA). 0b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
3 TIMER1_FLAG	Bandgap Timer 2 Enables the corresponding wake-up in Status A (STATUSA). 0b - Disable 1b - Enable
2 TIMER0_FLAG	Bandgap Timer 0 Enables the corresponding wake-up in Status A (STATUSA). 0b - Disable 1b - Enable
1 WAKEUP_FLAG	Wake-up Pin Flag Enables the corresponding wake-up in Status A (STATUSA). 0b - Disable 1b - Enable
0 POR_DET	POR Detect Enables the corresponding wake-up in Status A (STATUSA). 0b - Disable 1b - Enable

33.7.1.6 FRO16K Control A (FROCTLA)

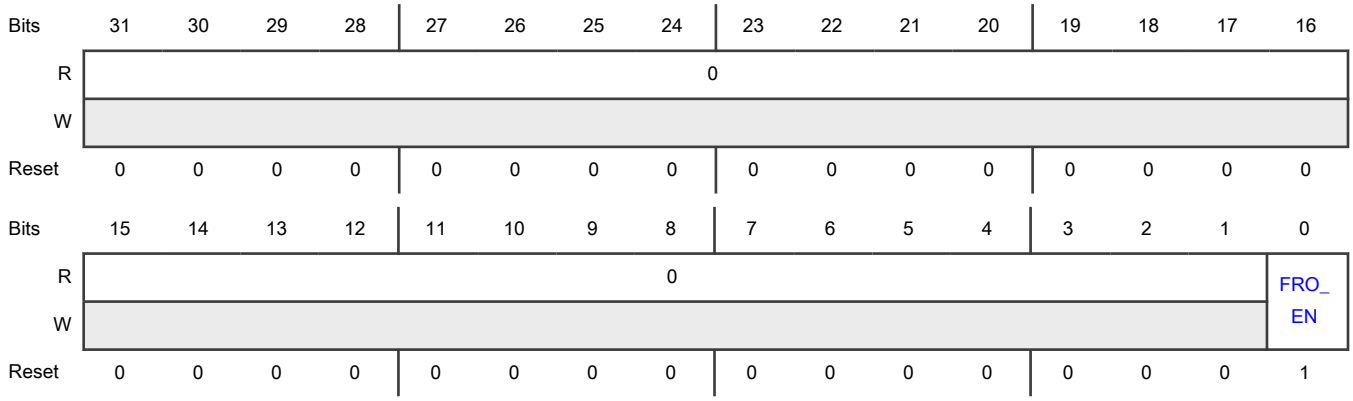
Offset

Register	Offset
FROCTLA	200h

Function

Controls FRO16K. Writes to this register are blocked when FROLCKA[LOCK] is 1.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FRO_EN	FRO16K Enable 0b - Disable 1b - Enable

33.7.1.7 FRO16K Lock A (FROLCKA)

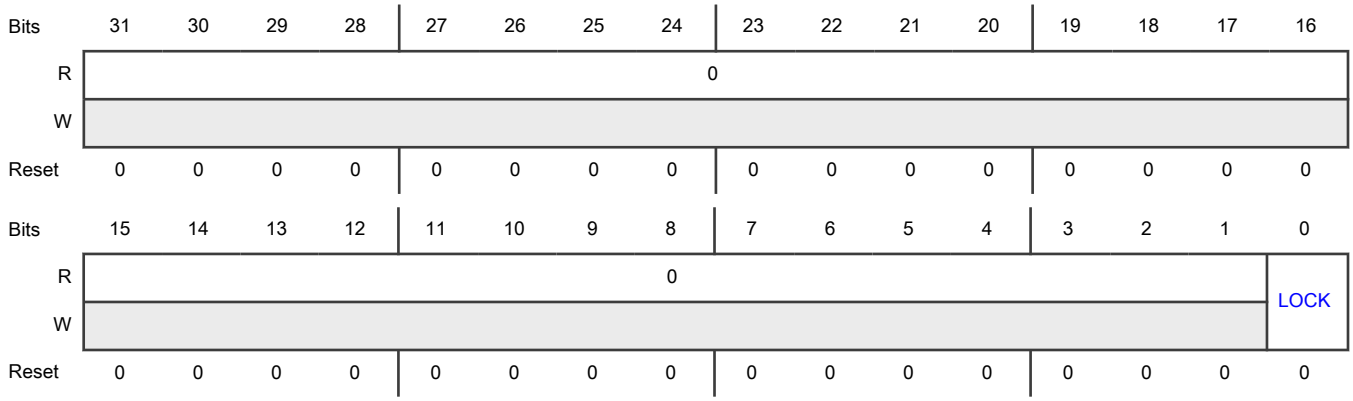
Offset

Register	Offset
FROLCKA	218h

Function

Contains the lock field. Writes to this register are blocked when FROLCKA[LOCK] is 1.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 LOCK	Lock Blocks any write to the FRO16K registers when you write 1 to this field. VBAT POR writes 0 to this field. 0b - Do not block 1b - Block

33.7.1.8 FRO16K Clock Enable (FROCLKE)

Offset

Register	Offset
FROCLKE	220h

Function

Contains clock gating register field to gate the FRO16K clock to other modules.

NOTE

FROCLKE cannot be locked (not affected by FRO16K Lock A (FROLCKA)).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															CLKE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 CLKE	Clock Enable Enables the corresponding FRO16 kHz output clock to other modules when you write 1 to the corresponding bit in this field. See the chip-specific VBAT information section for more information.

33.7.1.9 LDO_RAM Control A (LDOCTLA)

Offset

Register	Offset
LDOCTLA	300h

Function

Contains the LDO_RAM control field. When configuring the LDO_RAM, both control A and control B registers need to be programmed to the appropriate values. Writes to this register are blocked when LDOLCKA[LOCK] is 1.

NOTE

The FRO16K must be enabled before enabling the SRAM LDO or the bandgap

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													REFRESH...	LDO_EN	BG_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-3 —	Reserved
2 REFRESH_EN	Refresh Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the Bandgap Low-Power Refresh mode. The refresh mode must also be enabled for lowest power consumption. 0b - Refresh mode is disabled 1b - Refresh mode is enabled for low power operation
1 LDO_EN	LDO Enable Enables the backup SRAM regulator. LDOCTLA[BG_EN] must enable the bandgap when the backup SRAM regulator is enabled. After you enable the bandgap and backup SRAM regulator, you must wait until STATUSA[LDO_RDY] becomes 1 before the SRAM can enter a low-power state where the regulator supplies the array contents. 0b - Disable 1b - Enable
0 BG_EN	Bandgap Enable Enables the LDO_RAM bandgap. 0b - Disable 1b - Enable

33.7.1.10 LDO_RAM Lock A (LDOLCKA)

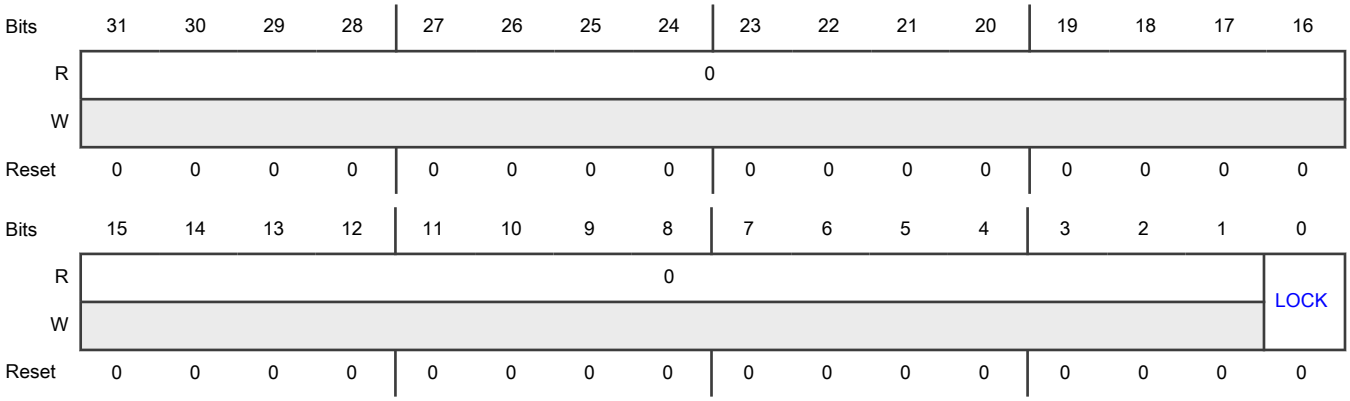
Offset

Register	Offset
LDOLCKA	318h

Function

Contains the lock field. Writes to this register are blocked when the LDO_RAM lock register is 1.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 LOCK	Lock Blocks any write to the LDO_RAM registers. VBAT POR writes 0 to this field. 0b - Do not block 1b - Block

33.7.1.11 RAM Control (LDORAMC)

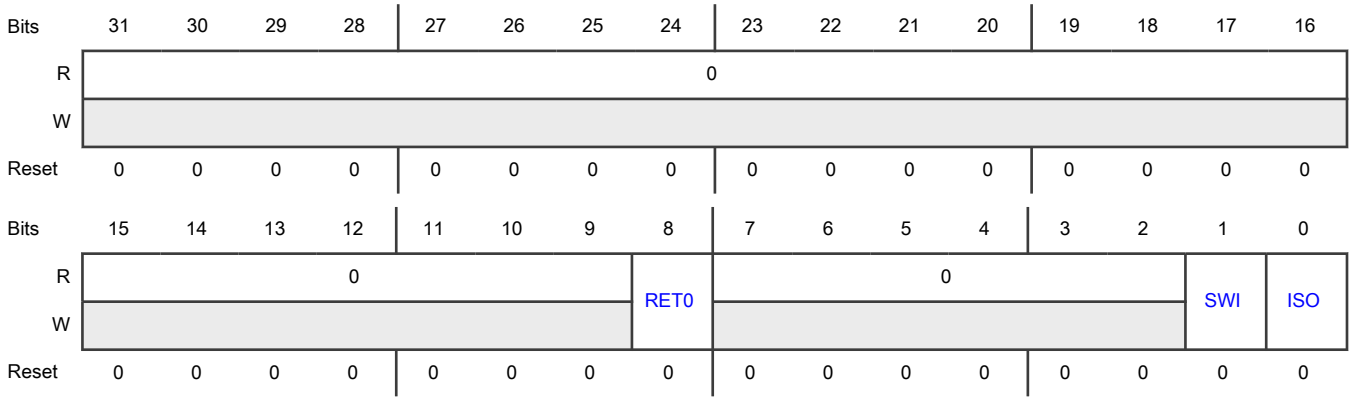
Offset

Register	Offset
LDORAMC	320h

Function

Contains software overrides for the SRAM isolation and backup switch. Always isolate the SRAM array before asserting or negating the switch control.

Diagram



Fields

Field	Function
31-9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-8 RETn	Retention Configures retention of each SRAM array in low power modes. See the VBAT chip-specific section for more details on these SRAM arrays. When LDO_RAM is enabled, at least one SRAM array must be retained. When LDO_RAM is disabled, this field specifies whether VDD_CORE continues to provide power to the SRAM array. 0b - Corresponding SRAM array is retained in low-power modes 1b - Corresponding SRAM array is not retained in low-power modes
7-2 —	Reserved
1 SWI	Switch SRAM Specifies how the SRAM array is powered. Change this field when LDORAMC[ISO] = 1. 0b - Supply follows the chip power modes 1b - LDO_RAM powers the array
0 ISO	Isolate SRAM Specifies how the SRAM array is isolated. 0b - State follows the chip power modes 1b - Isolates SRAM and places it in Low-Power Retention mode

33.7.1.12 Bandgap Timer 0 (LDOTIMER0)

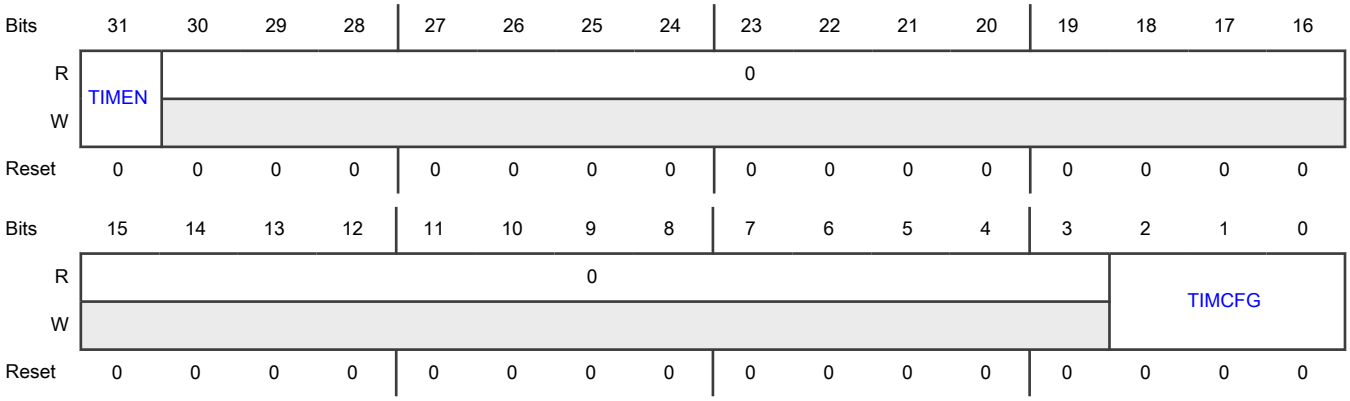
Offset

Register	Offset
LDOTIMER0	330h

Function

Controls one software-configurable timer when the bandgap is enabled and are clocked by FRO16K.

Diagram



Fields

Field	Function
31 TIMEN	Bandgap Timeout Period Enable 0b - Disable 1b - Enable
30-3 —	Reserved
2-0 TIMCFG	Timeout Configuration Configures the bandgap timeout 0 period. Changed when the timer is disabled. 000b - 1 s 001b - 500 ms 010b - 250 ms 011b - 125 ms 100b - 62.5 ms 101b - 31.25 ms 110b - 15.625 ms 111b - 7.8125 ms

33.7.1.13 Bandgap Timer 1 (LDOTIMER1)

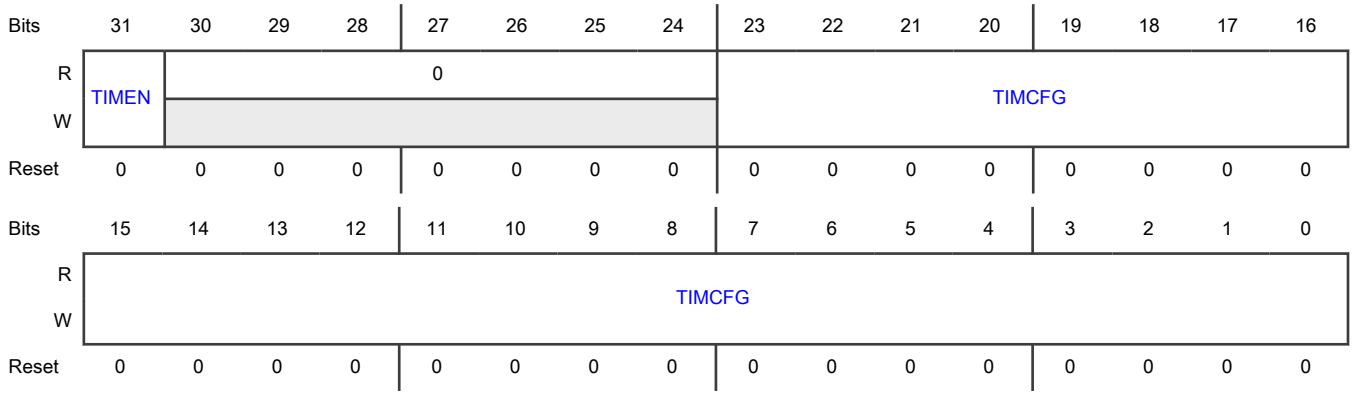
Offset

Register	Offset
LDOTIMER1	338h

Function

Controls the other software-configurable timer when the bandgap is enabled and are clocked by the FRO16K.

Diagram



Fields

Field	Function
31 TIMEN	Bandgap Timeout Period Enable 0b - Disable 1b - Enable
30-24 —	Reserved
23-0 TIMCFG	Timeout Configuration Configures the bandgap timeout 1 period. Configures timeout in number of seconds, ranging from 1 to 16,777,216 s. You can change this field when the timer is disabled.

Chapter 34
External Watchdog Monitor (EWM)

34.1 Chip-specific EWM information

Table 223. Reference links to related information

Topic	Related module	Reference
Full description	EWM	EWM
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

34.1.1 Module instances

This device has one instance of the EWM module, EWM0.

34.2 Overview

For safety purposes, a redundant watchdog system, EWM, is designed to monitor external circuits and the MCU software flow. This provides a backup mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The internal watchdog is used to monitor the flow and execution of the embedded software within the MCU. It consists of a counter that, if allowed to overflow, forces an internal, asynchronous reset to all on-chip peripherals. The counter also optionally asserts the RESET_B pin to reset external devices and circuits. The watchdog counter must not overflow if the software code works well and services the watchdog to restart the actual counter.

The EWM does not reset the MCU's CPU and peripherals, making it different from internal watchdog. The EWM module provides an independent ewm_out_b signal that, when asserted, resets or places an external circuit into a safe mode. The ewm_out_b signal asserts upon EWM counter timeout. An optional external input, ewm_in, allows additional control when asserting the ewm_out_b signal.

34.2.1 Block diagram

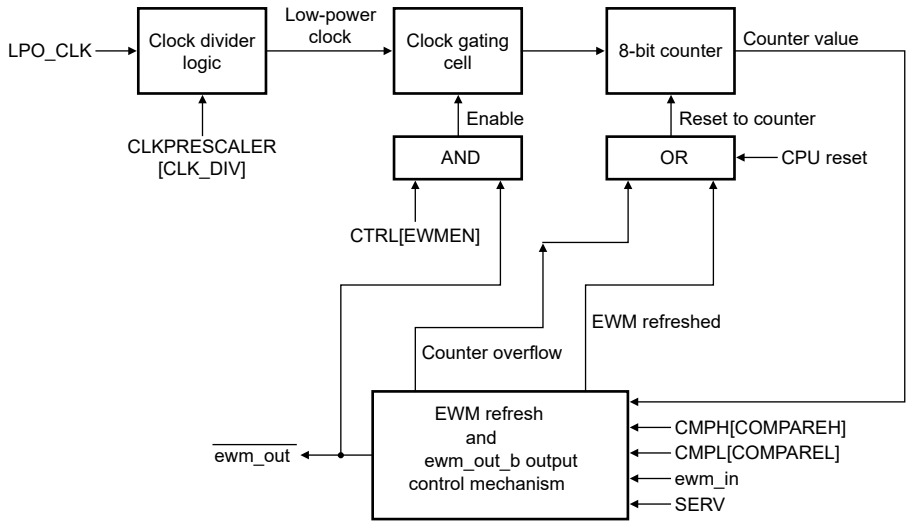


Figure 91. Block diagram

34.2.2 Features

- Independent LPO_CLK source
- Programmable timeout period, specified in terms of the number of EWM LPO_CLK cycles
- Windowed refresh option that provides:
  - A robust check to confirm that the program flow is faster than expected.
  - A programmable window.
  - Refresh outside the window, leading to assertion of the ewm_out_b signal.
- Robust refresh mechanism:
  - Write values of B4h and 2Ch to Service (SERV) within 15 peripheral bus clock cycles.
- One output port, ewm_out_b, which when asserted is used to reset or place the external circuit into Safe mode
- One input port, ewm_in, which allows an external circuit to control the assertion of the ewm_out_b signal

34.3 Functional description

The following sections discuss these aspects of EWM:

- Functional details
- Operating modes

NOTE

If the BUS_CLK is lost, EWM does not generate the ewm_out_b signal and no refresh operation is possible.

34.3.1 Modes of operation

34.3.1.1 Stop mode

When EWM is in Stop mode, the CPU cannot refresh EWM. After entering Stop mode, the EWM counter freezes.

Following are the possible ways to exit Stop mode:

- Through a reset: EWM remains disabled in this case.
- Through an interrupt: EWM is re-enabled and the counter continues to be clocked from the same value as prior to Stop mode entry.

NOTE

Consider the following if EWM enters Stop mode during the CPU refresh mechanism:

- While exiting Stop mode through an interrupt, the refresh mechanism starts from the previous state. That is, if you write the refresh command correctly and EWM enters Stop mode immediately, you must write the next command in *15* peripheral bus clocks after exiting Stop mode.
- You must mask all interrupts before executing the EWM refresh instructions.

34.3.1.2 Debug mode

EWM remains unimpacted when entering Debug mode:

- If EWM is enabled before entering Debug mode, it remains enabled.
- If EWM is disabled before entering Debug mode, it remains disabled.

34.3.2 Using the EWM counter

EWM uses an 8-bit ripple counter that is fed by a clock source independent of the peripheral bus clock source. As the preferred timeout is between 1 ms and 100 ms, the actual clock source must be in the kHz range.

The counter is reset to 0 in these conditions:

- After CPU reset
- After the EWM refresh action completes
- At counter overflow

The CPU cannot access the counter value.

34.3.3 Using compare registers

You can write to Compare Low (CMPL) and Compare High (CMPH) only once after a CPU reset and you cannot modify them until another CPU reset occurs. These registers are used to create a refresh window for the EWM module.

You cannot program Compare Low (CMPL) and Compare High (CMPH) with the same value. In case of any attempt, the ewm_out_b signal asserts as soon as the counter reaches the value of Compare Low (CMPL) + 1.

NOTE

You must update Compare Low (CMPL) and Compare High (CMPH) before enabling EWM, after which the counter resets to 0. Therefore, you must provide a reasonable time after POR for the external monitoring circuit to stabilize. You must also ensure that the ewm_in pin is deasserted.

34.3.4 Using the refresh mechanism

Other than the initial configuration of EWM, the CPU can access EWM only through Service (SERV). The CPU must access this register by correctly writing unique data within the windowed time frame, as determined by Compare Low (CMPL) and Compare High (CMPH) for the correct EWM refresh operation. The following table describes conditions that exist and the refresh mechanisms that apply to those conditions.

Table 224. Refresh mechanisms

Condition	Mechanism
The EWM refresh action completes when the value of Compare Low (CMPL) ≤ the counter value < the value of Compare High (CMPH).	The software behaves as expected and the EWM counter resets to 0. The ewm_out_b output signal remains in Deasserted state if, during the EWM refresh action, the ewm_in input is in Deasserted state.
The EWM refresh action completes when the counter value < the value of Compare Low (CMPL).	The software refreshes EWM before the windowed time frame, the counter resets to 0, and the ewm_out_b output signal asserts no matter what the value of the ewm_in input signal is.
The counter value reaches the value of Compare High (CMPH) prior to completion of the EWM refresh action.	The software does not refresh EWM. The EWM counter resets to 0 and the ewm_out_b output signal asserts no matter what the value of the ewm_in input signal is.

See Service (SERV) for more on the refresh mechanism.

34.3.5 Interrupt

When the ewm_out_b signal asserts, an interrupt request can be generated to indicate the assertion of the EWM reset out signal. The interrupt is enabled when CTRL[INTEN] = 1. Writing 0 to this field clears the interrupt request but does not affect the ewm_out_b signal, which can be deasserted only by forcing a system reset.

34.3.6 Clocking

The following table shows EWM clocks.

Table 225. EWM Clocks

Clock	Description
IPG_CLK	This is the system clock and should be turned on for EWM to be able to work properly. During low power modes in which the core is powered down, this clock is disabled,
IPG_CLK_S	This is the IPS clock and is synchronous with IPG_CLK. It is disabled except during IPS write accesses. it is enabled with EWM's IPS_MODULE_EN
IPO_CLOCK	This is a low power clock used for running EWM counter. This clock is gated when EWM is disabled or when ewm_out_b is asserted.

34.3.7 Using the counter clock prescaler

You can program CLKPRESCALER[CLK_DIV] to divide the EWM counter clock source. This divided clock is used to run the EWM counter.

NOTE

The divided clock used to run the EWM counter must not exceed half the frequency of the bus clock.

34.4 External signals

EWM includes external signals, as shown in the following table.

NOTE

All active-low signals are represented with the suffix "_b" throughout the chapter.

Table 226. Signal descriptions

Signal	Description	I/O
ewm_in	EWM's input for the safety status of external safety circuits. You can program the polarity of ewm_in by using CTRL[ASSIN]. The default polarity is active-low.	I
ewm_out_b	EWM's reset out signal	O

34.4.1 Using the ewm_out_b signal

The ewm_out_b signal is a digital output signal used to gate an external circuit (application-specific) that controls critical safety functions.

The ewm_out_b signal remains deasserted when the CPU regularly refreshes EWM within the programmable refresh window, indicating that the application code is executing as expected.

The ewm_out_b signal asserts in any of the following conditions:

- An EWM refresh action occurs when the counter value is less than the value of Compare Low (CMPL).
- The EWM counter value reaches the value of Compare High (CMPH) and no EWM refresh occurs.
- The functionality of the ewm_in pin is enabled and the ewm_in pin asserts when refreshing EWM.
- After any reset.

The ewm_out_b signal asserts after any reset by the virtue of the external pulldown mechanism on the ewm_out_b pin. To deassert the ewm_out_b signal, write 1 to CTRL[EWMEN] to enable EWM.

If the ewm_out_b signal shares its pad with a digital I/O pin, this actual pad defers to being an input signal on reset. The ewm_out_b signal controls the pad state only after CTRL[EWMEN] enables EWM.

NOTE

The ewm_out_b pad must be in Pulldown state when the EWM functionality is being used and EWM is under reset.

34.4.2 Using the ewm_out_b pin state in low-power modes

During Wait, Stop, and Power-Down modes, the ewm_out_b pin enters a high-impedance state. You have the option to control the logic state of the pin by using an external pull device or by configuring the internal pull device. When the CPU enters Run mode from Wait or Stop mode recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode after exiting Power-Down mode, the pin returns to its reset state.

34.4.3 Using the ewm_in signal

The ewm_in signal is a digital input signal for the safety status of external safety circuits. This signal allows an external circuit to control the assertion of the ewm_out_b signal. For example, in the application, an external circuit monitors a critical safety function, and if there is a fault with the safety function, the external circuit can actively initiate the ewm_out_b signal, which controls the gating circuit.

The ewm_in signal is ignored if EWM is disabled, or if CTRL[INEN] = 0 after any reset.

After you enable EWM (by writing 1 to CTRL[EWMEN]) and the ewm_in functionality (by writing 1 to CTRL[INEN]), the ewm_in signal must be in Deasserted state before the CPU starts refreshing EWM. This ensures that the ewm_out_b signal stays in Deasserted state; otherwise, the ewm_out_b output signal asserts.

34.5 Memory map and register definitions

This section contains the module memory map and registers.

NOTE

EWM supports only 8-bit register accesses; 16-bit and 32-bit accesses are not supported.

34.5.1 EWM register descriptions

34.5.1.1 EWM memory map

EWM0 base address: 4001_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	8	RW	00h
1h	Service (SERV)	8	W	00h
2h	Compare Low (CMPL)	8	RW	00h
3h	Compare High (CMPH)	8	RW	FFh
5h	Clock Prescaler (CLKPRESCALER)	8	RW	00h

34.5.1.2 Control (CTRL)

Offset

Register	Offset
CTRL	0h

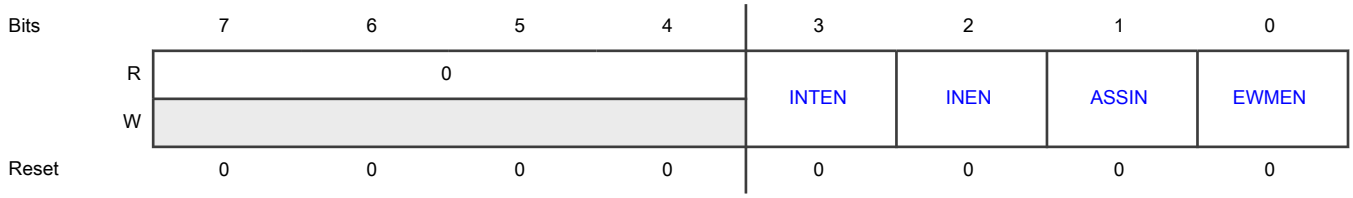
Function

Controls the functionality of EWM.

NOTE

You can write to CTRL[INEN], CTRL[ASSIN], and CTRL[EWMEN] only once after a CPU reset. Modifying these fields more than once generates a bus transfer error.

Diagram



Fields

Field	Function
7-4 —	Reserved
3 INTEN	Interrupt Enable Enables interrupt request generation. If this field = 1 and the ewm_out_b signal is asserted, an interrupt request is generated. To deassert interrupt requests, write 0 to this field. 0b - Deasserts interrupt requests 1b - Generates interrupt requests
2 INEN	Input Enable Enables the ewm_in port. When this field = 1, it enables the ewm_in port. 0b - Disables 1b - Enables
1 ASSIN	Assertion State Select Specifies the asserted state of the ewm_in signal. By default, the asserted state of the ewm_in signal is logic 0 (active-low), which is when this field = 0. When this field = 1, the ewm_in asserted state is logic 1 (active-high). You can use this field to change the expected polarity of the ewm_in signal. 0b - Logic 0 1b - Logic 1
0 EWMEN	EWM Enable Enables the EWM module. This field resets the EWM counter to 0 and deasserts the ewm_out_b signal. If this field = 1, it enables the EWM module, and if the field = 0, it disables the EWM module. You cannot re-enable this field until the next reset because of its write-once nature. 0b - Disables 1b - Enables

34.5.1.3 Service (SERV)

Offset

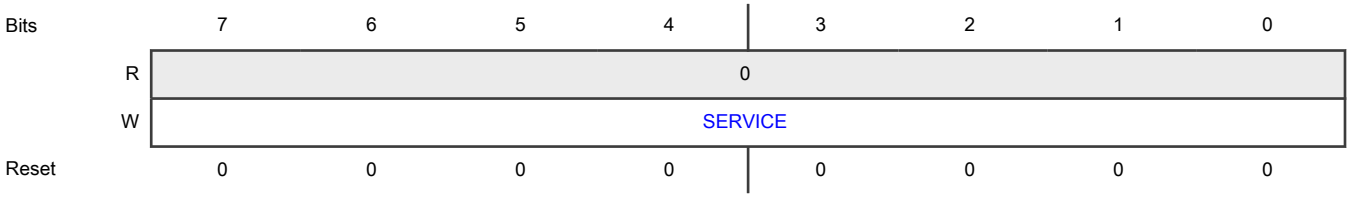
Register	Offset
SERV	1h

Function

Provides an interface from the CPU to the EWM module.

Attempted reads of this register return 0.

Diagram



Fields

Field	Function
7-0 SERVICE	Service Provides an interface from the CPU to the EWM module. The EWM refresh mechanism requires the CPU to write these values to this field: a first data byte of B4h, followed by a second data byte of 2Ch. The EWM refresh action is invalid if either of the following conditions is true: • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte, known as EWM_refresh_time. The number of peripheral bus clock cycles required for EWM_refresh_time is 15.

34.5.1.4 Compare Low (CMPL)

Offset

Register	Offset
CMPL	2h

Function

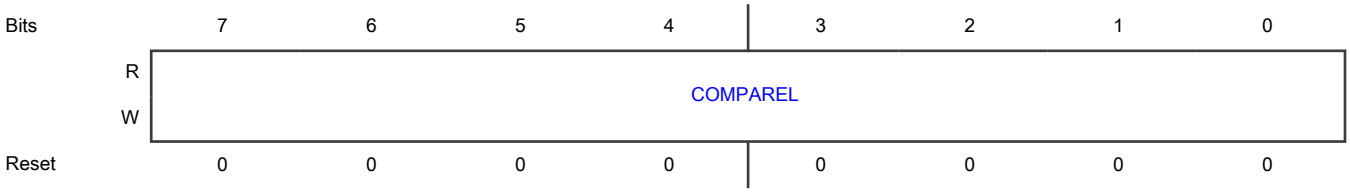
Determines the lower value of the windowed time frame for the correct EWM refresh operation.

This register is reset to 0 after a CPU reset. This provides no minimum time for the CPU to refresh the EWM counter.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error.

Diagram



Fields

Field	Function
7-0 COMPAREL	Compare Low Configures the minimum counter value when refreshes are allowed. If a refresh is attempted while the counter value is lower than COMPAREL, then the ewm_out_b signal is asserted. To prevent runaway code from changing the value of this field, you must write to this field after a CPU reset even if the (default) minimum refresh time is required.

34.5.1.5 Compare High (CMPH)

Offset

Register	Offset
CMPH	3h

Function

Determines the higher value of the windowed time frame for the correct EWM refresh operation.

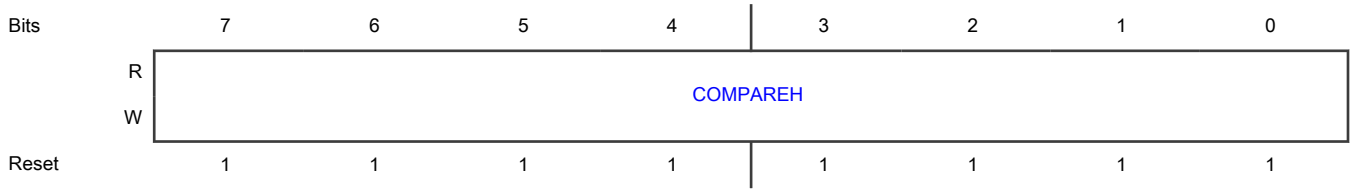
This register is reset to FFh after a CPU reset. This provides a maximum time of up to 256 clocks for the CPU to refresh the EWM counter.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error.

The valid values for this register are up to FEh because the EWM counter never expires when the value of COMPAREH = FFh. The expiration happens only if the EWM counter is greater than the value of COMPAREH.

Diagram



Fields

Field	Function
7-0 COMPAREH	Compare High Configures the maximum counter value till when refreshes are allowed. If a refresh is not attempted while the counter value is greater than COMPAREH, then the ewm_out_b signal is asserted. To prevent runaway code from changing the value of this field, you must write to this field after a CPU reset.

34.5.1.6 Clock Prescaler (CLKPRESCALER)

Offset

Register	Offset
CLKPRESCALER	5h

Function

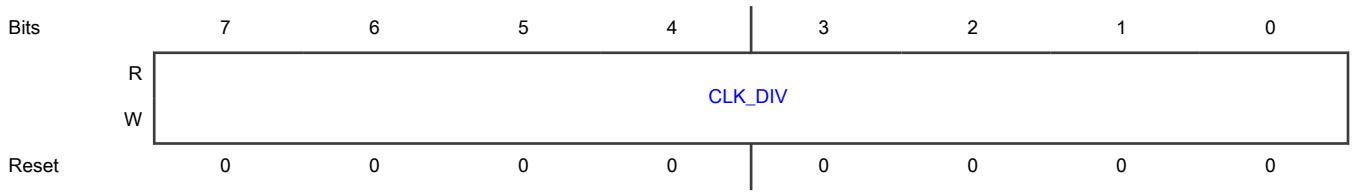
Prescales the EWM counter clock source by a clock divider.

This register is reset to 00h after a CPU reset.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error. You must write the required prescaler value before enabling EWM.

Diagram



Fields

Field	Function
7-0	Clock Divider Prescales the selected low-power clock source for running the EWM counter:

Table continues on the next page...

Field	Function
CLK_DIV	Prescaled clock frequency = low-power clock source frequency ÷ (1 + the value of CLK_DIV) See chip-specific information for low-power clock source frequency used in your device.

Chapter 35
Trigger Multiplexer (TRGMUX)

35.1 Chip-specific TRGMUX information

Table 227. Reference links to related information

Topic	Related module	Reference
Full description	TRGMUX	TRGMUX
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

35.1.1 Module instances

This device has one instance of the TRGMUX module, TRGMUX0.

35.1.2 TRGMUX0 inputs

Mux Select	Source Module	Source Function
0	-	Disabled
1	-	Always High
2	TRGMUX0	TRGMUX0_IN0
3	TRGMUX0	TRGMUX0_IN1
4	TRGMUX0	TRGMUX0_IN2
5	TRGMUX0	TRGMUX0_IN3
6	WUU0	Trigger Event
TIMER Modules (TMR)		
7	RTC0	Alarm Event
8	RTC0	Seconds Match
9	LPTMR0	Counter Match
10	LPTMR1	Counter Match
11	LPIT0	Channel 0
12	LPIT0	Channel 1
13	LPIT0	Channel 2
14	LPIT0	Channel 3
15	TPM0	Channel 0
16	TPM0	Channel 1

Table continues on the next page...

Table continued from the previous page...

Mux Select	Source Module	Source Function
17	TPM0	Channel 2
18	TPM0	Channel 3
19	TPM0	Channel 4
20	TPM0	Channel 5
21	TPM0	Overflow
22	TPM1	Channel 0
23	TPM1	Channel 1
24	TPM1	Channel 2
25	TPM1	Channel 3
26	TPM1	Channel 4
27	TPM1	Channel 5
28	TPM1	Overflow
COMMUNICATION Modules (COM)		
29	LPI2C0	Master End of Packet
30	LPI2C0	Slave End of Packet
31	LPI2C1	Master End of Packet
32	LPI2C1	Slave End of Packet
33	LPSPI0	End of Frame
34	LPSPI0	Received Data Word
35	LPSPI1	End of Frame
36	LPSPI1	Received Data Word
37	LPUART0	Received Data Word
38	LPUART0	Transmitted Data Word
39	LPUART0	Receive Line Idle
40	LPUART1	Received Data Word
41	LPUART1	Transmitted Data Word
42	LPUART1	Receive Line Idle
43	FLEXIO0	Channel 0
44	FLEXIO0	Channel 1
45	FLEXIO0	Channel 2
46	FLEXIO0	Channel 3
47	FLEXIO0	Channel 4

Table continues on the next page...

Table continued from the previous page...

Mux Select	Source Module	Source Function
48	FLEXIO0	Channel 5
49	FLEXIO0	Channel 6
50	FLEXIO0	Channel 7
HUMAN MACHINE INTERFACE Modules (HMI)		
51	GPIOA	Pin event Trigger 0
52	GPIOA	Pin event Trigger 1
53	GPIOB	Pin event Trigger 0
54	GPIOB	Pin event Trigger 1
55	GPIOC	Pin event Trigger 0
56	GPIOC	Pin event Trigger 1
57	GPIOD	Pin event Trigger 0
58	GPIOD	Pin event Trigger 1
ANALOG Modules (ANA)		
59	ADC-GP0	Trigger Output 0
60	ADC-GP0	Trigger Output 1
61	ADC-GP0	Trigger Output 2
62	ADC-GP0	Trigger Output 3
63	CMP-GP0	Comparator Output
64	CMP-GP1	Comparator Output
SYSTEM Modules (SYS)		
65	SPC0	DCDC Burst Trig
RADIO FREQUENCY Modules (RF)		
66	RF-2.4G	TOF TIMESTAMP TRIG
67	RF-2.4G	LANT_SW
SYSTEM Modules (SYS)		
68 - 255	Reserved	Reserved

35.1.3 TRGMUX0 outputs

Destination Module	Trigger Output 3	Trigger Output 2	Trigger Output 1	Trigger Output 0
TRGMUX0	TRGMUX_Output3	TRGMUX_Output2	TRGMUX_Output1	TRGMUX_Output0
LPIT0	Trigger	Trigger	Trigger	Trigger
TPM0	Reserved	Input Capture, Trigger, Output Modulation	Input Capture, Trigger, Output Modulation	Input Capture, Trigger, Output Modulation
TPM1	Reserved	Input Capture, Trigger, Output Modulation	Input Capture, Trigger, Output Modulation	Input Capture, Trigger, Output Modulation
COMMUNICATION Modules (COM)				
LPI2C0	Reserved	Reserved	Reserved	Input Trigger
LPI2C1	Reserved	Reserved	Reserved	Input Trigger
LPSPI0	Reserved	Reserved	Reserved	Host Request
LPSPI1	Reserved	Reserved	Reserved	Host Request
LPUART0	Reserved	Reserved	Reserved	RX Input, CTS Input, TXD Modulation
LPUART1	Reserved	Reserved	Reserved	RX Input, CTS Input, TXD Modulation
FlexIO0	Trigger	Trigger	Trigger	Trigger
ANALOG Modules (ANA)				
ADC-GP0	Trigger	Trigger	Trigger	Trigger
CMP-GP0	Reserved	Reserved	Reserved	Window, Trigger
CMP-GP1	Reserved	Reserved	Reserved	Window, Trigger

35.2 Overview

TRGMUX allows software to configure the trigger inputs for various peripherals.

35.2.1 Block diagram

The block diagram below shows the trigger selection logic of the TRGMUX module.

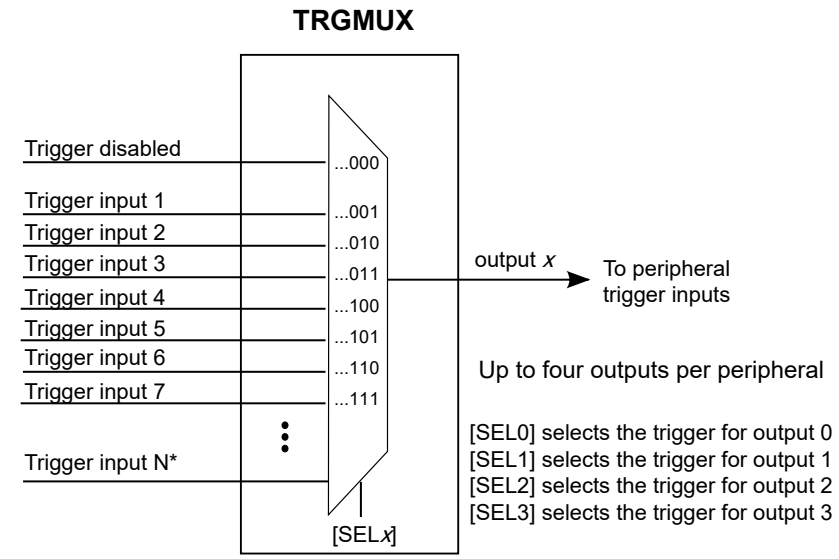


Figure 92. TRGMUX block diagram

35.2.2 Features

The TRGMUX module allows software to select the trigger source for peripherals.

Each peripheral has its own dedicated TRGMUX register. See each peripheral's TRGMUX register for details.

35.3 Functional description

The following sections describe functional details of the TRGMUX module.

35.3.1 Clocking

This module has no clocking considerations.

35.3.2 Interrupts

This module has no interrupts.

35.4 External signals

This module has no external signals.

35.5 Initialization

This module does not require initialization.

35.6 Memory map and register definition

The TRGMUX registers contain fields for selecting trigger sources for peripheral modules.

TRGMUX registers can be written only in supervisor mode.

35.6.1 TRGMUX register descriptions

35.6.1.1 TRGMUX memory map

Table 228. Select Bit Fields

Field	Description
SELx	This read/write field is used to configure the MUX select for the peripheral trigger inputs. 0h - Trigger function is disabled 1h - Trigger function is always high 2h - TRGMUX0 TRGMUX0_IN0 input is selected 3h - TRGMUX0 TRGMUX0_IN1 input is selected 4h - TRGMUX0 TRGMUX0_IN2 input is selected 5h - TRGMUX0 TRGMUX0_IN3 input is selected 6h - WUU0 Trigger Event input is selected 7h - RTC0 Alarm Event input is selected 8h - RTC0 Seconds Match input is selected 9h - LPTMR0 Counter Match input is selected Ah - LPTMR1 Counter Match input is selected Bh - LPIT0 Channel 0 input is selected Ch - LPIT0 Channel 1 input is selected Dh - LPIT0 Channel 2 input is selected Eh - LPIT0 Channel 3 input is selected Fh - TPM0 Channel 0 input is selected 10h - TPM0 Channel 1 input is selected 11h - TPM0 Channel 2 input is selected 12h - TPM0 Channel 3 input is selected 13h - TPM0 Channel 4 input is selected 14h - TPM0 Channel 5 input is selected 15h - TPM0 Overflow input is selected 16h - TPM1 Channel 0 input is selected 17h - TPM1 Channel 1 input is selected 18h - TPM1 Channel 2 input is selected 19h - TPM1 Channel 3 input is selected 1Ah - TPM1 Channel 4 input is selected 1Bh - TPM1 Channel 5 input is selected 1Ch - TPM1 Overflow input is selected

Table continues on the next page...

Table 228. Select Bit Fields

Field	Description
	1Dh - LPI2C0 Master End of Packet input is selected 1Eh - LPI2C0 Slave End of Packet input is selected 1Fh - LPI2C1 Master End of Packet input is selected 20h - LPI2C1 Slave End of Packet input is selected 21h - LPSPI0 End of Frame input is selected 22h - LPSPI0 Received Data Word input is selected 23h - LPSPI1 End of Frame input is selected 24h - LPSPI1 Received Data Word input is selected 25h - LPUART0 Received Data Word input is selected 26h - LPUART0 Transmitted Data Word input is selected 27h - LPUART0 Receive Line Idle input is selected 28h - LPUART1 Received Data Word input is selected 29h - LPUART1 Transmitted Data Word input is selected 2Ah - LPUART1 Receive Line Idle input is selected 2Bh - FLEXIO0 Channel 0 input is selected 2Ch - FLEXIO0 Channel 1 input is selected 2Dh - FLEXIO0 Channel 2 input is selected 2Eh - FLEXIO0 Channel 3 input is selected 2Fh - FLEXIO0 Channel 4 input is selected 30h - FLEXIO0 Channel 5 input is selected 31h - FLEXIO0 Channel 6 input is selected 32h - FLEXIO0 Channel 7 input is selected 33h - GPIOA Pin event Trigger 0 input is selected 34h - GPIOA Pin event Trigger 1 input is selected 35h - GPIOB Pin event Trigger 0 input is selected 36h - GPIOB Pin event Trigger 1 input is selected 37h - GPIOC Pin event Trigger 0 input is selected 38h - GPIOC Pin event Trigger 1 input is selected 39h - GPIOD Pin event Trigger 0 input is selected 3Ah - GPIOD Pin event Trigger 1 input is selected 3Bh - ADC-GP0 Trigger Output 0 input is selected 3Ch - ADC-GP0 Trigger Output 1 input is selected 3Dh - ADC-GP0 Trigger Output 2 input is selected 3Eh - ADC-GP0 Trigger Output 3 input is selected

Table continues on the next page...

Table 228. Select Bit Fields

Field	Description
	3Fh - CMP-GP0 Comparator Output input is selected 40h - CMP-GP1 Comparator Output input is selected 41h - SPC0 DCDC Burst Trig input is selected 42h - RF-2.4G TOF TIMESTAMP TRIG input is selected 43h - RF-2.4G LANT_SW input is selected 44h - Unused 45h - Unused 46h - Unused 47h - Unused 48h - Unused 49h - Unused 4Ah - Unused 4Bh - Unused 4Ch - Unused 4Dh - Unused 4Eh - Unused 4Fh - Unused 50h - Unused 51h - Unused 52h - Unused 53h - Unused 54h - Unused 55h - Unused 56h - Unused 57h - Unused 58h - Unused 59h - Unused 5Ah - Unused 5Bh - Unused 5Ch - Unused 5Dh - Unused 5Eh - Unused 5Fh - Unused 60h - Unused

Table continues on the next page...

Table 228. Select Bit Fields

Field	Description
	61h - Unused 62h - Unused 63h - Unused 64h - Unused 65h - Unused 66h - Unused 67h - Unused 68h - Unused 69h - Unused 6Ah - Unused 6Bh - Unused 6Ch - Unused 6Dh - Unused 6Eh - Unused 6Fh - Unused 70h - Unused 71h - Unused 72h - Unused 73h - Unused 74h - Unused 75h - Unused 76h - Unused 77h - Unused 78h - Unused 79h - Unused 7Ah - Unused 7Bh - Unused 7Ch - Unused 7Dh - Unused 7Eh - Unused 7Fh - Unused

TRGMUX0 base address: 4001_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX TRGMUX_OUT0 Register (TRGMUX_OUT0)	32	RW	0000_0000h
4h	TRGMUX LPIT0 Register (LPIT0)	32	RW	0000_0000h
8h	TRGMUX TPM0 Register (TPM0)	32	RW	0000_0000h
Ch	TRGMUX TPM1 Register (TPM1)	32	RW	0000_0000h
10h	TRGMUX LPI2C0 Register (LPI2C0)	32	RW	0000_0000h
14h	TRGMUX LPI2C1 Register (LPI2C1)	32	RW	0000_0000h
18h	TRGMUX LPSPI0 Register (LPSPI0)	32	RW	0000_0000h
1Ch	TRGMUX LPSPI1 Register (LPSPI1)	32	RW	0000_0000h
20h	TRGMUX LPUART0 Register (LPUART0)	32	RW	0000_0000h
24h	TRGMUX LPUART1 Register (LPUART1)	32	RW	0000_0000h
28h	TRGMUX FlexIO0 Register (FlexIO0)	32	RW	0000_0000h
2Ch	TRGMUX ADC_GP0 Register (ADC_GP0)	32	RW	0000_0000h
30h	TRGMUX CMP_GP0 Register (CMP_GP0)	32	RW	0000_0000h
34h	TRGMUX CMP_GP1 Register (CMP_GP1)	32	RW	0000_0000h

35.6.1.2 TRGMUX TRGMUX_OUT0 Register (TRGMUX_OUT0)

Offset

Register	Offset
TRGMUX_OUT0	0h

Function

This register is for the TRGMUX_OUT0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Source Select 3 This read/write bit field is used to configure the MUX source select for output 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Source Select 2 This read/write bit field is used to configure the MUX source select for output 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

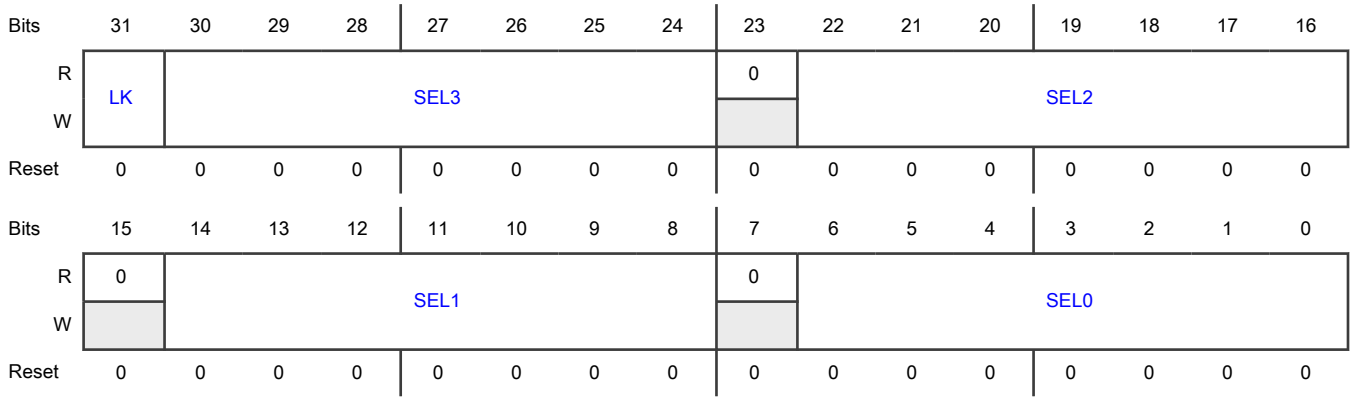
35.6.1.3 TRGMUX LPIT0 Register (LPIT0)

Offset

Register	Offset
LPIT0	4h

Function
This register is for the LPIT0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Source Select 3 This read/write bit field is used to configure the MUX source select for output 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Source Select 2 This read/write bit field is used to configure the MUX source select for output 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.4 TRGMUX TPM0 Register (TPM0)

Offset

Register	Offset
TPM0	8h

Function

This register is for the TPM0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Source Select 2 This read/write bit field is used to configure the MUX source select for output 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

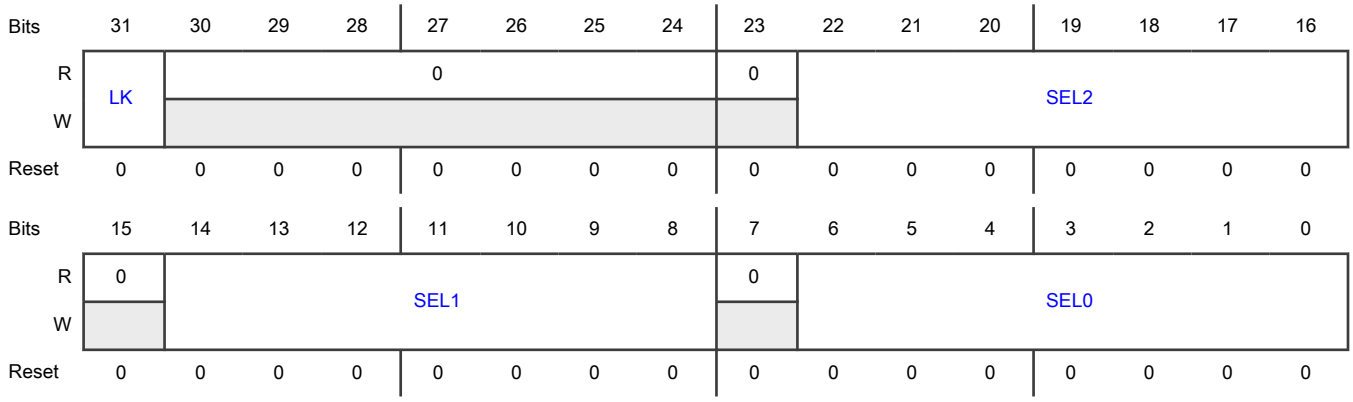
35.6.1.5 TRGMUX TPM1 Register (TPM1)

Offset

Register	Offset
TPM1	Ch

Function
This register is for the TPM1 module.

Diagram



Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Source Select 2 This read/write bit field is used to configure the MUX source select for output 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.6 TRGMUX LPI2C0 Register (LPI2C0)

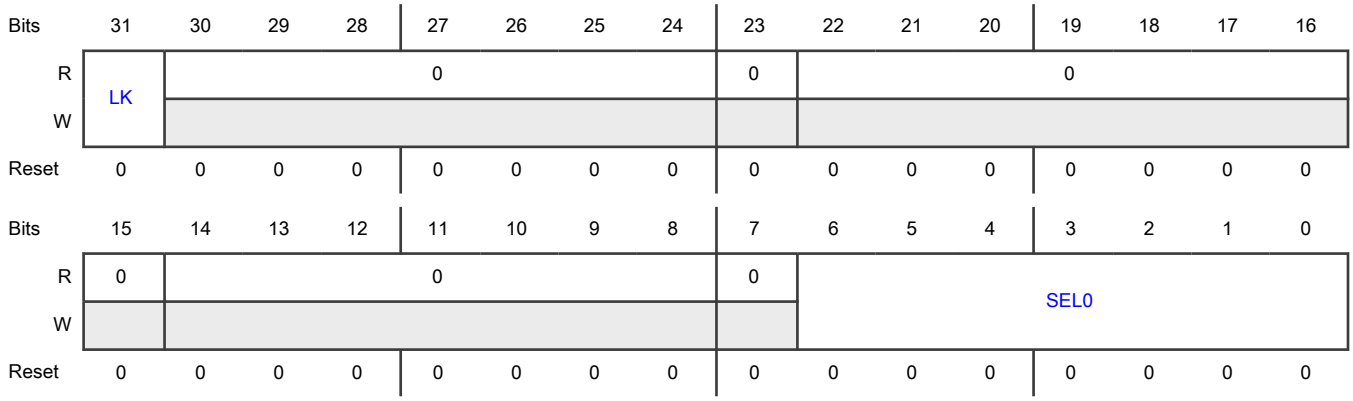
Offset

Register	Offset
LPI2C0	10h

Function

This register is for the LPI2C0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.7 TRGMUX LPI2C1 Register (LPI2C1)

Offset

Register	Offset
LPI2C1	14h

Function
This register is for the LPI2C1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.8 TRGMUX LPSPI0 Register (LPSPI0)

Offset

Register	Offset
LPSPI0	18h

Function

This register is for the LPSPI0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

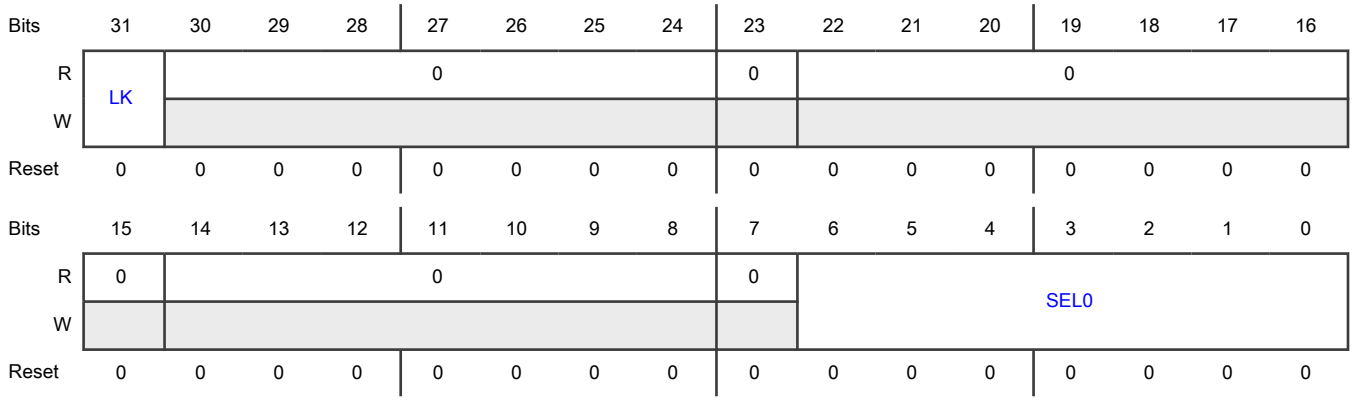
35.6.1.9 TRGMUX LPSPI1 Register (LPSPI1)

Offset

Register	Offset
LPSPI1	1Ch

Function
This register is for the LPSPI1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.10 TRGMUX LPUART0 Register (LPUART0)

Offset

Register	Offset
LPUART0	20h

Function
This register is for the LPUART0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.11 TRGMUX LPUART1 Register (LPUART1)

Offset

Register	Offset
LPUART1	24h

Function

This register is for the LPUART1 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

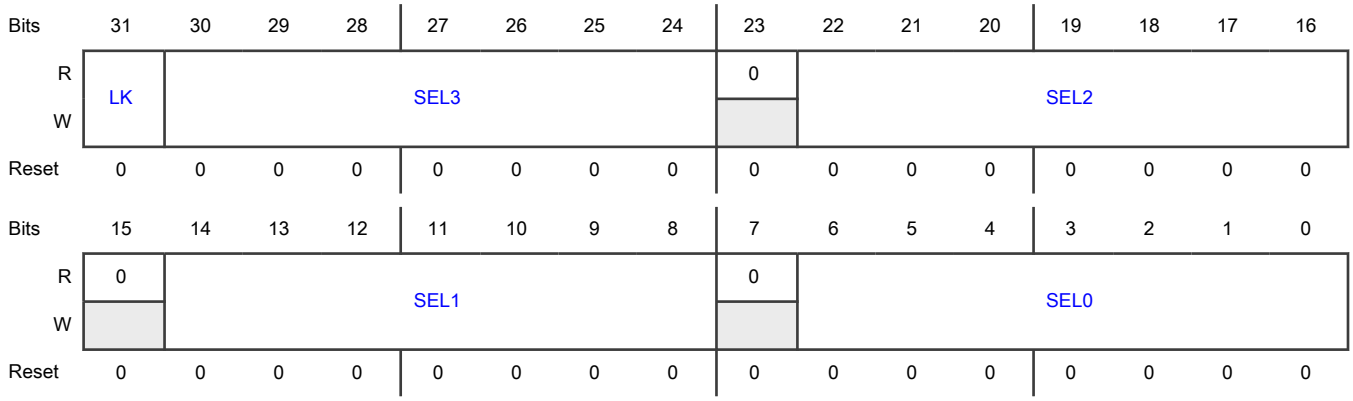
35.6.1.12 TRGMUX FlexIO0 Register (FlexIO0)

Offset

Register	Offset
FlexIO0	28h

Function
This register is for the FlexIO0 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Source Select 3 This read/write bit field is used to configure the MUX source select for output 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Source Select 2 This read/write bit field is used to configure the MUX source select for output 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.13 TRGMUX ADC_GP0 Register (ADC_GP0)

Offset

Register	Offset
ADC_GP0	2Ch

Function

This register is for the ADC_GP0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Source Select 3 This read/write bit field is used to configure the MUX source select for output 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Source Select 2 This read/write bit field is used to configure the MUX source select for output 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14-8 SEL1	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

35.6.1.14 TRGMUX CMP_GP0 Register (CMP_GP0)

Offset

Register	Offset
CMP_GP0	30h

Function
This register is for the CMP_GP0 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

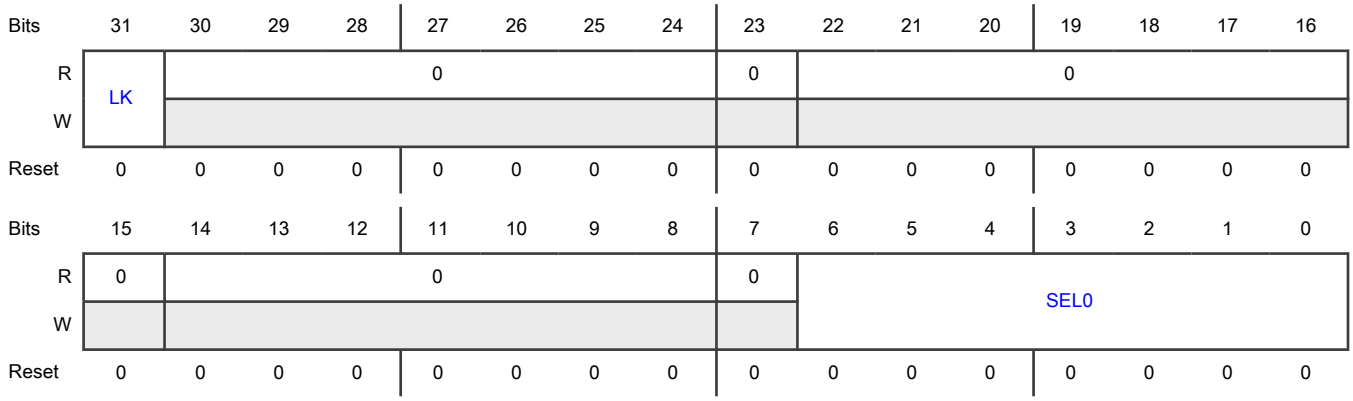
35.6.1.15 TRGMUX CMP_GP1 Register (CMP_GP1)

Offset

Register	Offset
CMP_GP1	34h

Function
This register is for the CMP_GP1 module.

Diagram



Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Source Select 1 This read/write bit field is used to configure the MUX source select for output 0. For the field setting definitions, see Memory map and register definition.

Chapter 36
Wakeup Unit (WUU)

36.1 Chip-specific WUU information

Table 229. Reference links to related information

Topic	Related module	Reference
Full description	WUU	WUU
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

36.1.1 Module instances

This device contains one instance of the WUU module, WUU0.

36.1.2 WUU0 sources

The device uses the following inputs as wakeup sources to the WUU0 module. The WUU0_Px inputs are connections to external physical pins. The WUU0_MxIF inputs are connections to the internal peripheral interrupt flags from different modules that can operate in low-power modes. The WUU0_MxDR inputs are connections to the internal peripheral Asynchronous DMA or module Triggers from different modules that can operate in low-power modes.

NOTE

In addition to the WUU0 wakeup sources, the device also wakes from low power modes when NMI_b or RESET_b pins are enabled and the respective pin is asserted.

Table 230. Wakeup sources for WUU0 inputs

WUU0 input	Source description
WUU0_P0	PTA0
WUU0_P1	—
WUU0_P2	PTA4
WUU0_P3	PTA17
WUU0_P4	PTA19
WUU0_P5	PTA21
WUU0_P6	—
WUU0_P7	PTC0
WUU0_P8	PTC1
WUU0_P9	PTC2
WUU0_P10	PTC4
WUU0_P11	PTC6

Table continues on the next page...

Table 230. Wakeup sources for WUU0 inputs (continued)

WUU0 input	Source description
WUU0_P12	PTC7
WUU0_P13	PTB0
WUU0_P14	PTB3
WUU0_P15	PTB4
WUU0_P16	—
WUU0_P17	—
WUU0_P18	—
WUU0_P19	—
WUU0_P20	—
WUU0_P21	—
WUU0_P22	—
WUU0_P23	—
WUU0_P24	—
WUU0_P25	—
WUU0_P26	—
WUU0_P27	CMP0 Output
WUU0_P28	CMP1 Output
WUU0_P29	—
WUU0_P30	—
WUU0_P31	—
WUU0_M0IF	LPTMR0, LPTMR1
WUU0_M1IF	GPIOD Interrupt 0
WUU0_M2IF	GPIOD Interrupt 1
WUU0_M3IF	Wake-From-Radio (WFR)
WUU0_M4IF	Wake From Smart Power Switch Domain
WUU0_M5IF	Digital Tamper Detect
WUU0_M6IF	Real Time Clock 0 Alarm
WUU0_M7IF	Real Time Clock 0 Seconds
WUU0_M0DR	LPTMR0 Asynchronous DMA
WUU0_M1DR	LPTMR1 Asynchronous DMA
WUU0_M2DR	Radio Asynchronous wakeup request
WUU0_M3DR	Reserved

Table continues on the next page...

Table 230. Wakeup sources for WUU0 inputs (continued)

WUU0 input	Source description
WUU0_M4DR	GPIOD Asynchronous DMA request 0
WUU0_M5DR	GPIOD Asynchronous DMA request 1
WUU0_M6DR	Reserved
WUU0_M7DR	Reserved
WUU0_M8DR	LPTMR0 Asynchronous Trigger
WUU0_M9DR	LPTMR1 Asynchronous Trigger

36.1.3 LPO clock

The LPO mentioned in this chapter is the 32 kHz clock from CCM32K, see 32 kHz Clock Control Module (CCM32K) for details.

36.2 Overview

WUU allows software to select external pins and on-chip modules as interrupt wake-up sources from Deep Sleep/Power Down power modes. External pins can also be configured as interrupt wake-up sources from all power modes. WUU lets modules generate a temporary wake-up from Deep Sleep to allow servicing of DMA or trigger events. WUU also allows pins to generate a temporary wake-up from Deep Sleep for servicing DMA or trigger events. Digital filtering is also available for the external wakeup pins.

Input sources to WUU are described in the chip-specific WUU information section.

36.2.1 Features

The WUU module features include:

- Support for external input pins and on-chip modules with individual enable bits to wake the chip from low leakage modes.
- Input sources may be external pins or from on-chip modules capable of running in Deep Sleep or Power Down. See the chip-specific WUU information for the wakeup input sources for this device.
- Support to configure on-chip modules as DMA or trigger sources for temporary wakeup from Deep Sleep.
- Support to configure external pins as DMA request or trigger sources for temporary wakeup from Deep Sleep.
- External input pins programmable for falling-edge, rising-edge, or any-edge detection.
- Optional digital filters provided to qualify an external pin detect. Note that when the LPO clock is disabled, filters are disabled and bypassed.
- All external input pins can be filtered, but only 8'd2 filters are available at a given time.
- Support to enable external input pin and filter detection in *all* power modes (not limited to Deep Sleep/Power Down).

36.2.2 Block diagram

The following figure is the block diagram for the WUU module.

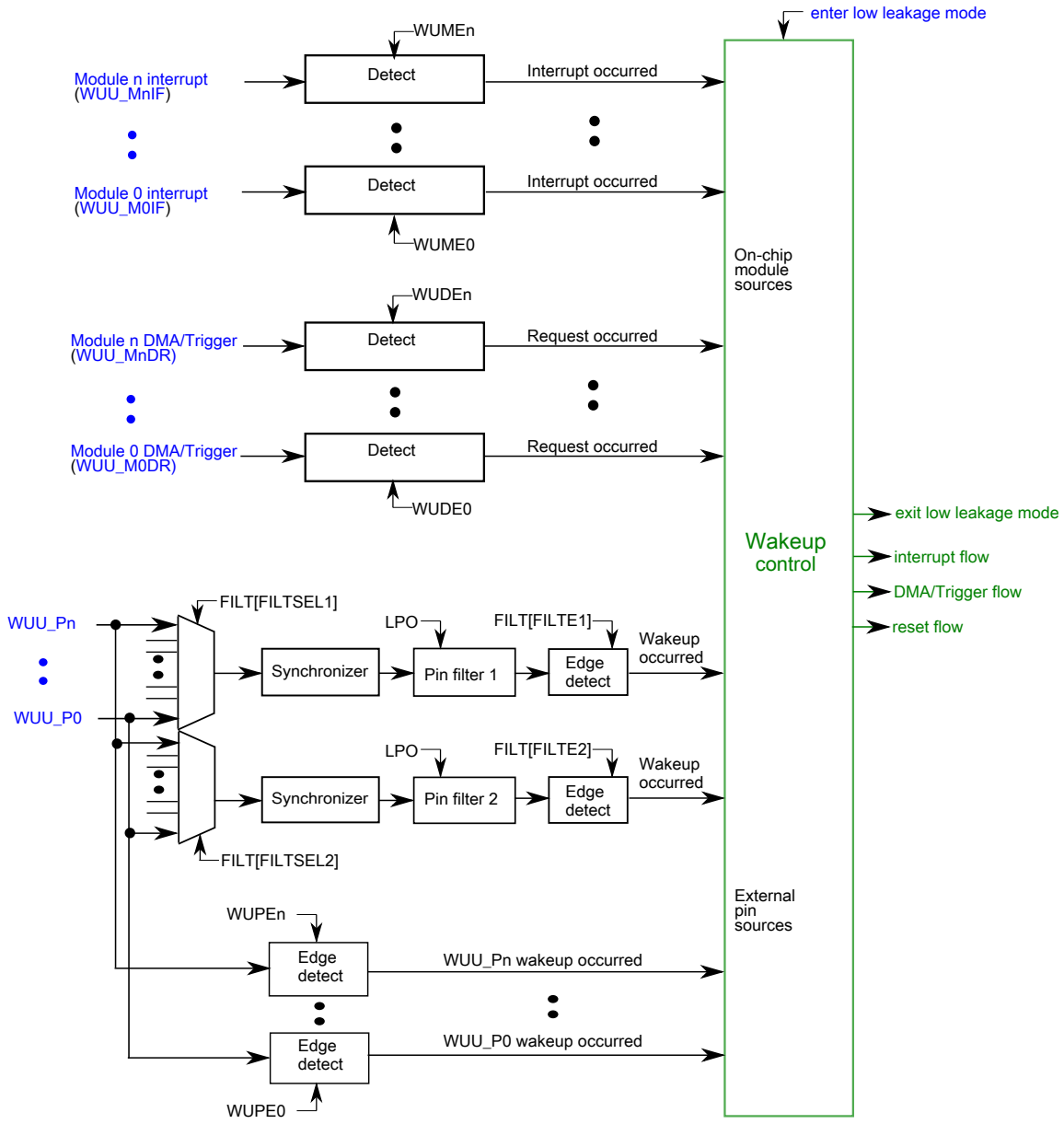


Figure 93. WUU block diagram

36.3 Functional description

The low-leakage wakeup unit (WUU) module allows on-chip modules and external input pins as a source of wakeup from low-leakage modes.

The WUU module contains pin enables for each external pin and on-chip module. For each external pin, the user can disable or select the edge type for the wakeup with the following options:

- Falling-edge
- Rising-edge
- Either-edge

Exit due to a pin event will trigger an WUU interrupt after wakeup is complete. When an external pin is enabled as a wakeup source, the pin must be configured as an input pin. A pin can also be configured to generate an interrupt from Deep Sleep/Power Down or a DMA/trigger request for temporary wakeup from Deep Sleep. Detection logic for a pin can optionally remain enabled during all power modes using the PMC register.

The WUU implements optional 3-cycle glitch filters, based on the LPO clock. A detected external pin is required to remain asserted until the enabled glitch filter times out. Additional latency of up to 2 cycles is due to synchronization, which results in a total of up to 5 cycles of delay before the detect circuit alerts the system to the wakeup event when the filter function is enabled. Two wakeup detect filters are available for selected external pins. Glitch filtering is not provided for the on-chip modules. A filter can be configured to generate an interrupt from Deep Sleep/Power Down or a DMA/trigger request for temporary wakeup from Deep Sleep. Detection logic for a filter can optionally remain enabled during all power modes using the FMC register.

For on-chip module interrupts, the WUMEx bit of the Module Interrupt Enable (ME) register enables the associated module interrupt as a wakeup source. Exit due to a module interrupt will not trigger an WUU interrupt.

For on-chip module DMA/trigger requests, the DE[WUDEx] bit enables the associated module DMA/trigger request as a wakeup source. Exit due to a module DMA/trigger request will not trigger a WUU interrupt.

36.3.1 Modes of operation

The WUU module becomes functional on entry into a low-leakage power mode. After recovery from Deep Sleep, the WUU is immediately disabled. After recovery from Power Down, the WUU continues to detect wake-up events until the user has reinitialized the system and deasserted isolation. Detection of external pin/filter events can also optionally remain enabled in all power modes using the PMC/FMC registers.

36.3.1.1 Deep Sleep mode

Wakeup events due to either an external pin input (WUU_Px) or an on-chip module interrupt (WUU_MxIF), result in a CPU interrupt flow to begin user code execution. The source of the wakeup determines the subsequent path of code execution:

- Pin events trigger the WUU interrupt service routine.
- Module interrupts trigger that same module's interrupt service routine.

NOTE

The WUU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully recover from Deep Sleep after an external pin event.

Wakeup events due to an on-chip module request (WUU_MxDR) DMA/trigger request result in a temporary exit from Deep Sleep to allow the request to be serviced while the CPU clock remains gated. After the request is serviced, the system re-enters Deep Sleep.

Wakeup events due to an external pin (WUU_Px) DMA/trigger request, result in a temporary exit from Deep Sleep to allow the request to be serviced while the CPU clock remains gated. After the request is serviced, the system re-enters Deep Sleep.

36.3.1.2 Power Down modes

For any WUU wakeup from Power Down, recovery is always via a reset flow. The source of the wakeup determines the subsequent path of code execution:

- Pin events trigger the WUU interrupt service routine.
- Module interrupts trigger that same module's interrupt service routine.

36.3.1.3 Non-low leakage modes

The WUU is inactive in all non-low leakage modes, with the exception of external pin/filter detection which can explicitly be kept enabled during all power modes using the PMC/FMC registers. The WUU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

36.3.2 Clocking

To provide the wakeup function for the chip, make sure clocking is available for WUU during chip low-power operation. See the chip-specific WUU information for the chip clock connection details.

WUU filters are clocked by the LPO clock.

36.3.3 Interrupts

On-chip module interrupts can be enabled as wakeup sources in Module Interrupt Enable (ME)

36.4 External signals

WUU signals are shown in the table found here.

Table 231. WUU signal descriptions

Signal	Description	I/O
WUU_Pn	External pin wakeup inputs. Can be enabled to detect a rising-edge, a falling-edge, or any change.	I
WUU_MnIF	On-chip module interrupt flags	I
WUU_MnDR	On-chip module DMA/trigger requests	I

36.5 Initialization

For an enabled module wakeup input, the module's flag must be cleared by software before entering Deep Sleep or Power Down mode to avoid an immediate exit from the mode.

Flags associated with external input pins, filtered and unfiltered, must also be cleared by software prior to entry to Deep Sleep or Power Down mode.

When an external wake-up pin filter is first enabled in the FILT register, filter operation begins immediately. After enabling an external pin filter or changing its source pin, wait at least five LPO clock cycles to allow the filter to initialize before entering Deep Sleep or Power Down mode or before enabling filter detection for all power modes in the FMC register.

NOTE

After recovering from a Power Down mode, software must restore the chip configuration before relaxing any power domain isolation controls. In particular, to avoid any WUU flag from being falsely set, restore the pin configuration for the enabled WUU wake-up pins before removing any voltage isolation between power domains.

36.6 Memory map/register definition

Information about WUU registers can be found here.

36.6.1 WUU register descriptions

When an external pin has been configured to be an active wakeup source in all power modes (PMC[WUPMCn] = 1), then the corresponding wakeup pin enable (PEx[WUPEn]) and wakeup pin configuration (PDCx[WUPDCn]) settings for that pin must not be changed. To modify the WUPEn or WUPDCn settings, follow these steps:

1. Clear the WUPMCn value for that pin.
2. Update the corresponding WUPEn or WUPDCn settings as needed.
3. Set the WUPMCn value for that pin back to 1.

NOTE

WUU registers can be written only in supervisor mode. Write accesses in user mode are blocked and result in a bus error.

All WUU registers are reset by Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. WUU registers are unaffected by reset types that do not trigger Chip Reset not Power Down.

36.6.1.1 WUU memory map

WUU0 base address: 4001_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0001h
4h	Parameter (PARAM)	32	R	2020_2002h
8h	Pin Enable 1 (PE1)	32	RW	0000_0000h
Ch	Pin Enable 2 (PE2)	32	RW	0000_0000h
18h	Module Interrupt Enable (ME)	32	RW	0000_0000h
1Ch	Module DMA/Trigger Enable (DE)	32	RW	0000_0000h
20h	Pin Flag (PF)	32	RW	0000_0000h
30h	Pin Filter (FILT)	32	RW	0000_0000h
38h	Pin DMA/Trigger Configuration 1 (PDC1)	32	RW	0000_0000h
3Ch	Pin DMA/Trigger Configuration 2 (PDC2)	32	RW	0000_0000h
48h	Pin Filter DMA/Trigger Configuration (FDC)	32	RW	0000_0000h
50h	Pin Mode Configuration (PMC)	32	RW	0000_0000h
58h	Pin Filter Mode Configuration (FMC)	32	RW	0000_0000h

36.6.1.2 Version ID (VERID)

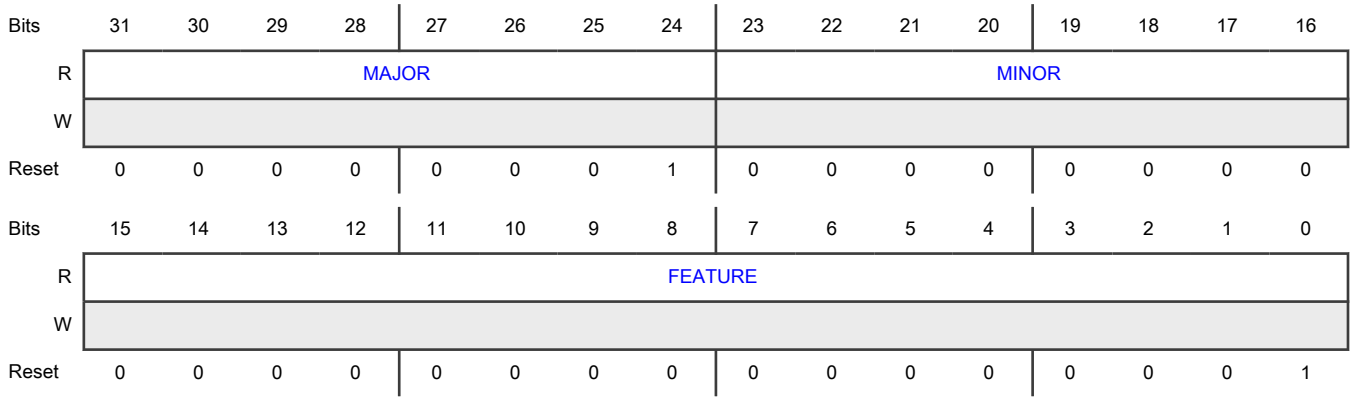
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0000b - Standard features implemented 0000_0000_0000_0001b - Support for DMA/Trigger generation from wakeup pins and filters enabled. Support for external pin/filter detection during all power modes enabled.

36.6.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values that were implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PINS								MODULES							
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMAS								FILTERS							
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-24 PINS	Pin Number Number of Pin wakeup sources supported
23-16 MODULES	Module Number Number of Module wakeup sources supported
15-8 DMAS	DMA Number Number of DMA wakeup sources supported
7-0 FILTERS	Filter Number Number of Pin Filters implemented

36.6.1.4 Pin Enable 1 (PE1)

Offset

Register	Offset
PE1	8h

Function

Contains the field to enable and select the edge detect type for the available external wakeup input pins in the range from WUU_P0 to WUU_P15.

NOTE

Do not modify the value of a WUPEn field when its corresponding PMC[WUPMCn] = 1.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WUPE15		WUPE14		WUPE13		WUPE12		WUPE11		WUPE10		WUPE9		WUPE8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WUPE7		WUPE6		WUPE5		WUPE4		WUPE3		WUPE2		WUPE1		WUPE0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 WUPE15	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
29-28 WUPE14	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
27-26 WUPE13	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
25-24 WUPE12	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
23-22 WUPE11	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
21-20 WUPE10	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
19-18 WUPE9	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
17-16 WUPE8	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
15-14 WUPE7	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
13-12 WUPE6	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
11-10 WUPE5	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
9-8	Wakeup pin enable for WUU_Pn

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUPE4	Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
7-6 WUPE3	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
5-4 WUPE2	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
3-2 WUPE1	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
1-0 WUPE0	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge

36.6.1.5 Pin Enable 2 (PE2)

Offset

Register	Offset
PE2	Ch

Function

Contains the field to enable and select the edge detect type for the available external wakeup input pins in the range from WUU_P16 to WUU_P31.

NOTE

Do not modify the value of a WUPEn field when its corresponding PMC[WUPMCn] = 1.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		WUPE28		WUPE27		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		0		0		0		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 Reserved31	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
29-28 Reserved30	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
27-26 Reserved29	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
25-24 WUPE28	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
23-22 WUPE27	Wakeup pin enable for WUU_Pn Enables the external input pin for wakeup and configures the edge detection. 00b - Disables as a wakeup pin 01b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enables as a wakeup pin. When configured as an interrupt/DMA request: Detect on any edge
21-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved26	00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
19-18 Reserved25	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
17-16 Reserved24	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
15-14 Reserved23	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
13-12 Reserved22	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
11-10 Reserved21	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
9-8 Reserved20	Reserved 00b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Not supported 10b - Not supported 11b - Not supported
7-6 Reserved19	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
5-4 Reserved18	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
3-2 Reserved17	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
1-0 Reserved16	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported

36.6.1.6 Module Interrupt Enable (ME)

Offset

Register	Offset
ME	18h

Function

Contains the bits to enable an on-chip module interrupt as a wakeup source.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	WUME7	WUME6	WUME5	WUME4	WUME3	WUME2	WUME1	WUME0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 Reserved31	Reserved 0b - Not supported 1b - Not supported
30 Reserved30	Reserved 0b - Not supported 1b - Not supported
29 Reserved29	Reserved 0b - Not supported 1b - Not supported
28 Reserved28	Reserved 0b - Not supported 1b - Not supported
27 Reserved27	Reserved 0b - Not supported 1b - Not supported
26 Reserved26	Reserved 0b - Not supported 1b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 Reserved25	Reserved 0b - Not supported 1b - Not supported
24 Reserved24	Reserved 0b - Not supported 1b - Not supported
23 Reserved23	Reserved 0b - Not supported 1b - Not supported
22 Reserved22	Reserved 0b - Not supported 1b - Not supported
21 Reserved21	Reserved 0b - Not supported 1b - Not supported
20 Reserved20	Reserved 0b - Not supported 1b - Not supported
19 Reserved19	Reserved 0b - Not supported 1b - Not supported
18 Reserved18	Reserved 0b - Not supported 1b - Not supported
17 Reserved17	Reserved 0b - Not supported 1b - Not supported
16 Reserved16	Reserved 0b - Not supported 1b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 Reserved15	Reserved 0b - Not supported 1b - Not supported
14 Reserved14	Reserved 0b - Not supported 1b - Not supported
13 Reserved13	Reserved 0b - Not supported 1b - Not supported
12 Reserved12	Reserved 0b - Not supported 1b - Not supported
11 Reserved11	Reserved 0b - Not supported 1b - Not supported
10 Reserved10	Reserved 0b - Not supported 1b - Not supported
9 Reserved9	Reserved 0b - Not supported 1b - Not supported
8 Reserved8	Reserved 0b - Not supported 1b - Not supported
7 WUME7	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables
6 WUME6	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables 1b - Enables
5 WUME5	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables
4 WUME4	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables
3 WUME3	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables
2 WUME2	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables
1 WUME1	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables
0 WUME0	Module iterrupt wakeup enable for module n Enables an on-chip module interrupt as a wakeup source input. 0b - Disables 1b - Enables

36.6.1.7 Module DMA/Trigger Enable (DE)

Offset

Register	Offset
DE	1Ch

Function

Contains the bits to enable an on-chip module DMA/trigger request as a wakeup source.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	WUDE 9	WUDE 8	0	0	WUDE 5	WUDE 4	0	WUDE 2	WUDE 1	WUDE 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 Reserved31	Reserved 0b - Not supported 1b - Not supported
30 Reserved30	Reserved 0b - Not supported 1b - Not supported
29 Reserved29	Reserved 0b - Not supported 1b - Not supported
28 Reserved28	Reserved 0b - Not supported 1b - Not supported
27 Reserved27	Reserved 0b - Not supported 1b - Not supported
26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved26	0b - Not supported 1b - Not supported
25 Reserved25	Reserved 0b - Not supported 1b - Not supported
24 Reserved24	Reserved 0b - Not supported 1b - Not supported
23 Reserved23	Reserved 0b - Not supported 1b - Not supported
22 Reserved22	Reserved 0b - Not supported 1b - Not supported
21 Reserved21	Reserved 0b - Not supported 1b - Not supported
20 Reserved20	Reserved 0b - Not supported 1b - Not supported
19 Reserved19	Reserved 0b - Not supported 1b - Not supported
18 Reserved18	Reserved 0b - Not supported 1b - Not supported
17 Reserved17	Reserved 0b - Not supported 1b - Not supported
16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved16	0b - Not supported 1b - Not supported
15 Reserved15	Reserved 0b - Not supported 1b - Not supported
14 Reserved14	Reserved 0b - Not supported 1b - Not supported
13 Reserved13	Reserved 0b - Not supported 1b - Not supported
12 Reserved12	Reserved 0b - Not supported 1b - Not supported
11 Reserved11	Reserved 0b - Not supported 1b - Not supported
10 Reserved10	Reserved 0b - Not supported 1b - Not supported
9 WUDE9	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables
8 WUDE8	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables
7 Reserved7	Reserved 0b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Not supported
6 Reserved6	Reserved 0b - Not supported 1b - Not supported
5 WUDE5	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables
4 WUDE4	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables
3 Reserved3	Reserved 0b - Not supported 1b - Not supported
2 WUDE2	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables
1 WUDE1	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables
0 WUDE0	DMA/Trigger wakeup enable for module n Enables an on-chip module DMA/trigger request as a wakeup source. 0b - Disables 1b - Enables

36.6.1.8 Pin Flag (PF)

Offset

Register	Offset
PF	20h

Function

Contains the wakeup flags indicating which wakeup source caused the MCU to exit Deep Sleep (if the corresponding pin-mode configuration bit PMC[WUPMCn] = 1) or Power Down mode. For Deep Sleep, this is the source causing the CPU interrupt flow. For Power Down, this is the source causing the MCU reset flow.

To clear a flag, write a 1 to the corresponding WUF*n* bit. If set, the wakeup flag (WUF*n*) remains set even if the associated pin wakeup is disabled in its WUPE*n* field.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	WUF28	WUF27	0	0	0	0	0	0	0	0	0	0	0
W				W1C	W1C											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WUF15	WUF14	WUF13	WUF12	WUF11	WUF10	WUF9	WUF8	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 Reserved31	Reserved 0b - Not supported 1b - Not supported
30 Reserved30	Reserved 0b - Not supported 1b - Not supported
29	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved29	0b - Not supported 1b - Not supported
28 WUF28	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
27 WUF27	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
26 Reserved26	Reserved 0b - Not supported 1b - Not supported
25 Reserved25	Reserved 0b - Not supported 1b - Not supported
24 Reserved24	Reserved 0b - Not supported 1b - Not supported
23 Reserved23	Reserved 0b - Not supported 1b - Not supported
22 Reserved22	Reserved 0b - Not supported 1b - Not supported
21 Reserved21	Reserved 0b - Not supported 1b - Not supported
20 Reserved20	Reserved 0b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Not supported
19 Reserved19	Reserved 0b - Not supported 1b - Not supported
18 Reserved18	Reserved 0b - Not supported 1b - Not supported
17 Reserved17	Reserved 0b - Not supported 1b - Not supported
16 Reserved16	Reserved 0b - Not supported 1b - Not supported
15 WUF15	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
14 WUF14	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
13 WUF13	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
12 WUF12	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
11	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUF11	0b - No 1b - Yes
10 WUF10	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
9 WUF9	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
8 WUF8	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
7 WUF7	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
6 WUF6	Reserved 0b - Not supported 1b - Not supported
5 WUF5	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
4 WUF4	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
3 WUF3	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
2 WUF2	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
1 WUF1	Reserved 0b - Not supported 1b - Not supported
0 WUF0	Wakeup flag for WUU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes

36.6.1.9 Pin Filter (FILT)

Offset

Register	Offset
FILT	30h

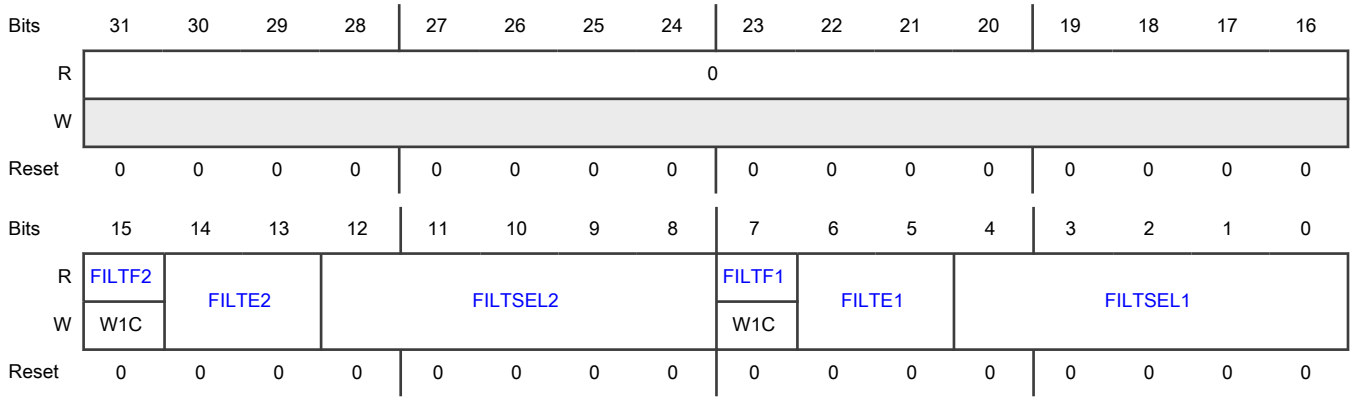
Function

Contains control and status fields to enable and configure the digital filter features for an external wakeup pin.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FILTF2	Filter 2 Flag Indicates that the filtered external pin was a source of exiting a low-leakage power mode. Write '1' to clear this flag. 0b - No 1b - Yes
14-13 FILTE2	Filter 2 Enable Enables the filter for wakeup and configures the edge detection. 00b - Disable filter 01b - Enable filter. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enable filter. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enable filter. When configured as an interrupt/DMA request: Detect on any edge
12-8 FILTSEL2	Filter 2 Pin Select Selects the external pin WUU_Pn to be filtered, where n equals the value programmed into FILTSEL2.
7 FILTF1	Filter 1 Flag Indicates that the filtered external pin was a source of exiting a low-leakage power mode. Write '1' to clear this flag. 0b - No 1b - Yes
6-5	Filter 1 Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FILTE1	Enables the filter for wakeup and configures the edge detection. 00b - Disable filter 01b - Enable filter. When configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level 10b - Enable filter. When configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level 11b - Enable filter. When configured as an interrupt/DMA request: Detect on any edge
4-0 FILTSEL1	Filter 1 Pin Select Selects the external pin WUU_Pn to be filtered, where n equals the value programmed into FILTSEL1.

36.6.1.10 Pin DMA/Trigger Configuration 1 (PDC1)

Offset

Register	Offset
PDC1	38h

Function

Configures the available external wakeup input pins in the range from WUU_P0 to WUU_P15 to generate an interrupt, DMA or a trigger request when detected.

- When configured as an interrupt, the interrupt is asserted when the edge programmed in the PE1 register is detected and remains asserted until the corresponding flag is cleared in the PF register.
- When configured as a DMA request, the request is asserted when the edge programmed in the PE1 register is detected and remains asserted until either the corresponding flag is cleared in the PF register or until the requested DMA transfer is complete.
- When configured as a trigger request, the trigger is asserted when the level programmed in the PE1 register is detected and remains asserted while the input pin is asserted. The corresponding flag in the PF register will not be set.

NOTE

Do not modify the value of a WUPDCn field when its corresponding PMC[WUPMCn] = 1.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WUPDC15		WUPDC14		WUPDC13		WUPDC12		WUPDC11		WUPDC10		WUPDC9		WUPDC8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WUPDC7		WUPDC6		WUPDC5		WUPDC4		WUPDC3		WUPDC2		WUPDC1		WUPDC0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 WUPDC15	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
29-28 WUPDC14	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
27-26 WUPDC13	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
25-24 WUPDC12	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Trigger event 11b - Reserved
23-22 WUPDC11	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
21-20 WUPDC10	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
19-18 WUPDC9	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
17-16 WUPDC8	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
15-14 WUPDC7	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 WUPDC6	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Reserved
11-10 WUPDC5	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
9-8 WUPDC4	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
7-6 WUPDC3	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
5-4 WUPDC2	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
3-2 WUPDC1	Reserved 00b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Not supported 10b - Not supported 11b - Reserved
1-0 WUPDC0	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved

36.6.1.11 Pin DMA/Trigger Configuration 2 (PDC2)

Offset

Register	Offset
PDC2	3Ch

Function

Configures the available external wakeup input pins in the range from WUU_P16 to WUU_P31 to generate an interrupt, DMA or a trigger request when detected.

- When configured as an interrupt, the interrupt is asserted when the edge programmed in the PE2 register is detected and remains asserted until the corresponding flag is cleared in the PF register.
- When configured as a DMA request, the request is asserted when the edge programmed in the PE2 register is detected and remains asserted until either the corresponding flag is cleared in the PF register or until the requested DMA transfer is complete.
- When configured as a trigger request, the trigger is asserted when the level programmed in the PE2 register is detected and remains asserted while the input pin is asserted. The corresponding flag in the PF register will not be set.

NOTE

Do not modify the value of a WUPDCn field when its corresponding PMC[WUPMCn] = 1.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		WUPDC28		WUPDC27		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		0		0		0		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 Reserved31	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
29-28 Reserved30	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
27-26 Reserved29	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
25-24 WUPDC28	Wakeup pin configuration for WUU_Pn Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
23-22	Wakeup pin configuration for WUU_Pn

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUPDC27	Configures an external pin as an interrupt, DMA or trigger wakeup source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
21-20 Reserved26	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
19-18 Reserved25	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
17-16 Reserved24	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
15-14 Reserved23	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
13-12 Reserved22	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
11-10 Reserved21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
9-8 Reserved20	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
7-6 Reserved19	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
5-4 Reserved18	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
3-2 Reserved17	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
1-0 Reserved16	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported

36.6.1.12 Pin Filter DMA/Trigger Configuration (FDC)

Offset

Register	Offset
FDC	48h

Function

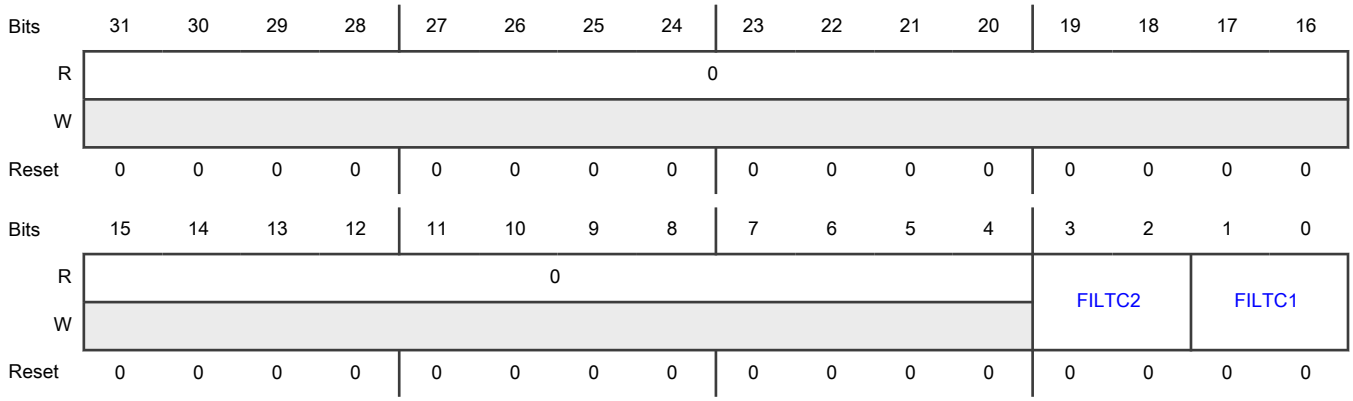
Configures the external pin filters to generate an interrupt, DMA or a trigger request when detected.

- When configured as an interrupt, the interrupt is asserted when the edge programmed in the FILTEn field is detected and remains asserted until the corresponding FILTFn flag is cleared.
- When configured as a DMA request, the request is asserted when the edge programmed in the FILTEn field is detected and remains asserted until either the corresponding FILTFn flag is cleared or until the requested DMA transfer is complete.
- When configured as a trigger request, the trigger is asserted when the level programmed in the FILTEn register is detected and remains asserted while the input pin is asserted. The corresponding FILTFn flag will not be set.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-2: FILTC2 1-0: FILTC1	Filter configuration for FILTn Configures a filter as an interrupt, DMA or trigger wakeup source. 00b - Interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - DMA request 10b - Trigger event 11b - Reserved

36.6.1.13 Pin Mode Configuration (PMC)

Offset

Register	Offset
PMC	50h

Function

Configures the detection logic for the available external wakeup input pins to remain enabled during all power modes, not just during Deep Sleep/Power Down.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	WUPMC28	WUPMC27	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WUPMC15	WUPMC14	WUPMC13	WUPMC12	WUPMC11	WUPMC10	WUPMC9	WUPMC8	WUPMC7	WUPMC6	WUPMC5	WUPMC4	WUPMC3	WUPMC2	WUPMC1	WUPMC0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 Reserved31	Reserved 0b - Not supported 1b - Not supported
30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved30	0b - Not supported 1b - Not supported
29 Reserved29	Reserved 0b - Not supported 1b - Not supported
28 WUPMC28	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
27 WUPMC27	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
26 Reserved26	Reserved 0b - Not supported 1b - Not supported
25 Reserved25	Reserved 0b - Not supported 1b - Not supported
24 Reserved24	Reserved 0b - Not supported 1b - Not supported
23 Reserved23	Reserved 0b - Not supported 1b - Not supported
22 Reserved22	Reserved 0b - Not supported 1b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 Reserved21	Reserved 0b - Not supported 1b - Not supported
20 Reserved20	Reserved 0b - Not supported 1b - Not supported
19 Reserved19	Reserved 0b - Not supported 1b - Not supported
18 Reserved18	Reserved 0b - Not supported 1b - Not supported
17 Reserved17	Reserved 0b - Not supported 1b - Not supported
16 Reserved16	Reserved 0b - Not supported 1b - Not supported
15 WUPMC15	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
14 WUPMC14	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
13 WUPMC13	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
12 WUPMC12	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
11 WUPMC11	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
10 WUPMC10	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
9 WUPMC9	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
8 WUPMC8	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
7	Wakeup pin mode configuration for WUU_Pn

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUPMC7	Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
6 WUPMC6	Reserved 0b - Not supported 1b - Not supported
5 WUPMC5	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
4 WUPMC4	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
3 WUPMC3	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
2 WUPMC2	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.
1 WUPMC1	Reserved 0b - Not supported 1b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 WUPMC0	Wakeup pin mode configuration for WUU_Pn Configures an external wakeup pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. Software can modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers. 1b - Active during all power modes. Software must not modify the corresponding fields within the Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn) registers.

36.6.1.14 Pin Filter Mode Configuration (FMC)

Offset

Register	Offset
FMC	58h

Function

Configures the detection logic for external pin filters to remain enabled during all power modes, not just during Deep Sleep/ Power Down.

NOTE

This register is reset on Chip Reset not Power Down and by reset types that trigger Chip Reset not Power Down. It is unaffected by reset types that do not trigger Chip Reset not Power Down.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														FILTM2	FILTM1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 FILTMn	Filter Mode for FILTn Configures an external wakeup pin filter to provide active detection during all power modes. 0b - Active only during Deep Sleep/Power Down mode 1b - Active during all power modes

Chapter 37
Watchdog Timer (WDOG)

37.1 Chip-specific WDOG information

Table 232. Reference links to related information

Topic	Related module	Reference
Full description	WDOG	WDOG
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

37.1.1 Module instances

This device has two instances of the WDOG module, WDOG0 and WDOG1.

NOTE

On reset, the WDOG0 is enabled and the WDOG1 is disable by default.

37.1.2 CS[CLK] fields

The CS[CLK] configures the clock source that feeds the watchdog counter as below.

- 00b - SLOW_CLK
- 01b - 32K_CLK
- 10b - FRO-6M
- 11b - OSC-RF

37.2 Overview

The Watchdog Timer (WDOG) module is an independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDOG module is not serviced (refreshed) within a certain period, it resets the MCU.

37.2.1 Block diagram

The following figure shows a block diagram of the WDOG module.

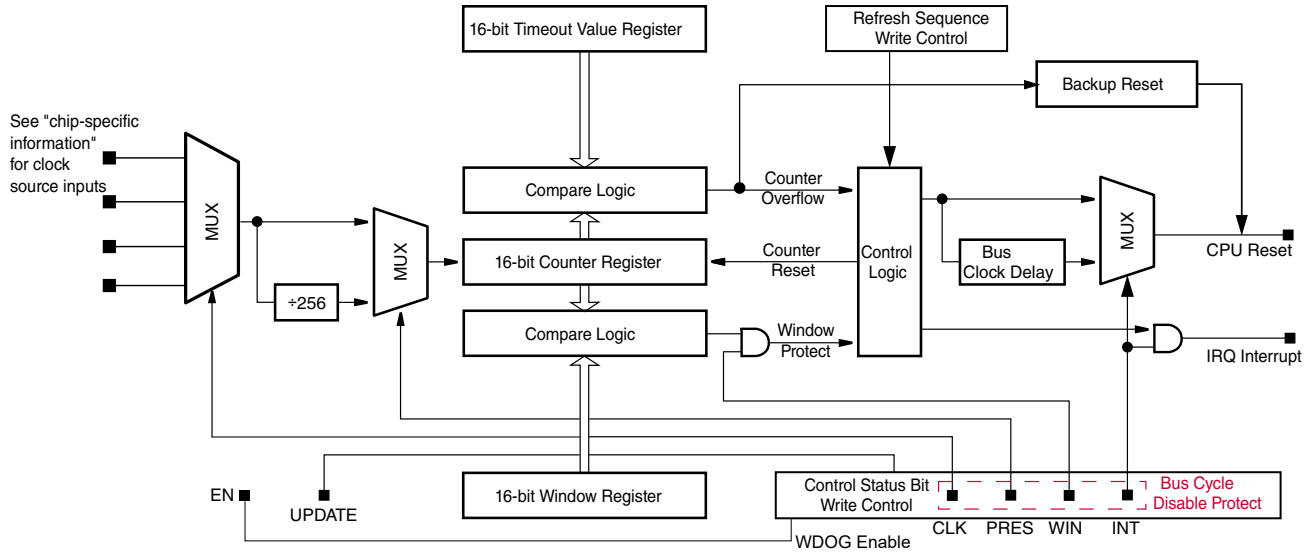


Figure 94. WDOG block diagram

37.2.2 Features

Features of the WDOG module include:

- Configurable clock source inputs independent from the bus clock
- Programmable timeout period
  - Programmable 16-bit timeout value
  - Optional fixed 256 clock prescaler when longer timeout periods are needed
- Robust write sequence for counter refresh
  - Refresh sequence of writing the watchdog counter register (CNT)
- Window mode option for the refresh mechanism
  - Programmable 16-bit window value
  - Provides robust check that program flow is faster than expected
  - Early refresh attempts trigger a reset
- Optional timeout interrupt to allow post-processing diagnostics
  - Interrupt request to CPU with interrupt vector for an interrupt service routine (ISR)
  - Forced reset occurs 128 bus clocks after the interrupt vector fetch
- Configuration bits are write-once-after-reset to ensure watchdog configuration cannot be mistakenly altered
- Robust write sequence for unlocking write-once configuration bits
  - Unlock sequence of writing the watchdog counter register (CNT), for allowing updates to write-once configuration bits
  - Software must make updates within 128 bus clocks after unlocking and before WDOG closing unlock window

37.3 Functional description

The WDOG module provides a fail safe mechanism to ensure the system can be reset to a known state of operation in case of system failure, such as the CPU clock stopping or there being a run away condition in the software code. The watchdog counter

runs continuously off a selectable clock source and expects to be serviced (refreshed) periodically. If it is not, it generates a reset triggering event.

37.3.1 Watchdog refresh mechanism

The watchdog resets the MCU if the watchdog counter is not refreshed. A robust refresh mechanism makes it very unlikely that the watchdog can be refreshed by runaway code.

To refresh the watchdog counter, software must execute a refresh write sequence before the timeout period expires. In addition, if window mode is used, software must not start the refresh sequence until after the time value set in the WIN register. See the following figure.

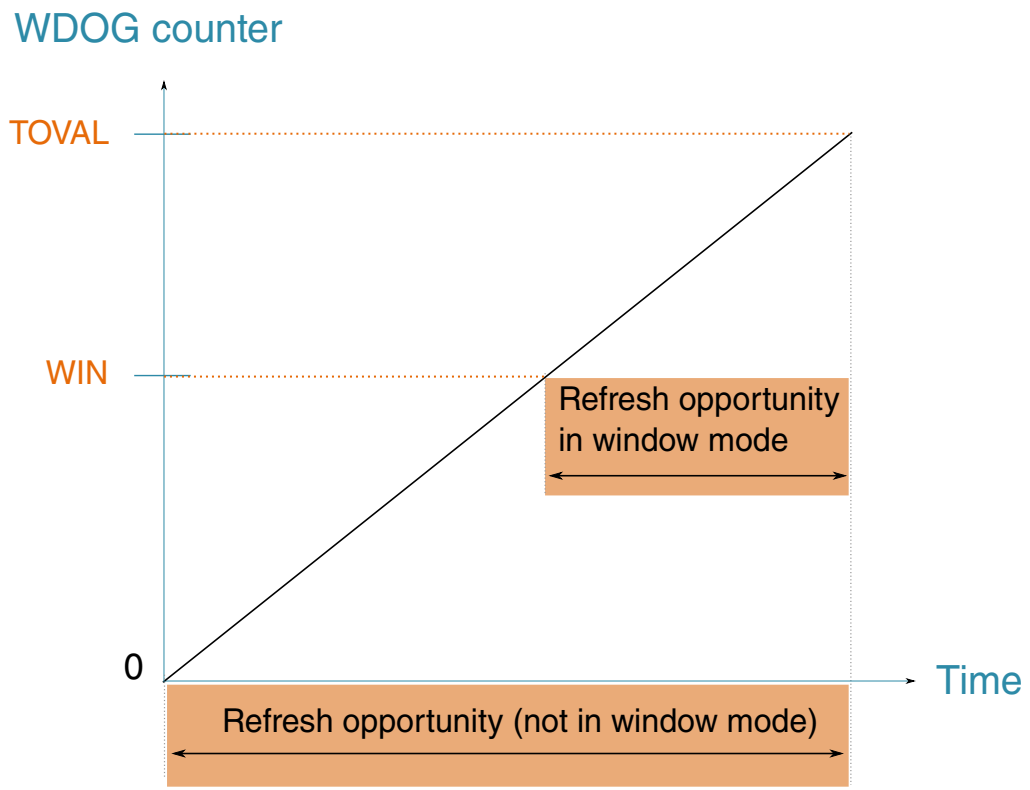


Figure 95. Refresh opportunity for the Watchdog counter

37.3.1.1 Window mode

Software finishing its main control loop faster than expected could be an indication of a problem. Depending on the requirements of the application, the WDOG can be programmed to force a reset when refresh attempts are early.

When Window mode is enabled, the watchdog must be refreshed after the counter has reached a minimum expected time value; otherwise, the watchdog resets the MCU. The minimum expected time value is specified in the WIN register. Setting CS[WIN] enables Window mode.

37.3.1.2 Refreshing the Watchdog

The refresh write sequence can be

- either two 16-bit writes (0xA602, 0xB480) or four 8-bit writes (0xA6, 0x02, 0xB4, 0x80) if WDOG_CS[CMD32EN] is 0;
- one 32-bit write (0xB480_A602) if WDOG_CS[CMD32EN] is 1.

to the CNT register. Both methods must occur before the WDG timeout; otherwise, the watchdog resets the MCU.

NOTE

Before starting the refresh sequence, disable the global interrupts. Otherwise, an interrupt could effectively invalidate the refresh sequence, if the interrupt occurs before the refresh writes finish. After the sequence finishes, restore the global interrupt control state.

For the example code snippets, see Application Information.

37.3.2 Configuring the Watchdog

37.3.2.1 Configuring the Watchdog Once

All watchdog control bits, timeout value, and window value are write-once after reset. This means that after a write has occurred they cannot be changed unless a reset occurs. This is guaranteed by the user configuring the window and timeout value first, followed by the other control bits, and ensuring that CS[UPDATE] is also set to 0.

This provides a robust mechanism to configure the watchdog and ensure that a runaway condition cannot mistakenly disable or modify the watchdog configuration after configured.

The new configuration takes effect only after all registers except CNT are written after reset. Otherwise, the WDOG uses the reset values by default. If window mode is not used (CS[WIN] is 0), writing to WIN is not required to make the new configuration take effect.

37.3.2.2 Reconfiguring the Watchdog

In some cases (like when supporting a bootloader function), you may want to reconfigure or disable the watchdog, *without forcing a reset first*.

- By setting CS[UPDATE] to 1 on the initial configuration of the watchdog after a reset, you can reconfigure the watchdog at any time by executing an unlock sequence.
- Conversely, if CS[UPDATE] remains 0, the only way to reconfigure the watchdog is by initiating a reset.

The unlock sequence is similar to the refresh sequence but uses different values.

37.3.2.2.1 Unlocking the Watchdog

The unlock sequence is

- either two 16-bit writes (0xC520, 0xD928) or four 8-bit writes (0xC5, 0x20, 0xD9, 0x28) if WDOG_CS[CMD32EN] is 0;
- one 32-bit write (0xD928_C520) if WDOG_CS[CMD32EN] is 1.

to the CNT register at any time after the watchdog has been configured. Improper unlock sequence causes the WDOG to reset. On completing the unlock sequence, the user must reconfigure the watchdog within 128 bus clocks; otherwise, the watchdog closes the unlock window.

NOTE

Due to the 128 bus clocks requirement for reconfiguring the watchdog, some delays must be inserted before executing STOP or WAIT instructions after reconfiguring the watchdog. This ensures that the watchdog's new configuration takes effect before the MCU enters low power mode. Otherwise, the MCU may not be waken up from low power mode.

37.3.4 Fast testing of the watchdog

Before executing application code in safety critical applications, users are required to test that the watchdog works as expected and resets the MCU. Testing every bit of a 16-bit counter by letting it run to the overflow value takes a relatively long time (64 k clocks).

To help minimize the startup delay for application code after reset, the watchdog has a feature to test the watchdog more quickly by splitting the counter into its constituent byte-wide stages. The low and high bytes are run independently and tested for timeout against the corresponding byte of the timeout value register. (For complete coverage when testing the high byte of the counter,

the test feature feeds the input clock via the 8th bit of the low byte, thus ensuring that the overflow connection from the low byte to the high byte is tested.)

Using this test feature reduces the test time to 512 clocks (not including overhead, such as user configuration and reset vector fetches). To further speed testing, use a faster clock (such as the bus clock) for the counter reference.

On a power-on reset, the POR bit in the system reset register is set, indicating the user should perform the WDOG fast test.

37.3.4.1 Testing each byte of the counter

The test procedure follows these steps:

1. Program the preferred watchdog timeout value in the TOVAL register during the watchdog configuration time period.
2. Select a byte of the counter to test using the CS[TST] = 10b for the low byte; CS[TST] = 11b for the high byte.
3. Wait for the watchdog to timeout. Optionally, in the idle loop, increment RAM locations as a parallel software counter for later comparison. Because the RAM is not affected by a watchdog reset, the timeout period of the watchdog counter can be compared with the software counter to verify the timeout period has occurred as expected.
4. The watchdog counter times out and forces a reset.
5. Confirm the WDOG flag in the system reset register is set, indicating that the watchdog caused the reset. (The POR flag remains clear.)
6. Confirm that CS[TST] shows a test (10b or 11b) was performed.

If confirmed, the count and compare functions work for the selected byte. Repeat the procedure, selecting the other byte in step 2.

NOTE

CS[TST] is cleared by a POR only and not affected by other resets.

37.3.4.2 Entering user mode

After successfully testing the low and high bytes of the watchdog counter, the user can configure CS[TST] to 01b to indicate the watchdog is ready for use in application user mode. Thus if a reset occurs again, software can recognize the reset trigger as a real watchdog reset caused by runaway or faulty application code.

As an ongoing test when using the default clock source, software can periodically read the CNT register to ensure the counter is being incremented.

37.3.5 Clocking

The watchdog counter has the clock source options selected by programming CS[CLK]. See the "chip-specific information" section for available clock inputs and the default option for this device.

The options allow software to select a clock source independent of the bus clock for applications that need to meet more robust safety requirements. Using a clock source other than the bus clock ensures that the watchdog counter continues to run if the bus clock is somehow halted; see Backup reset.

NOTE

The *default* clock source of WDOG should be enabled after its functional reset is deasserted, for the WDOG module to function properly.

An optional fixed prescaler for all clock sources allows for longer timeout periods. When CS[PRES] is set, the clock source is prescaled by 256 before clocking the watchdog counter.

The following table summarizes the different watchdog timeout periods which could be available, as an example.

Table 233. Watchdog timeout availability

Reference clock	Prescaler	Watchdog time-out availability
REF_CLK	Pass through	1×RCP to 65535×RCP [1]
	Enable	256×RCP to 16776960×RCP

1. RCP means Reference Clock Period.

NOTE

When the programmer switches clock sources during reconfiguration, the watchdog hardware holds the counter at zero for 2.5 periods of the previous clock source and 2.5 periods of the new clock source after the configuration time period (128 bus clocks) ends. This delay ensures a smooth transition before restarting the counter with the new configuration.

37.3.6 Backup reset

The backup reset function is a safeguard feature that independently generates a reset in case the main WDOG logic loses its clock (the bus clock) and can no longer monitor the counter. If the watchdog counter overflows twice in succession (without an intervening reset), the backup reset function takes effect and generates a reset.

NOTE

A clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the backup reset function is not available.

Backup reset becomes valid when interrupt is enabled and the watchdog clock is from non bus clock. If interrupt is enabled, once the bus clock is cut off before exiting interrupt routine, the normal watchdog reset will be blocked. Under this case, the second overflow will cause backup reset directly.

37.3.7 Reset and interrupts

An interrupt request can be generated to delay resets.

- **When interrupts are enabled (CS[INT] = 1):** After a reset-triggering event (like a counter timeout or invalid refresh attempt), the watchdog first generates an interrupt request. Next, the watchdog delays 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event) before forcing a reset, to allow the interrupt service routine (ISR) to perform tasks (like analyzing the stack to debug code).
- **When interrupts are disabled (CS[INT] = 0):** the watchdog does not delay before forcing a reset.

37.4 External signals

This module has no external signals.

37.5 Initialization

See Configuring the Watchdog Once.

37.6 Application Information

To disable or reconfigure the watchdog, it should be completed before the first watchdog timeout. Disabling or reconfiguring the watchdog should occur at the very beginning of the software code, e.g. the beginning of the startup or main function.

NOTE

When the watchdog is configured by user, it needs at least 2.5 periods of watchdog clock to take effect. This means the interval between the two configurations by user must be larger than 2.5 clocks.

NOTE

When the chip starts up from BOOT ROM and then jumps to flash, the watchdog should be disabled at the beginning of bootloader and enabled after it exits. To reconfigure the watchdog using the flash program, it also needs the interval of at least 2.5 watchdog clocks after the bootloader exits.

To disable or reconfigure the watchdog without forcing a reset, WDOG_CS[UPDATE] bit must be set during the initial configuration of the WDOG module. The unlock sequence can then be used at any time within the timeout limit to reconfigure the watchdog.

Disable Watchdog

To disable the watchdog, first do the unlock sequence, then unset the WDOG_CS[EN] bit. The code snippet below shows an example for 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
WDOG_CS &= ~WDOG_CS_EN_MASK; //disable watchdog
EnableInterrupts; //enable global interrupt
```

Disable Watchdog after Reset

All of the watchdog registers are unlocked by reset. Therefore, an unlock sequence is unnecessary but it needs to write all of watchdog registers to make the new configuration take effect. The code snippet below shows an example of disabling watchdog after reset.

```
DisableInterrupts; // disable global interrupt
WDOG_CS &= ~WDOG_CS_EN_MASK; // disable watchdog
WDOG_TOVAL= 0xFFFF;
while(WDOG_CS[ULK]);  // waiting for lock
while(~WDOG_CS[RCS]); // waiting for new configuration to take effect
EnableInterrupts; // enable global interrupt
```

Configure Watchdog

The watchdog can be configured once by setting the WDOG_CS[UPDATE]=0. After that, the watchdog cannot be reconfigured until a reset. To reconfigure without forcing a reset, set WDOG_CS[UPDATE]=1 when configuring the watchdog. The following example code shows how to configure the watchdog without window mode, clock source as LPO, interrupt enabled and timeout value to 256 clocks. The code snippet below shows an example for 32-bit write.

Configure once

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(0);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

Configure for reconfigurable

```
DisableInterrupts; //disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(1);
```

```
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

Refreshing the Watchdog

To refresh the watchdog and reset the watchdog counter to zero, a refresh sequence is required. The code snippet below shows an example for 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xB480A602; // refresh watchdog
EnableInterrupts; // enable global interrupt
```

37.7 Memory map and register definition

37.7.1 WDOG register descriptions

37.7.1.1 WDOG memory map

WDOG0 base address: 4001_A000h

WDOG1 base address: 4001_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control and Status Register (CS)	32	RW	See section
4h	Counter Register (CNT)	32	RW	0000_0000h
8h	Timeout Value Register (TOVAL)	32	RW	0000_FFFFh
Ch	Window Register (WIN)	32	RW	0000_0000h

37.7.1.2 Control and Status Register (CS)

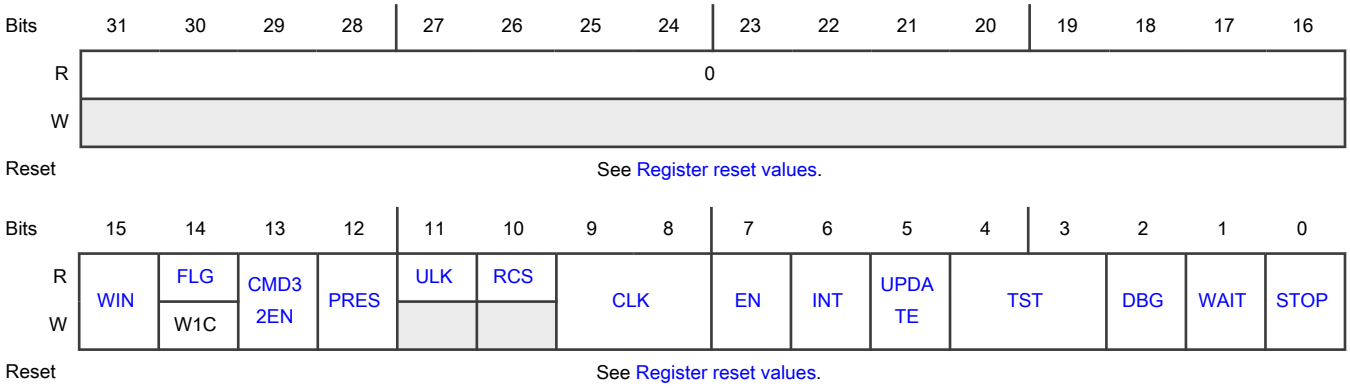
Offset

Register	Offset
CS	0h

Function

Implements control and indicates status of the module.

Diagram



Register reset values

Register	Reset value
CS	WDOG0: 0000_3A80h WDOG1: 0000_3A00h

Fields

Field	Function
31-16 —	Reserved
15 WIN	Watchdog Window This write-once bit enables window mode. See Window mode for details. 0b - Disables 1b - Enables
14 FLG	Watchdog Interrupt Flag This bit is an interrupt indicator when CS[INT] is set. Write 1 to clear it. 0b - No interrupt occurred. 1b - An interrupt occurred.
13 CMD32EN	Enables or disables WDOG support for 32-bit (otherwise 16-bit or 8-bit) refresh/unlock command write words This is write-once field, and the user needs to unlock WDOG after writing this field for reconfiguration. 0b - Disables support for 32-bit refresh/unlock command write words. Only 16-bit or 8-bit is supported. 1b - Enables support for 32-bit refresh/unlock command write words. 16-bit or 8-bit is NOT supported.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 PRES	Watchdog prescaler This write-once bit enables a fixed 256 pre-scaling of watchdog counter reference clock. (The block diagram shows this clock divider option.) 0b - 256 prescaler disabled. 1b - 256 prescaler enabled.
11 ULK	Unlock status Indicates whether WDOG is unlocked or not. 0b - Locked 1b - Unlocked
10 RCS	Reconfiguration Success Indicates whether the reconfiguration is successful or not. Default reset value is 0. This bit is set when new configuration takes effect, and is cleared by successful unlock command. 0b - Reconfiguring WDOG. 1b - Reconfiguration is successful.
9-8 CLK	Watchdog Clock This write-once field indicates the clock source that feeds the watchdog counter. See the "chip-specific information" section for clock source inputs.
7 EN	Watchdog Enable This write-once bit enables the watchdog counter to start counting. 0b - Watchdog disabled. 1b - Watchdog enabled.
6 INT	Watchdog Interrupt This write-once bit configures the watchdog to immediately generate an interrupt request upon a reset-triggering event (timeout or illegal write to the watchdog), before forcing a reset. After the interrupt vector fetch (which comes after the reset-triggering event), the reset occurs after a delay of 128 bus clocks. 0b - Watchdog interrupts are disabled. Watchdog resets are not delayed. 1b - Watchdog interrupts are enabled. Watchdog resets are delayed by 128 bus clocks from the interrupt vector fetch.
5 UPDATE	Allow updates This write-once bit allows software to reconfigure the watchdog without a reset. 0b - Updates not allowed. After the initial configuration, the watchdog cannot be later modified without forcing a reset. 1b - Updates allowed. Software can modify the watchdog configuration registers within 128 bus clocks after performing the unlock write sequence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-3 TST	Watchdog Test Enables the fast test mode. The test mode allows software to exercise all bits of the counter to demonstrate that the watchdog is functioning properly. See Fast testing of the watchdog. **NOTE** This write-once field is cleared (0:0) on POR only. Any other reset does not affect the value of this field. 00b - Watchdog test mode disabled. 01b - Watchdog user mode enabled (test mode disabled). After testing the watchdog, software should use this setting to indicate the watchdog is functioning normally in user mode. 10b - Watchdog test mode enabled, only the low byte is used. CNT[CNTLOW] is compared with TOVAL[TOVALLOW]. 11b - Watchdog test mode enabled, only the high byte is used. CNT[CNTHIGH] is compared with TOVAL[TOVALHIGH].
2 DBG	Debug Enable This write-once bit enables the watchdog to operate when the chip is in debug mode. 0b - Disables 1b - Enables
1 WAIT	Wait Enable This write-once bit enables the watchdog to operate when the chip is in wait mode. 0b - Disables 1b - Enables
0 STOP	Stop Enable This write-once bit enables the watchdog to operate when the chip is in stop mode. 0b - Disables 1b - Enables

37.7.1.3 Counter Register (CNT)

Offset

Register	Offset
CNT	4h

Function

Provides access to the value of the free-running watchdog counter. Software can read the counter register at any time.

Software cannot write directly to the watchdog counter; however, two write sequences to the register have special functions:

1. The *refresh sequence* resets the watchdog counter to 0x0000. See the "Refreshing the Watchdog" section in Application Information.
2. The *unlock sequence* allows the watchdog to be reconfigured without forcing a reset (when CS[UPDATE] = 1). See the "Configure for reconfigurable" section in Application Information.

NOTE

All other writes to this register are illegal and force a reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNTHIGH								CNTLOW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 CNTHIGH	High byte of the Watchdog Counter
7-0 CNTLOW	Low byte of the Watchdog Counter

37.7.1.4 Timeout Value Register (TOVAL)

Offset

Register	Offset
TOVAL	8h

Function

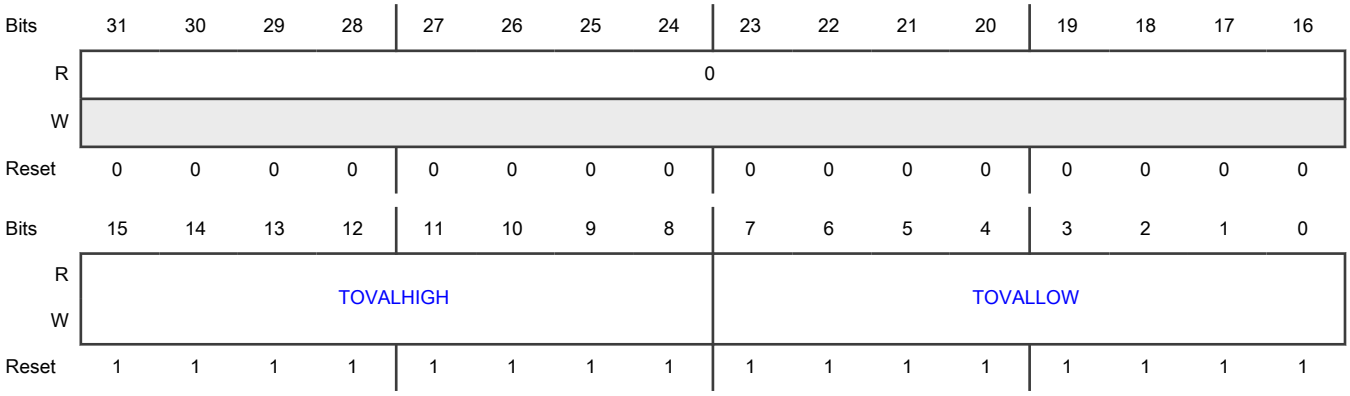
Contains the 16-bit value used to set the timeout period of the watchdog.

The watchdog counter (CNT) is continuously compared with the timeout value (TOVAL). If the counter reaches the timeout value, the watchdog forces a reset triggering event.

NOTE

Do not write 0 to the TOVAL register (if CS[TST]=11b, then TOVALHIGH cannot be written as 0; if CS[TST]=10b, then TOVALLOW cannot be 0); otherwise, the watchdog always generates a reset.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 TOVALHIGH	High byte of the timeout value
7-0 TOVALLOW	Low byte of the timeout value

37.7.1.5 Window Register (WIN)

Offset

Register	Offset
WIN	Ch

Function

When the window mode is enabled (CS[WIN] is set), this WIN register determines the earliest time that a refresh sequence is considered valid. See Watchdog refresh mechanism.

NOTE

The WIN register value must be less than the TOVAL register value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WINHIGH								WINLOW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 WINHIGH	High byte of Watchdog Window
7-0 WINLOW	Low byte of Watchdog Window

Chapter 38
EdgeLock Secure Enclave (ELE)

38.1 Introduction

While the emerging application space known as the Internet of Things (IoT) provides an incredible market and business opportunity, it also brings a variety of technical challenges for the microcontrollers located as edge nodes in massively-connected networks. Clearly, one of the most important technical issues remains the definition of an appropriate level of device security for the MCU devices - an application area where simple hardware acceleration of cryptographic functions is no longer an acceptable solution.

Indeed, the IoT market represents a perfect storm where the combined legacy security models for static microcontroller configurations and networked devices are no longer appropriate; instead a new paradigm for MCU security is emerging and NXP is using its industry-leading position as the preeminent supplier of secure solutions to lead the way. An EdgeLock secure enclave, or simply EdgeLock enclave (ELE) has been defined and implemented that significantly raises the level of security provided by a traditional connected MCU device.

The EdgeLock secure enclave is an independent black box processing element providing essential cryptographic services to the host system within a single microcontroller specifically in two areas: secret key storage and management, plus execution of symmetric and public key cryptographic services, including secure hashing, based on requests from the host. By design, it is intended to provide a secure environment within an otherwise non-secure device in which applications can execute secure cryptographic services.

38.1.1 Block Diagram

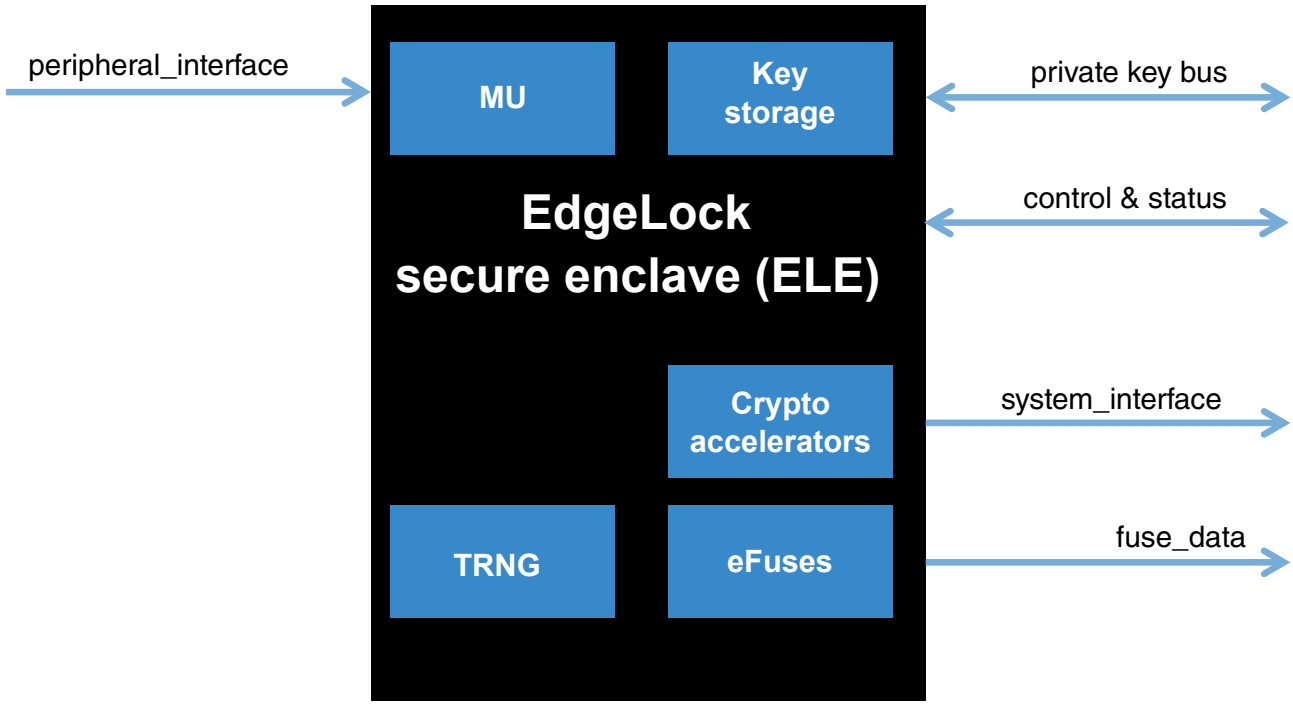


Figure 96. Block Diagram

From EdgeLock's perspective, there are two major interfaces:

```
   peripheral_interface              // host-to-peripheral interface (APB) to access the
messaging unit
```

```
    system_interface             // ELE-to-system interface (AHB) to access host memories
```

Additionally, there are three “minor” interfaces:

```
  control & status        // miscellaneous i/o connections for control & status
  private_key_bus         // secure output bus to distribute keys to host IP modules
  fuse_data               // output bus to distribute fuse values to host
```

38.1.2 Features

The key EdgeLock enclave architecture features include an independent processing element with optimized crypto IP for acceleration focused on two security functions:

1. Secret key generation, storage and management
   - Hardware key derivation from non-volatile memory within the SoC
   - Implements bus and memory obfuscation for cryptographic protection
2. Secure crypto services during secure boot and normal runtime
   - Simple peripheral interface with the host processor to minimize attack surface
     - Host application makes function calls to EdgeLock enclave via a messaging unit
     - Mailbox messages used to pass crypto services commands, pointers and data
     - EdgeLock enclave implements a run to completion execution model for crypto tasks
   - EdgeLock enclave has access to host's internal and external memories via a system interface
   - Operations and services controlled by CryptoLib software

38.1.2.1 Symmetric algorithms

“Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key (or, less commonly, in which their keys are different, but related in an easily computable way). This was the only kind of encryption publicly known until June 1976.

Symmetric key ciphers are implemented as either block ciphers or stream ciphers. A block cipher enciphers input in blocks of plaintext as opposed to individual characters, the input form used by a stream cipher.”[1]

[1] Taken from *https://en.wikipedia.org/wiki/Cryptography*

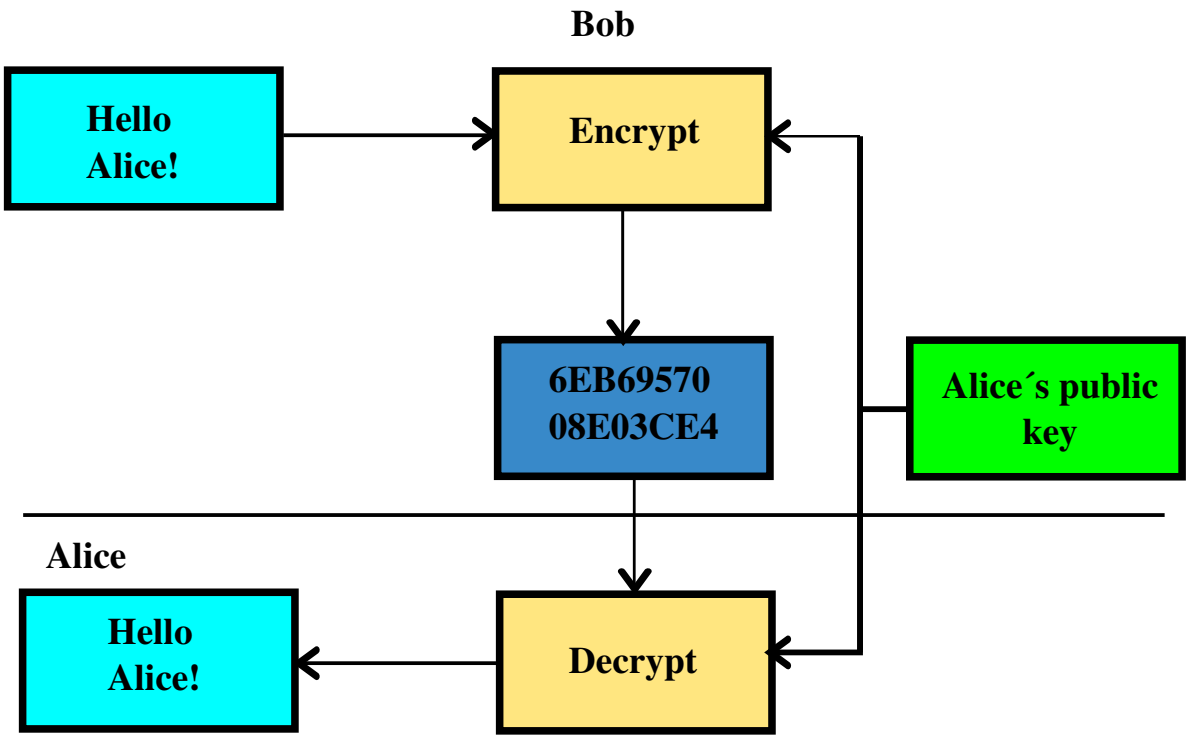


Figure 97. Symmetric-key cryptography, where a single key is used for encryption and decryption [1]

Advanced Encryption Standard (AES) - The AES algorithm is a symmetric block cipher. The algorithm processes 128-bit data blocks using the cipher key. The cipher key can be 128-, 192-, or 256-bits in length. These different versions of the AES algorithm are known as AES-128, AES-192, and AES-256. The amount of processing performed by the AES algorithm is determined by the length of the cipher key and affects the number of rounds used in the calculations.

The AES algorithm was standardized by the U.S. National Institute of Standards and Technology (NIST) in 2001. For more information on the details of the AES algorithm, please refer to Federal Information Processing Standard Publication 197 (FIPS 197) that can be downloaded from the NIST website at https://www.nist.gov.

EdgeLock enclave natively supports the following AES algorithms along with a variety of block cipher modes:

- Advanced Encryption Standard 128 (AES-128)
- Advanced Encryption Standard 192 (AES-192)
- Advanced Encryption Standard 256 (AES-256)

38.1.2.2 Asymmetric algorithms

“Symmetric-key cryptosystems use the same key for encryption and decryption of a message, although a message or group of messages can have a different key than others. A significant disadvantage of symmetric ciphers is the key management necessary to use them securely. Each distinct pair of communicating parties must, ideally, share a different key, and perhaps for each ciphertext exchanged as well. The number of keys required increases as the square of the number of network members, which very quickly requires complex key management schemes to keep them all consistent and secret.

A public key system is so constructed that calculation of one key (the 'private key') is computationally infeasible from the other (the 'public key'), even though they are necessarily related. Instead, both keys are generated secretly, as an interrelated pair. The historian David Kahn described public-key cryptography as "the most revolutionary new concept in the field since polyalphabetic substitution emerged in the Renaissance".

In public-key cryptosystems, the public key may be freely distributed, while its paired private key must remain secret. In a public-key encryption system, the public key is used for encryption, while the private or secret key is used for decryption. While Diffie and Hellman could not find such a system, they showed that public-key cryptography was indeed possible by presenting the Diffie–Hellman key exchange protocol, a solution that is now widely used in secure communications to allow two parties to secretly agree on a shared encryption key."[1]

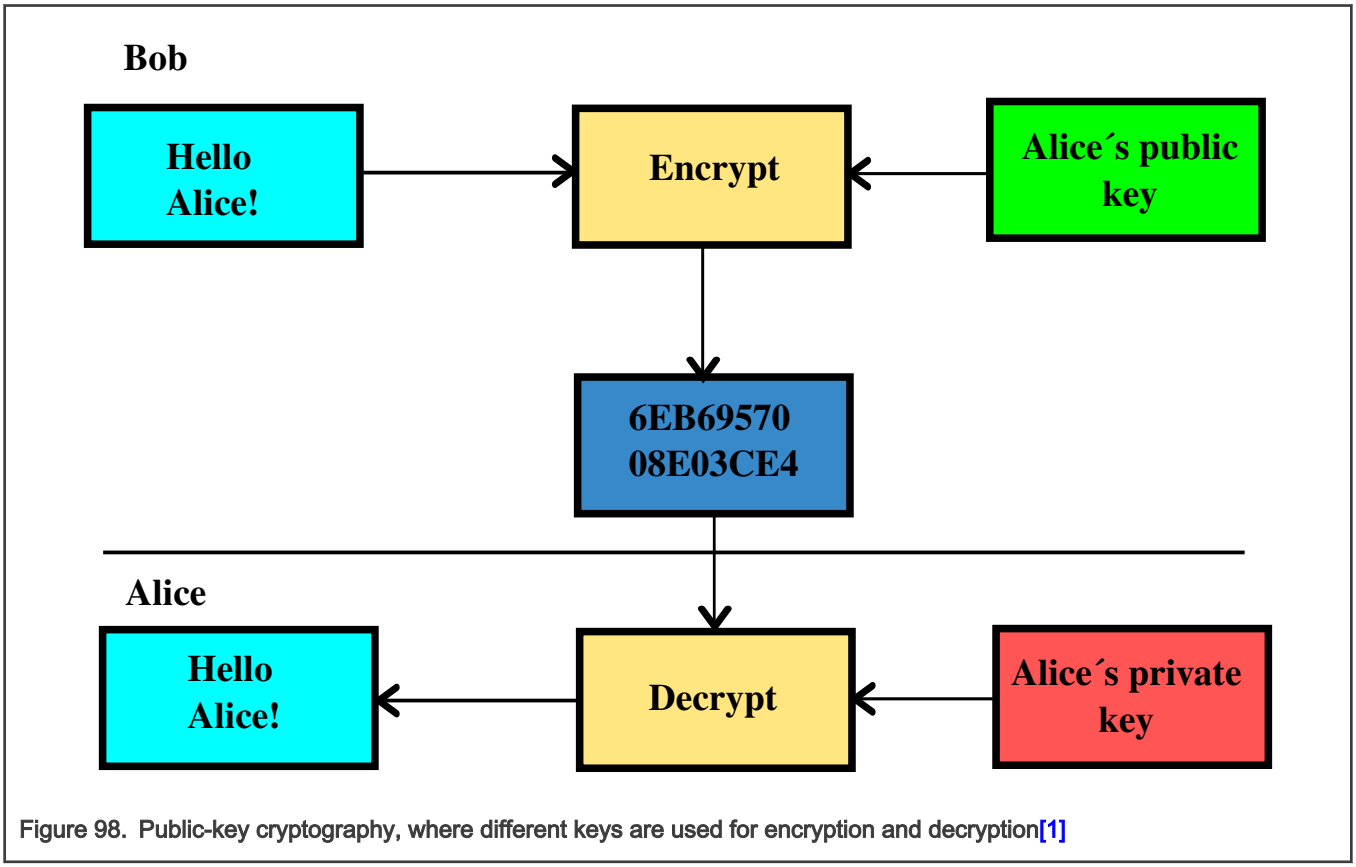


Figure 98. Public-key cryptography, where different keys are used for encryption and decryption[1]

Elliptic-Curve Cryptography (ECC) - The ECC algorithms are based on algebraic operations on elliptic curves over finite fields. The algorithms can be used for digital signatures, key agreement, plus encryption and can operate over prime fields or binary fields with varying cipher key lengths.

The ECC curves were originally standardized by the U.S. National Institute of Standards and Technology (NIST) in 2000. For more information on the details of the ECC curves, please refer to Federal Information Processing Standard Publication 186-4 (FIPS 186-4) that can be downloaded from the NIST website at https://www.nist.gov.

EdgeLock enclave natively support the following ECC curves:

- Elliptic Prime Curves (P-192, P-224, P-256, P-384, P-521)
- Elliptic Prime Curve 25519

38.1.2.3 Hash functions

"Cryptographic hash functions are a third type of cryptographic algorithm. They take a message of any length as input, and output a short, fixed length hash, which can be used in (for example) a digital signature. For good hash functions, an attacker cannot find two messages that produce the same hash.

Unlike block and stream ciphers that are invertible, cryptographic hash functions produce a hashed output that cannot be used to retrieve the original input data. Cryptographic hash functions are used to verify the authenticity of data retrieved from an untrusted source or to add a layer of security.

Message authentication codes (MACs) are much like cryptographic hash functions, except that a secret key can be used to authenticate the hash value upon receipt; this additional complication blocks an attack scheme against bare digest algorithms, and so has been thought worth the effort."[1]

Secure Hash Algorithm 2 (SHA-2) - The SHA-2 algorithms are cryptographic hash functions. The algorithms process the input information and generate a hash value. This hash value is known as a message digest. The hash value can be 224-, 256-, 384-, or 512-bits in length. These hashes are known as SHA-224, SHA-256, SHA-384, and SHA-512. The message digest can be used to verify the integrity of a set of information.

The SHA-2 algorithms were standardized by the U.S. National Institute of Standards and Technology (NIST) in 2001.

For more information on the details of the SHA-2 algorithms, please refer to Federal Information Processing Standard Publication 180-4 (FIPS 180-4) that can be downloaded from the NIST website at https://www.nist.gov.

EdgeLock enclave hardware supports:

- Secure Hash Algorithm 256 (SHA-256)

38.1.2.4 Nonce

A nonce is a random or pseudo-random number that is used only once in a cryptographic calculation. Nonces are important for initialization vectors and cryptographic hash functions.

EdgeLock Enclave natively supports:

- A True Random Number Generator (TRNG) which is a source of entropy that can be used to generate random numbers for use in cryptographic algorithms.

38.2 Messaging Unit (ELE_MU)

38.2.1 Overview

The Messaging Unit module enables two processing elements within the SoC to communicate and coordinate by passing messages (e.g., data, status and control) through its interfaces. The EdgeLock Enclave Messaging Unit (ELE_MU) is specifically targeted for use in Edgelock enclave. The ELE_MU also provides the ability for one processing element to signal the other processing element using interrupts based on data transmission status.

This module's design is based on synchronous clock domain crossings, that is, each side of the ELE_MU operates in a phase-aligned clock domain. The two clock domains may be operating at integer multipliers or dividers, but they are phase aligned so there is no need for logic to handle asynchronous clock domain crossings. The ELE_MU accomplishes synchronization using two sets of matching registers (Processor A-facing, Processor B-facing).

In the Edgelock enclave application, the ELE_MU "A-side" port corresponds to the SoC host processor while the ELE_MU "B-side" port is the security subsystem.

Throughout this chapter, "ELE_MUA" is used to denote hardware resources associated with the "A-side" port, and "ELE_MUB" denotes those associated with the "B-side" port. This manual only documents the MU "A-side" programming model as it is visible and accessible to the host processor, while the MU "B-side" programming model is restricted to the internal operation of the EdgeLock enclave and not visible nor accessible to the host processor.

Finally, the ELE_MUA port implements a secure semaphore (SEMA4) function to provide hardware enforcement of the Edgelock enclave serialization mechanism in multi-core/multi-host device configurations.

38.2.1.1 Block Diagram

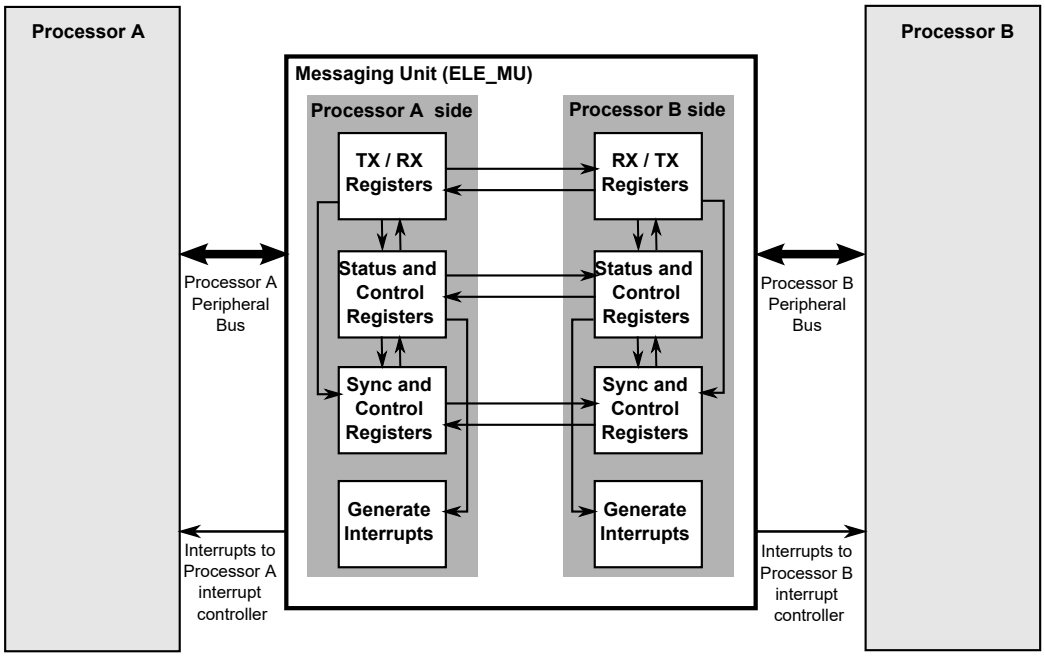


Figure 99. ELE_MU Block Diagram

38.2.1.2 Features

The ELE_MU includes the following features:

1. Memory-Mapped Registers
   - The ELE_MU is connected with separate peripheral buses on Processor A-side and Processor B-side.
2. Synchronous Message Transfers between Processing Elements
   - For sending data or messages between the processing elements, ELE_MUA provides 16 transmit registers and 2 receive registers, while ELE_MUB provides 2 transmit registers and 16 receive registers.
   - The transfer of data messages between processing elements uses transmit empty and receive full flags provided on both sides of the ELE_MU.
   - The update of these transmit and receive flags is accomplished using a fully synchronous mechanism. Upon a register read or write, the corresponding transmit or receive flags are updated at the completion of the register access cycle.
3. Secure Semaphore
   - The ELE_MUA port implements a semaphore function to support hardware enforcement of Edgelock enclave (ELE) serialization mechanisms in certain device configurations. This logic also provides read-only information on the ELE status.

38.2.2 Functional Description

The Messaging Unit (ELE_MU) enables two processing elements (Processor A and Processor B) to communicate with each other, by passing message/data information to each other, and by enabling one side to wake up the other side using data transmission interrupts.

The messaging, control, and status registers of the Processor A and Processor B sides for the ELE_MU are mapped to the processing element A and processing element B address spaces using a standard peripheral bridge memory slot. The peripheral data bus is 32 bits wide inside the ELE_MU module.

Most of the messaging mechanisms are symmetric. They are duplicated and are available on both the A-side and the B-side. The messaging mechanisms are:

- Sixteen 32-bit ELE_MUA transmit registers, which are each reflected in sixteen read-only ELE_MUB receive registers. These registers can be used to transfer 32-bit word messages or pass information for messages written to the shared memory (number of words, address pointers, and message type code).
- Two 32-bit ELE_MUB transmit registers, which are each reflected in two read-only ELE_MUA receive registers. These registers can be used to transfer 32-bit word messages or pass information for messages written to the shared memory (number of words, address pointers, and message type code).
- A write to a transmit register clears the corresponding “transmitter empty” bit in the Status Register on the transmitter side, and sets the appropriate “receiver full” bit in the Status Register on the receiver side. The setting of the bit at the receiver side can optionally trigger an interrupt at the receiver side (maskable receive interrupt).
- A read of one of the receive registers clears the corresponding “receiver full” bit in the Status Register at the receiver side, and sets the appropriate “transmitter empty” bit in the Status Register on the transmitter side. The setting of the “transmitter empty” bit can optionally trigger an interrupt at the transmitter side (maskable transmit interrupt).

A write to a transmit register signals the receiver side that data is ready for retrieval.

- Writing to the transmit register again without verifying that the data was retrieved is prohibited, because the transmitter side has no way of knowing the exact time that the receiver retrieves the data.
- Before attempting to write the transmit register again, the transmitter side waits for a “Transmitter Empty” interrupt, or polls the “Transmitter Empty” bit in the Transmit Status Register.

A read of a receive register signals the transmitter side that new data can be written to that register. In the same way, the receiver processor should not read a receive register before receiving a “Receiver Full” interrupt or polling the “Receiver Full” bit in the Receive Status Register.

- Reading the receive register again without verifying that the data was written is prohibited, because the receiver side has no way of knowing the exact time that the transmitter writes the data.
- Before attempting to read the receive register again, the receiver side waits for a “Receiver Full” interrupt, or polls the “Receiver Full” bit in the Receive Status Register.

38.2.2.1 Hardware Semaphore

The ELE_MU includes the following features:

To provide a mechanism to serialize ownership of the Edgelock enclave subsystem, ELE_MUA implements a hardware semaphore (SEMA4) register. Typically, and especially in multi-core host configurations, when a task wants to send crypto service requests to Edgelock enclave, it first establishes ownership of the subsystem by acquiring the semaphore. After the crypto task requests are completed, the host task relinquishes ownership of the semaphore, making the hardware available to other tasks.

Once the semaphore is acquired (locked), only writes to the ELE_MUA TR[0-14] registers from the semaphore owner are allowed; all other attempted writes are error terminated.

In addition to the physical 32-bit SEMA4 register, the ELE_MUA logic supports five read-only 32-bit virtual register programming model. The registers include a Edgelock enclave status register, the secure semaphore owner identity, operations to acquire and release the semaphore, plus a mechanism to force the release of the semaphore.

The physical register information includes the locked/unlocked semaphore state plus the domain identifier and the secure/nonsecure, privileged/user access mode attributes of the locking task.

Accesses to the five virtual registers can be classified into two groups:

1. Two simple register reads that return status (SEMA4_SR) and the secure 32-bit semaphore owner value (OWNR).

2. Three (atomic) compound W/R operations (ACQ, REL, FREL) that potentially update the semaphore state and then return the new secure owner value.

The three compound virtual register operations first perform a “write” and then return the standard 32-bit semaphore owner value. The owner value returns one of three possible values: 0x0000_0000 if the semaphore is unlocked, 0x0000_0055 if the semaphore is owned by the requesting host task (based on domain identifier and the 2-bit {secure/nonsecure, privileged/user} access mode attributes) and 0xFFFF_FFAA is the semaphore is locked but not by the requesting host task.

For the compound W/R operations, the (virtual) register IPS address encodes the write operation to be performed in the address. The actual bus transaction from the initiating host task is always a simple 32-bit data read.

38.2.3 Register Definition

The ELE_MU provides transmit and receive data registers for the communication between Processor A and Processor B. Some control and status registers to the Processor A and Processor B sides are for control operations (such as interrupts), and for status checking of the other ELE_MU-side. The following diagram shows the ELE_MU registers schematic.

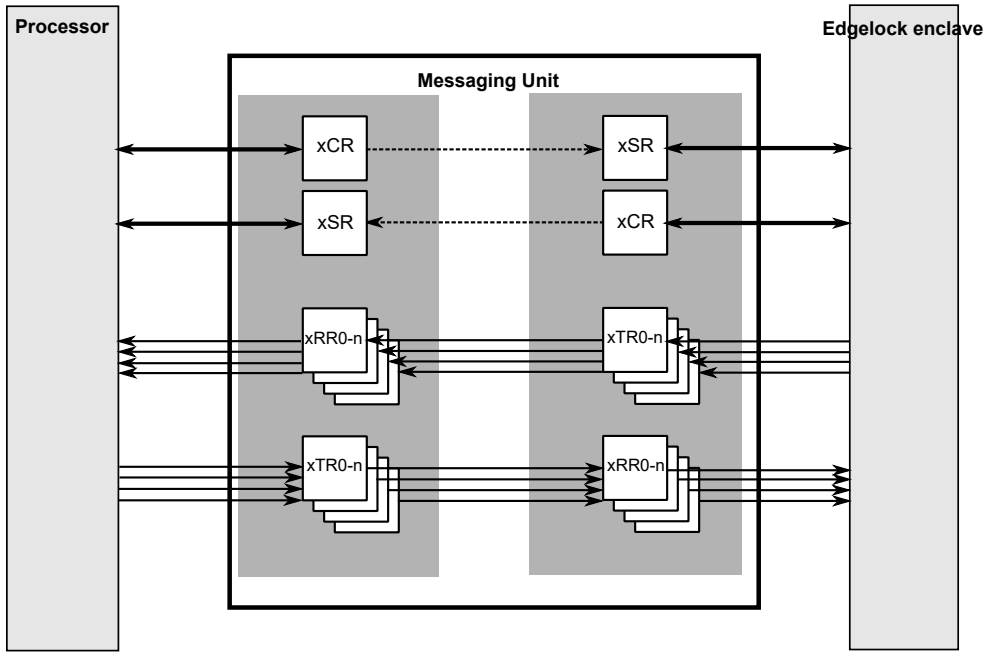


Figure 100. ELE_MU Registers

The detailed ELE_MU register definition can be found below.

NOTE

Read of a write-only, read zero (WORZ) register returns 0. Writes to a read-only (RO) register are ignored. Byte/halfword accesses are supported.

A read/write access to any illegal location of the ELE_MU generates a bus transfer error acknowledge to the Processor A or Processor B. Writes to "RSVD" registers are ignored and reads to "RSVD" registers return 0.

38.2.3.1 ELE_MUA register descriptions

This section contains the detailed register descriptions for the ELE_MUA registers.

38.2.3.1.1 ELE_MUA memory map

ELEMU.ELE_MUA base address: 4002_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	R	0100_0000h
4h	Parameter Register (PAR)	32	R	0000_0210h
8h	Unused Register 0 (UNUSED0)	32	R	0000_0000h
Ch	Status Register (SR)	32	R	0000_0020h
120h	Transmit Control Register (TCR)	32	RW	0000_0000h
124h	Transmit Status Register (TSR)	32	R	0000_0000h
12Ch	Receive Status Register (RSR)	32	R	0000_0000h
1FCh	Unused Register 1 (UNUSED1)	32	RW	0000_0000h
200h - 23Ch	Transmit Register (TR0 - TR15)	32	W	0000_0000h
280h - 284h	Receive Register (RR0 - RR1)	32	R	0000_0000h
400h	Semaphore Status Register (SEMA4_SR)	32	R	0000_0000h
474h	Semaphore Ownership Register (SEMA4_OWNR)	32	R	0000_0000h
998h	Semaphore Acquire Register (SEMA4_ACQ)	32	R	0000_0000h
ACCh	Semaphore Release Register (SEMA4_REL)	32	R	0000_0000h
BA4h	Semaphore Forced Release Register (SEMA4_FREL)	32	R	0000_0000h

38.2.3.1.2 Version ID Register (VER)

Offset

Register	Offset
VER	0h

Function

The VER register can be used to determine the version ID and feature set number of ELE_MUA.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major Version Number (0x01)
23-16 MINOR	Minor Version Number (0x00)
15-0 FEATURE	Feature Set Number 0000_0000_0000_0000b - Standard features are implemented.

38.2.3.1.3 Parameter Register (PAR)

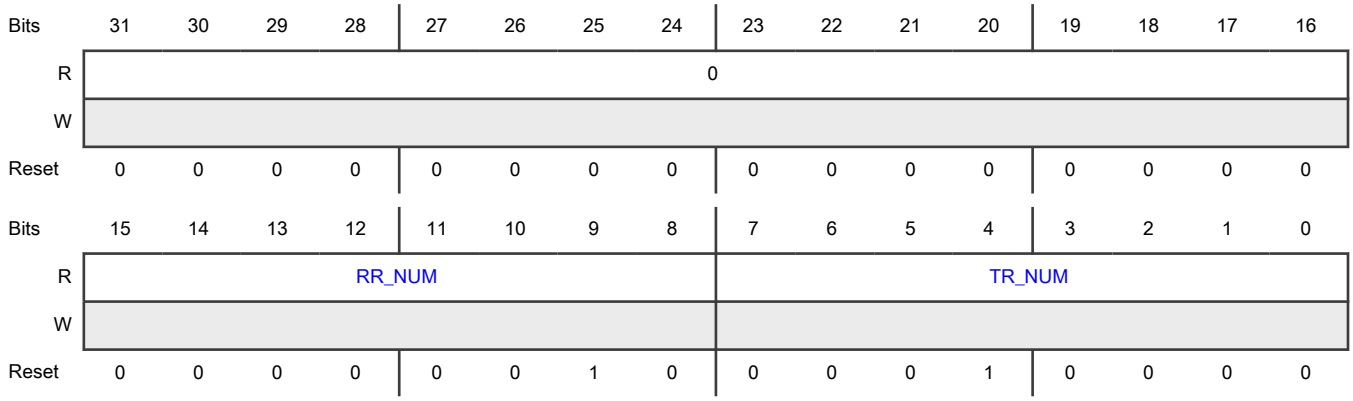
Offset

Register	Offset
PAR	4h

Function

The PAR register reports the number of Transmit (TR) registers and Receive (RR) registers.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 RR_NUM	Number of Receive (RRn) registers (8'd2)
7-0 TR_NUM	Number of Transmit (TRn) registers (8'd16)

38.2.3.1.4 Unused Register 0 (UNUSED0)

Offset

Register	Offset
UNUSED0	8h

Function

DO NOT WRITE

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 —	DO NOT WRITE TO THIS BIT FIELD.

38.2.3.1.5 Status Register (SR)

Offset

Register	Offset
SR	Ch

Function

The SR register reports the status of the Transmit Empty Pending and Receive Full Pending flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									RFP	TEP	0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved
6 RFP	Receive Full Pending Flag Indicates if any of the receive registers are ready to be read. When the RFP bit is set, then software should check the RSR[RFn] flags to determine which RRn register is ready to be read. 0b - No data is ready to be read. All RSR[RFn] bits are clear. 1b - Data is ready to be read. One or more RSR[RFn] bits are set.
5 TEP	Transmit Empty Pending • The TEP bit reads as "1" when any TSR[TEn] bit is set. • The TEP bit reads as "0" when all TSR[TEn] bits are clear. • When the TEP bit reads as "1", check the TSR[TEn] flags to determine which TRn register is ready to be written.
4-0 —	Reserved

38.2.3.1.6 Transmit Control Register (TCR)

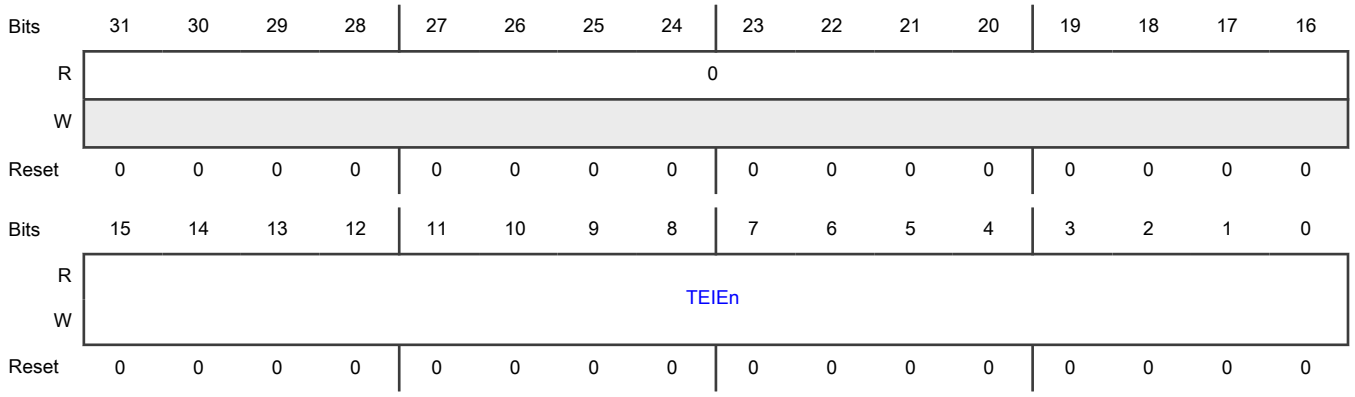
Offset

Register	Offset
TCR	120h

Function

The TCR register is used to configure which Transmit Empty interrupts are generated when the corresponding TSR[TEn] bits are set.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TEIEn	Transmit Register n Empty Interrupt Enable When set, causes a Transmit Empty interrupt to be generated when the corresponding TSR[TEn] bit is set.

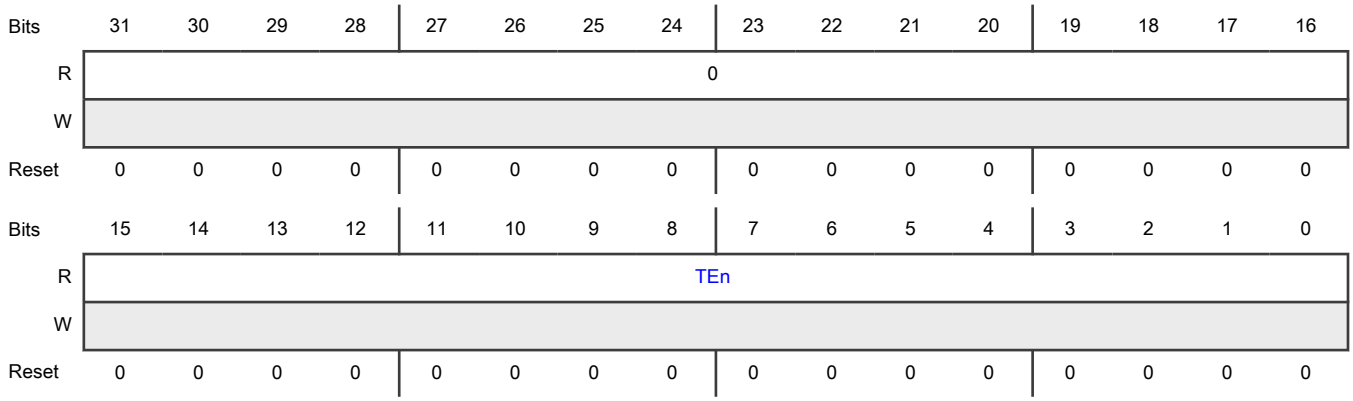
38.2.3.1.7 Transmit Status Register (TSR)

Offset

Register	Offset
TSR	124h

Function
The TSR register shows which TR registers are ready to be written.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TEn	Transmit Register n Empty • The TEn bit is used to indicate to the ELE_MUA processing element that the corresponding ELE_MUA TRn register is ready to be written.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The TEn bit is set after the corresponding ELE_MUB RRn register is read, indicating the ELE_MUA TRn register is empty. • The TEn bit is cleared after the ELE_MUA TRn register is written, indicating the ELE_MUA TRn register is full. • After the TEn bit is cleared, the Transmit Empty interrupt n (if the TCR[TEIEn] bit is set) is cleared on the ELE_MUA side.

38.2.3.1.8 Receive Status Register (RSR)

Offset

Register	Offset
RSR	12Ch

Function

The RSR register shows which RR registers are ready to be read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														RFn	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1-0 RFn	Receive Register n Full • The RFn bit is used to indicate to the ELE_MUA processing element that the corresponding ELE_MUA RRn register is ready to be read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The RFn bit is set after the corresponding ELE_MUB TRn register is written, indicating the ELE_MUA RRn register is full. • The RFn bit is cleared after the ELE_MUA RRn register is read, indicating the ELE_MUA RRn register is empty.

38.2.3.1.9 Unused Register 1 (UNUSED1)

Offset

Register	Offset
UNUSED1	1FCh

Function

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA16															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DATA16	Unused 16-bit Register

38.2.3.1.10 Transmit Register (TR0 - TR15)

Offset

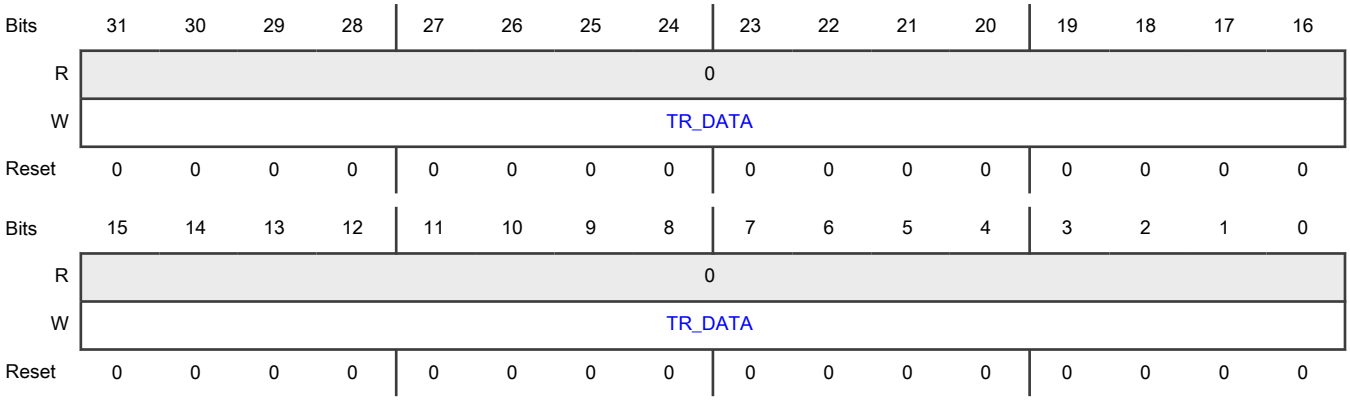
For n = 0 to 15:

Register	Offset
TRn	200h + (n × 4h)

Function

The TR registers contain Transmit Data.

Diagram



Fields

Field	Function
31-0 TR_DATA	Transmit Data • Data written to the TRn register is reflected in the ELE_MUB Receive Register n (RRn). The TRn and RRn registers are not double-buffered; a write to the TRn register overrides the data readable at the RRn register. • A write to the transmit register clears the ELE_MUA TSR[TEn] bit on the transmitter side, and sets the ELE_MUB RSR[RFn] bit on the receiver side. • TRn register should be written only when the ELE_MUA TSR[TEn] bit is set to "1". • Reading the TRn register returns all zeros.

38.2.3.1.11 Receive Register (RR0 - RR1)

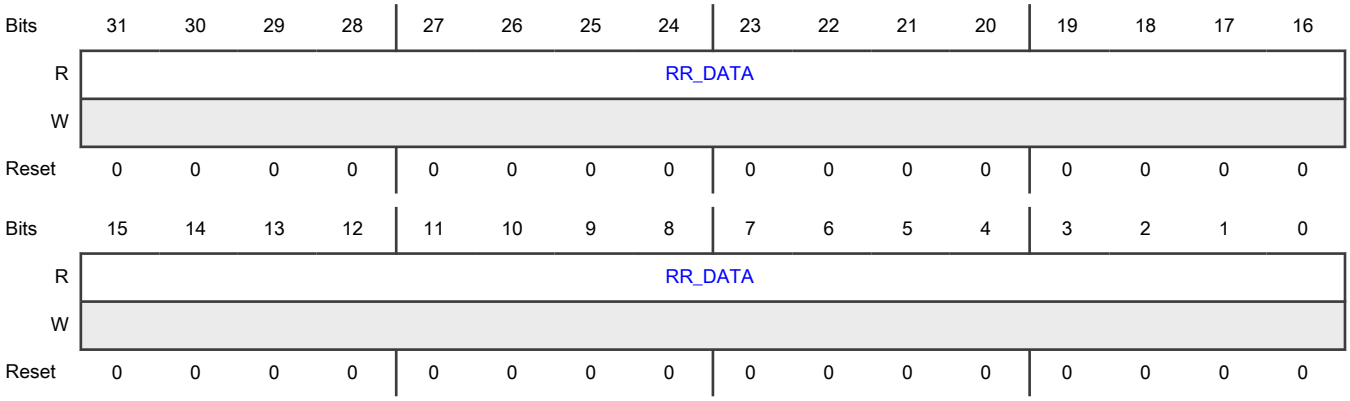
Offset

Register	Offset
RR0	280h
RR1	284h

Function

The RR registers contain Receive Data.

Diagram



Fields

Field	Function
31-0 RR_DATA	Receive Data • Reflects the data written to ELE_MUB Transmit Register (TRn). • Reading the RRn register clears the ELE_MUA RSR[RFn] bit on the receiver side, and sets the ELE_MUB TSR[TEn] bit on the transmitter side. • RRn register should be read only when the ELE_MUB RSR[RFn] bit is set to "1".

38.2.3.1.12 Semaphore Status Register (SEMA4_SR)

Offset

Register	Offset
SEMA4_SR	400h

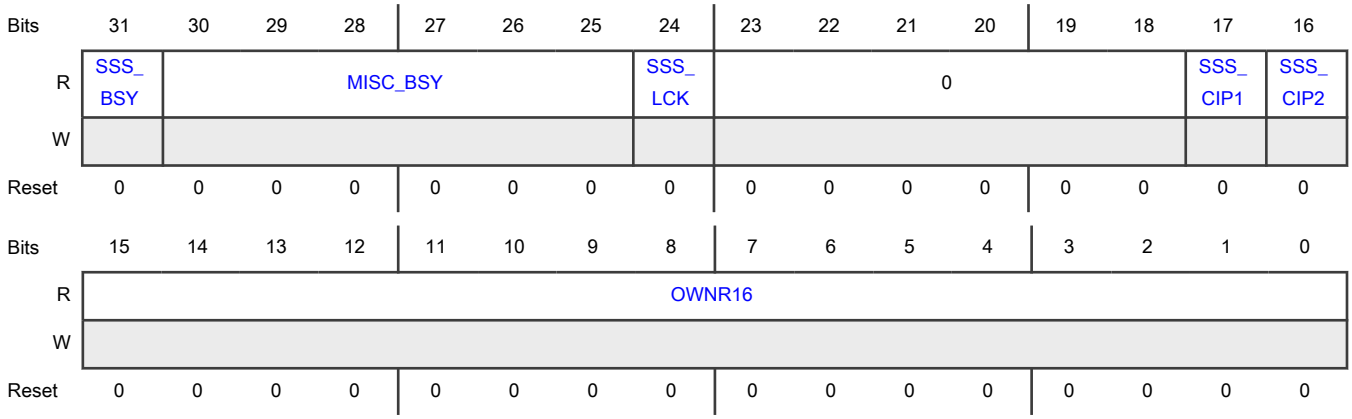
Function

Reads to this register return Edgelock enclave status and a 16-bit version of semaphore ownership.

NOTE

Bit field names with prefix "SSS" refer to the Security SubSystem, Edgelock enclave (ELE).

Diagram



Fields

Field	Function
31 SSS_BSY	Security SubSystem (ELE) Busy This field returns whether or not Edgelock enclave is busy. 0b - Edgelock enclave is not busy 1b - Edgelock enclave CPU is busy
30-25 MISC_BSY	Miscellaneous ELE Busy Indicators This field is a multi-bit collection of individual internal ELE module busy status bits, where an all-zero value indicates no modules are busy and a non-zero value indicates one or more internal ELE modules are busy.
24 SSS_LCK	Security SubSystem (ELE) lockup This field returns whether or not the Edgelock enclave is locked up. 0b - Edgelock enclave is not locked up 1b - Edgelock enclave is locked up in an unrecoverable state
23-18 —	Reserved
17 SSS_CIP1	Security SubSystem (ELE) command group 1 in progress This field returns whether or not a service request is being processed. 0b - Service request group 1 not being processed by ELE 1b - Service request group 1 being processed by ELE
16 SSS_CIP2	Security SubSystem (ELE) command group 2 in progress This field returns whether or not a service request is being processed 0b - Service request group 2 not being processed by ELE 1b - Service request group 2 being processed by ELE

Table continues on the next page...

Table continued from the previous page...

Field	Function

15-0
OWNR16
	Semaphore Owner
	This field returns one of three values: 0x0000 if the semaphore is unlocked, 0x0055 if the semaphore is locked and owned by the requesting host task (based on domain identifier and the 2-bit {secure/nonsecure, privileged/user} access mode attributes) and 0xFFAA if the semaphore is locked but not by the requesting host task. These values support a 3-way signed conditional branch (> 0 (owner), = 0 (unlocked), < 0 (not owner)) based on the return value.
	`if (SEMA4 == IDLE)`
	`  then OWNR16 = 0x0000`
	`  else if (SEMA4 == requesting_host)`
	`  then OWNR16 = 0x0055`
	`  else OWNR16 = 0xFFAA`

38.2.3.1.13 Semaphore Ownership Register (SEMA4_OWNR)

Offset

Register	Offset
SEMA4_OWNR	474h

Function

Reads to this virtual register return the semaphore ownership state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OWNR32															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OWNR32															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OWNR32	Semaphore Owner This field returns one of three values: 0x0000_0000 if the semaphore is unlocked, 0x0000_0055 if the semaphore is locked and owned by the requesting host task (based on domain identifier and the 2-bit

Table continues on the next page...

Field	Function

	{secure/nonsecure, privileged/user} access mode attributes) and 0xFFFF_FFAA if the semaphore is locked but not by the requesting host task. These values support a 3-way signed conditional branch (> 0 (owner), = 0 (unlocked), < 0 (not owner)) based on the return value.
	`if (SEMA4 == IDLE)`
	`  then OWNR32 = 0x0000_0000`
	`  else if (SEMA4 == requesting_host)`
	`  then OWNR32 = 0x0000_0055`
	`  else OWNR32 = 0xFFFF_FFAA`

38.2.3.1.14 Semaphore Acquire Register (SEMA4_ACQ)

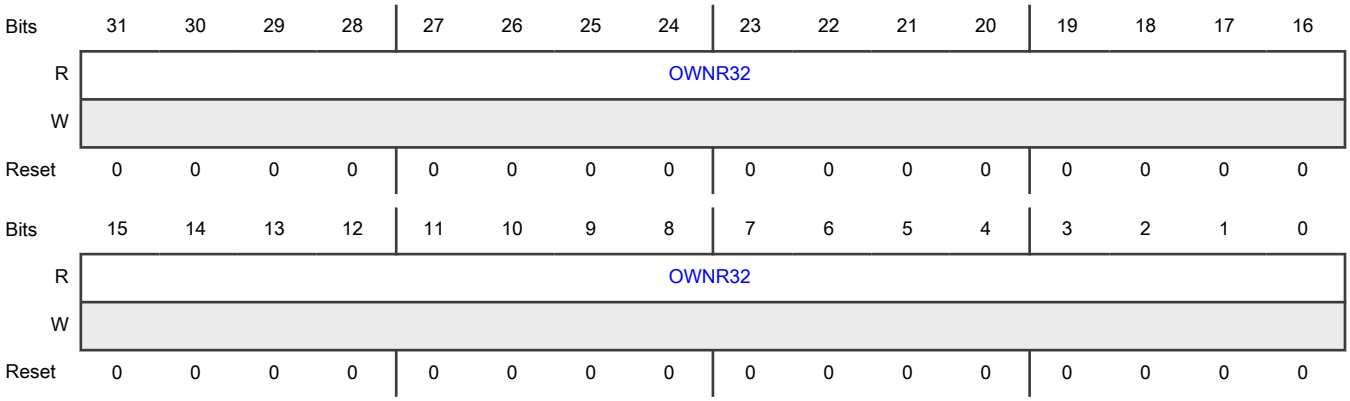
Offset

Register	Offset
SEMA4_ACQ	998h

Function

Reads to this virtual register attempt to acquire (lock) the semaphore for the requesting task.

Diagram



Fields

Field	Function
31-0 OWNR32	Semaphore Owner Reads of this virtual register perform a compound W/R operation. The first operation is an attempted lock of the semaphore. If unlocked, the semaphore state is changed to locked and the associated address attributes (domain identifier, secure/nonsecure and privileged/user attributes) are captured; else if already locked, then the attempted lock of the semaphore is ignored. The second operation is a secure read of the semaphore state returning one of two possible values: 0x0000_0055 if the semaphore is owned by the requesting host task (based on domain identifier and the 2-bit {secure/nonsecure, privileged/user} access mode attributes) and 0xFFFF_FFAA if the semaphore is locked but not by the requesting host task.

Table continues on the next page...

Field	Function
	if (SEMA4 == IDLE) then {SEMA4 = LOCKED by requesting host; OWNR32 = 0x0000_0055} else if (SEMA4 == requesting_host) then OWNR32 = 0x0000_0055 else OWNR32 = 0xFFFF_FFAA

38.2.3.1.15 Semaphore Release Register (SEMA4_REL)

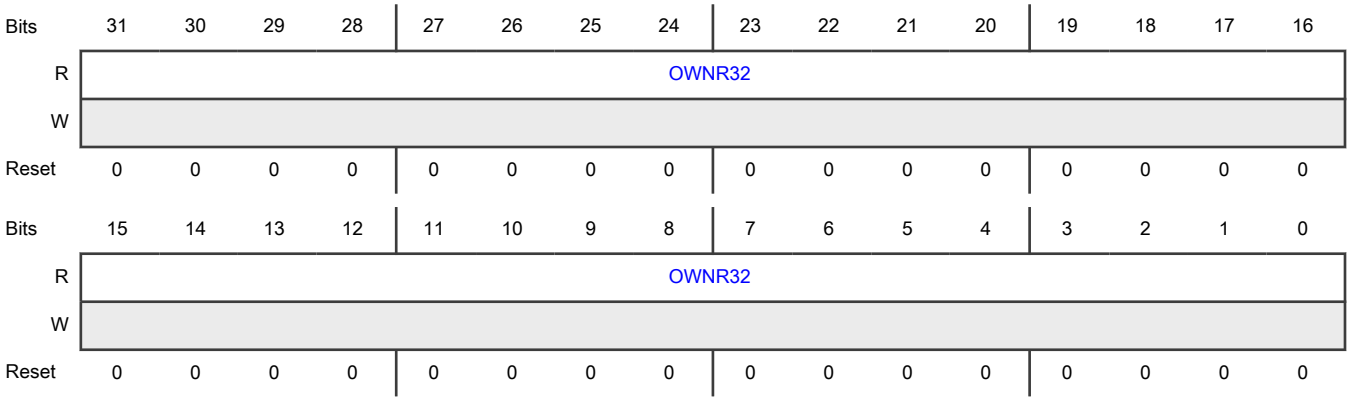
Offset

Register	Offset
SEMA4_REL	ACCh

Function

Reads to this virtual register attempt to release (unlock) the semaphore for the requesting task.

Diagram



Fields

Field	Function
31-0 OWNR32	Semaphore Owner Reads of this virtual register perform a compound W/R operation. The first operation is an attempted release (unlock) of the semaphore. If locked and the associated address attributes (domain identifier, secure/nonsecure and privileged/user attributes) of the transaction match the semaphore state, the state is changed to unlocked; else if the attributes do not match, or the semaphore is already unlocked, then the attempted unlock is ignored. The second operation is a secure read of the semaphore state returning one of two possible values: 0x0000_0000 if the semaphore is unlocked and 0xFFFF_FFAA is the semaphore is locked but not by the requesting host task. if (SEMA4 == LOCKED by requesting host) then {SEMA4 = UNLOCKED; OWNR32 = 0x0000_0000} else if (SEMA4 == LOCKED by a different host) then OWNR32 = 0xFFFF_FFAA else OWNR32 = 0x0000_0000

38.2.3.1.16 Semaphore Forced Release Register (SEMA4_FREL)

Offset

Register	Offset
SEMA4_FREL	BA4h

Function

Reads to this virtual register attempt to unconditionally release (unlock) the semaphore.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OWNR32															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OWNR32															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function

31-0
OWNR32
	Semaphore Owner
	Reads of this virtual register perform a compound W/R operation. The first operation is an attempted release (unlock) of the semaphore. If locked and the associated secure/nonsecure and privileged/user attributes of the transaction signals SecurePrivileged, the state is unconditionally forced to unlocked; else if the attributes define an access mode different from SecurePrivileged, or the semaphore is already unlocked, then the attempted forced unlock is ignored. The second operation is a secure read of the semaphore state returning one of two possible values: 0x0000_0000 if the semaphore is unlocked and 0xFFFF_FFAA if the host task requesting the forced release was not in the SecurePrivileged mode.
	`if ((SEMA4 == LOCKED) && (request == SecurePrivileged))`
	`  then {SEMA4 = UNLOCKED; OWNR32 = 0x0000_0000}`
	`  else if (SEMA4 == UNLOCKED)`
	`  then OWNR32 = 0x0000_0000`
	`  else OWNR32 = 0xFFFF_FFAA`

38.3 Software Architecture and API Examples

The software abstractions of EdgeLock enclave services are defined and used in the form of a standard set of application-callable API functions. In the context of the software application programming interfaces (APIs), the EdgeLock enclave Security SubSystem is simply referenced as the SSS. Accordingly, the software interface detailed here is called the SSS API.

As the SSS is not customer programmable, to use the SSS-provided services, the user programmable host core must initialize, and access said services by interacting with the SSS through a specified interface. This interface is exposed as a low-level

Application Programming Interface that operates with the SSS Messaging Unit (ELE_MU). Serialization of the SSS API is also defined as the Security Subsystem Communication Protocol (SSCP).

Selected requirements for the SSS API include:

- Stateful function context is kept on the host side and is transferred to/from subsystem for stateful operation
- Session context to establish a virtual connection between an application/user context and specific security subsystem and functions
- Key store (secure storage of persistent and transient keys and other attributes)
- Key management (key revocations, key usage constraints)
- Crypto operations (symmetric cipher, authenticated encryption with additional data, message digest, message authentication code, asymmetric crypto, key derivation)
- Integrated with mbedTLS in MCUXpresso SDK

The SSS API is intended to easily integrate with generic software crypto libraries. As an out of box example, the MCUXpresso SDK delivers a version of Arm mbedTLS library integrated with the SSS API, thus enabling the EdgeLock enclave in applications with TLS/DTLS protocol.

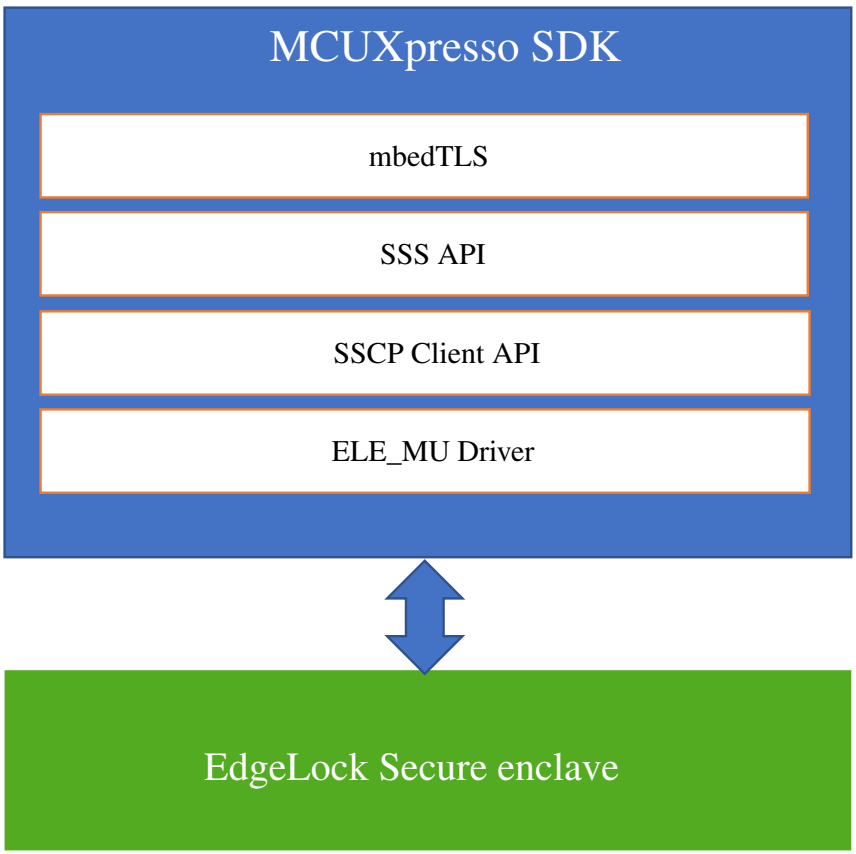


Figure 101. Host SW Security Stack

38.3.1 AES128-ECB encryption examples

In this example, consider the operation of performing a 128-bit AES encryption on a single 16-byte block of data using ECB (Electronic Code Book) mode, the simplest symmetric key mode of operation.

At https://tls.mbed.org/api/aes_8h.html, the Arm mbed site, the following API is defined:

```
int mbedtls_aes_crypt_ecb
(
 mbedtls_aes_context  *ctx,
 int                  mode,
 const unsigned char  input[16],
 unsigned char        output[16]
)
```

This function performs an AES single-block encryption or decryption operation.

It performs the operation defined in the mode parameter (encrypt or decrypt) on the input data buffer defined in the input [*] pointer.

mbedtls_aes_init(), and either mbedtls_aes_setkey_enc() or mbedtls_aes_setkey_dec() must be called before the first call to this API with the same context.

Table 234. mbedtls_aes_crypt_ecb parameters

Parameters	Description
ctx	AES context to use for encryption and decryption It must be initialized and bound to a key
mode	AES operation: MBEDTLS_AES_ENCRYPT (1), MBEDTLS_AES_DECRYPT (0)
input	Buffer holding the input data It must be readable and at least 16 bytes long
output	Buffer where the output data will be written Must be writeable and at least 16 bytes long
ReturnValues	
0	Operation completed successfully

```
sss_status_t sss_cipher_one_go
           (
             sss_symmetric_t *context,
             uint8_t         *iv,
             size_t          ivLen,
             const uint8_t   *srcData,
             uint8_t         *destData,
             size_t          dataLen
           )
```

This function performs an assortment of symmetric key calculations including an AES single-block encryption or decryption operation. Since this function supports a superset of the required AES_ECB computation, some of the input parameters are unused in this instance.

Table 235. status_t sss_cipher_one_go parameters

Parameters	Description
context	Pointer to the symmetric crypto context

Table continues on the next page...

Table 235. status_t sss_cipher_one_go parameters (continued)

Parameters	Description
	Must be initialized and bound to a key
iv	Pointer to the symmetric operation Initialization Vector Unused for AES_ECB
ivLen	Length of the Initialization Vector [bytes] Unused for AES_ECB
srcData	Pointer to buffer containing the input data Must be readable and at least 16 bytes long
destData	Pointer to buffer containing the output data Must be writeable and at least 16 bytes long
dataLen	Size of the input and output data buffer [bytes]
ReturnValues	
kStatus_SSS_Success	Operation completed successfully (0)
kStatus_SSS_Fail	Operation failed (!0)

The ELE_MU driver running on the host processor executes a series of register writes to the ELE_MUA_TRn registers to transmit the service request's parameters to the EdgeLock enclave. When the EdgeLock enclave execution is complete, the ELE_MUB_TRn registers are written with status information which is retrieved by the host processor by reading the ELE_MUA_RRn registers.

The mbedTLS arguments and the SSS API are designed to very efficiently connect with the EdgeLock enclave's ELE_MU hardware interface to maximize performance of the crypto service request's initiation and termination response.

38.3.2 AES128-CBC encryption/decryption example

This is an example of a more complex block cipher mode, consider the mbedTLS and SSS APIs for an AES-CBC (Cipher Block Chaining) service request:

```
int mbedtls_aes_crypt_cbc
                    (
                     mbedtls_aes_context  *ctx,
                     int                  mode,
                     size_t               length,
                     unsigned char        iv[16],
                     const unsigned char  input[16],
                     unsigned char        output[16]
                    )
```

This function performs an AES-CBC encryption or decryption operation on full blocks.

It performs the operation defined in the mode parameter (encrypt or decrypt) on the input data buffer defined in the input parameter.

It can be called as many times as needed, until all the input data is processed. mbedtls_aes_init(), and either mbedtls_aes_setkey_enc() or mbedtls_aes_setkey_dec() must be called before the first call to this API with the same context.

NOTE

This function operates on full blocks, that is, the input size must be a multiple of the AES block size of 16 bytes.

Upon exit, the content of the IV is updated so that you can call the same function again on the next block(s) of data and get the same result as if it was encrypted in one call. This allows a "streaming" usage, that is, the cipher block chaining supports the hardware's ability to continue to add new data blocks into the crypto calculation as the new data becomes available.. If you need to retain the contents of the IV, you should either save it manually or use the cipher module instead.

Table 236. mbedtls_aes_crypt_cbc parameters

Parameters	Description
ctx	AES context to use for encryption and decryption It must be initialized and bound to a key
mode	AES operation: MBEDTLS_AES_ENCRYPT (1) or MBEDTLS_AES_DECRYPT (0)
length	The length of the input data in bytes. This must be a multiple of the block size (16 bytes).
iv	Initialization vector (updated after use). It must be a readable and writeable buffer of 16 bytes.
input	Buffer holding the input data It must be readable and of size of length bytes.
output	Buffer where the output data will be written It must be writeable and of size length bytes.
Return Values	
0	Operation completed successfully
!0	MBEDTLS_ERR_AES_INVALID_INPUT_LENGTH on failure

The SSS API is the same one previously used in the AES_ECB function call.

```
sss_status_t sss_cipher_one_go
                    (
                      sss_symmetric_t *context,
                      uint8_t         *iv,
                      size_t          ivLen,
                      const uint8_t   *srcData,
                      uint8_t         *destData,
                      size_t          dataLen
                     )
```

For this function, all the input parameters are used (unlike the simpler AES_ECB call).

To summarize, the combination of the ELE_MU hardware module plus the security software stack defining the interface from the mbedTLS APIs through the SSS API and eventually the EdgeLock enclave software provides an efficient, secure mechanism to process host crypto service requests.

Chapter 39
Cyclic Redundancy Check (CRC)

39.1 Chip-specific CRC information

Table 237. Reference links to related information

Topic	Related module	Reference
Full description	CRC	CRC
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

39.1.1 Module instances

This device has one instance of the CRC module, CRC0.

39.2 Overview

CRC generates 16-bit and 32-bit CRC codes for error detection. You calculate these codes for 32 bits of data at a time.

This module provides a programmable polynomial and other parameters that you require to implement the 16-bit and 32-bit CRC standards.

39.2.1 Block diagram

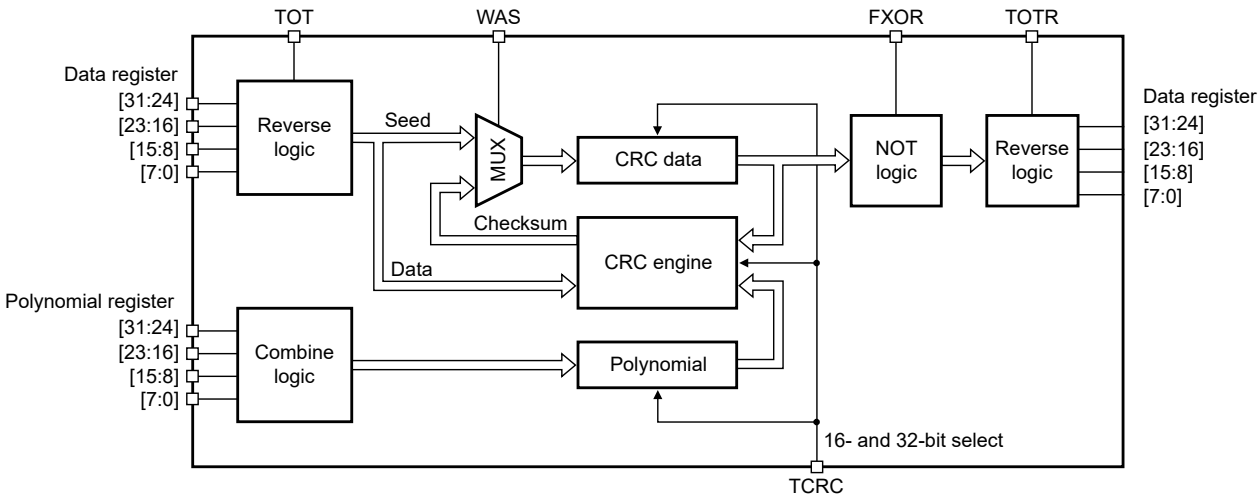


Figure 102. Block diagram

39.2.2 Features

- Hardware CRC generator circuit using 16-bit and 32-bit programmable shift registers.
- Programmable initial seed value and polynomial.

- Option to transpose input or output data (the CRC result) bitwise or bytewise: this option is required for certain CRC standards. You cannot perform a bytewise transpose operation when accessing CRC Data (DATA) via 8-bit accesses. You can only perform the bytewise transpose function.
- Option to invert the final CRC result.
- 32-bit CPU register programming interface.

39.3 Memory map and register descriptions

NOTE

Write accesses to the register addresses that are not mapped to the peripheral but included in the address space of the peripheral may result in unpredictable functionality. You must reconfigure CRC in case a transfer error occurs.

39.3.1 CRC register descriptions

39.3.1.1 CRC memory map

CRC0 base address: 4002_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	CRC Data (DATA)	32	RW	FFFF_FFFFh
4h	CRC Polynomial (GPOLY)	32	RW	0000_1021h
8h	CRC Control (CTRL)	32	RW	0000_0000h

39.3.1.2 CRC Data (DATA)

Offset

Register	Offset
DATA	0h

Function

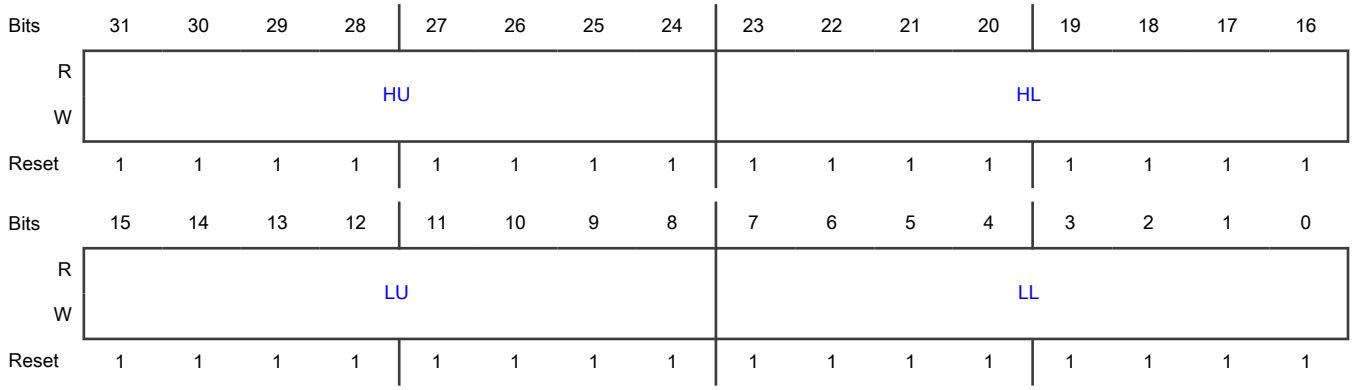
Contains the value of the seed, data, and checksum. When CTRL[WAS] = 1 , any write to this register is regarded as the seed value. When CTRL[WAS] becomes 0, any write to this register is regarded as data for general CRC computation.

In 16-bit CRC mode, the DATA[HU] and DATA[HL] are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, you can write 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous; with the MSB of data value written first.

After writing all data values, you can read the CRC result from this register. In 16-bit CRC mode, the CRC result is available in DATA[LU] and DATA[LL]. In 32-bit CRC mode, all fields contain the result. Reads of this register, at any time, return the intermediate CRC value, if the CRC module is configured.

Diagram



Fields

Field	Function
31-24 HU	CRC High Upper Byte Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 0. • In 16-bit CRC mode (CTRL[TCRC] = 0), this field is not used for programming a seed value. • In 32-bit CRC mode (CTRL[TCRC] = 1), values written to this field are part of the seed value when CTRL[WAS] = 1.
23-16 HL	CRC High Lower Byte Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 0. • In 16-bit CRC mode (CTRL[TCRC] = 0), this field is not used for programming a seed value. • In 32-bit CRC mode (CTRL[TCRC] = 1), values written to this field are part of the seed value when CTRL[WAS] = 1.
15-8 LU	CRC Low Upper Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 1, values written to this field are part of the seed value.
7-0 LL	CRC Low Lower Byte Generates CRC checksum when CTRL[WAS] = 0 . When CTRL[WAS] = 1 , values written to this field are part of the seed value.

39.3.1.3 CRC Polynomial (GPOLY)

Offset

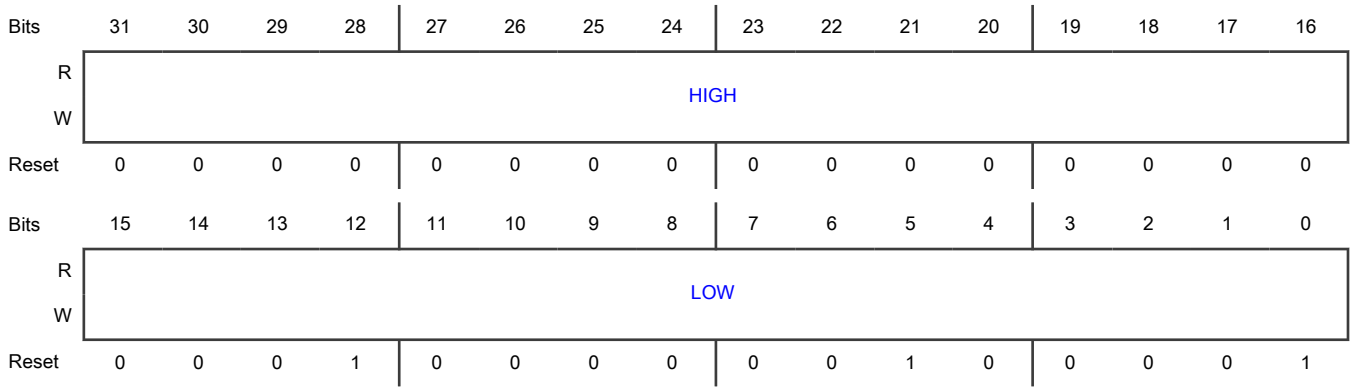
Register	Offset
GPOLY	4h

Function

Contains the value of the polynomial for CRC calculation.

- GPOLY[HIGH] contains the upper 16 bits of the CRC polynomial that are used only in 32-bit CRC mode. Writes to this field are ignored in 16-bit CRC mode.
- GPOLY[LOW] contains the lower 16 bits of the CRC polynomial that are used in both 16-bit and 32-bit CRC modes.

Diagram



Fields

Field	Function
31-16 HIGH	High Polynominal Half-Word Writable and readable in 32-bit CRC mode (CTRL[TCRC] = 1). You cannot write to this field in 16-bit CRC mode (CTRL[TCRC] = 0).
15-0 LOW	Low Polynominal Half-Word Writable and readable in both 16-bit and 32-bit CRC modes.

39.3.1.4 CRC Control (CTRL)

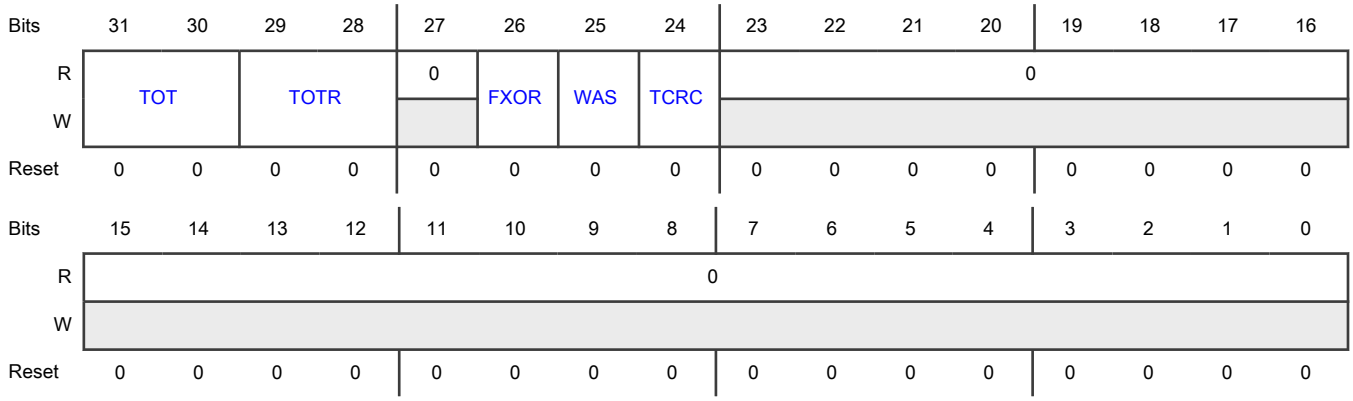
Offset

Register	Offset
CTRL	8h

Function

Controls the configuration and working of CRC module. You must write 1 to the appropriate fields of this register before starting a new CRC calculation, which you can initialize by writing 1 to CTRL[WAS] and then writing the seed into CRC Data (DATA).

Diagram



Fields

Field	Function
31-30 TOT	Transpose Type for Writes Defines the transpose configuration of the data written to CRC Data (DATA). See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed; bytes are not transposed 10b - Both bits in bytes and bytes are transposed 11b - Only bytes are transposed; no bits in a byte are transposed
29-28 TOTR	Transpose Type for Read Identifies the transpose configuration of the value read from CRC Data (DATA). See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed; bytes are not transposed 10b - Both bits in bytes and bytes are transposed 11b - Only bytes are transposed; no bits in a byte are transposed
27 —	Reserved
26 FXOR	Complement Read of CRC Data Register Enables on-the-fly complementing of read data. Some CRC protocols require the final checksum to be XORed with FFFFFFFFh or FFFFh. 0b - No XOR on reading 1b - Inverts or complements the read value of the CRC Data
25	Write as Seed Specifies whether writes to CRC Data (DATA) are data values or seed values.

Table continues on the next page...

Table continued from the previous page...

Field	Function
WAS	When this field = 1, the value that you write to CRC Data (DATA) is considered as seed value. When this field = 0, the value that you write to CRC Data (DATA) is considered as data for CRC computation. 0b - Data values 1b - Seed values
24 TCRC	TCRC Defines the width of the CRC protocol. 0b - 16-bit 1b - 32-bit
23-0 —	Reserved

39.4 Functional description

39.4.1 Modes of operation

The following sections discuss the various modes that affect the functionality of CRC.

39.4.1.1 Run mode

This is the basic mode of operation.

39.4.1.2 Low-Power mode

An in-progress CRC calculation stops when the chip enters Low-Power mode that disables the module clock. The calculation resumes after the clock is enabled or via system reset for exiting Low-Power mode. Clock gating for the module is dependent on the chip.

39.4.2 CRC calculations

In 16-bit and 32-bit CRC modes, data values can be programmed as 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes can lead to an incorrect CRC computation.

39.4.2.1 Computing a 16-bit CRC

Perform these steps to compute a 16-bit CRC:

1. Write 0 to CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in CRC Control (CTRL) as required for the CRC calculation. See Transpose feature and Result complement for details.
3. Write a 16-bit polynomial to GPOLY[LOW].

   GPOLY[HIGH] is is not usable in 16-bit CRC mode.
4. Write 1 to CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to DATA[LU] and DATA[LL].

   DATA[HU] and DATA[HL] are not used.

6. Write 0 to CTRL[WAS] to write data values.
7. Write data values into DATA[HU], DATA[HL], DATA[LU], and DATA[LL].

   CRC is computed on every data value write and the intermediate CRC result is stored back into DATA[LU] and DATA[LL].
8. Read the final CRC result from DATA[LU] and DATA[LL].

The transpose and complement operations are performed on-the-fly when reading or writing values. See Transpose feature and Result complement for details.

39.4.2.2 Computing a 32-bit CRC

Perform these steps to compute a 32-bit CRC:

1. Write 1 to CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in CRC Control (CTRL) as required for CRC calculation. See Transpose feature and Result complement for details.
3. Write a 32-bit polynomial to GPOLY[HIGH] and GPOLY[LOW].
4. Write 1 to CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to DATA [HU], DATA [HL], DATA [LU], and DATA [LL].
6. Write 0 to CTRL[WAS] to start writing data values.
7. Write data values DATA[HU], DATA[HL], DATA[LU], and DATA[LL].

   CRC is computed on every data value write and the intermediate CRC result is stored back into DATA[HU], DATA[HL], DATA[LU], and DATA[LL].
8. Read the final CRC result from DATA[HU], DATA [HL], DATA [LU], and DATA [LL].

   CRC is calculated bytewise and two clocks are required to complete one CRC calculation.

The transpose and complement operations are performed on-the-fly when reading or writing values. See Transpose feature and Result complement for details.

39.4.3 Transpose feature

The transpose feature is not enabled by default. However, the CRC standards require the input data and/or the final checksum to be transposed. You have an option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on-the-fly while being read or written.

Some protocols use the little-endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

39.4.3.1 Types of transpose

CRC provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] and CTRL[TOTR] according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from CRC Data (DATA).

1. CTRL[TOT] or CTRL[TOTR] is 0.

   No transposition occurs.
2. CTRL[TOT] or CTRL[TOTR] is 1.

   Bits in a byte are transposed when bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}. See the following figure.

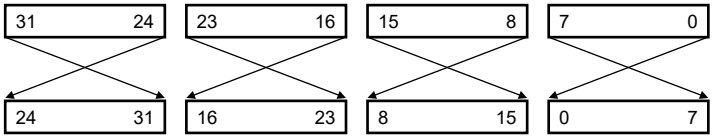


Figure 103. Transpose type 1b

3. CTRL[TOT] or CTRL[TOTR] is 10b.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes {reg[0:7], reg[8:15], reg[16:23], reg[24:31]}. See the following figure.

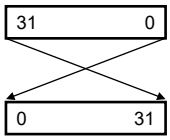


Figure 104. Transpose type 10b

4. CTRL[TOT] or CTRL[TOTR] is 11b.

   Bytes are transposed but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}. See the following figure.

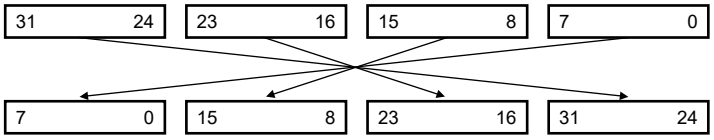


Figure 105. Transpose type 11b

NOTE

For 8-bit and 16-bit write accesses to CRC Data (DATA), the data is transposed with 0s on the unused byte or bytes (taking 32 bits as a whole), but CRC is calculated on the valid byte(s) only. When reading the CRC Data (DATA) for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in DATA[HU] and DATA[HL]. You must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

39.4.4 Result complement

When CTRL[FXOR] = 1, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in CRC Data (DATA) every time CRC Data (DATA) is read. When CTRL[FXOR] = 0, reading CRC Data (DATA) accesses the raw checksum value.

39.5 Use cases

The following tables use the little-endian format.

39.5.1 CTRL programming

The following table shows CRC Control (CTRL) programming for 16-bit CRC.

Table 238. CTRL programming for 16-bit CRC

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-16_CCITT_FALSE	1021h	FFFFh	0	0	0000h	0h	0h	0h
CRC-16_ARC	8005h	0000h	1	1	0000h	1h	2h	0h
CRC-16_AUG_CCITT	1021h	1D0Fh	0	0	0000h	0h	0h	0h
CRC-16_BUYPASS	8005h	0000h	0	0	0000h	0h	0h	0h
CRC-16_CCITT_ZERO	1021h	0000h	0	0	0000h	0h	0h	0h
CRC-16_CDMA2000	C867h	FFFFh	0	0	0000h	0h	0h	0h
CRC-16_DDS_110	8005h	800Dh	0	0	0000h	0h	0h	0h
CRC-16_DECT_R	589h	0000h	0	0	FFFFh	1h	2h	1h
CRC-16_DECT_X	589h	0000h	0	0	0000h	0h	0h	0h
CRC-16_DNP	3D65h	0000h	1	1	FFFFh	1h	2h	1h
CRC-16_EN_13757	3D65h	0000h	0	0	FFFFh	1h	2h	1h
CRC-16_GENIBUS	1021h	FFFFh	0	0	FFFFh	1h	2h	1h
CRC-16_MAXIM	8005h	0000h	1	1	FFFFh	1h	2h	1h
CRC-16_MCRF4XX	1021h	FFFFh	1	1	0000h	1h	2h	0h
CRC-16_RIELLO	1021h	B2AAh	1	1	0000h	1h	2h	0h
CRC-16_T10_DIF	8BB7h	0000h	0	0	0000h	0h	0h	0h
CRC-16_TELEDISK	A097h	0000h	0	0	0000h	0h	0h	0h
CRC-16_TMS37157	1021h	89ECh	1	1	0000h	1h	2h	0h
CRC-16_USB	8005h	FFFFh	1	1	FFFFh	1h	2h	1h
CRC-16_A	1021h	C6C6h	1	1	0000h	1h	2h	0h
CRC-16_KERMIT	1021h	0000h	1	1	0000h	1h	2h	0h
CRC-16_MODBUS	8005h	FFFFh	1	1	0000h	1h	2h	0h
CRC-16_X_25	1021h	FFFFh	1	1	FFFFh	1h	2h	1h
CRC-16_XMODEM	1021h	0000h	0	0	0000h	0h	0h	0h

The following table shows CRC Control (CTRL) programming for 32-bit CRC.

Table 239. CTRL programming for 32-bit CRC

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-32	04C11DB7h	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32_BZIP2	04C11DB7h	FFFFFFFFh	0	0	FFFF_FFFFh	0h	0h	1h

Table continues on the next page...

Table 239. CTRL programming for 32-bit CRC (continued)

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-32C	1EDC6F41h	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32D	A833982Bh	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32_MPEG-2	04C11DB7h	FFFFFFFFh	0	0	0000_0000h	0h	0h	0h
CRC-32_POSIX	04C11DB7h	00000000h	0	0	FFFF_FFFFh	0h	0h	1h
CRC-32Q	814141ABh	00000000h	0	0	0000_0000h	0h	0h	0h
CRC-32_JAMCRC	04C11DB7h	FFFFFFFFh	1	1	0000_0000h	1h	2h	0h
CRC-32_XFER	000000AFh	00000000h	0	0	0000_0000h	0h	0h	0h

39.5.2 Expected read data fields

The following table shows the expected read data fields for 16-bit CRC.

Table 240. Expected read data fields for 16-bit CRC

Algorithm	CRC Data (DATA)
CRC16_CCITT_FALSE	[31:16] = Unknown[15:0] = Valid data
CRC16_ARC	[31:16] = Valid data [15:0] = Unknown
CRC16_AUG_CCITT	[31:16] = Unknown [15:0] = Valid data
CRC16_BUYPASS	[31:16] = Unknown [15:0] = Valid data
CRC16_CCITT_ZERO	[31:16] = Unknown [15:0] = Valid data
CRC16_CDMA2000	[31:16] = Unknown [15:0] = Valid data
CRC16_DDS_110	[31:16] = Unknown [15:0] = Valid data
CRC16_DECT_R	[31:16] = Valid data [15:0] = Unknown
CRC16_DECT_X	[31:16] = Unknown [15:0] = Valid data
CRC16_DNP	[31:16] = Valid data [15:0] = Unknown
CRC-16_EN_13757	[31:16] = Valid data [15:0] = Unknown
CRC-16_GENIBUS	[31:16] = Valid data [15:0] = Unknown
CRC-16_MAXIM	[31:16] = Valid data [15:0] = Unknown
CRC-16_MCRF4XX	[31:16] = Valid data [15:0] = Unknown
CRC-16_RIELLO	[31:16] = Valid data [15:0] = Unknown
CRC-16_T10_DIF	[31:16] = Unknown [15:0] = Valid data
CRC-16_TELEDISK	[31:16] = Unknown [15:0] = Valid data
CRC-16_TMS37157	[31:16] = Valid data [15:0] = Unknown

Table continues on the next page...

Table 240. Expected read data fields for 16-bit CRC (continued)

Algorithm	CRC Data (DATA)
CRC-16_USB	[31:16] = Valid data [15:0] = Unknown
CRC-16_A	[31:16] = Valid data [15:0] = Unknown
CRC-16_KERMIT	[31:16] = Valid data [15:0] = Unknown
CRC-16_MODBUS	[31:16] = Valid data [15:0] = Unknown
CRC-16_X_25	[31:16] = Valid data [15:0] = Unknown
CRC-16_XMODEM	[31:16] = Unknown [15:0] = Valid data

The following table shows the expected read data fields for 32-bit CRC.

Table 241. Expected read data fields for 32-bit CRC

Algorithm	CRC Data (DATA)
CRC-32	[31:0] = Valid data
CRC-32_BZIP2	[31:0] = Valid data
CRC-32C	[31:0] = Valid data
CRC-32D	[31:0] = Valid data
CRC-32_MPEG-2	[31:0] = Valid data
CRC-32_POSIX	[31:0] = Valid data
CRC-32Q	[31:0] = Valid data
CRC-32_JAMCRC	[31:0] = Valid data
CRC-32_XFER	[31:0] = Valid data

39.6 External signals

There is no CRC signal that connects off chip.

39.7 CRC initialization and reinitialization

To enable CRC calculation, you must program:

- CTRL[WAS].
- CRC Polynomial (GPOLY).
- Parameters for transposition and CRC result inversion in the applicable registers.

Writing 1 to CTRL[WAS] enables you to program the seed value into CRC Data (DATA).

After a CRC calculation completes, you can reinitialize the module for a new CRC computation by again writing 1 to CTRL [WAS] and programming a new, or previously used, seed value. You must set all other parameters before programming the seed value and subsequent data values.

Chapter 40
16-bit Analog-to-Digital Converter (ADC)

40.1 Chip-specific ADC information

Table 242. Reference links to related information

Topic	Related module	Reference
Full description	ADC	ADC
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

40.1.1 Module instances

This device has one instance of the ADC module, ADC0.

40.1.2 ADC input connections

ADC Channel (CMDLn[ADCH])	A Connection	B Connection	Input Type
0	ADC0_A0		Dedicated, high speed
1	ADC0_A5	ADC0_B5	RTC domain, dedicated
2	ADC0_A6	ADC0_B6	RTC domain, dedicated
3	ADC0_A7		RTC domain, dedicated
4	VSS_ANA	VSS_ANA	Dedicated, high speed
5	ADC0_A8		GPIO domain, muxed
6	ADC0_A9	ADC0_B9	GPIO domain, muxed
7	ADC0_A10	ADC0_B10	GPIO domain, muxed
8	ADC0_A11	ADC0_B11	GPIO domain, muxed
9	ADC0_A12	ADC0_B12	GPIO domain, muxed
10	ADC0_A13	ADC0_B13	GPIO domain, muxed
11	ADC0_A14		
12	ADC0_A15		
13			
14			
15			
16			

Table continues on the next page...

Table continued from the previous page...

ADC Channel (CMDLn[ADCH])	A Connection	B Connection	Input Type
17			
18			
19			
20			
21			
22			
23	VREFO	VREFL	Dedicated VREF connection
24			Reserved
25	VREF BG+	VREF BG-	VREF bandgap
26	Temperature+	Temperature-	Temperature Sensor (inside ADC, no top level connection)
27	PMC BG+	PMC BG-	PMC Bandgap (spc0_pmc_1vbuf_ana_1p8v)
28	VDD_LDO_CORE / 4	VDD_CORE / 4	PMC Resistive dividers
29	VDD_LDO_SYS / 4	VDD_SYS / 4	PMC Resistive dividers
30	ATX0	ATX1	Analog test bus
31	ATX0	ATX2	Analog test bus

40.1.3 ADC voltage reference options

The ADC voltage references are:

- CFG[REFSEL]=00, VDD_ANA supply pin
- CFG[REFSEL]=01, VREFO voltage reference driven from the VREF block
- CFG[REFSEL]=10, VREFH reference pin

40.1.4 ADC trigger inputs

ADC Trigger sources get routed through the Trigger Multiplexer (TRGMUX). See the TRGMUX chapter for available trigger sources.

40.1.5 ADC-VREF dependency

The ADC module requires biasing a voltage from VREF module. For proper ADC operation, the Low-Power Bandgap Buffer Enable and Low-Power Bandgap Enable bitfields (bit 2, LPBG_BUF_EN and bit 1, LPBGEN) in the VREF Control and Status Register (VREF[CSR]) must be enabled prior to use the ADC peripheral. Attempting to use the ADC without the VREF biasing voltage leads to unexpected behavior of the ADC.

40.2 Overview

The 16-bit analog-to-digital converter (ADC) is a dual successive approximation ADC designed for operation within an integrated microcontroller system-on-a-chip.

NOTE

For chip-specific modes of operation, see the power management information for the device.

40.2.1 Block diagram

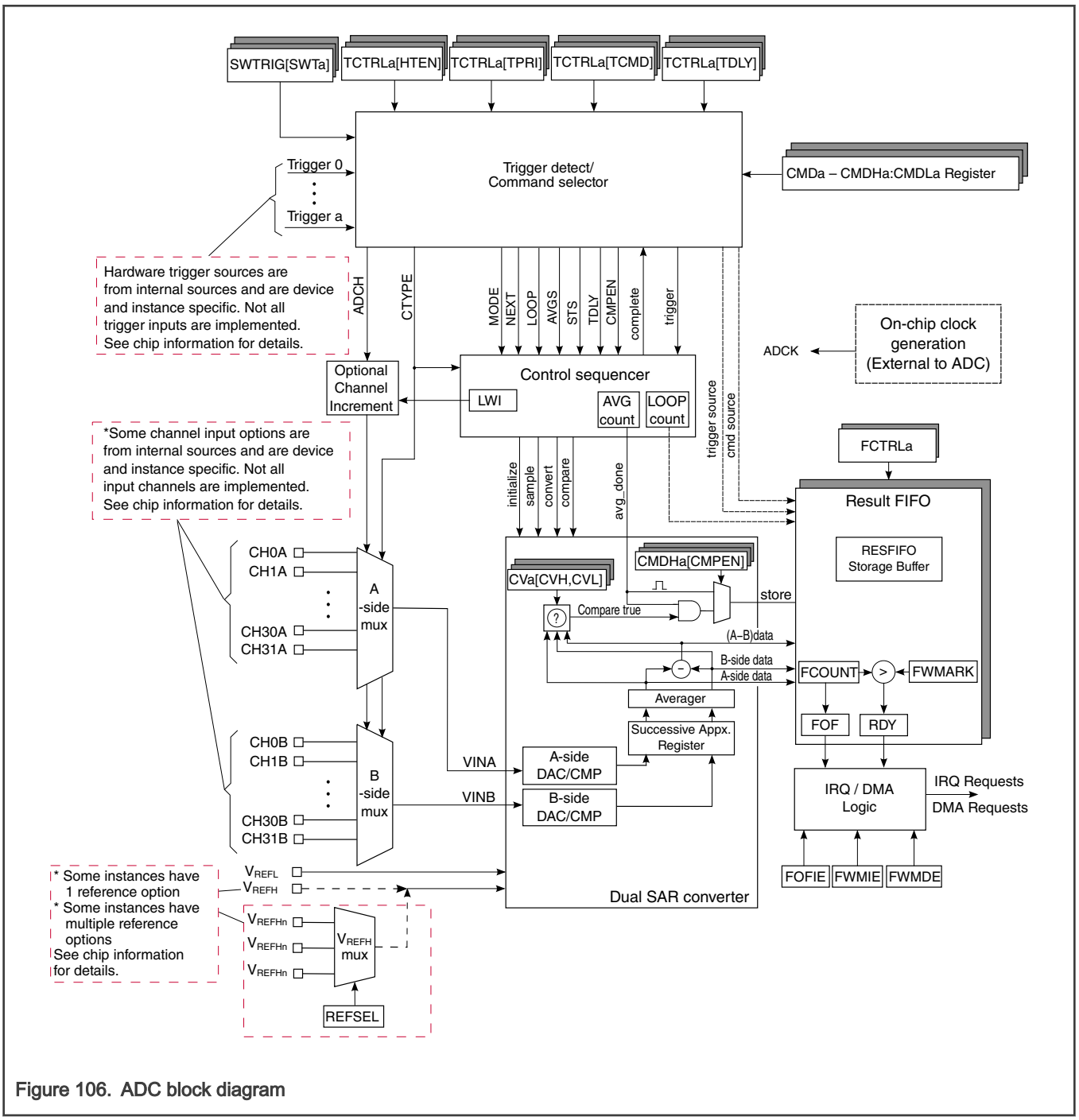


Figure 106. ADC block diagram

40.2.2 Features

ADC has these features:

- Linear successive approximation algorithm
  - Differential operation with 16-bit or 13-bit resolution
  - Single-ended operation with 16-bit or 12-bit resolution
  - Support for two simultaneous single-ended conversions
- Configurable analog input sample time
- Configurable speed options to accommodate operation in low-power modes of SoC
- Trigger detection with up to 4 trigger sources with priority level configuration. Software or hardware trigger option for each.
- 15 command buffers, to allow independent options selection and channel sequence scanning
- Automatic comparisons for less-than, greater-than, within range, or out-of-range with "store on true" and "repeat until true" options
- 2 independent result FIFOs, each containing 16 entries. Each FIFO has configurable watermark and overflow detection.
- Interrupt, DMA, or polled operation
- Linearity and gain adjustment calibration logic

40.3 Functional description

ADC performs analog-to-digital conversions on any of the software-selectable analog input channels via a successive approximation algorithm.

The ADC module can average the result of multiple conversions on a channel before storing the calculated result. The hardware average function is enabled by setting CMDHn[AVGS] to a non-zero value. The function operates in any conversion mode or configuration.

ADC can compare the result of a conversion with the contents of two value registers for less-than, greater-than, inside-range, or outside-range detection. The compare function operates in any conversion mode or configuration.

When the conversion and averaging loops finish, the resulting data is placed in one of 2 available FIFO data buffers . The data includes tag information associated with the result. When the number of stored data words exceeds the setting, a configurable watermark level supports interrupts or DMA requests. Interrupts can also be enabled to indicate when FIFO overflow errors occur.

The module initializes to its lowest power state during reset.

The ADC analog circuits can be pre-enabled to begin conversions sooner at the expense of higher idle currents. Conversions are initiated by selectable trigger events from software or hardware sources.

The trigger-detection logic includes a configurable enable and priority scheme for available trigger sources.

ADC includes multiple command buffers to provide flexibility for channel scanning and independent channel selections for different trigger sources. These command buffers can be configured for:

- Differential or single-ended conversion
- Sample time
- Averaging on a per-channel basis

ADC includes offset and linearity calibration logic. A request for calibration should be made upon reset or power up. Each successive approximation register (SAR) conversion uses calibration data calculated during the calibration routine.

The sequencing of an ADC command is summarized in the flow diagram below.

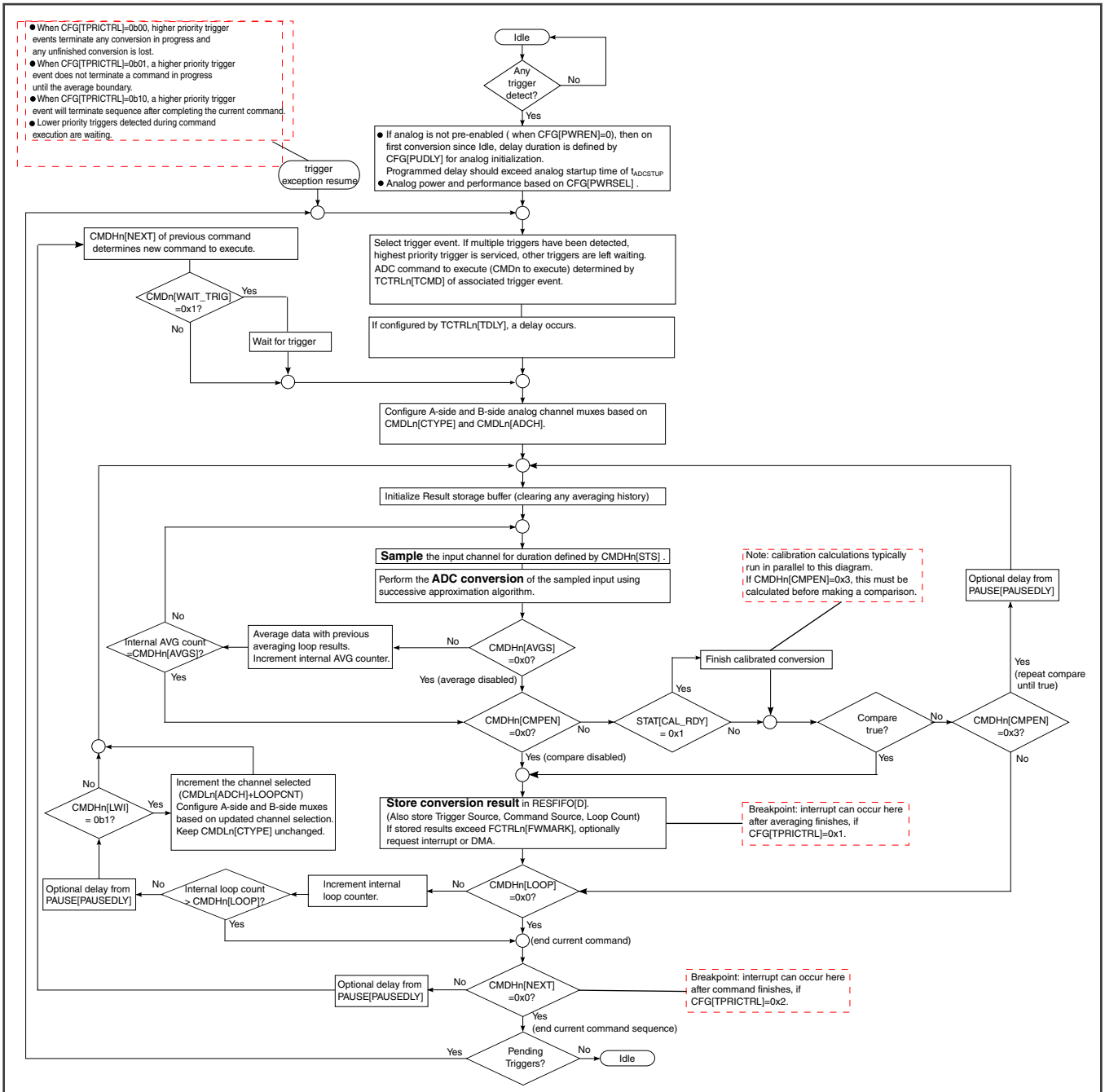


Figure 107. ADC command execution flow diagram

40.3.1 Power control mode

By default, the ADC analog circuits are disabled while ADC is in its idle state. When a trigger is detected and ADC command processing is initiated, the analog circuits are enabled. These circuits require a period of initialization before the first conversion cycle.

The value of CFG[PUDLY] should be set to incur a delay longer than $t_{ADCSTUP}$. The accuracy of initial conversions after activation is degraded if the value of CFG[PUDLY] is too low.

You can achieve faster conversion startup times by setting CFG[PWREN] to pre-enable the analog circuits of ADC. This faster conversion consumes extra power, even while ADC is in an idle state. When CFG[PWREN] is 1, the Power Enable timer is

activated. The timer enforces the minimum time required (configured by CFG[PUDLY]) before detected triggers can initiate ADC conversions.

ADC also has options for controlling power and performance summarized in the table below. See the device data sheet for specification of the available power modes.

Table 243. Power option settings

CFG[PWRSEL]	Description
0xb (Default)	Slow speed and low power
1xb	Fast speed and high power

40.3.2 ADC modes of operation

The ADC module supports the chip low power modes described in the following table. See section Clock operation for more information.

Table 244. Chip modes supported by the ADC module

Chip mode	ADC Operation
Run	Normal operation
Deepsleep/Sleep	When the Doze Enable bit (CTRL[DOZEN]) is 0, the ADC can continue to operate and the module is using an external or internal clock source which remains operating during Deepsleep/Sleep modes. ADC is allowed to continue operation while the system is transitioned to the low power state. Any conversion in progress is not disrupted. External hardware trigger detect and active conversions are operational. When the DOZEN bit is 1, the module waits for the current averaging iteration/FIFO storage to complete before acknowledging Deepsleep or Sleep mode entry. There is no associated ack/handshake and the system transitions to low power state without ADC interaction. ADC operation should terminate after completion of any conversion in progress.
Deep Powerdown	The Doze Enable (CTRL[DOZEN]) bit is ignored and the module waits for the current transfer to complete any pending operation before acknowledging Deep Powerdown mode entry.

40.3.3 Voltage reference

The voltage reference high (V_{REFH}) used by ADC is supplied from either an on-chip voltage reference source or from an off-chip source via external pins. V_{REFL} is always from an external pin and must be at the same voltage as V_{SSA}.

This block supports a programmable selection of the Voltage Reference High used for ADC conversions (via CFG[REFSEL]).

See the chip configuration information on the voltage reference options specific to this packaged device.

40.3.4 Trigger detect and command execution

See Figure 107 for a flow diagram of command execution sequencing.

ADC command execution is initiated from up to 4 trigger sources. Each trigger can be software-generated by writing 1b to the corresponding SWTRIG[SWTn] field. Alternatively, asynchronous input sources at the periphery of the module can generate hardware triggers. The number and sources of hardware triggers implemented is device-specific. See the chip-specific ADC information for descriptions of available hardware trigger sources for this device.

Each hardware trigger source is enabled by setting the associated enable field (TCTRLn[HTEN]). Each trigger source is assigned a priority via the associated priority control field (TCTRLn[TPRI]). Each of the trigger sources is associated with a command buffer via the associated command select field (TCTRLn[TCMD]).

When a hardware trigger input is enabled, hardware trigger events are detected on the rising edge of the associated hardware trigger source.

Each trigger source has an associated priority field TCTRLn[TPRI] which allows arbitration between trigger sources. Arbitration selects two things: which trigger sequence to execute next, and how to handle a trigger exception. Trigger exceptions are defined as allowing a higher-priority trigger sequence to interrupt operation of a lower-priority sequence. When a trigger exception occurs, programmable arbitration allows the configurable stop and resume points for low-priority sequences. The fields affecting arbitration are CFG[HPT_EXDI], CFG[TCMDRES], CFG[TRES], and CFG[TPRICTRL].

1. When CFG[HPT_EXDI] = 1b, trigger exceptions are disabled and any higher priority triggers are left pending until the current sequence completes. New triggers are accepted based on priority.
2. When CFG[HPT_EXDI] = 0b (default), exceptions are enabled and the higher-priority sequence begins executing at a user-specified breakpoint.

Breakpoint locations are determined by CFG[TPRICTRL], which affects latency for accepting trigger exceptions.

1. When CFG[TPRICTRL] = 0h, a higher-priority trigger causes an immediate command abort and the new command specified by the trigger is immediately started.
2. When CFG[TPRICTRL] = 1h, the current conversion is allowed to complete (including averaging) before the higher-priority exception starts. In this mode, if the command is running through a series of averages, this series completes. However, there is no requirement to finish the entire command before being interrupted. For example, if the command consists of four loop iterations, there is no requirement to complete all four iterations before the interrupt occurs.
3. When CFG[TPRICTRL] = 2h, a higher-priority trigger begins once the current command is completed. If a command consists of five loop iterations each containing eight averages, then all 40 conversions must be completed before accepting the trigger exception.

CFG[TCMDRES] and CFG[TRES] determine what ADC does after accepting a trigger exception. The module can be programmed to resume commands after returning from a trigger exception.

1. When CFG[TRES] = 0h, commands are not automatically resumed after being stopped by an exception. However, an interrupt is set to indicate that this case has occurred. The flag TSTAT[TEXC_NUM] can be used to resolve which trigger was stopped by the exception.
2. When CFG[TRES] = 1h, ADC automatically resumes commands after they were stopped by an exception.

Using CFG[TRES] with CFG[TCMDRES], the module can be programmed to resume commands at one of two possible locations.

1. When CFG[TCMDRES] = 0h, the trigger which was stopped by an exception is resumed from the beginning of its associated command sequence. Triggers that are waiting to be resumed take the same priority programmed to TCTRLn[TPRI].
2. When CFG[TCMDRES] = 1h, the trigger is resumed from the command that it was executing before being interrupted by an exception.

If a trigger occurs with priority lower than the priority of the currently executing command, trigger detection is left pending until the current command sequence finishes. Lower-priority trigger events cannot be serviced until a higher-priority triggered command (or command sequence) completes.

When a conversion is completed (including hardware averaging when CMDHn[AVGS] is non-zero), the result is placed in a RESFIFO buffer. When an ADC command selects looping (when CMDHn[LOOP] is non-zero) a command stores multiple conversion results to the FIFO during execution of that command.

At the end of command execution, CMDHn[NEXT] selects the next command to be executed. Multiple commands can be executed sequentially by configuring the CMDHn[NEXT] field of each command. Setting the next command to 0h causes conversions to terminate at the completion of the current command. Unending circular command execution is allowed by setting the CMDHn[NEXT] field of the last command in a sequence to the first command in the sequence.

By default, command sequences execute automatically in the order in which CMDHn[NEXT] fields are programmed. However, by using CMDHn[WAIT_TRIG], command execution can be stalled and launched based on trigger inputs. For example, if TRIGGER2 is programmed to start the command sequence CMD1, CMD2, CMD3, then this sequence is run once unconditionally to

completion upon receiving TRIGGER2. If CMDH2[WAIT_TRIG] = 1h, however, the sequence pauses after CMD1 until TRIGGER2 is received again. In this way, sequences can be stalled until a trigger assertion is received.

Disabling ADC by writing 0b to CTRL[ADCEN] terminates any active ADC command processing. Writing 0b to CTRL[ADCEN] causes the current command (or command sequence) to terminate, clears any pending triggers, and sends the ADC module to an idle state.

40.3.5 Pause option

When an application does not require the maximum conversion rate, the effective conversion rate can be reduced:

- By implementing periodic trigger events to initiate ADC conversions, or
- By selecting a reduced-frequency clock as the ADCK source.

Both options are chip-specific and are dependent on ADC triggering and clocking options external to the ADC module. The latency associated with ADC analog power-up delays limits the maximum conversion rate when using periodic triggering.

Conversion rates can also be reduced by inserting a pause of a programmable duration:

- Between loop iterations
- Between commands in a sequence
- Between conversions, when a command is executing in the Compare Until True configuration

When PAUSE[PAUSEEN] = 1, PAUSE[PAUSEDLY] controls the duration of pausing during command execution sequencing. The pause delay is a count of (PAUSE[PAUSEDLY] * 4) ADCK cycles.

NOTE

Do not change the PAUSE register while CTRL[ADCEN] = 1. Writes to the PAUSE register while CTRL[ADCEN] is 1 can lead to metastable operation.

See Figure 107 for the places during command execution sequencing where the pause can be inserted.

40.3.6 Temperature Sensor

The ADC module has a dedicated input channel for an on-chip temperature sensor. See the chip specific channel definition to determine which channel is connected to the on-chip temperature sensor.

To calculate the temperature, application software must first execute a conversion of the temperature sensor channel with configuration requirements. The sequence to convert the temperature sensor channel is:

1. Configure a command buffer register to convert the temperature sensor channel (CMDLn[ADCH]).
2. Program the command buffer with the following parameters:
   - Differential: CMDLn[CTYPE] = 2h
   - Max averaging: CMDHn[AVGS] = 7h
   - Max sample time: CMDHn[STS] = 7h
   - LOOP set to 1: CMDHn[LOOP] = 1h
   - Loop with increment disabled: CMDHn[LWI] = 0h
   - Compare function disabled: CMDHn[CMPEN] = 0h
3. Configure a trigger control register TCTRLn with the following parameters:
   - TCTRLn[TCMD] = command buffer used in steps 2 and 3.
   - TCTRLn[FIFO_SEL_A] directs conversion results to FIFO0 or FIFO1
4. Trigger a conversion of the temperature sensor channel. Software triggers the conversion by writing 1 to the associated bit in Software Trigger Register (SWTRIG) (that is, the trigger configured in the previous step).

After completing the conversion of the temperature sensor channel, two results are stored in the FIFO selected by TCTRLn[FIFO_SEL_A]. Each result corresponds to a component of the overall temperature value. Application software reads these values from the FIFO and uses the equation below to calculate the ambient temperature.

$$\text{Temp} = A\left[\frac{\alpha(\text{Vbe8} - \text{Vbe1})}{\text{Vbe8} + (\alpha(\text{Vbe8} - \text{Vbe1}))}\right] - B$$

Figure 108. Temperature sensor function

Where:

- Vbe1 is the first value stored to the FIFO as a result of the temperature sensor channel conversion.
- Vbe8 is the second value stored to the FIFO as a result of the temperature sensor channel conversion.
- A is the slope factor.
- B is the offset factor.
- α is the bandgap coefficient.

A, B, and α are specified constant values from the ADC electrical information in the device datasheet.

40.3.7 Result FIFO operation

ADC includes 2 16-entry FIFOs in which the result of ADC conversions are stored. In addition, a valid indicator bit, the trigger source, the source command, and the loop count are also stored with the data. FCTRLn[FCOUNT] indicates how many valid data words are stored in each RESFIFO.

A programmable watermark threshold supports configurable notification of data availability. When FCTRLn[FCOUNT] is greater than FCTRLn[FWMARK], the associated RDY flag is asserted. When IE[FWMIEn] = 1, a watermark interrupt request is issued. When DE[FWMDEn] = 1, a DMA request is issued. Reading RESFIFO provides the oldest unread data word entry in the FIFO and decrements FCTRLn[FCOUNT]. When FCTRLn[FCOUNT] falls equal to or below FCTRLn[FWMARK], the RDY flag is cleared.

Each FIFO can be emptied by successive reads of RESFIFOn. When RESFIFOn[VALID] is 1, the associated FIFO entry is valid. Reading RESFIFOn when the FIFO is empty (when RESFIFOn[VALID] = 0 and FCTRLn[FCOUNT] = 0h) provides an undefined data word. All FIFOs are reset by writing 1b to CTRL[RSTFIFOn].

If ADC attempts to store a data word to the FIFO when the FIFO is full, the FIFO overflow flag (FCTRLn[FOF]) is set. When IE[FOFIEn] = 1, an overflow interrupt request is issued. The FOF flag is cleared by writing 1 to STAT[FOFn]. When overflow events occur, no new data is stored and the data associated with the storage event that triggered the overflow is lost.

Conversion results can be steered to any FIFO in the design. TCTRLn[FIFO_SEL_A] and TCTRLn[FIFO_SEL_B] determine into which FIFO the final result is written. Depending on which trigger is executing, the results can be steered to different locations. Depending on the type of conversion selected, the FIFO destination register fields are interpreted differently. During either differential or single-ended mode (CMDLn[CTYPE] != 3h) only one result is produced. The destination during these modes is determined from TCTRLn[FIFO_SEL_A]. In dual-single-ended mode, both TCTRLn[FIFO_SEL_A] and TCTRLn[FIFO_SEL_B] determine the Channel A and Channel B destinations respectively.

40.3.8 Sampling Modes

The ADC module supports three different sampling modes: dual single-ended, single-ended, and differential. The sampling mode is determined by the currently executing command using CMDLn[CTYPE].

When executing a command in dual single-ended mode, two independent conversion results are calculated and stored in selectable FIFO destinations. Command processing, however, is not individually controlled for each independent channel being sampled. When operating in dual single-ended mode both channels are sampled and processed simultaneously. The input channels selected for dual single-ended mode are controlled by CMDLn[ADCH] and are always fixed pairs of inputs (CH0A/CH0B, CH1A/CH1B, and so on).

If comparisons are enabled in dual single-ended mode, only the A-side channels (CH0A, CH1A, and so on) are used for the comparison. The A-side comparison determines whether both the A-side and B-side results are written to the FIFOs.

- If the A-side comparison passes, both the A-side and B-side results are stored.
- If the A-side comparison fails, A-side and B-side results are not written to the FIFOs.

Single-ended mode is configurable to allow either A-side or B-side channels to be sampled. In single-ended mode, the results from each conversion can be written to a selectable FIFO using TCTRLn[FIFO_SEL_A].

In differential mode, the final SAR calculation is equivalent to V(CHA) - V(CHB). If the result is negative, then the value is stored in sign-extended two's complement format. TCTRLn[FIFO_SEL_A] also determines where differential conversion results are stored.

In dual single-ended mode, however, two independent results are produced. Individual control is provided by using TCTRLn[FIFO_SEL_A] and TCTRLn[FIFO_SEL_B] to select a FIFO destination for both results during this mode. Both single-ended results may be written to the same destination by programming TCTRLn[FIFO_SEL_A] = TCTRLn[FIFO_SEL_B]. In this case, the CH_A result is always stored before the CH_B result.

40.3.9 Compare Function

After the input is sampled and converted and any averaging iterations are performed, CMDHn[CMPEN] determines:

- Whether to use the automatic compare function, storing the conversion result when true.
- Whether to repeat the channel acquisition until the automatic compare function returns a true result.

The command-sequencing options related to the compare function are summarized in the table below.

NOTE

Latency is added to the end of a compare-until-true conversion to resolve the next command or loop in a sequence. This latency is necessary to calibrate the SAR data before resolving the result of a comparison. Delay for this feature is only added when resolving the result of a conversion. Intermediate samples during averaging do not include extra latency; only loop and command boundaries experience this delay. The latency is always less than or equal to five ADC clock cycles.

Table 245. Compare modes

CMDHn[CMPEN]	Compare function	Description
00b	Compare disabled	Do not perform compare operation. Always store the conversion result to the FIFO.
01b	Reserved	
10b	Store on true	Perform compare operation. Store conversion result to FIFO after averaging only when the result of the comparison is true. Regardless of the comparison result, the LOOP setting is considered. The LOOP counter is incremented before deciding whether the current command has completed or additional LOOP iterations are required.
11b	Repeat compare until true	Perform compare operation. Store conversion result to FIFO after averaging only when the result of the comparison is true. Once a comparison is true, the LOOP setting is considered. The LOOP counter is incremented before deciding whether the current command has completed or additional LOOP iterations are required. When a comparison is false, the conversion is repeated without considering the LOOP setting, and the LOOP counter is not incremented.

The compare operation checks the result based on the values of CVn[CVH] and CVn[CVL], as shown in the following table.

Table 246. Compare operations

CVn[CVL] vs. CVn[CVH]	Operation	Description
set CVn[CVL] < CVn[CVH]	Outside range (General form)	True if the result is less than CVn[CVL] **or** greater than CVn[CVH].
set CVn[CVH] to maximum value set CVn[CVL] to compare point	Less than	True if the result is less than CVn[CVL].
set CVn[CVL] to minimum value set CVn[CVH] to compare point	Greater than	True if the result is greater than CVn[CVH].
set CVn[CVL] > CVn[CVH]	Inside range	True if the result is less than CVn[CVL] **and** greater than CVn[CVH].

NOTE

When ADC continues operating in a low-power mode, the compare function can monitor the voltage and wake the device only when the compare condition is met.

40.3.10 Clock operation

The ADC operates from the ADCK clock input provided from an on-chip clock select block and is used by the SAR conversion control sequencing logic and the FIFO storage buffer. The ADCK frequency must be within the specified frequency range for ADCK and varies based on configuration of CFG[PWRSEL]. Refer to the device datasheet for supported frequency ranges.

The ADC continues operating in Deepsleep and Sleep modes provided the Doze Enable bit (CTRL[DOZEN]) is clear and the on-chip clock select block continues to supply an ADCK clock source. Note, in Deepsleep mode with CTRL[DOZEN] == 0b0, the bus clock can be shut off, and asynchronous interrupts and DMA requests can be configured. In addition, the ADC continues processing commands and writing data to the internal FIFO. The ADC has four sources for async interrupts during Deepsleep mode: watermark, FIFO overflow, TCOMP, and TEXC. To enable them, properly configure the bits IE[FWMIEx], IE[FOFIEx], IE[TCOMP_IE], and IE[TEXC_IE] before entering Deepsleep mode.

When the DOZEN bit is set in Deepsleep and Sleep modes, the ADC waits for the current averaging iteration/FIFO storage to complete before acknowledging Deepsleep or Sleep mode entry. Any pending triggers are dropped when a Deepsleep/Sleep mode request is made with DOZEN set. The ADC is forced into its lowest power setting after acknowledging the DOZEN Deepsleep/Sleep mode request. The same behavior is observed when entering a Deep Powerdown Mode.

40.3.11 Resets

Table 247. ADC Resets

Reset source	Description
Chip reset	The logic and registers for the ADC are reset to their default state on a chip reset.
Software reset	The ADC implements a software reset bit in its Control Register. The CTRL[RST] bit resets all logic and registers to their default state, except for the CTRL register itself.
FIFO reset	The ADC implements write-only control that reset the FIFO0 (CTRL[RSTFIFO0]) and FIFO1 (CTRL[RSTFIFO1]). After a FIFO is reset, that FIFO is empty.

40.3.12 Interrupts and DMA requests

The ADC includes several sources for interrupts and/or DMA requests. The table below summarizes these sources.

A programmable watermark threshold supports configurable notification of data availability and can generate either an interrupt exception or a DMA request. When FCTRLn[COUNT] is greater than FCTRLn[FWMARK], the associated RDYn flag is asserted. Masking for this exception is controlled by the IE[FWMIEn] and DE[FWMDEn] control bits. When RDYn is asserted and mask control bit IE[FWMIEn] is set, a watermark interrupt request is issued. When RDYn is asserted and mask control bit DE[FWMDEn] is set, a DMA request is issued.

The other exception sources can only generate interrupts and do not have a DMA request option. Each of these sources has a mask control bit in the IE register and no corresponding bit in the DE register.

Table 248. ADC Interrupts and DMA Requests

Status Register (STAT)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
RDYn	Result FIFO n Ready Flag	Conversion result data is written to Result FIFO and has a watermark configurable trigger level to generate an exception request as controlled by FCTRLn[FWMARK].	Y	Y	Y
FOFn	Result FIFO n Overflow Flag	Attempts to store data to the FIFO when the FIFO is full is an error condition.	Y	N	Y
TCOMP_INT	Trigger Completion Flag	A trigger sequence has been completed (all associated commands have been run).	Y	N	Y
TEXC_INT	High Priority Trigger Flag	A high priority trigger exception has occurred.	Y	N	Y

40.4 External signals

The ADC module supports analog channel inputs with differential and single-ended conversion options for all channels. ADC requires supply and ground connections.

Table 249. ADC signal descriptions

Signal	Description	I/O
V_{DDA}	Analog Power Supply	I
V_{SSA}	Analog Ground	I
CHnA - CH0A[1]	A-side Analog Channel Inputs	I
CHnB - CH0B[1]	B-side Analog Channel Inputs	I

1. where n is the maximum channel number supported in the chip. See the chip-specific information for the number of channels supported on your device.

40.4.1 Analog channel inputs (CHnA and CHnB)

CMDLn[ADCH] and CMDLn[CTYPE] control selection of paired or individual input channels. Each ADC command independently makes a channel and conversion type selection. Each CMDLn[ADCH] channel selection has an associated A side and an associated B side input. Each CMDLn[ADCH] pair can be converted in differential mode, but only limited pairs should be converted as differential channels (for example, adjacent pins designed with matched impedance). For the pin pairings available for differential conversions for your device, see the Chip Configuration details.

NOTE

Some input channels from on-chip sources, such as temperature sensors and reference voltage sources, may only be connected to individual instances of ADC. Some input channel options in the field-setting descriptions may not be available for your device. See chip-specific information for the channels supported on this device.

40.5 Initialization

40.5.1 Calibration functions

The ADC module has multiple calibration functions that must be executed as part of ADC setup to achieve the specified accuracy. Calibration must be run after any reset and before a conversion is initiated. Before offset calibration or calibration, the user must configure the clock source and frequency of ADC for the clock source availability and needs of the application. ADC must be enabled (CTRL[ADCEN] = 1h) before a calibration function runs. If calibration is requested while ADC is actively converting, that sequence completes before starting calibration.

Averaging multiple conversions can achieve improved accuracy during the calibration routines. CTRL[CAL_AVGS] is used during a calibration routine to control how many samples are averaged together. If the application uses ADC in a wide variety of configurations, the configuration for which the highest accuracy is required should be selected. Alternatively, multiple calibrations can be done for the different configurations.

40.5.1.1 Offset calibration

The Offset Trim Register (OFSTRIM) is used to trim for ADC comparator offset voltage. The ADC supports an offset calibration function in which the OFSTRIM register is automatically updated. To perform offset calibration:

1. Configure for the desired averaging via CTRL[CAL_AVGS].
2. Initiate Offset Calibration by setting CTRL[CALOFS].
3. Poll the STAT[CAL_RDY] flag. When STAT[CAL_RDY] is asserted, the offset calibration function has completed, and the OFSTRIM register has updated.

40.5.1.2 ADC Calibration

ADC includes hardware calibration logic in which the A-side and B-side converters are calibrated for gain error and linearity error correction of the raw conversion result. GCR0[GCALR] and the CAL_GAR registers control calibration for the A-side converter. GCR1[GCALR] and the CAL_GBR registers control calibration for the B-side converter. ADC supports a calibration function in which the CAL_GAR and CAL_GBR registers are automatically updated. The calibration function also updates GCC0[GAIN_CAL] (associated with A-side calibration) and GCC1[GAIN_CAL] (associated with B-side calibration). For A-side calibration, a software calculation is required to derive GCR0[GCALR] from GCC0[GAIN_CAL]. For B-side calibration, a corresponding calculation is required to derive GCR1[GCALR] from GCC1[GAIN_CAL].

To complete calibration setup:

1. Execute Offset calibration steps. The Offset Trim Register (OFSTRIM) is used during calibration to trim for comparator offset voltage.
2. Configure the averaging via CTRL[CAL_AVGS].
3. Initiate the calibration routine by writing 1 to CTRL[CAL_REQ]. CTRL[CAL_REQ] remains 1 until the CAL routine has been accepted by the ADC. After acceptance, CTRL[CAL_REQ] automatically becomes 0.
4. The A-side calibration data is updated first. Poll the GCR0[RDY] flag. When it is asserted, the hardware controlled A-side calibration operation is complete, and CAL_GAR and GCC0[GAIN_CAL] registers are updated. The updated value in GCC0[GAIN_CAL] is required for further software processing described in the following steps.
5. Read GCC0[GAIN_CAL] and store for use in the gain_adjustment calculation.
6. Calculate the A-side gain_adjustment: (131072)/(131072-GCC0[GAIN_CAL]). GCC0[GAIN_CAL] is a 16-bit unsigned value. This calculation results in a floating point value between 1 and 2.

7. The integer value of the gain_adjustment calculation is always 1 and can be discarded. The fractional component must be stored to GCR0[GCALR]. Write the fractional component value to GCR0[GCALR].
8. Execute the same calculation for the B-side converter. Poll the GCC1[RDY] flag. When it is asserted, the hardware controlled B-side calibration operation is complete, and CAL_GBR and GCC1[GAIN_CAL] registers are updated. The updated value in GCC1[GAIN_CAL] is needed for further software processing.
9. Read the GCC1[GAIN_CAL] register and store for use in the gain_adjustment calculation.
10. Calculate the B-side gain_adjustment = (131072)/(131072-GCC1[GAIN_CAL]). Formatting for the value is the same as the A-side.
11. Like the A-side, round the fractional component of the B-side gain_adjustment to 16-bits and store it in GCR1[GCALR].
12. Once GCR0[GCALR] and GCR1[GCALR] contain the results from the gain_adjustment calculations, set the GCR0[RDY] and GCR1[RDY] flags to indicate they are valid. It is acceptable for the GCRn[GCALR] and corresponding GCRn[RDY] field to be updated on the same write cycle.

After completing the steps above, the calibration sequence is complete and the STAT[CAL_RDY] flag is set. The STAT[CAL_RDY] flag remains set until the user resets the system or requests a new calibration sequence.

When STAT[CAL_RDY] is set, ADC is configured to run in calibrated mode. Each conversion uses a combination of linearity and gain calibration results to correct SAR data. Calibration conversion latency is required to process each sample. However, due to the pipelined nature of data and control sequences, each conversion can still be initiated without experiencing this calibration delay.

40.5.1.3 Calibration General A-Side and B-Side Widths

The general calibration value registers CAL_GARn and CAL_GBRn have non-uniform widths. The following table defines the bit widths of each register.

Table 250. Calibration General Widths

Element CAL_GxR[N]	Width (Bits)
N = 00h, 20h	11
N = 01h	12
N = 02h–03h	13
N = 0x04–07h	14
N = 0x08–0Fh	15
N = 0x10–1Fh	16

These registers are typically updated automatically during the self-calibration sequence. To reduce the latency associated with ADC setup, the CAL_GxR values from a calibration sequence can be stored in non-volatile memory after an initial calibration. These values can then be written to the CAL_GxR registers via software prior to the first ADC conversion. If these registers are set to values not generated by the calibration function, the linearity error specifications may not be met.

NOTE

These values can only be written in a single access, byte accesses are not supported.

40.6 ADC register descriptions

This section describes the ADC registers.

40.6.1 ADC memory map

ADC0 base address: 4004_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0200_2C0Bh
4h	Parameter Register (PARAM)	32	R	0F0F_1004h
10h	Control Register (CTRL)	32	RW	0000_0000h
14h	Status Register (STAT)	32	RW	0000_0000h
18h	Interrupt Enable Register (IE)	32	RW	0000_0000h
1Ch	DMA Enable Register (DE)	32	RW	0000_0000h
20h	Configuration Register (CFG)	32	RW	0080_0000h
24h	Pause Register (PAUSE)	32	RW	0000_0000h
34h	Software Trigger Register (SWTRIG)	32	RW	0000_0000h
38h	Trigger Status Register (TSTAT)	32	RW	0000_0000h
40h	Offset Trim Register (OFSTRIM)	32	RW	0000_0000h
A0h - ACh	Trigger Control Register (TCTRL0 - TCTRL3)	32	RW	0000_0000h
E0h - E4h	FIFO Control Register (FCTRL0 - FCTRL1)	32	RW	0000_0000h
F0h - F4h	Gain Calibration Control (GCC0 - GCC1)	32	R	0000_0000h
F8h - FCh	Gain Calculation Result (GCR0 - GCR1)	32	RW	0000_0000h
100h	Command Low Buffer Register (CMDL1)	32	RW	0000_0000h
104h	Command High Buffer Register (CMDH1)	32	RW	0000_0000h
108h	Command Low Buffer Register (CMDL2)	32	RW	0000_0000h
10Ch	Command High Buffer Register (CMDH2)	32	RW	0000_0000h
110h	Command Low Buffer Register (CMDL3)	32	RW	0000_0000h
114h	Command High Buffer Register (CMDH3)	32	RW	0000_0000h
118h	Command Low Buffer Register (CMDL4)	32	RW	0000_0000h
11Ch	Command High Buffer Register (CMDH4)	32	RW	0000_0000h
120h	Command Low Buffer Register (CMDL5)	32	RW	0000_0000h
124h	Command High Buffer Register (CMDH5)	32	RW	0000_0000h
128h	Command Low Buffer Register (CMDL6)	32	RW	0000_0000h
12Ch	Command High Buffer Register (CMDH6)	32	RW	0000_0000h
130h	Command Low Buffer Register (CMDL7)	32	RW	0000_0000h
134h	Command High Buffer Register (CMDH7)	32	RW	0000_0000h
138h	Command Low Buffer Register (CMDL8)	32	RW	0000_0000h
13Ch	Command High Buffer Register (CMDH8)	32	RW	0000_0000h
140h	Command Low Buffer Register (CMDL9)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
144h	Command High Buffer Register (CMDH9)	32	RW	0000_0000h
148h	Command Low Buffer Register (CMDL10)	32	RW	0000_0000h
14Ch	Command High Buffer Register (CMDH10)	32	RW	0000_0000h
150h	Command Low Buffer Register (CMDL11)	32	RW	0000_0000h
154h	Command High Buffer Register (CMDH11)	32	RW	0000_0000h
158h	Command Low Buffer Register (CMDL12)	32	RW	0000_0000h
15Ch	Command High Buffer Register (CMDH12)	32	RW	0000_0000h
160h	Command Low Buffer Register (CMDL13)	32	RW	0000_0000h
164h	Command High Buffer Register (CMDH13)	32	RW	0000_0000h
168h	Command Low Buffer Register (CMDL14)	32	RW	0000_0000h
16Ch	Command High Buffer Register (CMDH14)	32	RW	0000_0000h
170h	Command Low Buffer Register (CMDL15)	32	RW	0000_0000h
174h	Command High Buffer Register (CMDH15)	32	RW	0000_0000h
200h - 238h	Compare Value Register (CV1 - CV15)	32	RW	0000_0000h
300h - 304h	Data Result FIFO Register (RESFIFO0 - RESFIFO1)	32	R	0000_0000h
400h	Calibration General A-Side Registers (CAL_GAR0)	32	RW	0000_0000h
404h	Calibration General A-Side Registers (CAL_GAR1)	32	RW	0000_0000h
408h - 40Ch	Calibration General A-Side Registers (CAL_GAR2 - CAL_GAR3)	32	RW	0000_0000h
410h - 41Ch	Calibration General A-Side Registers (CAL_GAR4 - CAL_GAR7)	32	RW	0000_0000h
420h - 43Ch	Calibration General A-Side Registers (CAL_GAR8 - CAL_GAR15)	32	RW	0000_0000h
440h - 47Ch	Calibration General A-Side Registers (CAL_GAR16 - CAL_GAR31)	32	RW	0000_0000h
480h	Calibration General A-Side Registers (CAL_GAR32)	32	RW	0000_0000h
500h	Calibration General B-Side Registers (CAL_GBR0)	32	RW	0000_0000h
504h	Calibration General B-Side Registers (CAL_GBR1)	32	RW	0000_0000h
508h - 50Ch	Calibration General B-Side Registers (CAL_GBR2 - CAL_GBR3)	32	RW	0000_0000h
510h - 51Ch	Calibration General B-Side Registers (CAL_GBR4 - CAL_GBR7)	32	RW	0000_0000h
520h - 53Ch	Calibration General B-Side Registers (CAL_GBR8 - CAL_GBR15)	32	RW	0000_0000h
540h - 57Ch	Calibration General B-Side Registers (CAL_GBR16 - CAL_GBR31)	32	RW	0000_0000h
580h	Calibration General B-Side Registers (CAL_GBR32)	32	RW	0000_0000h

40.6.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

Indicates the version integrated for this instance on the device and the inclusion of optional features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	NUM_FIFO			NUM_SEC	CALOFSI	IADCKI	VR1RNGI	0	CSW			MVI	0	DIFFEN	RES
W																
Reset	0	0	1	0	1	1	0	0	0	0	0	0	1	0	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module specification.
15 —	Reserved
14-12 NUM_FIFO	Number of FIFOs Indicates the number of result FIFOs implemented in the design. 000b - N/A 001b - One 010b - Two 011b - Three 100b - Four

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 NUM_SEC	Number of Single-Ended Outputs Supported Indicates the number of single-ended channels which can be processed simultaneously. 0b - One 1b - Two
10 CALOFSI	Calibration Function Implemented Indicates whether ADC contains hardware calibration functions. When supported, CTRL[CAL_REQ] can be used to request the calibration routine. 0b - Not implemented 1b - Implemented
9 IADCKI	Internal ADC Clock Implemented Indicates whether this implementation of the ADC block includes an internal clock source. 0b - Not implemented 1b - Implemented
8 VR1RNGI	Voltage Reference 1 Range Control Bit Implemented Indicates whether a control bit is implemented to select the input voltage range on Voltage Reference Option 1. 0b - Range control not required. 1b - Range control required.
7 —	Reserved
6-4 CSW	Channel Scale Width Indicates whether channel scaling is supported. When supported, each command buffer has a control field (CMDLn[CSCALE]) for setting input scaling. 000b - Not supported. 001b - Supported with one-bit CSCALE control field. 110b - Supported with six-bit CSCALE control field.
3 MVI	Multiple Vref Implemented Indicates whether multiple voltage reference high (VREFH) inputs are supported. When multiple voltage references are supported, CFG[REFSEL] selects voltage reference high options. 0b - Single VREFH input supported. 1b - Multiple VREFH inputs supported.
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 DIFFEN	Differential Supported Indicates whether differential operation is supported. When supported, each command buffer has control fields (CMDLn[CTYPE]) for configuring for differential operation and expanding the number of supported channels from 32 to 64. 0b - Not supported 1b - Supported. CMDLn[CTYPE] controls fields implemented.
0 RES	Resolution Indicates the maximum accuracy supported. 0b - Up to 13-bit differential or 12-bit single-ended resolution supported. 1b - Up to 16-bit differential or 16-bit single-ended resolution supported. CMDLn[MODE] available for selecting the resolution of conversions for the associated command.

40.6.3 Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Function

The Parameter register indicates the size of several variable integration options for this instance on the device.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMD_NUM								CV_NUM							
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFOSIZE								TRIG_NUM							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 CMD_NUM	Command Buffer Number Indicates number of command buffers implemented.
23-16 CV_NUM	Compare Value Number Indicates number of compare value registers implemented.
15-8 FIFOSIZE	Result FIFO Depth Indicates the maximum number of conversion data words that can be stored in the result FIFO before an overflow occurs. 0000_0001b - 2 0000_0100b - 4 0000_1000b - 8 0001_0000b - 16 0010_0000b - 32 0100_0000b - 64
7-0 TRIG_NUM	Trigger Number Number of Triggers implemented.

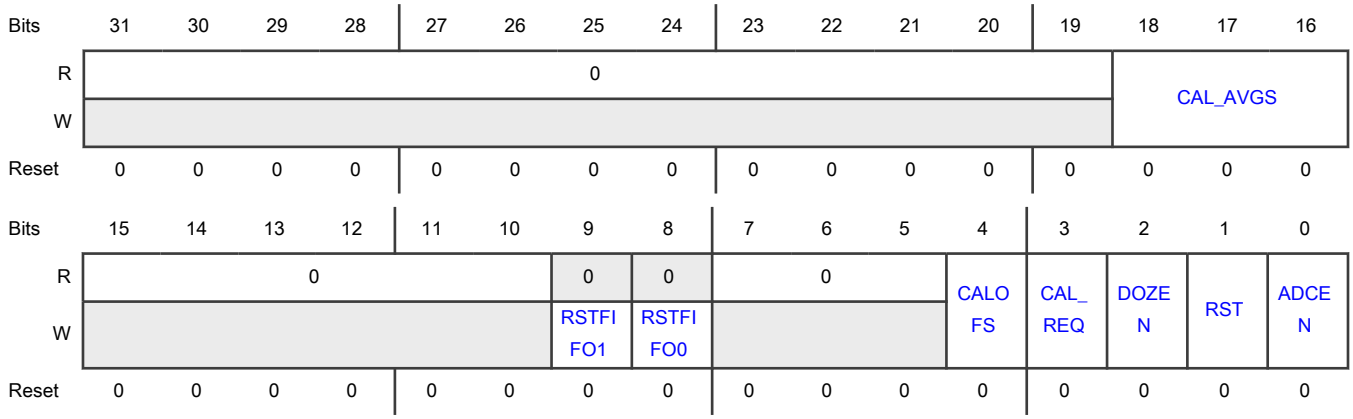
40.6.4 Control Register (CTRL)

Offset

Register	Offset
CTRL	10h

Function
Includes primary control bits.

Diagram



Fields

Field	Function
31-19 —	Reserved
18-16 CAL_AVGS	Auto-Calibration Averages Selects how many ADC conversions are averaged to calculate each calibration value. Selecting a higher number of averages will lead to more accurate conversions after completing calibration. The CAL_AVGS bit field applies to both CALOFS and CAL_REQ calibration types. This value should be fixed when requesting and running calibration with CTRL[CAL_REQ] or CTRL[CALOFS]. 000b - Single conversion. 001b - 2 conversions averaged. 010b - 4 conversions averaged. 011b - 8 conversions averaged. 100b - 16 conversions averaged. 101b - 32 conversions averaged. 110b - 64 conversions averaged. 111b - 128 conversions averaged.
15-10 —	Reserved
9 RSTFIFO1	Reset FIFO 1 0b - No effect. 1b - FIFO 1 is reset.
8 RSTFIFO0	Reset FIFO 0 0b - No effect. 1b - FIFO 0 is reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 CALOFS	Offset Calibration Request Indicates whether a request for a hardware calibration calculation has been made. Writing 1 to this field initiates a hardware offset calibration calculation. Any conversions in progress are completed before launching the offset calibration function. After being accepted, ADC calculates the OFSTRIM[OFSTRIM] value and updates the OFSTRIM register automatically. This field becomes 0 upon completion of the offset calibration calculation. STAT[CAL_RDY] indicates when the offset calibration routine has completed. 0b - Calibration function disabled 1b - Request for offset calibration function
3 CAL_REQ	Auto-Calibration Request Indicates whether a request for the hardware calibration routine has been made. Automatically becomes 0 when the system accepts the hardware calibration request. STAT[CAL_RDY] indicates when this calibration routine has been completed. 0b - No request made. 1b - Request has been made.
2 DOZEN	Doze Enable Controls system transition to Deep-sleep/ and Sleep power modes while ADC is converting. When 0, immediate entries to Deep-sleep or Sleep modes are allowed and ADC conversion functions remain enabled. Any conversion in progress is not disrupted. The selected clock source provided from the on-chip clock source must be able to continue operating. When 1, the ADC waits for the current averaging iteration or FIFO storage to complete before acknowledging the low-power entry request. After entering the low-power state, ADC is kept inactive until the system exits the low-power state. When the system enters Deep Power-down mode, the DOZEN bit is ignored and ADC waits for the current transfer to complete any pending operation. 0b - ADC is enabled in low-power mode. 1b - ADC is disabled in low-power mode.
1 RST	Software Reset Resets all internal logic and registers, except the CTRL register. Remains 1 until cleared by software. 0b - ADC logic is not reset. 1b - ADC logic is reset.
0 ADCEN	ADC Enable 0b - Disabled 1b - Enabled

40.6.5 Status Register (STAT)

Offset

Register	Offset
STAT	14h

Function

Provides the status of the ADC module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CMDACT				0						TRGACT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ADC_ACT...	CAL_RDY	TCOMP_I...	TEXC_INT	0				FOF1	RDY1	FOF0	RDY0
W							W1C	W1C					W1C		W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 CMDACT	Command Active Indicates the command actively being processed. Commands are only shown here when they are actively being processed by the SAR routine. Use with STAT[ADC_ACTIVE] to determine which conversion is running. 0000b - No command currently in progress. 0001b - Command 1 currently being executed. 0010b - Command 2 currently being executed. 0011b-1111b - Associated command number currently being executed.
23-18 —	Reserved
17-16 TRGACT	Trigger Active

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates the trigger actively being processed. This field can be used to determine which trigger is running through a trigger delay or being converted by the SAR routine. The command associated with this field is running through a conversion when STAT[CMDACT] is not zero. 00b - Command (sequence) associated with Trigger 0 currently being executed. 01b - Command (sequence) associated with Trigger 1 currently being executed. 10b - Command (sequence) associated with Trigger 2 currently being executed. 11b - Command (sequence) associated with Trigger 3 currently being executed.
15-12 —	Reserved
11 ADC_ACTIVE	ADC Active Indicates whether the module is processing a conversion, or has pending triggers to service. 0b - ADC is idle. There are no pending triggers to service and no active commands are being processed. 1b - ADC is processing a conversion, running through the power-up delay, or servicing a trigger.
10 CAL_RDY	Calibration Ready Indicates whether the ADC request for calibration (CTRL[CAL_REQ] or CTRL[CALOFS]) has been completed and the results are ready. This flag is automatically cleared when a new hardware calibration or OFSTRIM request is made. 0b - Calibration is incomplete or has not been run. 1b - ADC is calibrated.
9 TCOMP_INT	Interrupt Flag For Trigger Completion Indicates when a trigger sequence has been completed (all associated commands have been run). IE[TCOMP_IE] must be 1 for the specific trigger source to flag an interrupt upon completion. 0b - Either IE[TCOMP_IE] = 0, or no trigger sequences have run to completion. 1b - Trigger sequence has been completed and all data is stored in the associated FIFO.
8 TEXC_INT	Interrupt Flag For High-Priority Trigger Exception Indicates when a high-priority trigger exception has occurred and CFG[TRES] = 0. This flag only asserts if trigger exception interrupts are enabled (IE[TEXC_IE] = 1h). This flag can be used with TSTAT[TEXC_NUM] to resolve which trigger source was interrupted. 0b - No trigger exceptions have occurred. 1b - A trigger exception has occurred and is pending acknowledgment.
7-4 —	Reserved
3	Result FIFO1 Overflow Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
FOF1	Indicates that more data has been written to the Result FIFO 1 than it can hold. The newer data is not stored and the FIFO holds the original contents. This flag asserts regardless of the value of IE[FOFIE1]. However, an interrupt request is issued only if IE[FOFIE1] is 1. 0b - No result FIFO1 overflow has occurred since the last time that the flag was cleared. 1b - At least one result FIFO1 overflow has occurred since the last time that the flag was cleared.
2 RDY1	Result FIFO1 Ready Flag Indicates when the number of valid data words in the result FIFO 1 is greater than the level set in FCTRL0[FWMARK]. This flag asserts regardless of the value of IE[FWMIE1]. However, an interrupt request or DMA request occurs only when the associated control bit (IE[FWMIE1] and DE[FWMDE1]) is set. This flag is cleared when FCTRL0[FCOUNT] (which decrements on each read of the RESFIFO register) is less than or equal to the level set in FCTRL0[FWMARK]. 0b - Not above watermark 1b - Above watermark
1 FOF0	Result FIFO 0 Overflow Flag Indicates that more data has been written to the Result FIFO 0 than it can hold. The newer data is not stored and the FIFO holds the original contents. This flag asserts regardless of the value of IE[FOFIE0]. However, an interrupt request is issued only if IE[FOFIE0] is 1. 0b - No result FIFO 0 overflow has occurred since the last time that the flag was cleared. 1b - At least one result FIFO 0 overflow has occurred since the last time that the flag was cleared.
0 RDY0	Result FIFO 0 Ready Flag Indicates when the number of valid data words in the result FIFO 0 is greater than the watermark set in FCTRL0[FWMARK]. This flag asserts regardless of the value of IE[FWMIE0]. However, an interrupt request or DMA request occurs only when the associated control field (IE[FWMIE0] and DE[FWMDE0]) is 1. This flag is cleared when the FCTRL0[FCOUNT] (which decrements on each read of the RESFIFO register) is less than or equal to the level set in FCTRL0[FWMARK]. 0b - Not above watermark 1b - Above watermark

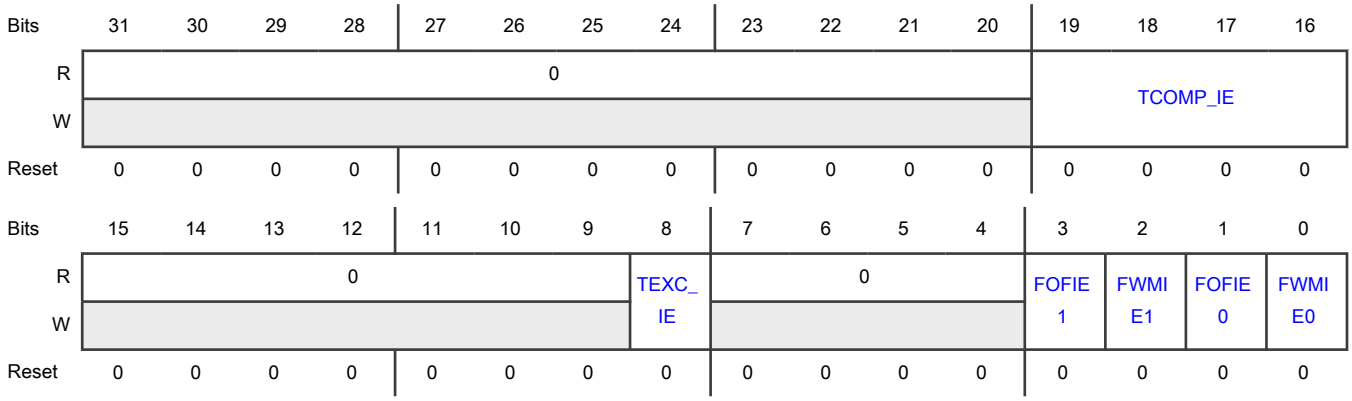
40.6.6 Interrupt Enable Register (IE)

Offset

Register	Offset
IE	18h

Function
Includes system interrupt masking control bits.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 TCOMP_IE	Trigger Completion Interrupt Enable Enables generation of interrupt requests to indicate when complete trigger command sequences are executed. Each bit in TCOMP_IE corresponds to a trigger source. (TCOMP_IE[0] corresponds to trigger 0, and so on.) All results are stored in the FIFO when a sequence is considered complete. 0000b - All disabled 0001b - Trigger completion interrupts are enabled for trigger source 0 only. 0010b - Trigger completion interrupts are enabled for trigger source 1 only. 0011b-1110b - Associated trigger completion interrupts are enabled. 1111b - All enabled
15-9 —	Reserved
8 TEXC_IE	Trigger Exception Interrupt Enable Enables ADC to assert an interrupt request when a high-priority trigger exception occurs. TSTAT[TEXC_NUM] contains the value of the corresponding trigger affected by the exception. 0b - Disabled 1b - Enabled
7-4 —	Reserved
3 FOFIE1	Result FIFO1 Overflow Interrupt Enable Enables generation of overflow interrupt requests when FOF1 flag is asserted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
2 FWMIE1	FIFO1 Watermark Interrupt Enable Enables generation of watermark interrupt requests when STAT[RDY1] flag is asserted. 0b - Disabled 1b - Enabled
1 FOFIE0	Result FIFO 0 Overflow Interrupt Enable Enables generation of overflow interrupt requests when FOF flag is asserted. 0b - Disabled 1b - Enabled
0 FWMIE0	FIFO 0 Watermark Interrupt Enable Enables generation of watermark interrupt requests when STAT[RDY0] flag is asserted. 0b - Disabled 1b - Enabled

40.6.7 DMA Enable Register (DE)

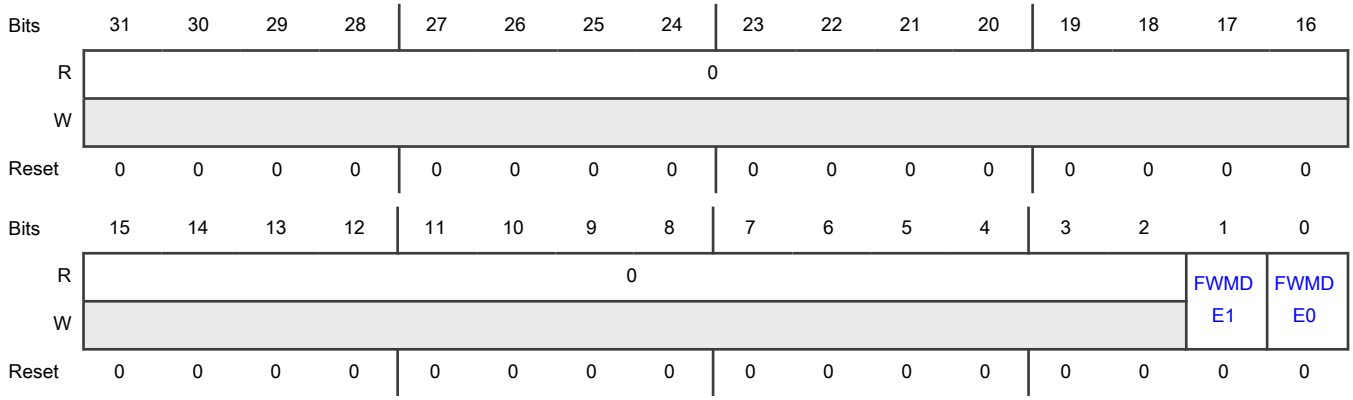
Offset

Register	Offset
DE	1Ch

Function

Includes DMA request-masking control bits.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 FWMDE1	FIFO1 Watermark DMA Enable Enables generation of DMA requests when STAT[RDY1] flag is asserted. 0b - Disabled 1b - Enabled
0 FWMDE0	FIFO 0 Watermark DMA Enable Enables generation of DMA requests when STAT[RDY0] flag is asserted. 0b - Disabled 1b - Enabled

40.6.8 Configuration Register (CFG)

Offset

Register	Offset
CFG	20h

Function

Controls ADC functions common to all commands. When CTRL[ADCEN] = 1, this register cannot be changed and writes to this register are ignored.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	PWREN	0				PUDLY							
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0			HPT_EXDI	TCMDRES	TRES		REFSEL		PWRSEL		0		TPRICTRL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 PWREN	ADC Analog Pre-Enable Enables the ADC analog circuits. When setting this field, user code should delay for a period exceeding the analog startup time of $t_{ADCSTUP}$ before enabling ADC for operation. The module is still operational even when this field is 0, but command execution start is delayed for a period defined by CFG[PUDLY]. See the device data sheet for ADC idle and active power consumption parameters. 0b - ADC analog circuits are only enabled while conversions are active. Analog startup delays affect performance. 1b - ADC analog circuits are pre-enabled and ready to execute conversions without startup delays, at the cost of higher DC current consumption. A single power-up delay (CFG[PUDLY]) is executed immediately once PWREN is set. No detected triggers begin ADC operation until the power-up delay time has passed. After this initial delay expires, the analog circuits remain pre-enabled, and no additional delays are executed.
27-24 —	Reserved
23-16 PUDLY	Power-up Delay Defines the power-up delay executed after an initial trigger wakes ADC from its idle state. Must be programmed to a non-zero value to enable the ADC. Depending on the value of CFG[PWREN], the delay is executed in one of two modes: • When CFG[PWREN] = 0, the ADC analog circuits are only powered on while the module is active. The power-up delay allows time for the analog circuits to stabilize after being powered on. The startup delay count of (PUDLY * 4) ADCK cycles must result in a longer delay than the analog startup time of $t_{ADCSTUP}$. Accuracy of the initial conversions after activation is degraded if CFG[PUDLY] is set to too small a value. After active conversions, if no subsequent conversions are pending, the ADC analog circuits are automatically reverted to their low-power idle state. When ADC is awakened with a later trigger, the power-up delay runs again. • When CFG[PWREN] = 1 prior to ADC activation, the analog circuits are pre-enabled and the activation delay defined by CFG[PUDLY] is executed immediately. After the delay has been executed, the analog circuits remain enabled regardless of ADC activity. This configuration begins conversions immediately after trigger event detection, at the cost of increased DC power consumption in the analog circuits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE ADC does not begin an initial conversion until the power-up delay has executed, regardless of the value of CFG[PWREN].
15 —	Reserved
14-11 —	Reserved
10 HPT_EXDI	High-Priority Trigger Exception Disable Disables high-priority trigger exceptions. See Trigger detect and command execution for a detailed description on trigger exception handling. When 1, exceptions are disabled and CFG[TCMDRES], CFG[TRES], and CFG[TPRICTRL] are ignored. 0b - Enabled 1b - Disabled
9 TCMDRES	Trigger Command Resume Determines where a trigger sequence resumes when interrupted by a high-priority trigger exception. To use this field, CFG[TRES] must be 1. 0b - Trigger sequence automatically restarted. 1b - Trigger sequence resumed from the command that was executed prior to the exception.
8 TRES	Trigger Resume Enable Determines whether trigger sequences interrupted by a high-priority exception are automatically resumed or restarted. If 1, the sequence can be resumed along command boundaries or restarted from the beginning of a sequence, as determined by CFG[TCMDRES]. If 0, interrupted triggers can be resumed via software by monitoring STAT[TEXC_INT] (or via ISR handling for trigger exception interrupt when IE[TEXC_IE] = 1). Software determines which trigger was interrupted by reading TSTAT[TEXC_NUM] and restarts the trigger by writing 1 to the corresponding field in the SWTRIG register. 0b - Not automatically resumed or restarted 1b - Automatically resumed or restarted
7-6 REFSEL	Voltage Reference Selection Selects the voltage reference high used for conversions. **NOTE** See the chip-specific ADC information for voltage reference options specific to this packaged device.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Option 1 01b - Option 2 10b - Option 3 11b - Reserved
5-4 PWRSEL	Power Configuration Select Configures the module for power and performance. In the high-power setting, the highest conversion rates are possible. See the device data sheet for power and performance capabilities for each setting. 0xb - Low power 1xb - High power
3-2 —	Reserved
1-0 TPRICTRL	ADC Trigger Priority Control Controls how higher-priority trigger exceptions are handled when they are received during command processing. See Trigger detect and command execution for a detailed explanation of trigger event handling. 00b - Current conversion is aborted and the new command specified by the trigger is started. 01b - Current command is stopped after completing the current conversion. If averaging is enabled, the averaging loop is completed. CMDHn[LOOP] is ignored and the higher-priority trigger is serviced. 10b - Current command is completed (averaging, looping, compare) before servicing the higher-priority trigger. 11b - Reserved

40.6.9 Pause Register (PAUSE)

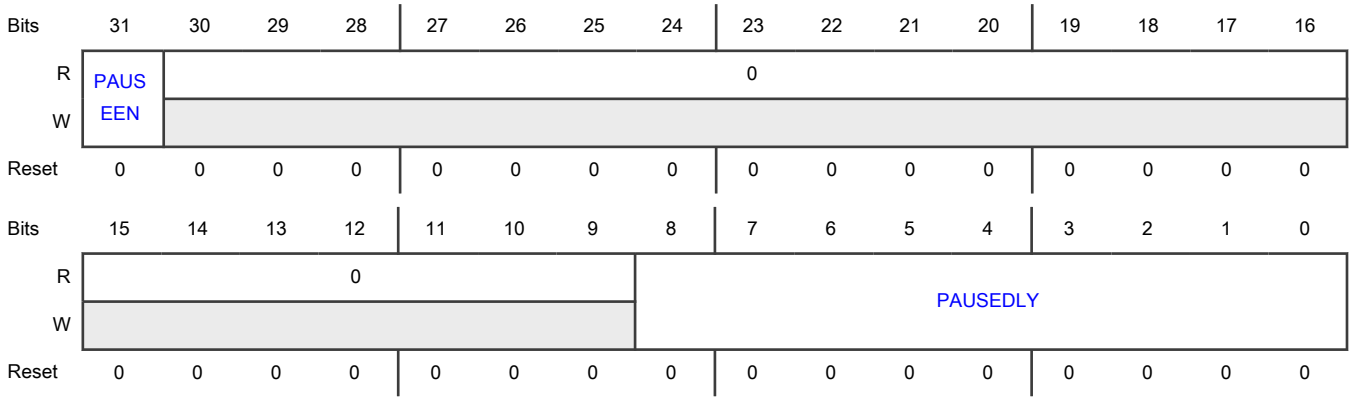
Offset

Register	Offset
PAUSE	24h

Function

Controls the optional inserted delay between conversions. When CTRL[ADCEN] = 1, this register should not be modified.

Diagram



Fields

Field	Function
31 PAUSEEN	Pause Enable Enables the ADC pausing function. When enabled, a programmable delay is inserted during command execution sequencing: • Between LOOP iterations. • Between commands in a sequence. • Between conversions, when a command is executing in compare-until-true configuration. **NOTE** CMDHn[WAIT_TRIG] and PAUSE are mutually exclusive. CMDHn[WAIT_TRIG] takes priority over PAUSE between commands when both are enabled. 0b - Disabled 1b - Enabled
30-9 —	Reserved
8-0 PAUSEDLY	Pause Delay Controls the duration of pauses during command execution sequencing when PAUSE[PAUSEEN] = 1. The pause delay is a count of (PAUSEDLY * 4) ADCK cycles.

40.6.10 Software Trigger Register (SWTRIG)

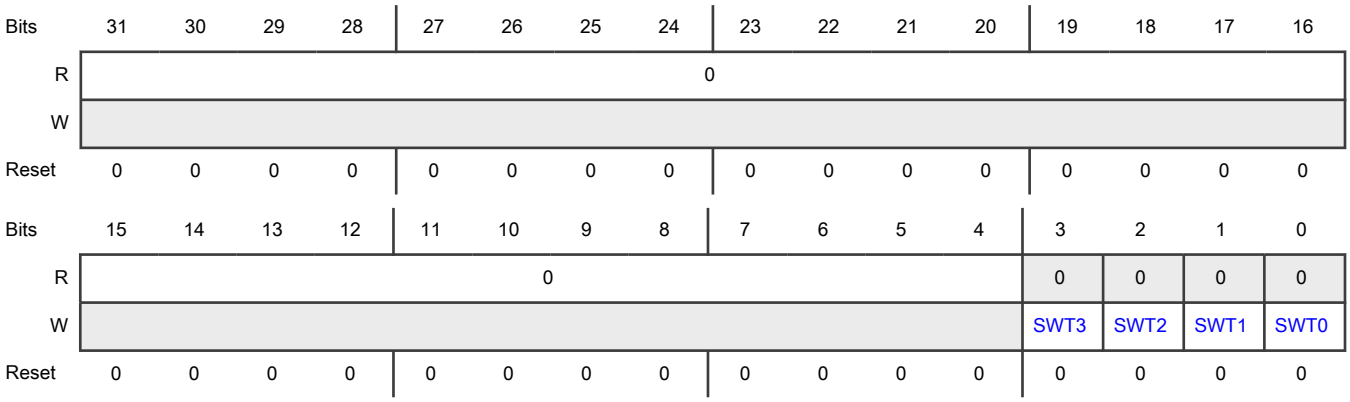
Offset

Register	Offset
SWTRIG	34h

Function

Initiates software-triggered conversions when written to. Writes to this register are ignored while CTRL[ADCEN] = 0. There is an approximately three ADC-clock-cycle synchronization delay between asserting CTRL[ADCEN] and when SWTRIG can be accepted.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 SWT3	Software Trigger 3 Write 1 to SWT3 generates a trigger 3 event. Writing 1 to SWT3 is ignored while the trigger 3 event is being serviced or is pending. 0b - No trigger 3 event generated. 1b - Trigger 3 event generated.
2 SWT2	Software Trigger 2 Write 1 to SWT2 generates a trigger 2 event. Writing 1 to SWT2 is ignored while the trigger 2 event is being serviced or is pending. 0b - No trigger 2 event generated. 1b - Trigger 2 event generated.
1 SWT1	Software Trigger 1 Write 1 to SWT1 generates a trigger 1 event. Writing 1 to SWT1 is ignored while the trigger 1 event is being serviced or is pending. 0b - No trigger 1 event generated. 1b - Trigger 1 event generated.
0 SWT0	Software Trigger 0 Generates a trigger 0 event. Writing 1 to this field is ignored while the trigger 0 event is being serviced or is pending.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No trigger 0 event generated. 1b - Trigger 0 event generated.

40.6.11 Trigger Status Register (TSTAT)

Offset

Register	Offset
TSTAT	38h

Function

Contains status flags to indicate when trigger sequences have been completed or interrupted by a high-priority trigger exception. Each field in this register is set by hardware and cleared by software.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												TCOMP_FLAG			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												TEXC_NUM			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 TCOMP_FLAG	Trigger Completion Flag Indicates which triggers have been completed. Each bit in this field corresponds to a trigger. When a trigger sequence has completed, the corresponding bit in TCOMP_FLAG is set. For example, if trigger 3 has been completed, then bit 19 is set in the TSTAT register. This register is active only if the corresponding bit in IE[TCOMP_IE] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE A synchronization delay may be added to the end of the final conversion in a sequence, prior to this flag being set. This delay can range from two to four ADC CLK cycles. 0000b - No triggers have been completed. Trigger completion interrupts are disabled. 0001b - Trigger 0 has been completed and trigger 0 has enabled completion interrupts. 0010b - Trigger 1 has been completed and trigger 1 has enabled completion interrupts. 0011b-1110b - Associated trigger sequence has completed and has enabled completion interrupts. 1111b - Every trigger sequence has been completed and every trigger has enabled completion interrupts.
15-4 —	Reserved
3-0 TEXC_NUM	Trigger Exception Number Indicates which triggers have been interrupted by exceptions. Each bit in this field corresponds to a trigger. When the corresponding trigger sequence is interrupted by a high-priority trigger exception, the corresponding bit is set. For example, if trigger 3 has been interrupted, then bit 3 is set in this register. This register is active regardless of the value in IE[TEXC_IE]. 0000b - No triggers have been interrupted by a high-priority exception. Or CFG[TRES] = 1. 0001b - Trigger 0 has been interrupted by a high-priority exception. 0010b - Trigger 1 has been interrupted by a high-priority exception. 0011b-1110b - Associated trigger sequence has interrupted by a high-priority exception. 1111b - Every trigger sequence has been interrupted by a high-priority exception.

40.6.12 Offset Trim Register (OFSTRIM)

Offset

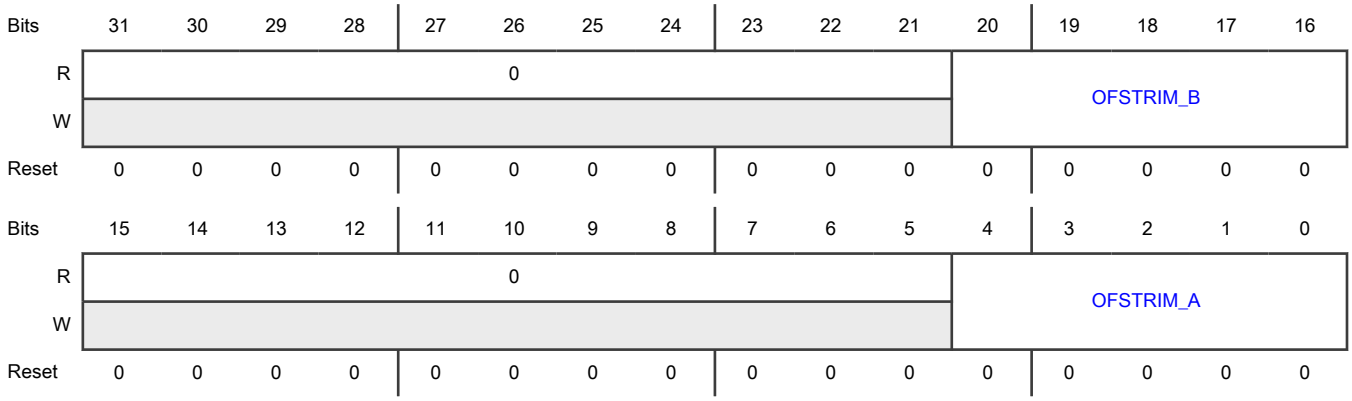
Register	Offset
OFSTRIM	40h

Function

Used to trim for offset. ADC supports a calibration step in which the ADC determines the value needed in the OFSTRIM register.

To determine the value to put in the OFSTRIM register, write 1 to CTRL[CALOFS]. This setting automatically begins a sequence to calculate this value. Once the sequence has completed, the OFSTRIM register is updated with a signed value between -16 and 15. This value is used to minimize offset during normal operation.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 OFSTRIM_B	Trim for Offset OFSTRIM is a 5-bit signed value between -16 and 15. This value applies to the ADC B-side conversion results.
15-5 —	Reserved
4-0 OFSTRIM_A	Trim for Offset OFSTRIM_A is a 5-bit signed value between -16 and 15. This value applies to the ADC A-side conversion results.

40.6.13 Trigger Control Register (TCTRL0 - TCTRL3)

Offset

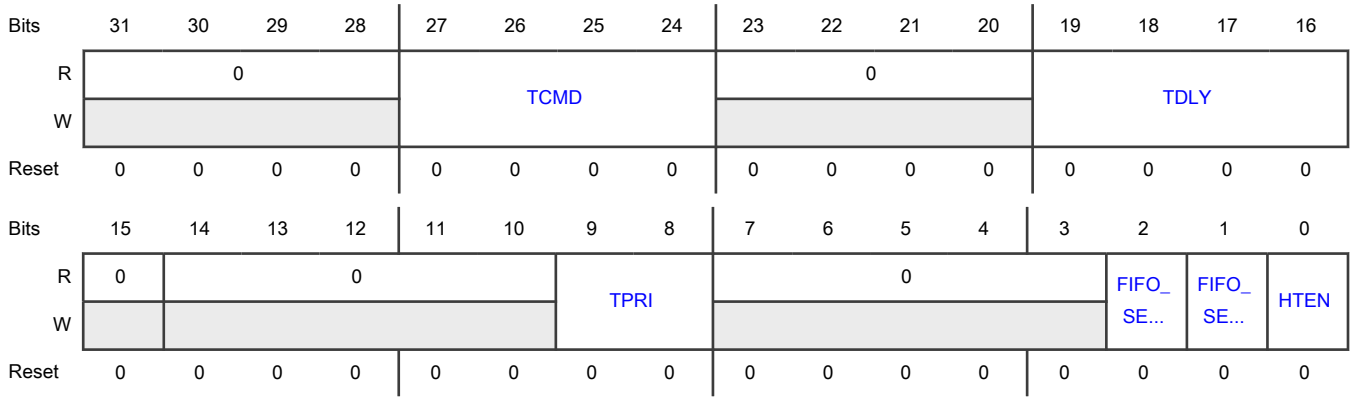
Register	Offset
TCTRL0	A0h
TCTRL1	A4h
TCTRL2	A8h
TCTRL3	ACh

Function

Implements control fields associated with each trigger source. When ADC is executing commands, only one TCTRLn register controls ADC conversions.

Do not update the controlling TCTRLn register while ADC is active. Writing to a TCTRLn register while that trigger control register controls the ADC operation may cause unpredictable behavior.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 TCMD	Trigger Command Select Selects the command from the command buffer to execute upon detection of the associated trigger event. When a higher-priority trigger is received while ADC is converting, the command in this register associated with the new trigger is executed. This execution is done under the control of CFG[TPRICTRL]. See Trigger detect and command execution for details about the relationship between CFG[TPRICTRL] and TCMD. 0000b - Not a valid selection from the command buffer. Trigger event is ignored. 0001b - CMD1 0010b-1110b - Corresponding CMD is executed 1111b - CMD15
23-20 —	Reserved
19-16 TDLY	Trigger Delay Select Selects the trigger delay duration for the start of servicing a trigger event. Each trigger source has an associated programmable delay prior to beginning an initial conversion. When this field is 0, no delay is incurred. When this field is set to a non-zero value, the duration of the delay is 2^{TDLY} ADCK cycles.
15 —	Reserved
14-10 —	Reserved
9-8	Trigger Priority Setting

Table continues on the next page...

Table continued from the previous page...

Field	Function
TPRI	Sets the priority of the associated trigger source. If two or more triggers have the same priority level, the lower-order trigger event has the higher priority. If Trigger 0 and Trigger 1 are pending triggers and TCTRL0[TPRI] is configured the same as TCTRL1[TPRI], the Trigger 0 command is serviced first. 00b - Highest priority, Level 1 01b-10b - Set to corresponding priority level. 11b - Lowest priority, Level 4
7-3 —	Reserved
2 FIFO_SEL_B	SAR Result Destination for Channel B Indicates the FIFO to which SAR results are written for Channel B. This field is only used in dual single-ended mode (CMDLn[CTYPE] = 11b). In this mode, the SAR result from Channel B is written to the FIFO number specified by this field. See Sampling Modes for more information. 0b - FIFO 0 1b - FIFO 1
1 FIFO_SEL_A	SAR Result Destination for Channel A Indicates the FIFO to which SAR results are written for Channel A. This field provides the destination for all single-ended and differential conversions and for all A-side conversions in dual single-ended mode. Conversion results are stored in the FIFO number specified, under the following conditions: • Single-ended mode: CMDLn[CTYPE] = 00b or CMDLn[CTYPE] = 01b. • Differential mode: CMDLn[CTYPE] = 10b. • Dual single-ended mode: CMDLn[CTYPE] = 11b. 0b - FIFO 0 1b - FIFO 1
0 HTEN	Trigger Enable Enables hardware trigger source to initiate conversion on the rising edge of the input trigger source. **NOTE** Enabling the hardware trigger does not disable software triggers. 0b - Disabled 1b - Enabled

40.6.14 FIFO Control Register (FCTRL0 - FCTRL1)

Offset

Register	Offset
FCTRL0	E0h
FCTRL1	E4h

Function

Contains control and status fields for each FIFO in the design. A programmable watermark can be set for each FIFO, which can be used to trigger an interrupt. In addition, the number of entries stored in each FIFO can be monitored by reading FCTRLn[FCOUNT].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FWMARK			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 FWMARK	Watermark Level Selection Selects the storage threshold for the ADC Result FIFO. When the number of data words stored in the FIFO is greater than this value, the STAT[RDY0] flag is asserted. When IE[FWMIEn] = 1, an interrupt request is generated. When DE[FWMDEn] = 1, a DMA request is generated.
15-5 —	Reserved
4-0 FCOUNT	Result FIFO Counter Indicates the number of data words stored in the result FIFO. This value may be used with PARAM[FIFOSIZE] to calculate how much room is left in the result FIFO. This field is incremented with each storage of new data into the result FIFO and decrements with each read of the result FIFO. The FIFO is reset by writing to CTRL[RSTFIFOn], which initializes FCTRLn[FCOUNT] to 0h.

40.6.15 Gain Calibration Control (GCC0 - GCC1)

Offset

Register	Offset
GCC0	F0h
GCC1	F4h

Function

Holds intermediate values used as part of the calibration steps. GCC0 is associated with the A-side converter and GCC1 is associated with the B-side converter. GCCn[GAIN_CAL] is calculated with hardware and automatically updated during the calibration steps. Once the hardware calibration sequence has updated, GCCn[RDY] is asserted automatically.

Requesting a calibration routine automatically clears the GCCn[RDY] flag until the new GCCn[GAIN_CAL] value is calculated. To complete calibration, further software processing is needed, where GCCn[GAIN_CAL] is used to calculate the gain adjustment. The result is stored to GCRn[GCALR]. See Calibration functions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							RDY	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GAIN_CAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24 RDY	Gain Calibration Value Valid Indicates whether the data stored in GCCn[GAIN_CAL] is valid and should be used to derive GCRn[GCALR]. If the data in GCCn[GAIN_CAL] is invalid, you can run the hardware calibration routine to correct this issue. 0b - Invalid 1b - Valid
23-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 GAIN_CAL	Gain Calibration Value Indicates the calculated value of the gain calibration. As part of the hardware calibration steps, this field is automatically updated with a 16-bit unsigned number. It is used in the gain adjustment calculations to derive the value written to GCRn[GCALR]. See Calibration functions.

40.6.16 Gain Calculation Result (GCR0 - GCR1)

Offset

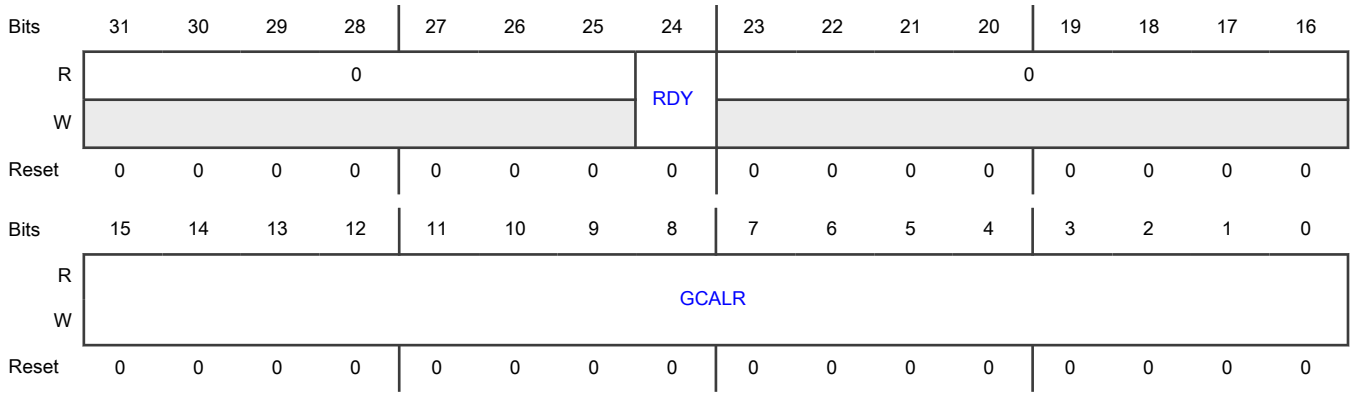
Register	Offset
GCR0	F8h
GCR1	FCh

Function

Used during ADC conversions for automated gain error adjustment. GCR0 is associated with the A-side converter and GCR1 is associated with the B-side converter. Determining the values to store to GCR0[GCALR] and GCR1[GCALR] is the result of software calculations described in setup steps in Calibration functions. Upon writing the GCRn[GCALR] value, the user should also write 1 to GCRn[RDY] to indicate that the gain adjustment result is valid.

A calibration sequence begins by writing 1 to CTRL[CAL_REQ], and the calibration sequence is not complete until GCRn[GCALR] is calculated and GCRn[RDY] is 1. The gain adjustment calculation produces a floating-point value between 1 and 2. GCRn[GCALR] holds the 16-bit fractional component of the gain adjustment calculation. To convert the GCRn[GCALR] value to the gain adjustment value in decimal format, use this formula: 1 + 0.5 * GCRn[15] + 0.25 * GCRn[14] + 0.125 * GCRn[13] + (and so on)

Diagram



Fields

Field	Function
31-25 —	Reserved
24 RDY	Gain Calculation Ready Indicates whether the data stored in GCRn[GCALR] is valid and is used for gain adjustment. GCRn[GCALR] must be written from memory or calculated each time the calibration routine is run. The user must write 1 to this GCRn[RDY] after writing valid data to GCRn[GCALR]. This field becomes 0 automatically when requesting a new calibration sequence. 0b - Invalid 1b - Valid
23-16 —	Reserved
15-0 GCALR	Gain Calculation Result Holds the 16-bit fractional component generated from the gain adjustment calculation during the auto-calibration routine.

40.6.17 Command Low Buffer Register (CMDL1 - CMDL15)

Offset

For a = 1 to 15:

Register	Offset
CMDLa	F8h + (a × 8h)

Function

Controls channel selection and conversion options. There are 15 command buffers (CMDn), each constructed from two 32-bit registers (CMDHn:CMDLn) that can be configured for different channel selection and varying conversion options. Any command buffer is selected and used as the controlling command by association with a trigger event via configuration of TCTRLn[TCMD]. When ADC is executing commands, only one of the CMD buffers controls ADC conversions. Do not update the controlling CMD buffer while ADC is active. A write to a CMD buffer while that CMD buffer controls the ADC operation may cause unpredictable behavior.

NOTE

The 15 command buffers are numbered [CMDH1:CMDL1] through [CMDH15:CMDL15]. In NXP-supplied header files, these buffers are likely to be defined as two 15-element arrays that are indexed from 0. For example, the type declaration would be:

```
unsigned int CMDH[15];
unsigned int CMDL[15];
```

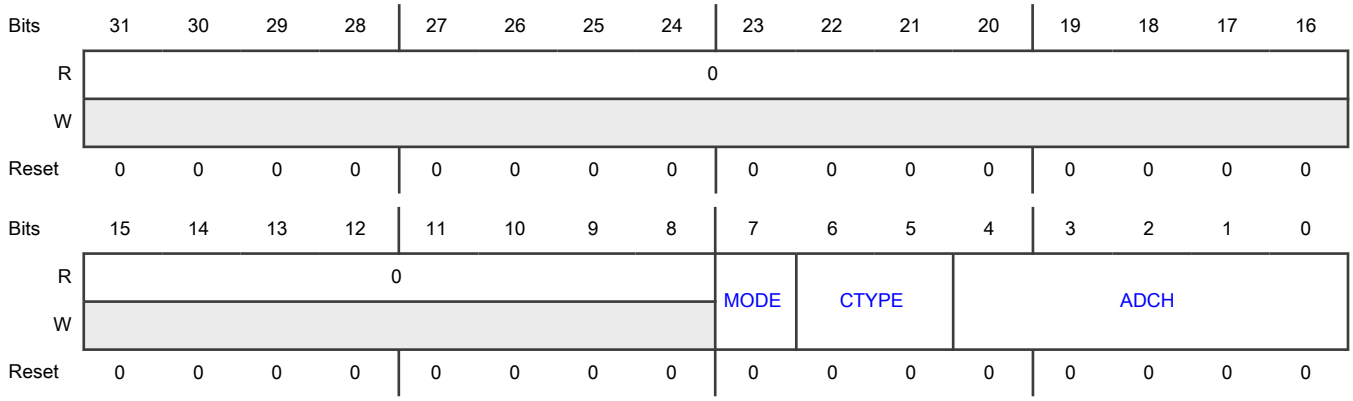
and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL15 as `ADC0->CMDL[14]`.

The CMDLn[ADCH] and CMDLn[CTYPE] fields control selection of paired or individual input channels. Each ADC command independently makes a channel and conversion type selection. Each ADCH channel selection has an associated A-side and B-side input. Each ADCH pair can be converted in differential mode, but only limited pairs should be converted as differential channels (for example, adjacent pins designed with matched impedance). For the pin pairings available for differential conversions for your device, see the Chip Configuration details.

NOTE

Some input channels are from on-chip sources such as temperature sensors and reference voltage sources and may only be connected to individual instances of ADC. Some input channel options in the field descriptions may not be available for your device. See the chip-specific information for the channels supported on this device.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 —	Reserved
7 MODE	Select Resolution of Conversions Selects the ADC resolution. 0b - Standard resolution. Single-ended 12-bit conversion; differential 13-bit conversion with 2's complement output. 1b - High resolution. Single-ended 16-bit conversion; differential 16-bit conversion with 2's complement output.
6-5 CTYPE	Conversion Type Chooses how a pair of A and B channels are converted. 00b - Single-Ended mode. Only A-side channel is converted. 01b - Single-Ended mode. Only B-side channel is converted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Differential mode. A-B. 11b - Dual-Single-Ended mode. Both A-side and B-side channels are converted independently.
4-0 ADCH	Input Channel Select Selects the input from one of the channel inputs or one of the input pairs, with CMDLn[CTYPE]. Each ADCH channel selection has an associated A-side and B-side input. **NOTE** Not all pairs should be converted as differential channels. See the chip-specific information for the pin pairings available for differential conversions for your device. **NOTE** Some input channels are from internal resources such as temperature sensors and band gap voltage sources and may only be connected to individual instances of ADC. Some input channel options in field descriptions might not be available for your device. See the chip-specific information for the ADC channel assignments for your device. 0_0000b - CH0A or CH0B or CH0A/CH0B pair. 0_0001b - CH1A or CH1B or CH1A/CH1B pair. 0_0010b - CH2A or CH2B or CH2A/CH2B pair. 0_0011b - CH3A or CH3B or CH3A/CH3B pair. 0_0100b-1_1101b - Select corresponding channel CHnA or CHnB or CHnA/CHnB pair. 1_1110b - CH30A or CH30B or CH30A/CH30B pair. 1_1111b - CH31A or CH31B or CH31A/CH31B pair.

40.6.18 Command High Buffer Register (CMDH1 - CMDH15)

Offset

For a = 1 to 15:

Register	Offset
CMDHa	FCh + (a × 8h)

Function

Controls channel selection and conversion options. There are 15 command buffers (CMDn), each constructed from two 32-bit registers (CMDHn:CMDLn) that can be configured for different channel selection and conversion options. Any command buffer is selected and used as the controlling command by association with a trigger event via configuration of TCTRLn[TCMD]. When ADC is executing commands, only one of the CMD buffers controls ADC conversions. Do not update the controlling CMD buffer while ADC is active. A write to a CMD buffer while that CMD buffer controls the ADC operation may cause unpredictable behavior.

NOTE

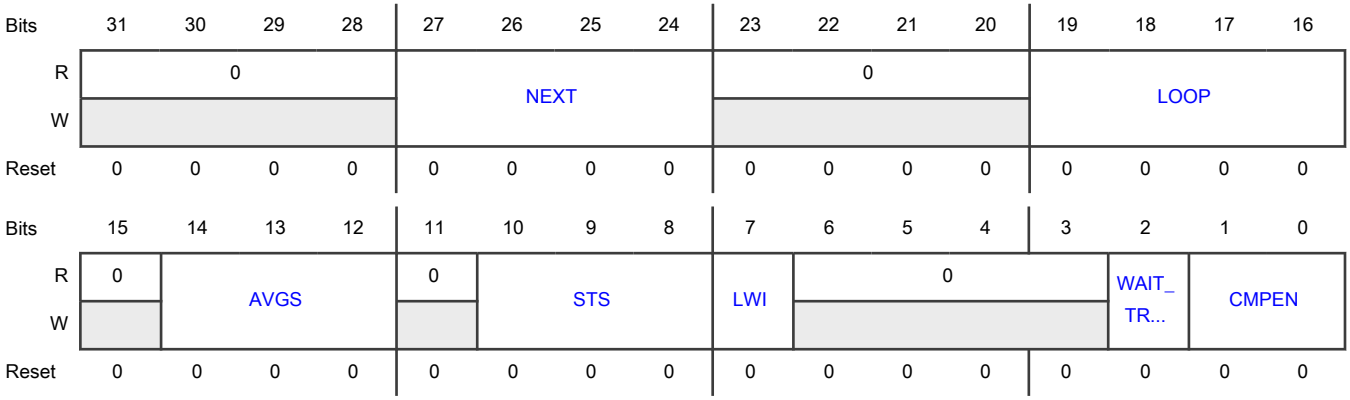
The 15 command buffers are numbered [CMDH1:CMDL1] through [CMDH15:CMDL15]. In NXP-supplied header files, these buffers are likely to be defined as two 15-element arrays that are indexed from 0. For example, the type declaration would be:

```
unsigned int CMDH[15];

unsigned int CMDL[15];
```

and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL15 as `ADC0->CMDL[14]`.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 NEXT	Next Command Select Selects the next command to be executed after this command completes. Multiple commands can be configured in a scan configuration by linking the next command in a daisy-chain sequence. The command buffer number is not indicative of any particular order. The order of execution is strictly controlled by this field. For example, a sequence of commands could be CMD2 to CMD1 to CMD3. Unending circular command execution can be configured by setting the NEXT field in the last command in a sequence to the first command in the sequence. It is also allowed for a command to set the next command to itself, resulting in a continuous conversion configuration. Setting the next command to 0h causes conversions to terminate at the completion of the command. Lower-priority trigger events cannot be serviced until a higher-priority triggered command (or sequence of commands) completes. 0000b - No next command defined. Terminate conversions at completion of current command. If lower priority trigger pending, begin command associated with lower priority trigger. 0001b - CMD1 0010b-1110b - Select corresponding CMD command buffer register as next command 1111b - CMD15
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-16 LOOP	Loop Count Select Selects the number of times that this command executes (and stores conversion result to RESFIFO) before finishing and transitioning to the next command or idle state. CMDHn[LWI] controls whether a single channel is converted on each iteration or an automatic channel increment results in channel scanning functionality. 0000b - Looping not enabled. Command executes one time. 0001b - Loop one time. Command executes two times. 0010b - Loop two times. Command executes three times. 0011b-1110b - Loop corresponding number of times. Command executes LOOP + 1 times. 1111b - Loop 15 times. Command executes 16 times.
15 —	Reserved
14-12 AVGS	Hardware Average Select Selects how many ADC conversions are averaged to create the ADC result (2^{AVGS}). An internal storage buffer captures temporary results while the averaging iterations are executed. Hardware averaging is a nested loop control and does not extend across LOOP boundaries. See Functional description for usage of AVGS, LOOP, and NEXT fields in command execution sequencing. 000b - Single conversion 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 128
11 —	Reserved
10-8 STS	Sample Time Select Selects the total sampling time. When this field contains a non-zero value, the sample time is ($3.5 + 2^{STS}$) ADCK cycles. The shortest sample time maximizes conversion speed for lower-impedance inputs. Extending sample time allows higher-impedance inputs to be sampled accurately. Longer sample times can lower overall power consumption when command looping and sequencing are configured and high conversion rates are not required. 000b - Minimum sample time of 3.5 ADCK cycles.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - 5.5 ADCK cycles. (3.5 + 2^1 ADCK cycles) 010b - 7.5 ADCK cycles. (3.5 + 2^2 ADCK cycles) 011b - 11.5 ADCK cycles. (3.5 + ^{3} ADCK cycles) 100b - 19.5 ADCK cycles. (3.5 + 2^4 ADCK cycles) 101b - 35.5 ADCK cycles. (3.5 + 2^5 ADCK cycles) 110b - 67.5 ADCK cycles. (3.5 + 2^6 ADCK cycles) 111b - 131.5 ADCK cycles. (3.5 + 2^7 ADCK cycles)
7 LWI	Loop with Increment Enables automatic channel incrementing. When 0, the LOOP field selects the number of times the selected channel is converted consecutively. When 1, automatic channel incrementing is enabled and the LOOP field defines how many consecutive channels are converted as part of the command execution. Example 1: LOOP = 8h, LWI = 0b, CMDLn[CTYPE] = 0h, CMDLn[ADCH] = Dh. Convert on channel 13A 9 times. Example 2: LOOP = 8h, LWI = 1b, CMDLn[CTYPE] = 0h, CMDLn[ADCH] = Dh. Run channels 13A-21A each one time. Maximum channel scanning using a single command buffer is defined by the maximum value of the LOOP field (16). 0b - Disabled 1b - Enabled
6-3 —	Reserved
2 WAIT_TRIG	Wait for Trigger Assertion Before Execution Selects whether commands execute automatically or a trigger must be received before execution. When 0, wait states are added before the command until the active trigger is asserted again. When WAIT_TRIG = 0, each command is automatically executed when called. 0b - Command executes automatically. 1b - Active trigger must be asserted again before executing this command.
1-0 CMPEN	Compare Function Enable Enables the automatic compare function. Selects whether to store only when the comparison operation is true, following ADC channel input sampling, conversion, and averaging. When the compare function is enabled, the conversion result is compared to the compare value registers (CVn[CVH] and CVn[CVL]). See Compare Function for details about the options for command sequencing related to the compare function. 00b - Disabled 01b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Enabled. Store on true. 11b - Enabled. Repeat channel acquisition (sample, convert, and compare) until true.

40.6.19 Compare Value Register (CV1 - CV15)

Offset

For a = 1 to 15:

Register	Offset
CVa	1FCh + (a × 4h)

Function

Contains values used to compare the conversion result when the compare function is enabled. This register is formatted like the D field in Data Result FIFO Register (RESFIFO0 - RESFIFO1), which has in different definitions for bit position and value format in different modes. There is a direct association of each compare value register to a specific command buffer register. For example, CV1 is only used during execution of CMD1 command.

When ADC is executing commands, do not update the CVn register associated with the active command (CMDn). Writes to associated CVn register during this time may result in unpredictable behavior.

NOTE

The 15 compare value registers are numbered [CV1] through [CV15]. In NXP-supplied header files, these registers are likely to be defined as a 15-element array indexed from 0. For example, the type declaration would be:

```
unsigned int CV[15];
```

and software would access ADC0 CV1 as `ADC0->CV[0]` and ADC0 CV15 as `ADC0->CV[14]`.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CVH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CVL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 CVH	Compare Value High Determines the high compare value. The compare function can be configured to check whether the result: • Is less than comparison values • Is greater than comparison values • Falls within a range of two comparison values • Falls outside a range of two comparison values. After the input is sampled and converted and any averaging iterations are performed, CVL and CVH can be used in a compare operation on the result. See Compare Function for a description of CVH usage.
15-0 CVL	Compare Value Low Determines the low compare value. The compare function can be configured to check whether the result: • Is less than comparison values • Is greater than comparison values • Falls within a range of two comparison values • Falls outside a range of two comparison values. After the input is sampled and converted and any averaging iterations are performed, CVL and CVH can be used in a compare operation on the result. See Compare Function for a description of CVL usage.

40.6.20 Data Result FIFO Register (RESFIFO0 - RESFIFO1)

Offset

Register	Offset
RESFIFO0	300h
RESFIFO1	304h

Function

Stores the data result of ADC conversions in a 16-entry FIFO. Several tag fields of source command and trigger information are stored with the data. FCTRLn[FCOUNT] indicates how many valid data words are stored in the RESFIFO. Reading RESFIFO provides the oldest unread data word entry in the FIFO and decrements FCTRLn[FCOUNT]. The FIFO can be emptied by successive reads of RESFIFO. The FIFO is reset by writing 0b1 to CTRL[RSTFIFOn].

The following table describes the format of data in the result FIFO in different modes of operation. The sign bit is the MSB in signed 2's complement modes. For example, when configured for 12-bit single-ended mode, D[15] and D[2:0] become 0. When configured for 13-bit differential mode, D[15] is the sign bit, and D[2:0] becomes 0.

Table 251. Data result register format description

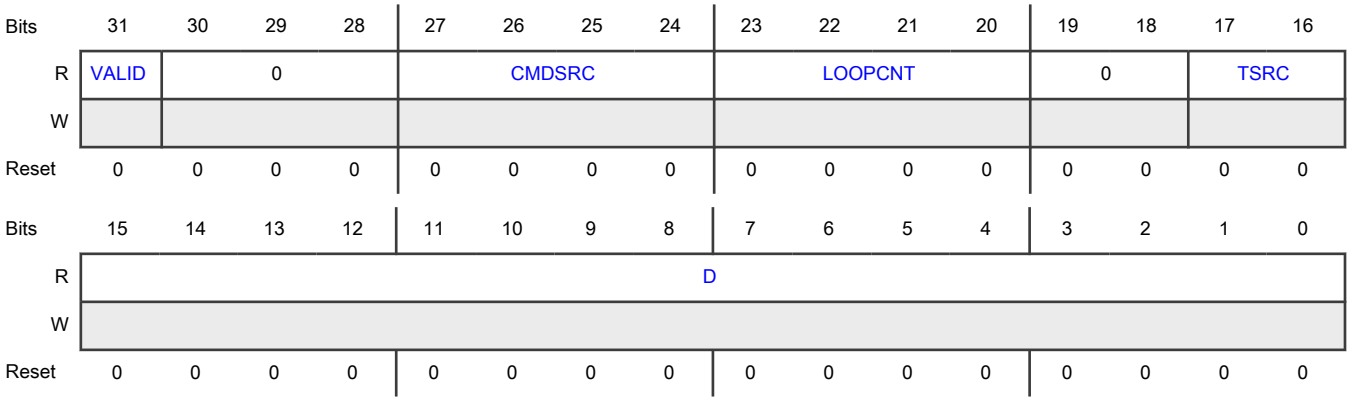
Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Signed 2's complement
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned, 16-bit magnitude
13-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Signed 2's complement, left justified, zero extended
12-bit single-ended	0	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Unsigned, zero in D[15] and D[2:0]

NOTE

S: Sign bit;

D: Data, 2's complement data when indicated

Diagram



Fields

Field	Function
31 VALID	FIFO Entry is Valid Indicates whether the FIFO entry is valid, which determines what happens to reads from RESFIFO. 0b - FIFO is empty. Discard any read from RESFIFO. 1b - FIFO contains data. FIFO record read from RESFIFO is valid.
30-28 —	Reserved
27-24	Command Buffer Source

Table continues on the next page...

Table continued from the previous page...

Field	Function
CMDSRC	Indicates the executed command buffer that generated this result. 0000b - Not a valid value CMDSRC value for a data word in RESFIFO. 0h is only found in the initial FIFO state, prior to the storage of an ADC conversion result into a RESFIFO buffer. 0001b - CMD1 0010b-1110b - Corresponding command buffer used as control settings for this conversion. 1111b - CMD15
23-20 LOOPCNT	Loop Count Value Indicates the loop count value during the command that generated this result. When CMDHn[LOOP] is non-zero, results are stored multiple times during command execution at the loop boundary. 0000b - Result is from initial conversion in command. 0001b - Result is from second conversion in command. 0010b-1110b - Result is from (LOOPCNT + 1) conversion in command. 1111b - Result is from 16th conversion in command.
19-18 —	Reserved
17-16 TSRC	Trigger Source Indicates the trigger source that initiated a conversion and generated this result. When multiple commands are chained together using CMDHn[NEXT], this field indicates the trigger source that started the command sequence. 00b - Trigger source 0 01b - Trigger source 1 10b-10b - Corresponding trigger source initiated this conversion. 11b - Trigger source 3
15-0 D	Data Result Contains the result of an ADC conversion. The formatting for the data in D is summarized in Table 251.

40.6.21 Calibration General A-Side Registers (CAL_GAR0)

Offset

Register	Offset
CAL_GAR0	400h

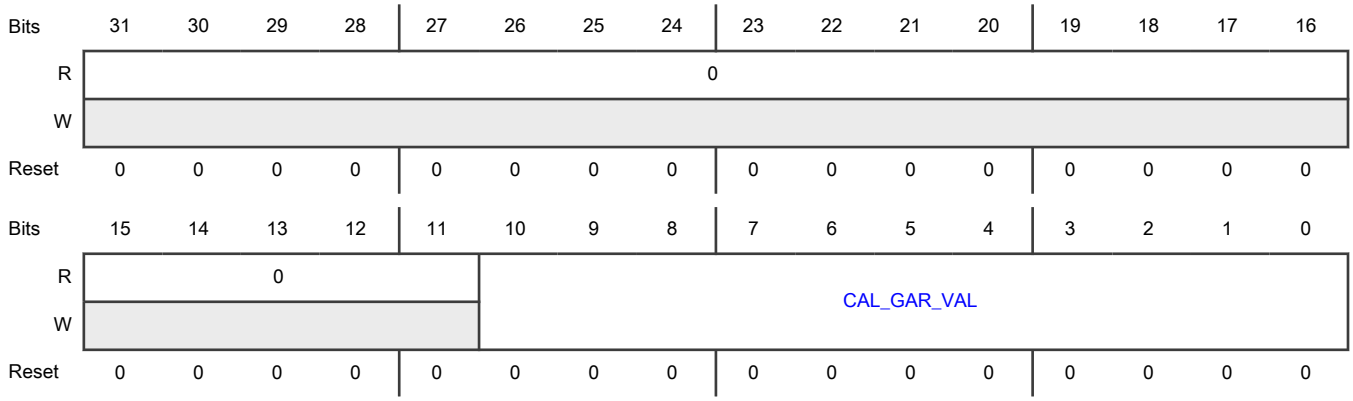
Function

Corrects for linearity errors of the A-side converter. All 33 A-side general calibration value registers (CAL_GAR0-CAL_GAR32) contain calibration information that is automatically updated during the self-calibration sequence. See Calibration functions for more information on completing ADC calibration steps.

The calibration values in the CAL_GAR registers affect the conversion result by conditionally being subtracted from the conversion before the result is transferred into the FIFOs. Calibration must be run each time ADC is powered down or a hard reset is issued. To reduce the latency required to run calibration, the CAL_GAR values can be stored in non-volatile memory after an initial calibration. These values can be recovered prior to the first ADC conversion. If these registers are set to values not generated by the calibration function, the linearity error specifications may not be met.

The CAL_GAR registers are only read-and-write accessible when ADC is disabled with CTRL[ADCEN] = 0. Access time when writing to these registers is larger than three ADC clock cycles. Wait states are inserted on the bus to meet synchronization timing to the associated CAL_GAR register. The width of each register in this array is non-uniform. The exact width of each register is summarized in Calibration General A-Side and B-Side Widths.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 CAL_GAR_VAL	Calibration General A Side Register Element

40.6.22 Calibration General A-Side Registers (CAL_GAR1)

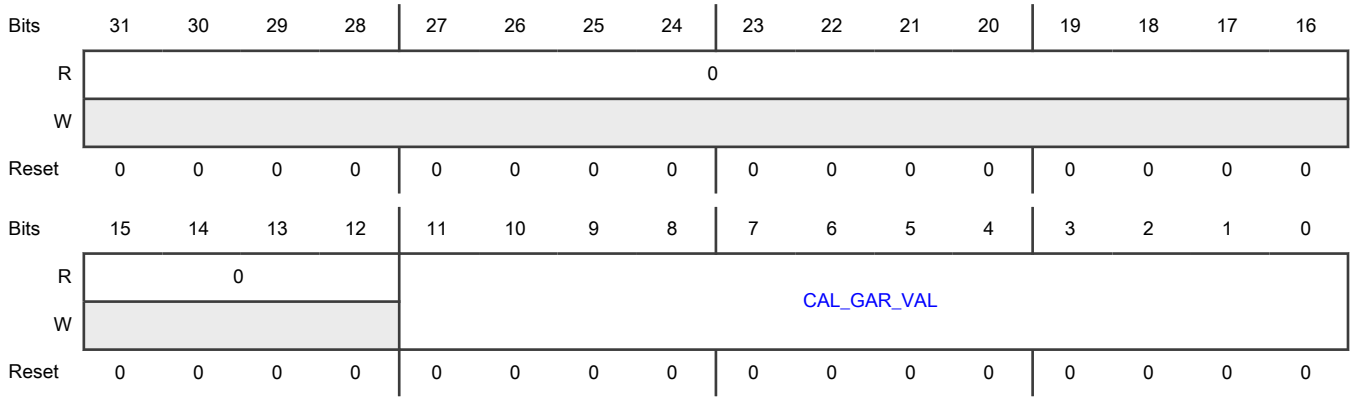
Offset

Register	Offset
CAL_GAR1	404h

Function

Calibration register CAL_GAR1. For more detail, see Calibration General A-Side Registers (CAL_GAR0).

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 CAL_GAR_VAL	Calibration General A Side Register Element

40.6.23 Calibration General A-Side Registers (CAL_GAR2 - CAL_GAR3)

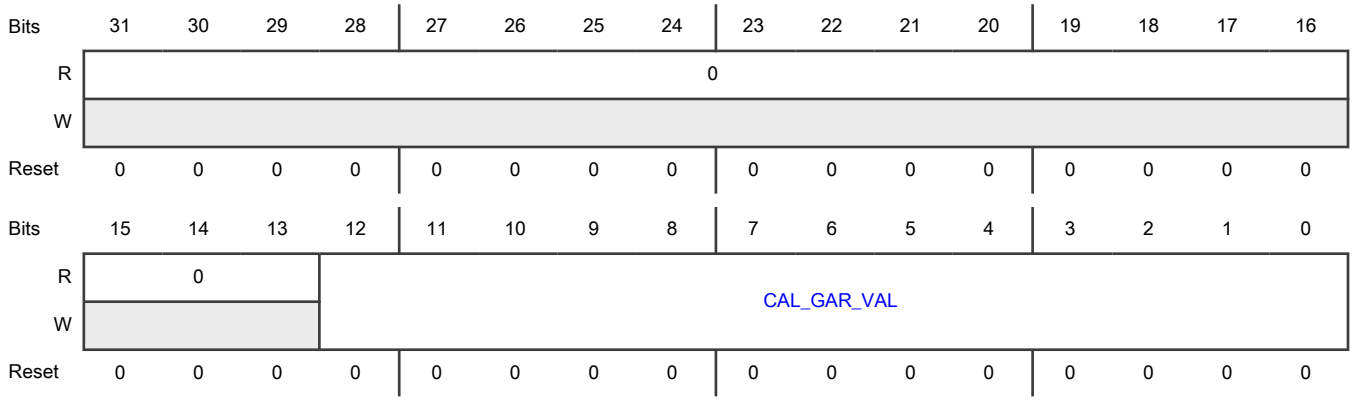
Offset

Register	Offset
CAL_GAR2	408h
CAL_GAR3	40Ch

Function

Calibration registers CAL_GAR2 and CAL_GAR3. For more detail, see Calibration General A-Side Registers (CAL_GAR0).

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 CAL_GAR_VAL	Calibration General A Side Register Element

40.6.24 Calibration General A-Side Registers (CAL_GAR4 - CAL_GAR7)

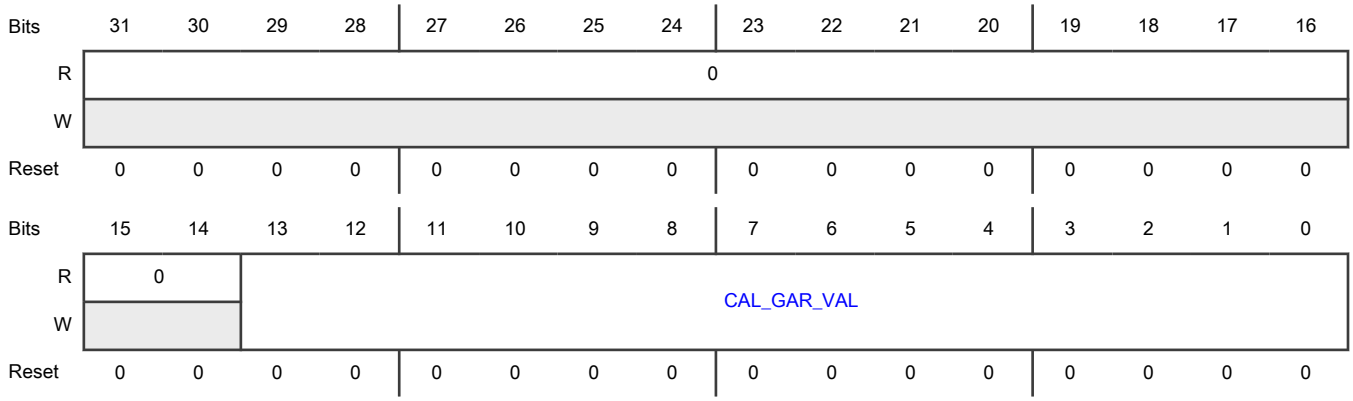
Offset

Register	Offset
CAL_GAR4	410h
CAL_GAR5	414h
CAL_GAR6	418h
CAL_GAR7	41Ch

Function

Calibration registers CAL_GAR4 through CAL_GAR7. For more detail, see Calibration General A-Side Registers (CAL_GAR0).

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0	Calibration General A Side Register Element

Table continues on the next page...

Table continued from the previous page...

Field	Function
CAL_GAR_VAL	

40.6.25 Calibration General A-Side Registers (CAL_GAR8 - CAL_GAR15)

Offset

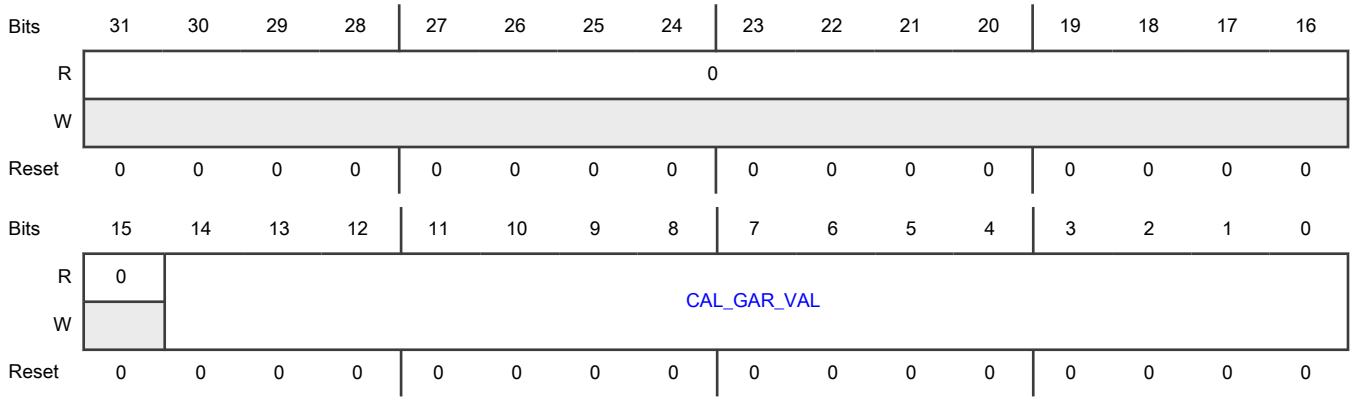
For a = 8 to 15:

Register	Offset
CAL_GARa	400h + (a × 4h)

Function

Calibration registers CAL_GAR8 through CAL_GAR15. For more detail, see Calibration General A-Side Registers (CAL_GAR0).

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 CAL_GAR_VAL	Calibration General A Side Register Element

40.6.26 Calibration General A-Side Registers (CAL_GAR16 - CAL_GAR31)

Offset

For a = 16 to 31:

Register	Offset
CAL_GARa	400h + (a × 4h)

Function

Calibration registers CAL_GAR16 through CAL_GAR31. For more detail, see Calibration General A-Side Registers (CAL_GAR0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAL_GAR_VAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 CAL_GAR_VAL	Calibration General A Side Register Element

40.6.27 Calibration General A-Side Registers (CAL_GAR32)

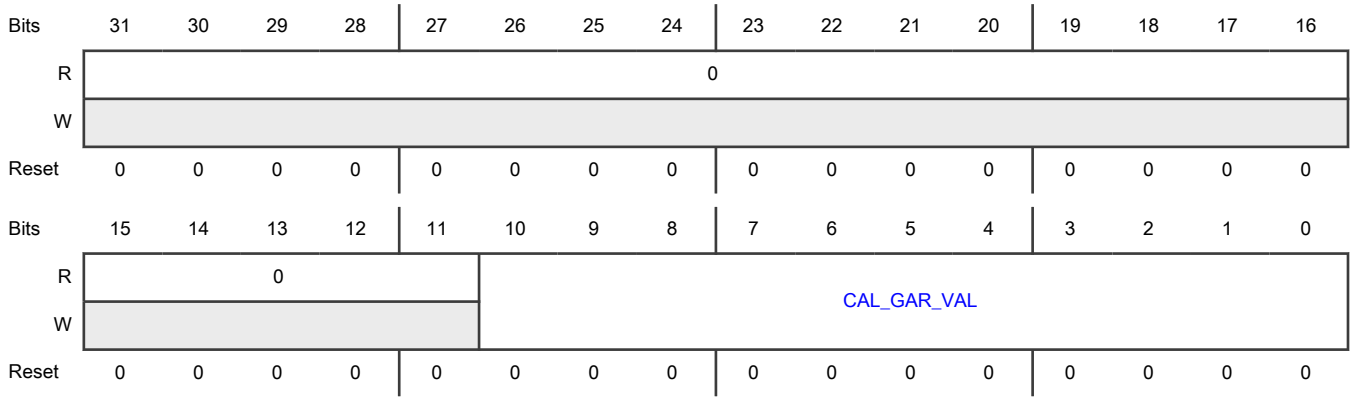
Offset

Register	Offset
CAL_GAR32	480h

Function

Calibration register CAL_GAR32. For more detail, see Calibration General A-Side Registers (CAL_GAR0).

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 CAL_GAR_VAL	Calibration General A Side Register Element

40.6.28 Calibration General B-Side Registers (CAL_GBR0)

Offset

Register	Offset
CAL_GBR0	500h

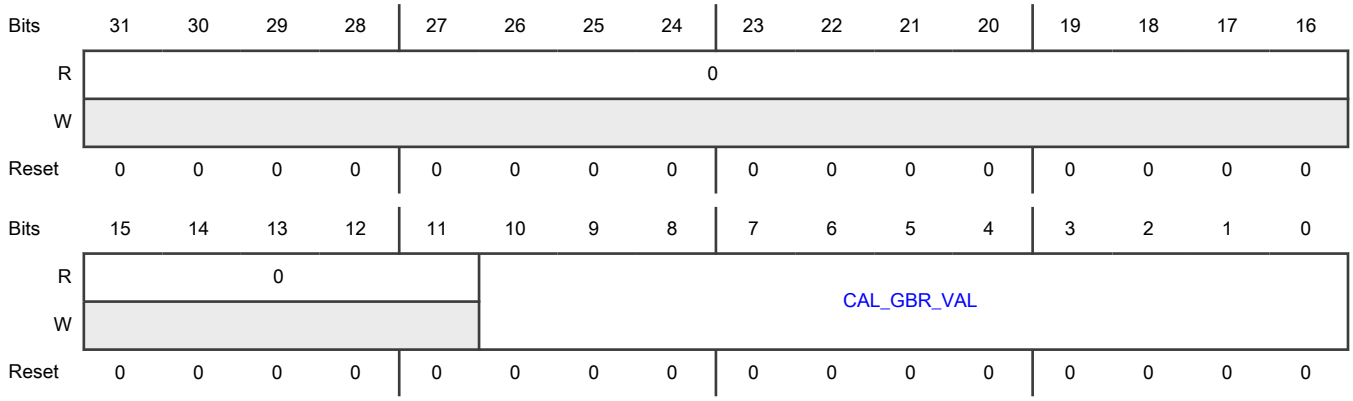
Function

Corrects for linearity errors of the B-side converter. All 33 B-side general calibration value registers (CAL_GBR0-CAL_GBR32) contain calibration information that is automatically updated during the self-calibration sequence. See Calibration functions for more information on completing ADC calibration steps.

The calibration values in the CAL_GBR registers affect the conversion result by conditionally being subtracted from the conversion before the result is transferred into the FIFOs. Calibration must be run each time ADC is powered down or a hard reset is issued. To reduce the latency required to run calibration, the CAL_GBR values can be stored in non-volatile memory after an initial calibration. These values can be recovered prior to the first ADC conversion. If these registers are set to values not generated by the calibration function, the linearity error specifications may not be met.

The CAL_GBR registers are only read-and-write accessible when ADC is disabled with CTRL[ADCEN] = 0. Access time when writing to these registers is larger than three ADC clock cycles. Wait states are inserted on the bus to meet synchronization timing to the associated CAL_GBR register. The width of each register in this array is non-uniform. The exact width of each register is summarized in Calibration General A-Side and B-Side Widths.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 CAL_GBR_VAL	Calibration General B Side Register Element

40.6.29 Calibration General B-Side Registers (CAL_GBR1)

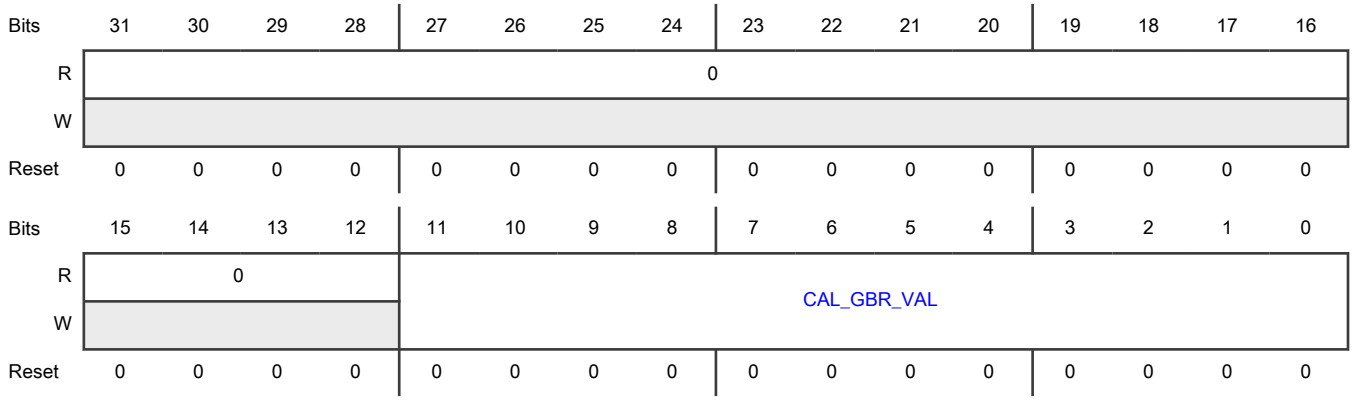
Offset

Register	Offset
CAL_GBR1	504h

Function

Calibration register CAL_GBR1. For more detail, see Calibration General B-Side Registers (CAL_GBR0).

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 CAL_GBR_VAL	Calibration General B Side Register Element

40.6.30 Calibration General B-Side Registers (CAL_GBR2 - CAL_GBR3)

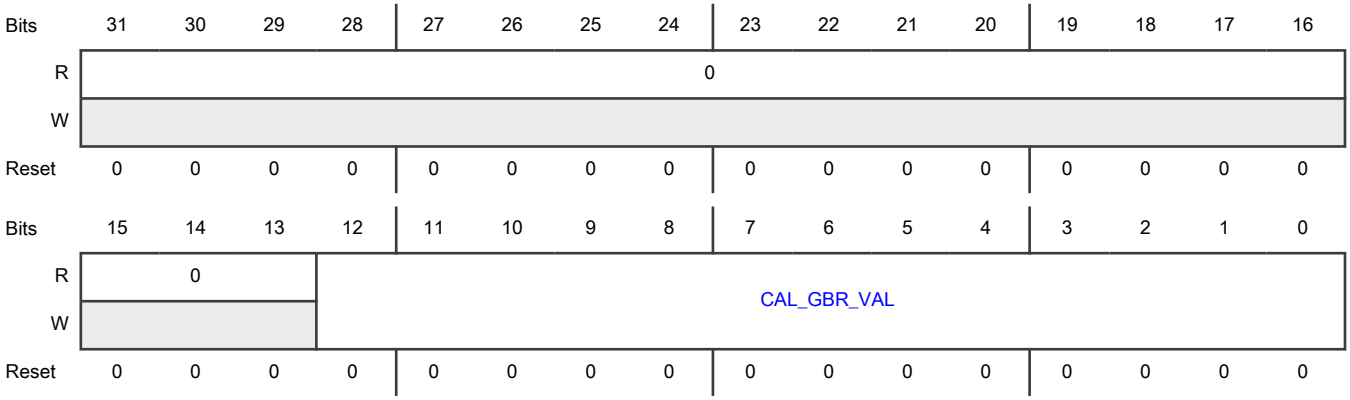
Offset

Register	Offset
CAL_GBR2	508h
CAL_GBR3	50Ch

Function

Calibration registers CAL_GBR2 and CAL_GBR3. For more detail, see Calibration General B-Side Registers (CAL_GBR0).

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 CAL_GBR_VAL	Calibration General B Side Register Element

40.6.31 Calibration General B-Side Registers (CAL_GBR4 - CAL_GBR7)

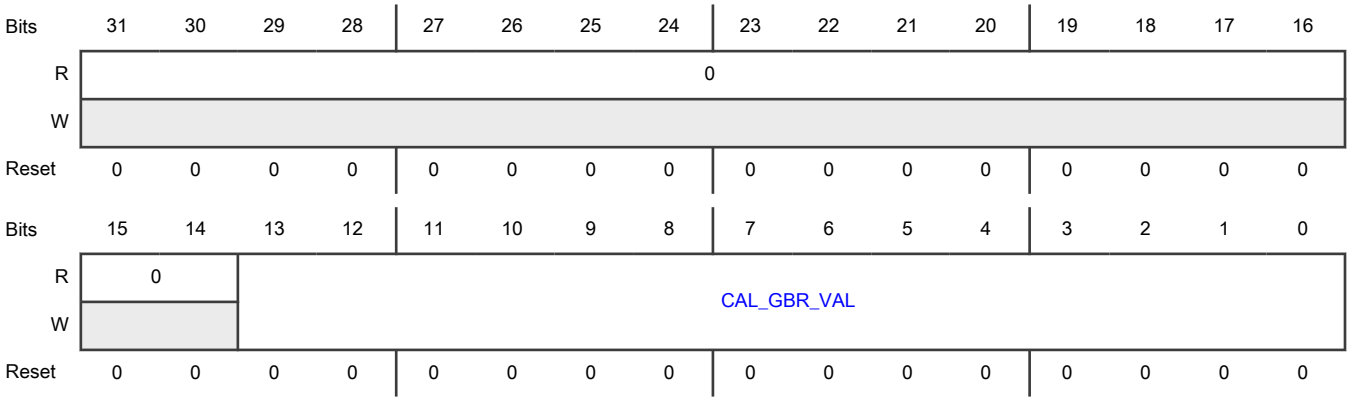
Offset

Register	Offset
CAL_GBR4	510h
CAL_GBR5	514h
CAL_GBR6	518h
CAL_GBR7	51Ch

Function

Calibration registers CAL_GBR4 through CAL_GBR7. For more detail, see Calibration General B-Side Registers (CAL_GBR0).

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 CAL_GBR_VAL	Calibration General B Side Register Element

40.6.32 Calibration General B-Side Registers (CAL_GBR8 - CAL_GBR15)

Offset

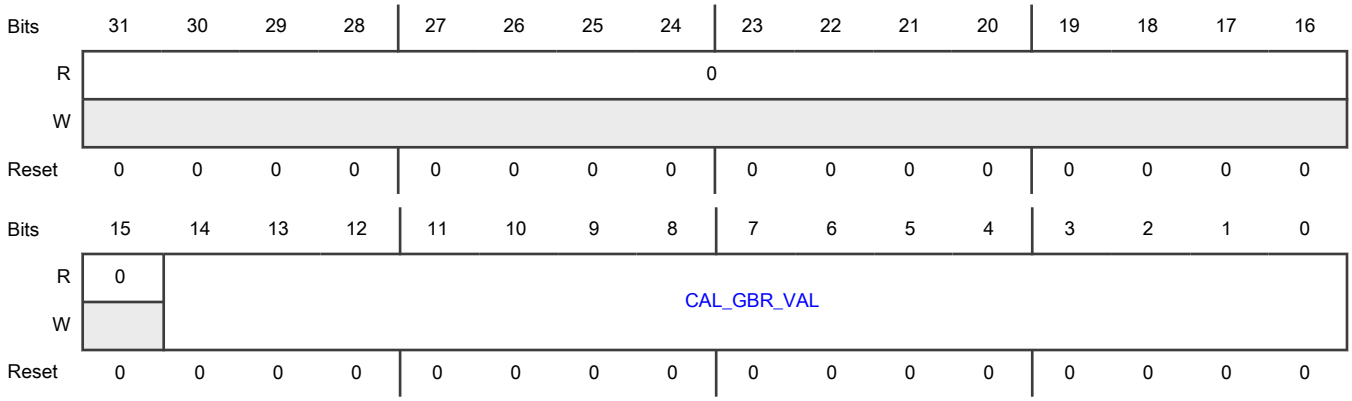
For a = 8 to 15:

Register	Offset
CAL_GBRa	500h + (a × 4h)

Function

Calibration registers CAL_GBR8 through CAL_GBR15. For more detail, see Calibration General B-Side Registers (CAL_GBR0).

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 CAL_GBR_VAL	Calibration General B Side Register Element

40.6.33 Calibration General B-Side Registers (CAL_GBR16 - CAL_GBR31)

Offset

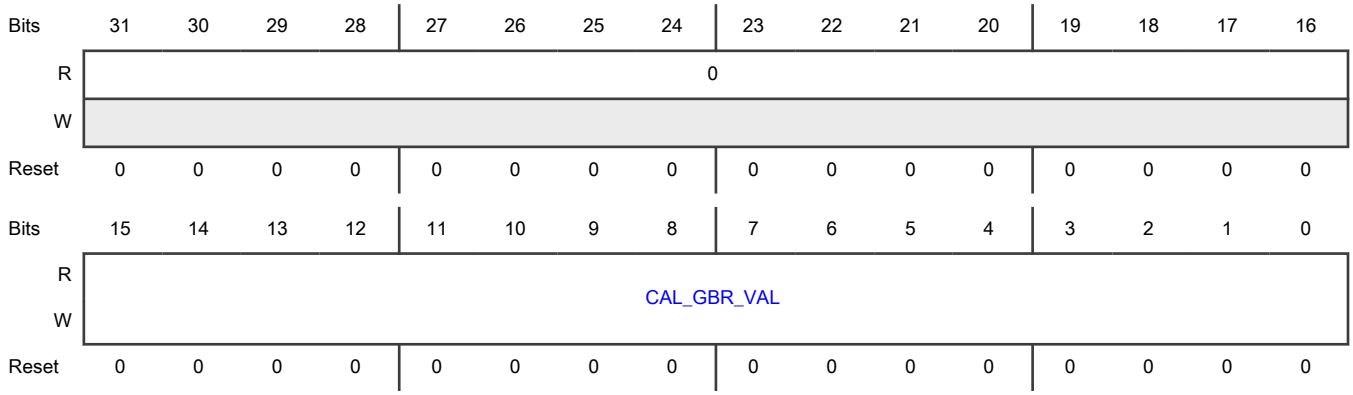
For a = 16 to 31:

Register	Offset
CAL_GBRa	500h + (a × 4h)

Function

Calibration registers CAL_GBR16 through CAL_GBR31. For more detail, see Calibration General B-Side Registers (CAL_GBR0).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 CAL_GBR_VAL	Calibration General B Side Register Element

40.6.34 Calibration General B-Side Registers (CAL_GBR32)

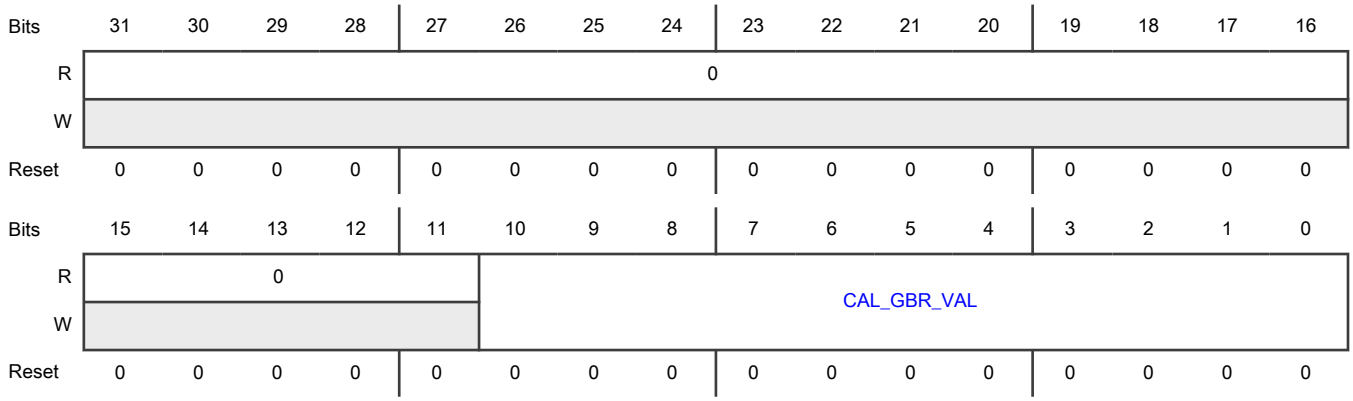
Offset

Register	Offset
CAL_GBR32	580h

Function

Calibration register CAL_GBR32. For more detail, see Calibration General B-Side Registers (CAL_GBR0).

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 CAL_GBR_VAL	Calibration General B Side Register Element

Chapter 41
Low Power Comparator (LPCMP)

41.1 Chip-specific LPCMP information

Table 252. Reference links to related information

Topic	Related module	Reference
Full description	LPCMP	LPCMP
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

41.1.1 Module instances

This device has two instances of the LPCMP module, LPCMP0 and LPCMP1.

41.1.2 Input connections

Table 253. Input connections

Index	CMP0	CMP1	Description
0	CMP0_IN0	CMP1_IN0	Pinned out in VDD_IO
1	—	CMP1_IN1	VDD_IO domain
2	CMP0_IN2	—	VDD_IO domain
3	CMP0_IN3	—	VDD_IO domain
4	PMC BG	PMC BG	VDD_SYS domain
5	VREFO	VREFO	VDD_ANA domain
7	CMP0 8-bit DAC	CMP1 8-bit DAC	Internal DAC

41.1.3 VREFH_EXT

The DCR[VRSEL] is used to configure the VREFH_EXT:

- VRSEL=0, VREFH0 is selected, which is VDD_IO_ABC
- VRSEL=1, VREFH1 is selected, which is VREF_OUT (must be in VREFH / VDD_ANA domain)

41.2 Overview

The LPCMP module provides a circuit to compare two analog input voltages. It includes the following:

- A comparator (CMP)
- A DAC
- An analog mux (ANMUX)

See Block diagram for more information.

CMP can operate across the full range of the supply voltage, known as rail-to-rail operation.

DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. DAC divides the supply reference V_{in} into 256 voltage levels. An 8-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/256$.

You can select V_{in} from the following voltage sources:

- vrefh0
- vrefh1

See the chip-specific LPCMP information for more information on source of vrefh0 and vrefh1.

NOTE

The LPCMP's internal DAC output is available as an on-chip internal signal only and is not available for an external chip pin.

ANMUX allows you to select an analog input signal from among eight channel options. One channel option is the DAC output. Other chip resources are connected to the other channels. See the chip-specific LPCMP information section for more information. ANMUX can operate across the full range of the supply voltage.

41.2.1 Block diagram

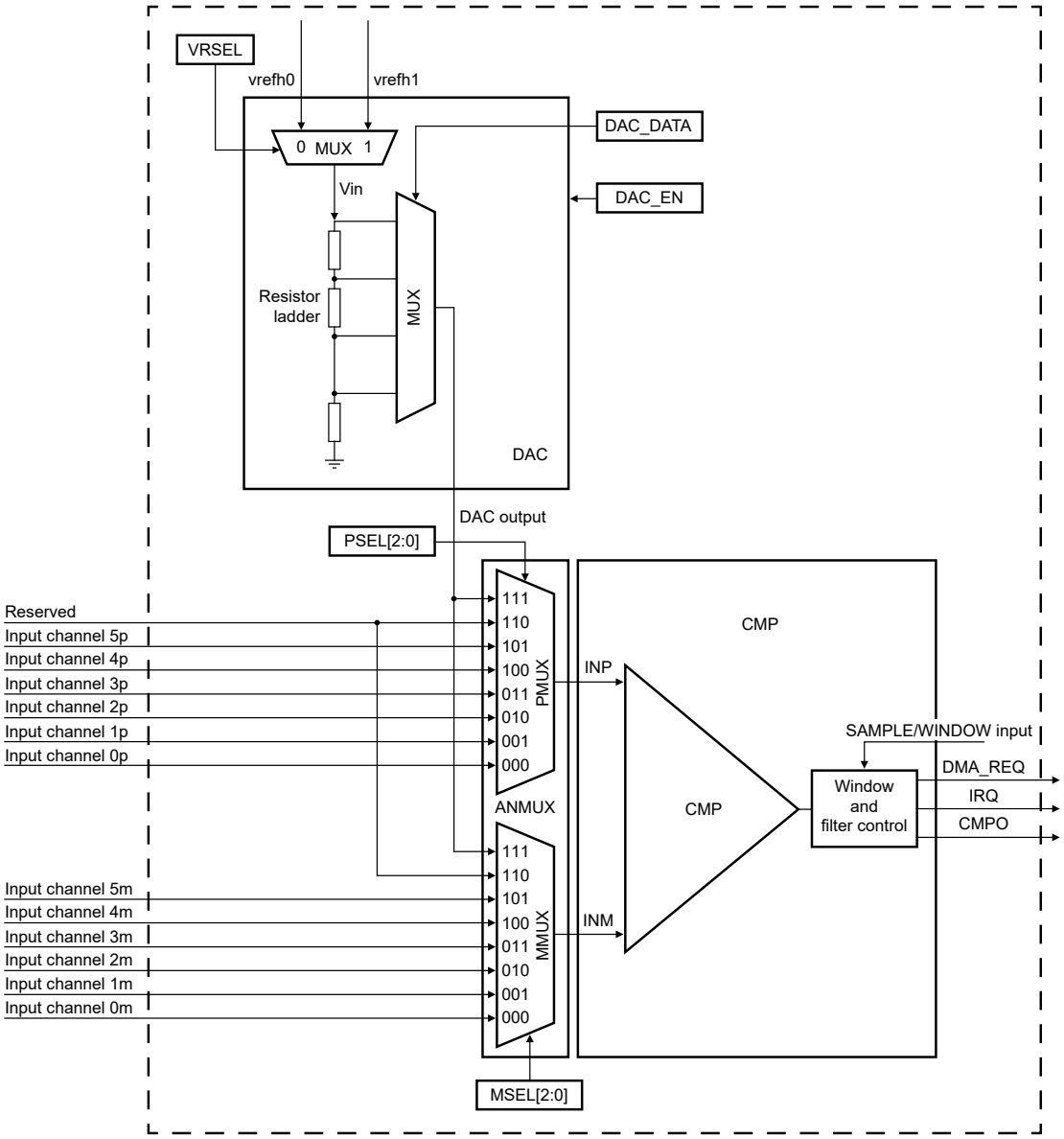


Figure 109. Block diagram

41.2.2 Features

The features of the LPCMP module include：

- Includes two 8-to-1 channel MUXes to select input signal from eight channels
- Supports multiple operation modes to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered
- Provides the following advance features for window and sample:

- Window and sample signals can be inverted.
- CMPO rising, falling or both edges closes the window.
- CMPO level can be defined when window is closed.

- Provides selectable performance levels:
  - Nano-Power mode
  - Low-Power (speed) mode
  - High-Power (speed) mode
- Supports programmable hysteresis control
- Provides a selectable inversion on comparator output
- Uses an external hysteresis at the same time the output filter is used for internal functions
- Provides interrupt and DMA support
- Includes an 8-bit resolution DAC
- Provides a selectable supply reference source for DAC
- Provides a configurable Low-Power (speed) mode or High-Power (speed) mode for DAC

41.3 Functional description

41.3.1 Functional block diagram

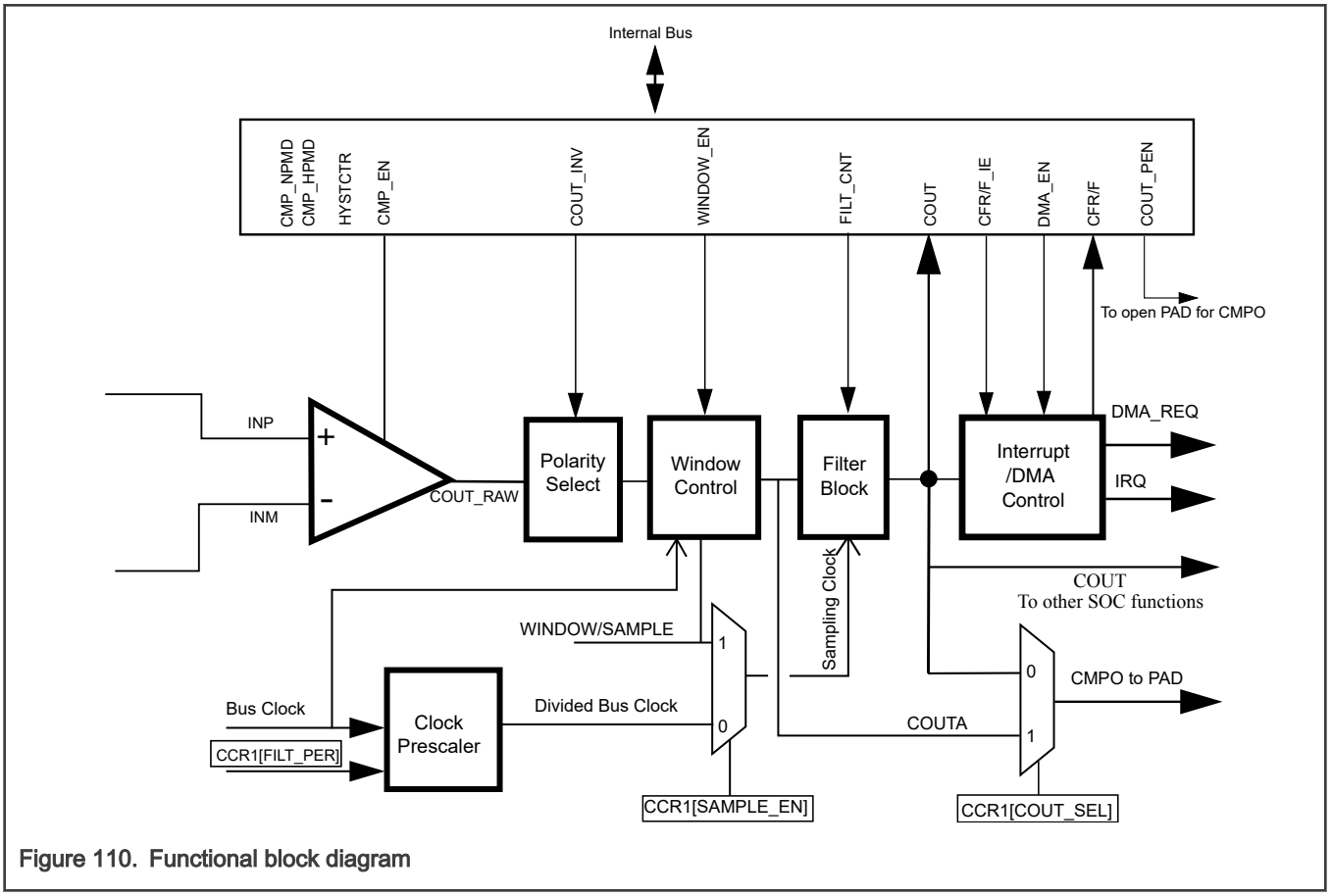


Figure 110. Functional block diagram

As shown in the block diagram, the functions are:

- Compared two analog input voltages applied to INP and INM, COUT_RAW is high when the INP input voltage is greater than the INM input voltage, and COUT_RAW is low when the INP input voltage is less than the INM input voltage.
- The COUT_RAW signal can be inverted by enabling CCR1[COUT_INV].
- The optionally inverted comparator output COUT_RAW is sampled on every bus clock when you enable the CCR1[WINDOW_EN] to generate COUTA. In this case, the comparator output is ignored during time periods when the input voltages are not valid. This is useful when you implement zero-crossing-detection for certain PWM applications.
- The window control block is bypassed when CCR1[WINDOW_EN] is disabled.
- The filter block acts as a simple sampler when CCR1[FILT_CNT] is set to 01h.
- The filter block acts as a filter based on multiple samples when CCR1[FILT_CNT] is set to be greater than 01h.
  - If CCR1[SAMPLE_EN] is set to 1, use the external SAMPLE input as the sampling clock.
  - If CCR1[SAMPLE_EN] is set to 0, use the divided bus clock as the sampling clock.
- Bypasses the filter block when it is not in use.

```
Bypass_Filter_Block = (FILT_CNT == 0x00) |  (~SAMPLE_EN & (FILT_PER == 0x00))
```

- Both COUTA and COUT can be configured as module output CMPO by configuring CCR1[COUT_SEL], and are used for different purposes within the system.
- The optionally filtered COUT can be read directly in CSR[COUT].
- The SAMPLE/WINDOW signal can be inverted by setting CCR1[WINDOW_INV].
- The SAMPLE/WINDOW signal can be closed by CMPO's falling edge and/or rising edge by setting CCR1[WINDOW_CLS] in Window mode.
- In Window mode, when window is closed, define the COUTA value as CCR1[COUTA_OW] by setting CCR1[COUTA_OWEN]. If CCR1[COUTA_OWEN] is not set, COUTA holds the last sampled value.

NOTE

See the chip configuration section for the source of SAMPLE/WINDOW input.

41.3.2 Low-pass filter mode

The low-pass filter mode operates on an unfiltered, optionally inverted comparator output COUTA, and generates the filtered and synchronized output COUT. You can configure both COUTA and COUT as module outputs and use for different purposes within the system.

Synchronization and edge detection determine the bit values of status register. They also apply to COUT for all sampling and windowed modes. You can perform filtering using an internal timebase defined by CCR1[FILT_PER], or use an external sample input to determine sample time.

The need for digital filtering and the amount of filtering depends on your requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, generate a high-frequency oscillations at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

41.3.2.1 Enabling low-pass filter mode

You can enable low-pass filter mode by setting the following:

- CCR1[FILT_CNT] > 01h
- CCR1[FILT_PER] to a nonzero value or writing 1 to CCR1[SAMPLE_EN].

If you use the divided bus clock to drive the low-pass filter, it samples COUTA every CCR1[FILT_PER] bus clock cycle.

If CCR1[SAMPLE_EN] is set to 1, the low-pass filter samples COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CCR1[FILT_CNT] samples agree that the output value has changed.

41.3.2.2 Latency issues

Program the value of CCR1[FILT_PER] or sample period such that the sampling period is longer than the period of the expected noise, ensuring that a given noise spike corrupts only one sample. You must choose the value of CCR1[FILT_CNT] to reduce the probability of noisy samples causing an incorrect transition to recognize. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CCR1[FILT_CNT].

You must trade off the values of CCR1[FILT_PER] or sample period and CCR1[FILT_CNT] against the need for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CCR1[FILT_CNT].

Table 255 summarizes maximum latency values for the various modes of operation in the absence of noise. Filtering latency restarts each time the noise masks an actual output transition.

41.3.3 Low power mode operation

Below table introduces the mode of operation of lower power.

Table 254. Low power mode operation

Mode of operation	Description
Sleep	LPCMP can operate only in Continuous mode.

41.4 Functional modes

You can combine the comparator window and filter features as shown in the following table.

Table 255. Functional modes

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
1	0	X	X	X	X	See the Disabled mode (#1).	N/A
2A	1	0	X	0x00	X	See the Continuous mode (#2A and #2B).	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	See the Sampled, non-filtered mode (#3A and #3B).	$T_{PD} + T_{SAMPLE} + 3T_{per}$
3B	1	0	0	0x01	> 0x00		$T_{PD} + (FILT\_PER * T_{per}) + 3T_{per}$
4A	1	0	1	> 0x01	X	See the Sampled, filtered mode (#4A and #4B).	$T_{PD} + (FILT\_CNT * T_{SAMPLE}) + 3T_{per}$
4B	1	0	0	> 0x01	> 0x00		$T_{PD} + (FILT\_CNT * FILT\_PER \times T_{per}) + 3T_{per}$
5A	1	1	0	0x00	X	See the Windowed mode (#5A and #5B).	$T_{PD} + 2T_{per}$
5B	1	1	0	X	0x00		

Table continues on the next page...

Table 255. Functional modes (continued)

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
6	1	1	0	0x01	> 0x00	See the Windowed/ Resampled mode (#6).	T_{PD} + (FILT_PER * T_{per}) + 3T_{per}
7	1	1	0	> 0x01	> 0x00	See the Windowed/ Filtered mode (#7).	T_{PD} + (FILT_CNT * FILT_PER x T_{per}) + 3T_{per}
All other combinations of CMP_EN, WINDOW_EN, SAMPLE_EN, FILT_CNT, and FILT_PER are illegal.							

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

41.4.1 Disabled mode (#1)

In this mode:

- The analog comparator is non-functional and consumes no power.
- CSR[COUT] and CMPO are the same as CCR1[COUT_INV].

41.4.2 Continuous mode (#2A and #2B)

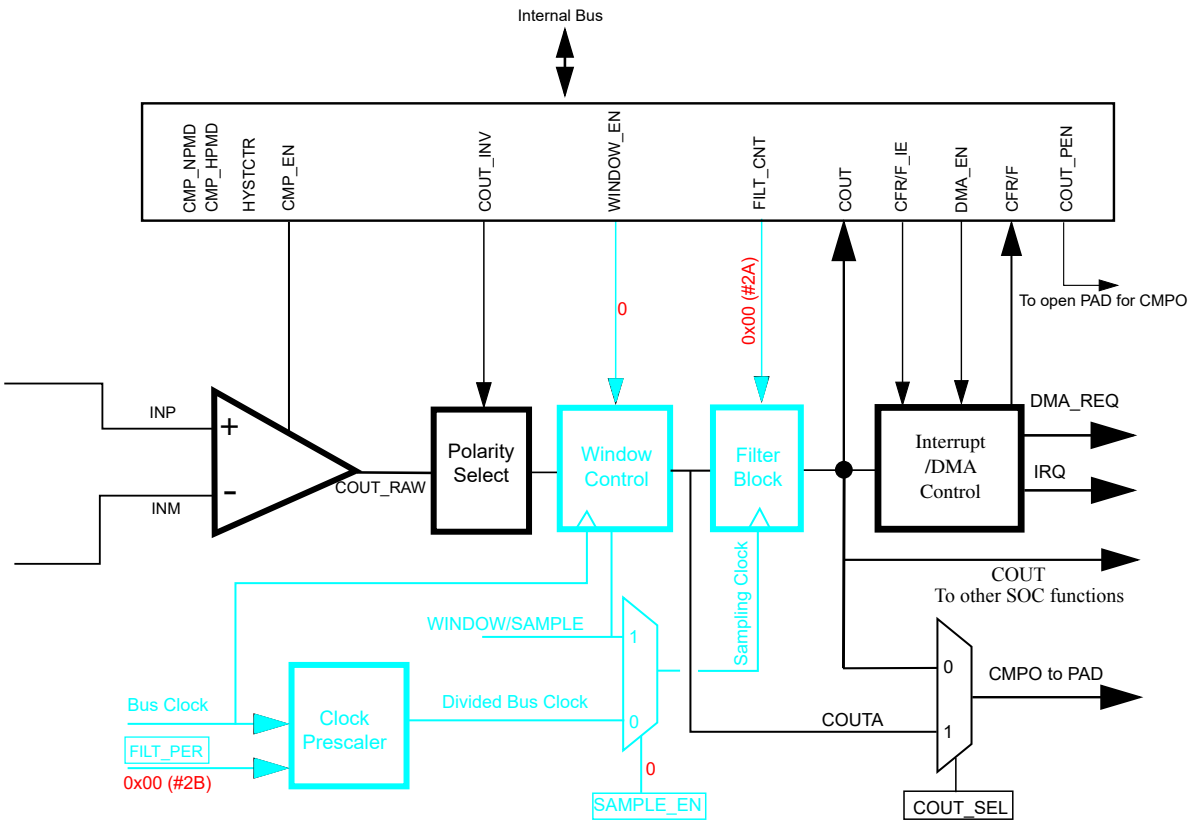


Figure 111. Comparator operation in continuous mode

COUT_RAW is optionally inverted in this mode but is not subject to external sampling or filtering. Both window control and filter blocks bypass completely, and CSR[COUT] updates continuously. The path from comparator input pins to output pins operates in a combinational (unclocked) mode. COUT and COUTA are identical in this mode.

For cases where a comparator drives a fault input, you must configure it to operate in Continuous mode so that an external fault can immediately pass to the target fault circuitry through the comparator.

41.4.3 Sampled, non-filtered mode (#3A and #3B)

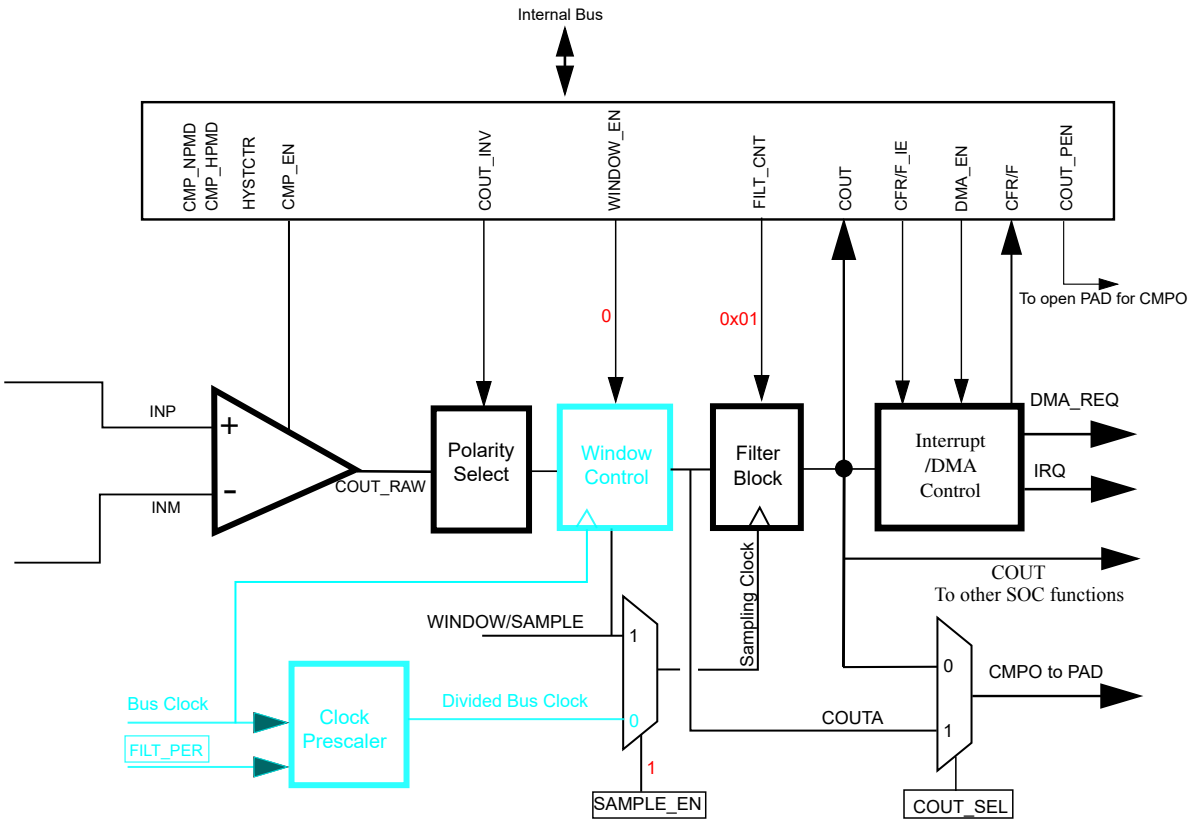


Figure 112. Sampled, non-filtered (#3A): sampling point externally driven

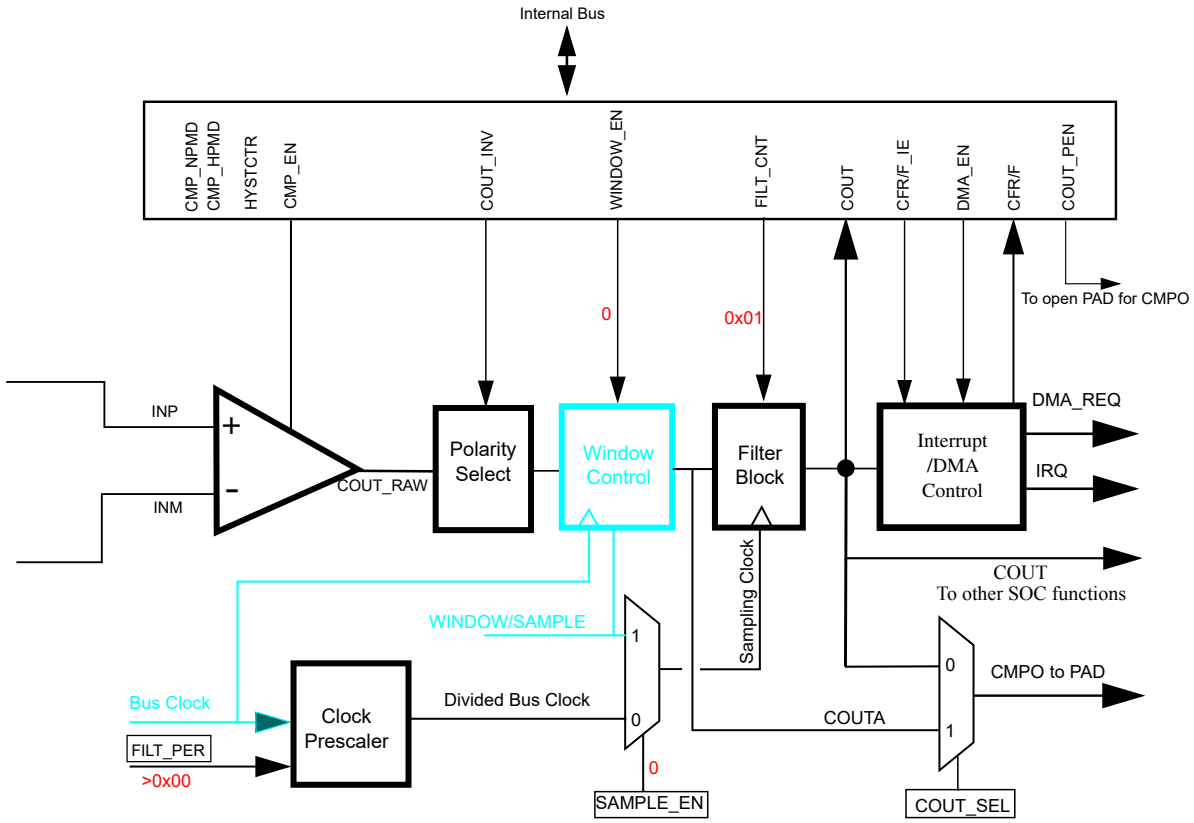


Figure 113. Sampled, Non-Filtered (#3B): sampling interval internally derived

In this mode, the path from analog inputs to COUTA is combinational (unclocked). Windowing control bypasses completely. You can sample COUTA whenever you detect a rising edge on the sampling clock.

The difference in two operation modes (#3A and #3B) of sampled, non-filtered mode is that how you drive the clock to the filter block. In #3A, the clock to filter block drives externally, and in #3B, the clock to filter block drives internally.

The filter block has no other function than sample or hold of the comparator output in this mode.

The following figure shows the comparator operation in this mode, assuming that the polarity select sets to a non-inverting state.

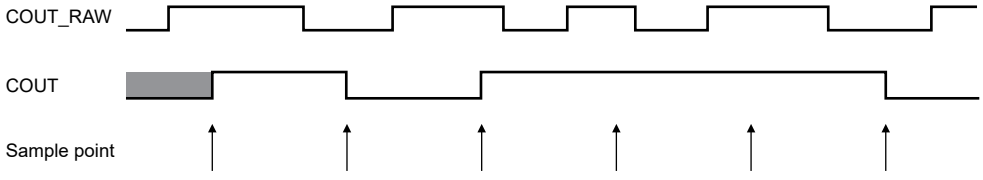


Figure 114. Sampled, Non-Filtered mode timing diagram

41.4.4 Sampled, filtered mode (#4A and #4B)

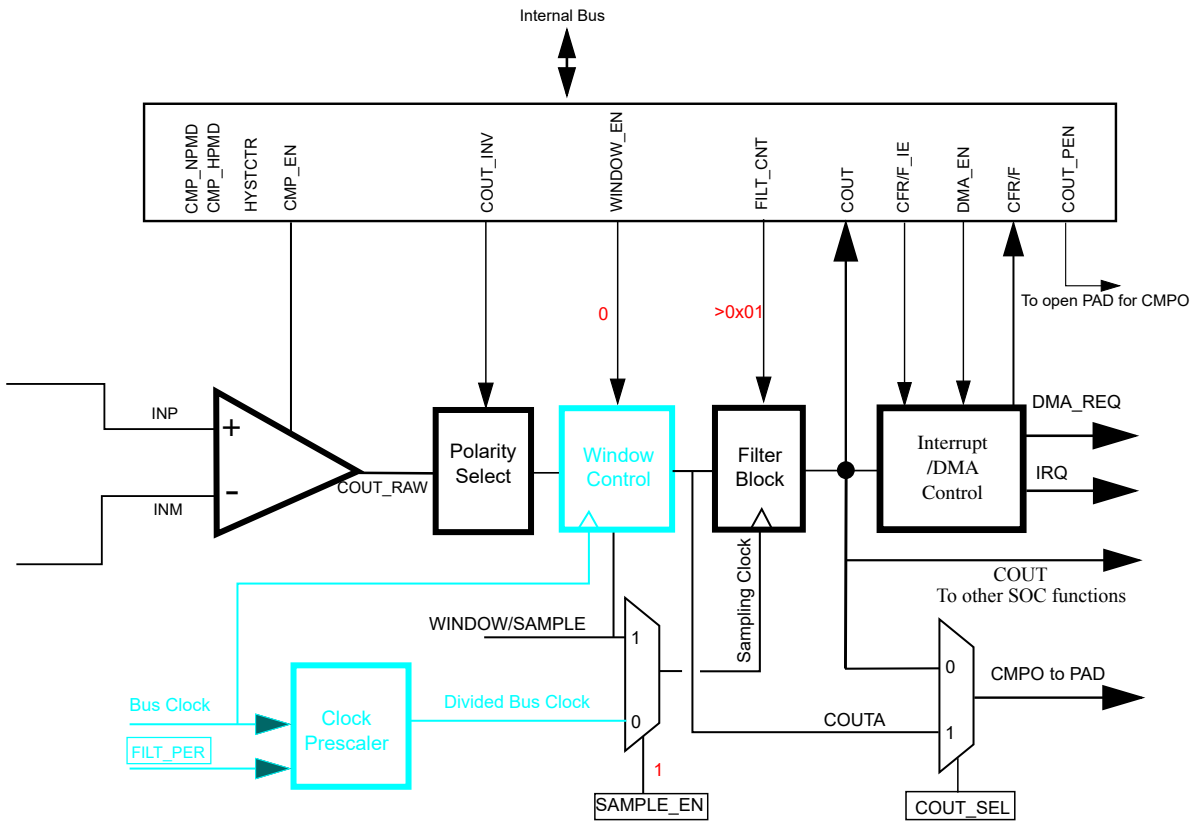


Figure 115. Sampled, filtered (#4A): sampling point externally driven

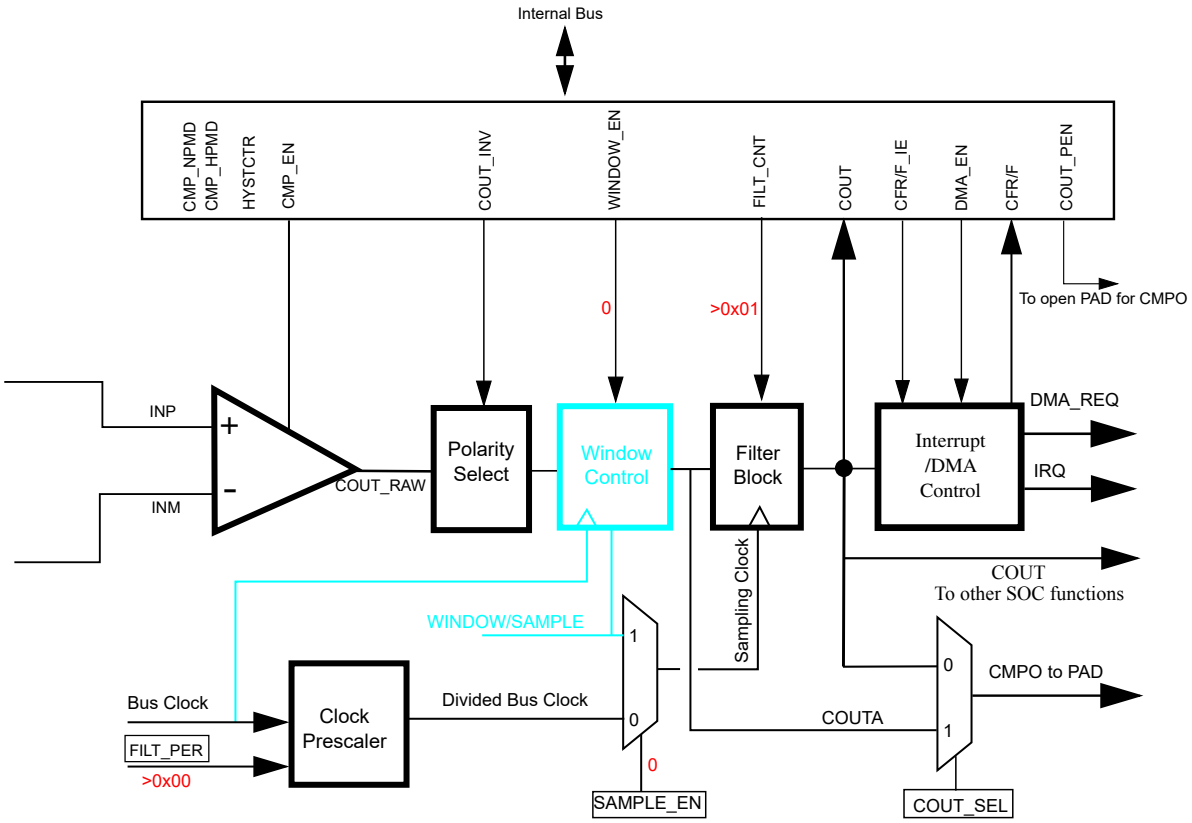


Figure 116. Sampled, filtered (#4B): sampling point internally derived

In this mode, the path from the analog inputs to COUTA is combinational(unclocked). Windowing control bypasses completely. You can sample COUTA whenever you detect a rising edge on the sampling clock.

The only difference in operation between sampled, non-filtered (#3A) mode and sampled, filtered (#4A) mode is that CCR1[FILT_CNT] is larger than 1, which activates filter operation.

The only difference in operation between sampled, non-filtered (#3B) mode and sampled, filtered (#4B) mode is that CCR1[FILT_CNT] is larger than 1, which activates filter operation.

41.4.5 Windowed mode (#5A and #5B)

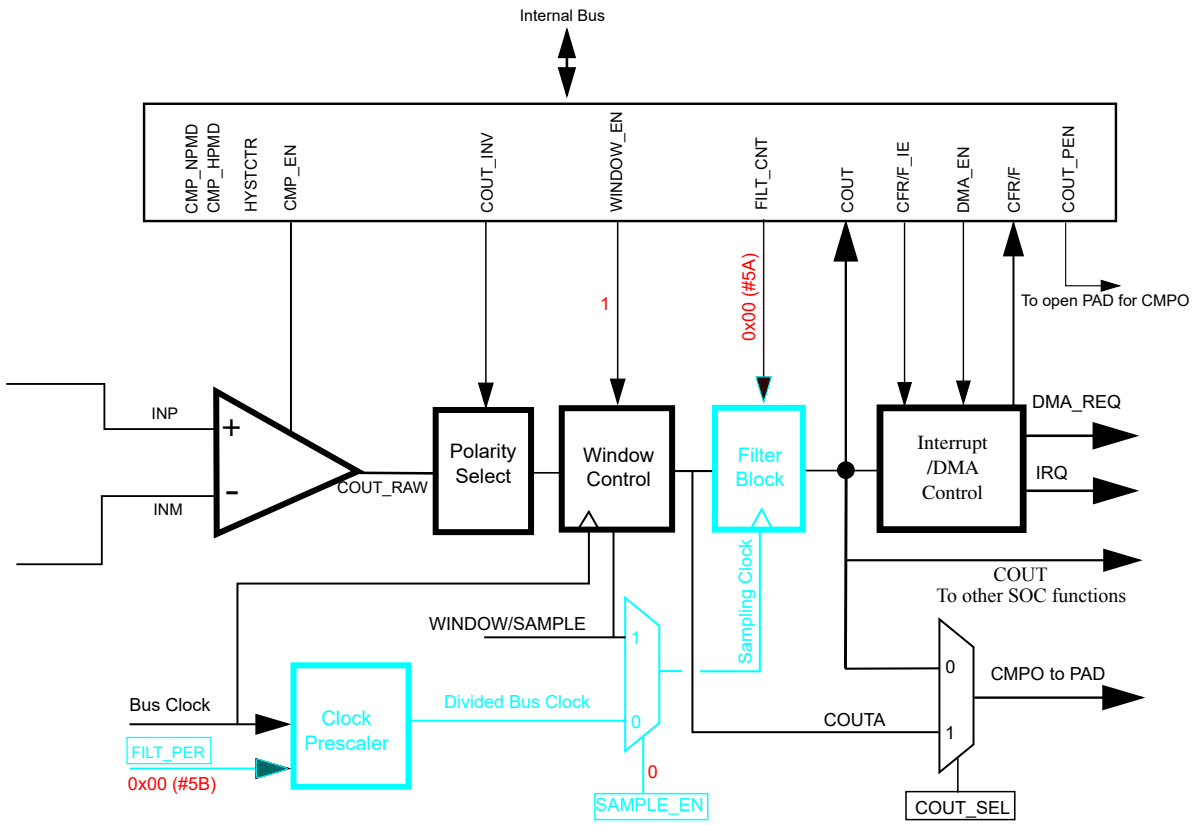


Figure 117. Windowed mode

The bus clock clocks COUTA whenever you enable the window in this mode. The last latched value holds after you disable the window and the filter block is bypassed.

The following figure shows the comparator operation in this mode, ignoring the latency of the analog comparator, polarity select, and window control block. The polarity select sets to a non-inverting state.

COUTA may lag the analog inputs by up to two functional clock cycles plus the combinational path delay through the comparator and polarity select logic in the actual operation.

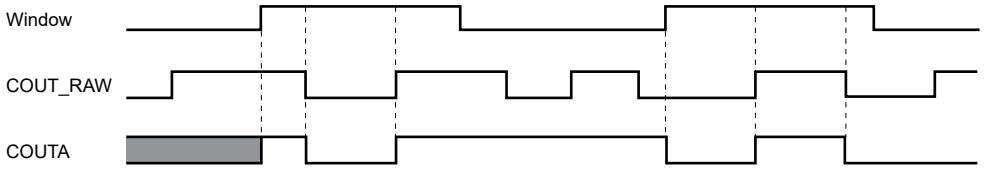


Figure 118. Windowed mode timing diagram

The following figure shows that if CCR1[COUTA_OWEN] becomes 1, you can define COUTA level as CCR1[COUTA_OW], after you closes the window.

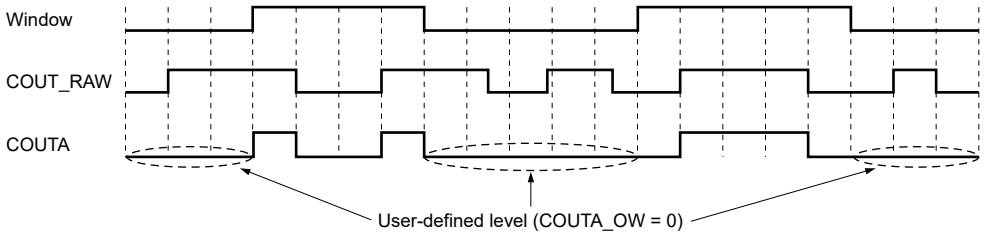


Figure 119. Windowed mode timing diagram with user defined value 0 outside window

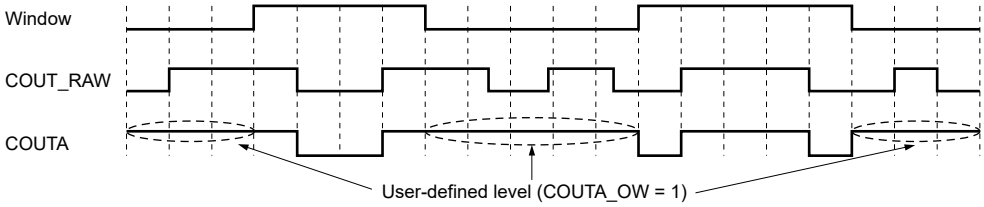


Figure 120. Windowed mode timing diagram with user defined value 1 outside window

NOTE

When the window is open, COUT_A will switch from COUTA_OW to COUT_RAW. When the window is closed, COUT_A will switch from COUT_RAW to COUTA_OW. This may generate unnecessary transition flags, for instance, CFR or CFF. User needs to choose COUTA_OW carefully according to the actual application, and select the appropriate flag CFR or CFF to generate interrupt.

If CCR1[WINDOW_CLS] becomes 1, you can define the CMPO event (rising edge, falling edge or both edges that CCR1[EVT_SEL] selects) to close the window. The external window signal has to go to zero and back to one to enable the internal window again. The following figure shows an example that CMPO rising edge closes the internal window.

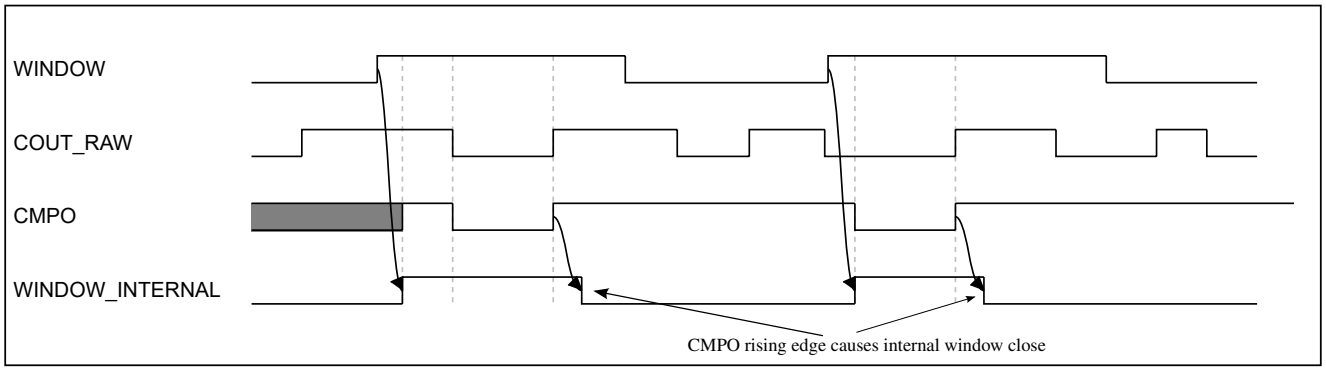


Figure 121. Windowed mode timing diagram with CMPO rising edge close window

The following figure shows that if CCR1[WINDOW_INV] becomes 1, you can invert the window signal before you use it.

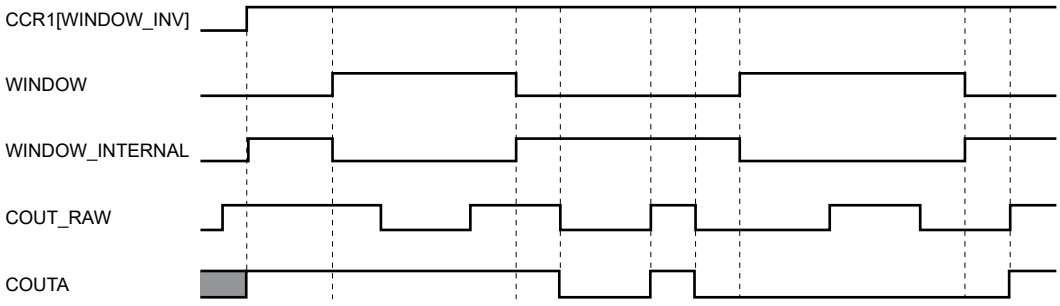


Figure 122. Windowed mode timing diagram with window signal inverted

41.4.6 Windowed/Resampled mode (#6)

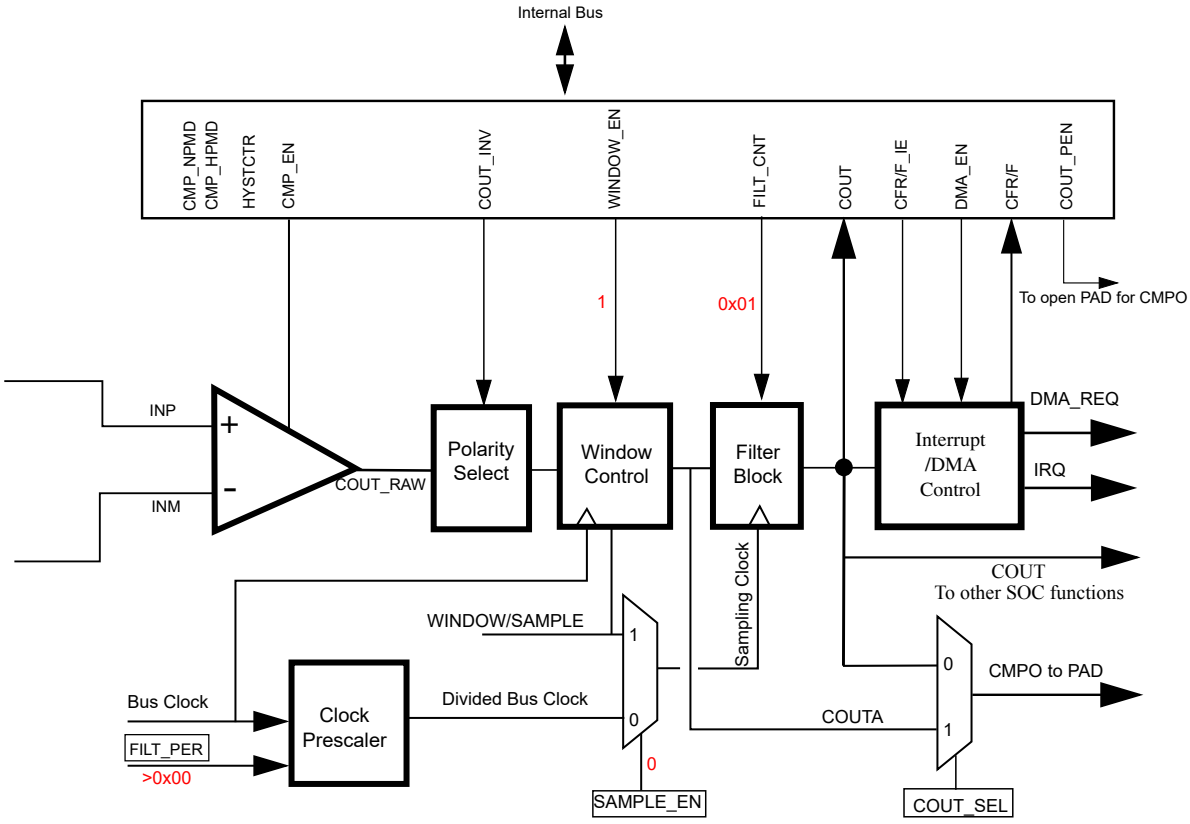


Figure 123. Windowed/Resampled mode

This mode of operation results in an unfiltered string of comparator samples where CCR1[FILT_PER] and the bus clock rate determines the interval between the samples. The following section shows that the configuration for this mode is virtually identical to that for the Windowed/Filtered mode. The only difference is that the value of CCR1[FILT_CNT] must be 1 in this mode.

The following figure uses the same input stimulus shown in Figure 118, and adds resampling of COUTA to generate COUT. The arrows in the figure indicate the time points at which the samples are taken. You can ignore prop delays and latency for clarity.

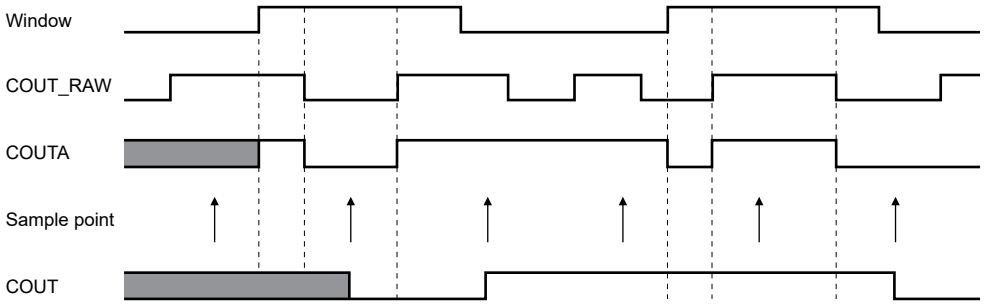


Figure 124. Windowed/Resampled mode operation

This example demonstrates the operation of the comparator in Windowed/Resampled mode, and does not reflect any specific application. Based on the sampling rate and window placement, COUT may not see zero-crossing events that the analog comparator detects. You must carefully consider the sampling period and/or window placement for a given application.

41.4.7 Windowed/Filtered mode (#7)

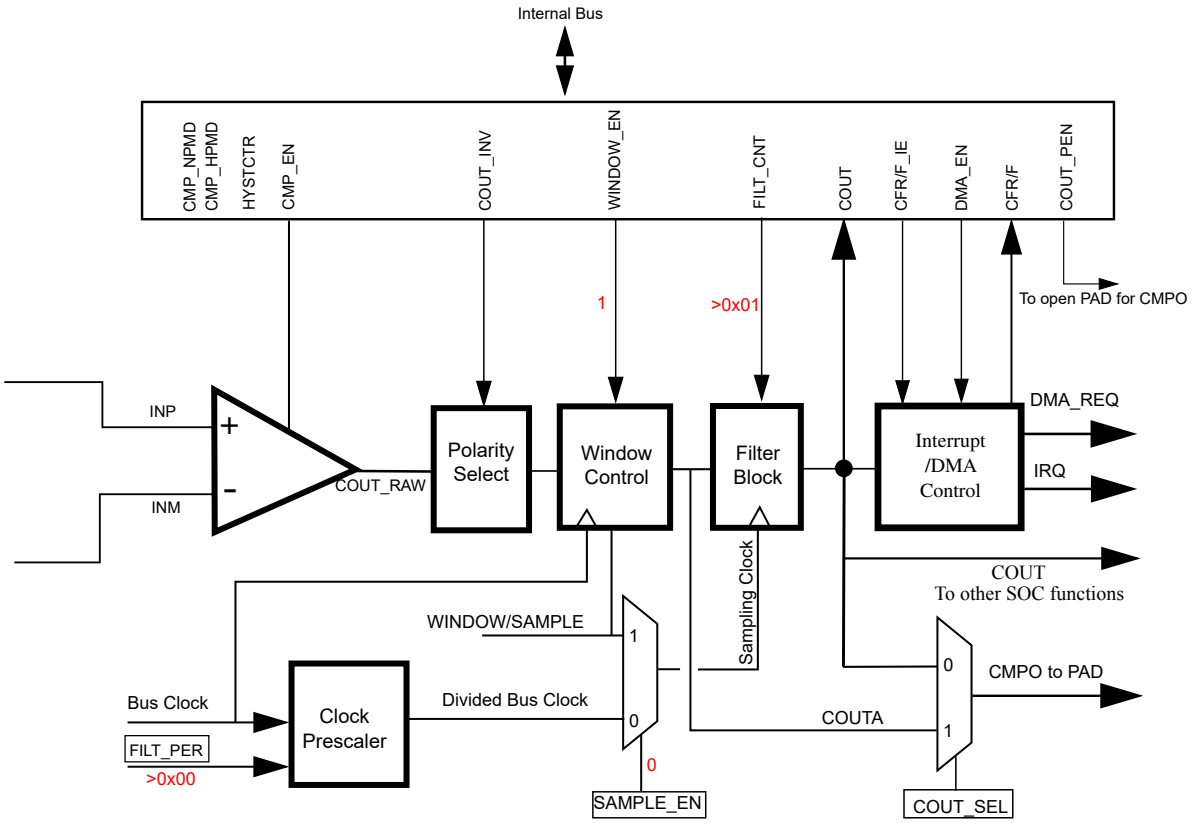


Figure 125. Windowed/Filtered mode

The only difference in operation between Windowed/Resampled mode (#6) and Windowed/Filtered mode (#7) is that CCR1[FILT_CNT] is >1, which activates filter operation.

This mode is the most complex mode of operation for the comparator block, as it utilizes both windowing and filtering features. It also has the highest latency of any of the modes. This is approximately: up to 2 peripheral clock synchronization in the window function + ((CCR1[FILT_CNT] x CCR1[FILT_PER]) + 1) x peripheral clock for the filter function.

41.5 DMA

After the DMA support enables by writing 1 to CCR1[DMA_EN] and the interrupt enables by writing 1 to IER[CFR_IE], IER[CFF_IE], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt. After the DMA completes the transfer, it sends a transfer completing indicator signal that deasserts the DMA transfer request and clears the flags (both CSR[CFR] and CSR[CFF]) to allow a subsequent change on comparator output to occur and forces another DMA request.

The comparator can remain functional in Sleep modes if CCR0[CMP_STOP_EN] becomes 1. A DMA transfer request wakes up the system from Sleep mode. After completing the data transfer, the system go back again to Sleep mode. See the DMA chapters in this document for more information on the asynchronous DMA function.

41.6 Clocking

LPCMP requires the following clocks to operate:

- Uses bus clock for register access and window/filter function.

41.7 Resets

The global chip reset signal resets LPCMP.

41.8 Interrupts

After the corresponding IER becomes 1, CSR[CFR], CSR[CFF], and can generate an interrupt, assuming that CCR1[DMA_EN] is not 1. You can clear either the flag or IER to deassert the interrupt.

41.9 External signal descriptions

Below table introduces external signals.

Table 256. External signal descriptions

Signal	Description	I/O
CMPO	Filtered or unfiltered comparator output	O
Input_Analog_Channels	Analog input channels (see the chip-specific information for more on the connections).	I
VREFH_EXT	External reference voltage for the CMP-DAC (see the chip-specific information for more on the connections).	I
RR_ACTIVE	Round-robin trigger mode enabled.	O

41.10 Initialization

You can enable LPCMP by writing 1 to CCR0[CMP_EN], and then configuring the control registers (CCR1, CCR2, DCR, and so on).

To disable LPCMP, write 0 to CCR0[CMP_EN]. Switching operation modes or changing control register fields on-the-fly (when CCR0[CMP_EN] is set to 1) may cause noise on the COUT or COUTA signals. To avoid unwanted signal noise, you must ensure to disable the module before switching modes or changing control fields.

The time required to stabilize COUT is the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function, and filter, see the datasheet for more information on power-on delays of the comparators. Table 255 specifies the delay, which the windowing and filter function causes.

During operation, you must always consider the propagation delay of the selected data paths. It can take many bus clock cycles for COUT and CSR[CFR]/CSR[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

41.11 Application information

41.11.1 Round-robin trigger mode programing recommendation

Configure the Round-robin trigger mode as follows:

1. Configure the comparison cycles by RRCR0[RR_NSAM]. Note: It is a mandatory request that the round robin cycling period must set longer than the time that all the active channels complete the specified comparison cycles set by RRCR0[RR_NSAM].
2. Configure CMP initialization delay by RRCR0[RR_INITMOD]. Note: In programming RRCR0[RR_INITMOD], the RR_INITMOD x round robin clock period must be longer than the initialization delay, see the chip datasheet for more information.
3. Configure RRCR1[FIXP] to select the fixed port of CMP and RRCR1[FIXCH] to select the fixed channel.
4. Configure channels for comparison by RRCR1[RR_CHnEN].
5. Write RRCSR[RR_CHnOUT] to define the pre-set state of channel n.
6. Clear channel flags RRSR[RR_CHnF].
7. Enable round robin interrupt by IER[RRF_IE] (disable IER[CFR_IE] and IER[CFF_IE]).
8. Enable round-robin trigger mode by setting RRCR0[RR_EN] to 1.

41.11.2 Hysteresis

The following diagram illustrates implementation of an external hysteresis resistor bridge between the asynchronous comparator output and the INP input of the Comparator. Since positive feedback is required, CCR1[COUT_INV] must be set to 0 when the hysteresis resistor bridge is added to the INP input of the comparator. CCR1[COUT_INV] must be set to 1 when the hysteresis resistor bridge is added to the INM input of the comparator.

The option of adding an external resistor bridge for the purpose of adding hysteresis to the comparator and the amount of hysteresis depends on the user's individual requirements. Hysteresis can be important in some system designs. In the absence of hysteresis, the continuous comparison of nearly identical analog inputs may add noise and waste power by generating high-frequency oscillations at CMPO.

If external hysteresis is added to the comparator, the bridge must be designed to consider other issues than simply how much hysteresis is needed. The resistor values must be sufficiently high so that they do not cause the drive strength of the digital output driver in the CMPO IO cell to be exceeded. Also, if any digital function other than CMPO must operate on the CMPO pad in the presence of such a bridge, the resistor values must be sufficiently high so that the IO cell can function appropriately in its digital role.

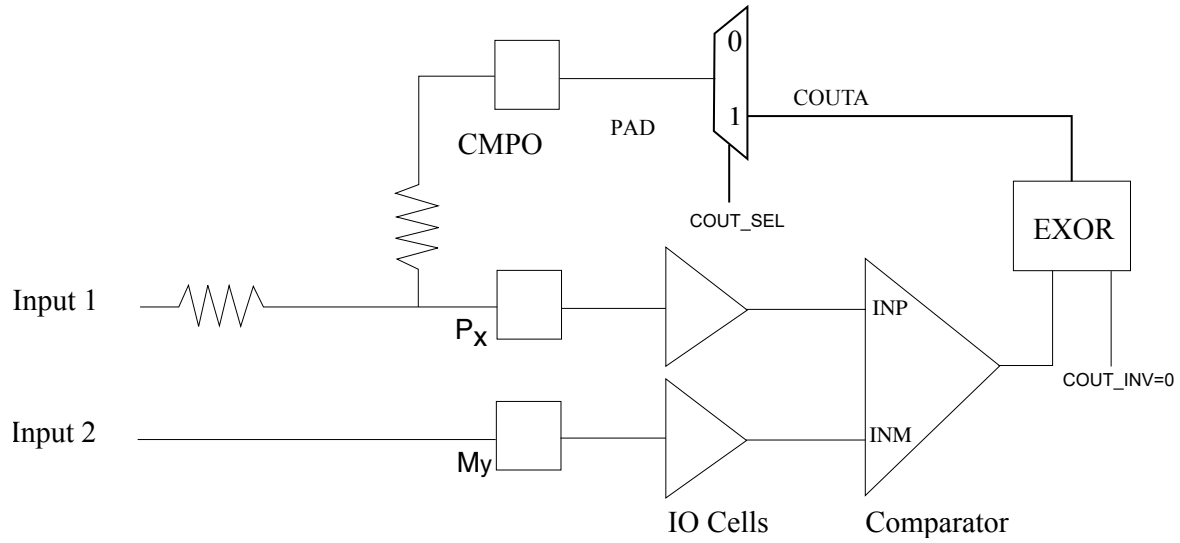


Figure 126. External hysteresis circuit

41.12 LPCMP register descriptions

The memory map comprises of 32-bit aligned registers, which you can access via 8-, 16- or 32-bit reads and 32-bit write. Attempted accesses using unsupported write data sizes, writes to read-only resources, or to reserved spaces terminate with an error. Read access to reserved address generates a transfer error and the read data bus shows all 0s.

41.12.1 LPCMP memory map

LPCMP0 base address: 4004_8000h

LPCMP1 base address: 4004_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0000h
4h	Parameter (PARAM)	32	R	0000_0002h
8h	Comparator Control Register 0 (CCR0)	32	RW	0000_0002h
Ch	Comparator Control Register 1 (CCR1)	32	RW	0000_0000h
10h	Comparator Control Register 2 (CCR2)	32	RW	0000_0000h
18h	DAC Control (DCR)	32	RW	0000_0000h
1Ch	Interrupt Enable (IER)	32	RW	0000_0000h
20h	Comparator Status (CSR)	32	RW	0000_0000h

41.12.2 Version ID (VERID)

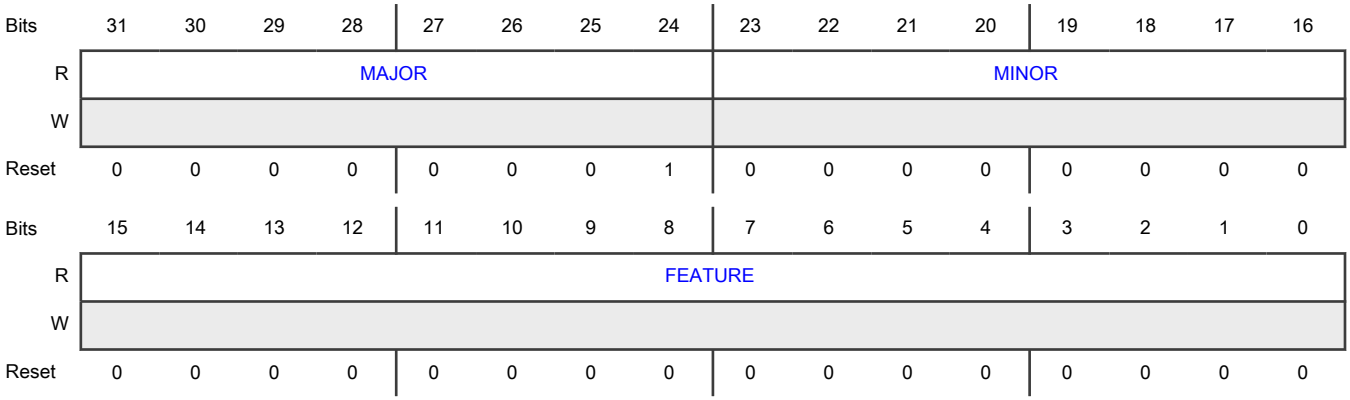
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0001b - Round robin feature

41.12.3 Parameter (PARAM)

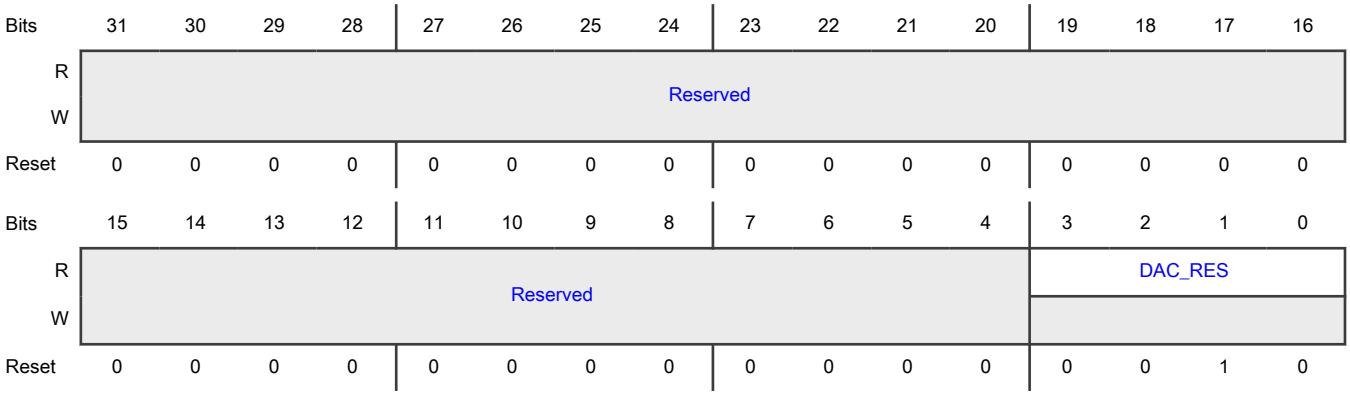
Offset

Register	Offset
PARAM	4h

Function

Contains parameter values that are implemented in the module.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 DAC_RES	DAC Resolution Indicates the DAC resolution, which this implementation supports. 0000b - 4-bit DAC 0001b - 6-bit DAC 0010b - 8-bit DAC 0011b - 10-bit DAC 0100b - 12-bit DAC 0101b - 14-bit DAC 0110b - 16-bit DAC

41.12.4 Comparator Control Register 0 (CCR0)

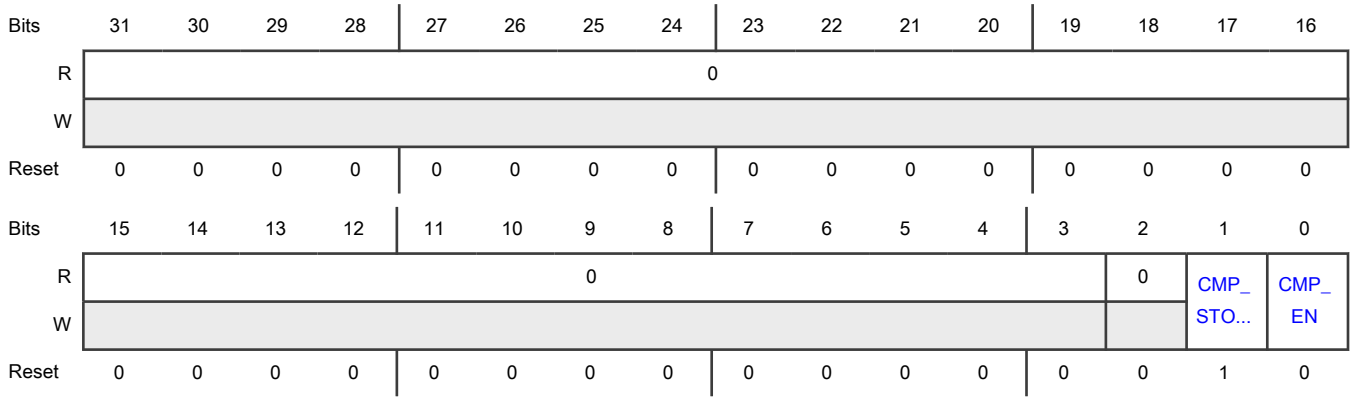
Offset

Register	Offset
CCR0	8h

Function

Contains configuration options for enabling the analog comparator.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 —	Reserved
1 CMP_STOP_EN	Comparator Sleep Mode Enable Helps you enable the analog comparator or the DAC when the chip is in Sleep mode. 0b - Disable the analog comparator regardless of CMP_EN. 1b - Allows CMP_EN to enable the analog comparator.
0 CMP_EN	Comparator Enable Enables the analog comparator. 0b - Disables (The analog logic remains off and consumes no power.) 1b - Enables

41.12.5 Comparator Control Register 1 (CCR1)

Offset

Register	Offset
CCR1	Ch

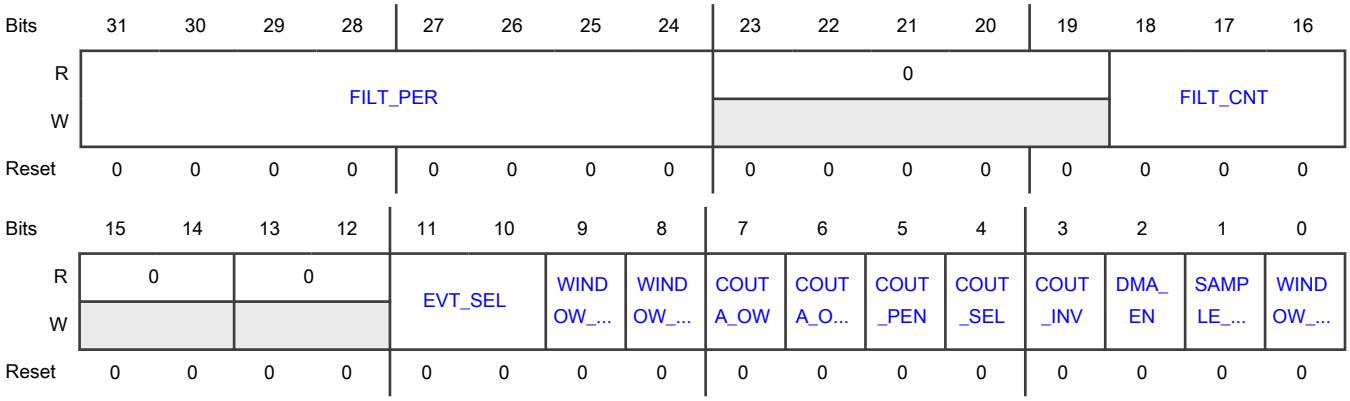
Function

Contains configuration options for the comparator operation, such as enabling Sampling or Windowing mode.

NOTE

You cannot enable Sampling and Windowing modes both at the same time. Sampling mode takes precedence over Windowing mode. If you write 1 to both SAMPLE_EN and WINDOW_EN, only SAMPLE_EN becomes 1.

Diagram



Fields

Field	Function
31-24 FILT_PER	Filter Sample Period Specifies the sampling period (in bus clock cycles) of the comparator output filter. Programming this field to 00h bypasses the filter. See Functional description for more information on filter programming and latency. **NOTE** FILT_PER has no effect in Sampling mode (CCR1[SAMPLE_EN] = 1).
23-19 —	Reserved
18-16 FILT_CNT	Filter Sample Count Specifies the number of consecutive samples that must agree before the comparator output filter accepts the sample as a new valid output state. See Functional description for more information on filter programming and latency. 000b - Filter is bypassed: COUT = COUTA 001b - 1 consecutive sample (Comparator output is simply sampled.) 010b - 2 consecutive samples 011b - 3 consecutive samples 100b - 4 consecutive samples 101b - 5 consecutive samples 110b - 6 consecutive samples 111b - 7 consecutive samples
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13-12 —	Reserved
11-10 EVT_SEL	CMPO Event Select Selects which CMPO signal edge (rising, falling, or both) defines a CMPO event. **NOTE** Valid only in Windowing mode. 00b - Rising edge 01b - Falling edge 1xb - Both edges
9 WINDOW_CLS	CMPO Event Window Close Enables a CMPO event (defined as a CMPO rising edge, falling edge, or both) to close an active window. See EVT_SEL to configure the CMPO event. **NOTE** The WINDOW signal has to go to zero and back to one again to re-activate the window. Valid only in Windowing mode. 0b - CMPO event cannot close the window 1b - CMPO event can close the window
8 WINDOW_INV	WINDOW/SAMPLE Signal Invert Inverts the window/sample signal. 0b - Do not invert 1b - Invert
7 COUTA_OW	COUTA Output Level for Closed Window Defines the COUTA signal value when the window is closed. **NOTE** Valid only in Windowing mode and when COUTA_OWEN=1. 0b - COUTA is 0 1b - COUTA is 1
6 COUTA_OWEN	COUTA_OW Enable Enables the COUTA signal value to be defined by COUTA_OW when the window is closed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Valid only in Windowing mode. 0b - COUTA holds the last sampled value 1b - COUTA_OW defines COUTA
5 COUT_PEN	Comparator Output Pin Enable Enables the comparator output to become an available signal option for a selected package pin. 0b - Not available 1b - Available
4 COUT_SEL	Comparator Output Select Selects which comparator output option, COUT or COUTA, to use for CMPO. 0b - Use COUT (filtered) 1b - Use COUTA (unfiltered)
3 COUT_INV	Comparator Invert Selects the polarity of the analog comparator function, affecting the value driven to the COUT output (on both the chip pin and as CSR[COUT]) when CCR0[CMP_EN] is 0. 0b - Do not invert 1b - Invert
2 DMA_EN	DMA Enable Enables DMA transfers triggered from the LPCMP module. After this field and the corresponding interrupt enable field becomes 1, a DMA request is asserted when CFR or CFF becomes 1. 0b - Disables 1b - Enables
1 SAMPLE_EN	Sampling Enable Enables Sampling mode. 0b - Disables 1b - Enables
0 WINDOW_EN	Windowing Enable Enables Windowing mode. **NOTE** Valid only when SAMPLE_EN = 0. 0b - Disables 1b - Enables

41.12.6 Comparator Control Register 2 (CCR2)

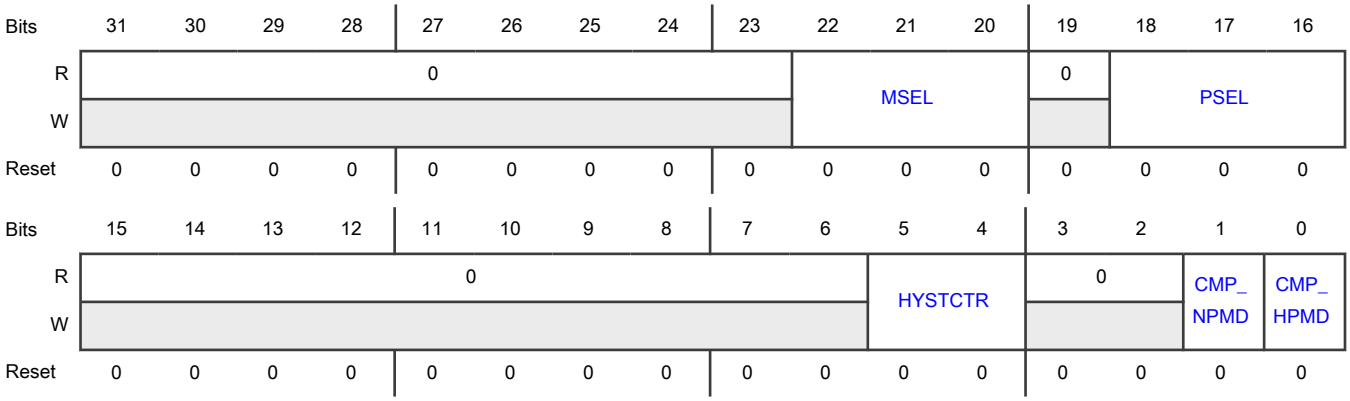
Offset

Register	Offset
CCR2	10h

Function

Contains the configuration options for the comparator operation, such as selecting the plus and minus comparator inputs and the hysteresis levels.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 MSEL	Minus Input MUX Select Selects the input used for the negative mux. See the chip-specific LPCMP information for more on connections. 000b - Input 0m 001b - Input 1m 010b - Input 2m 011b - Input 3m 100b - Input 4m 101b - Input 5m 110b - Reserved 111b - Internal DAC output

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 —	Reserved
18-16 PSEL	Plus Input MUX Select Selects the input used for the positive mux. See the chip-specific LPCMP information for more on connections. 000b - Input 0p 001b - Input 1p 010b - Input 2p 011b - Input 3p 100b - Input 4p 101b - Input 5p 110b - Reserved 111b - Internal DAC output
15-6 —	Reserved
5-4 HYSTCTR	Comparator Hysteresis Control Selects the level of internally generated hysteresis for the comparator output. See the chip datasheet to get the specific values for each hysteresis level. 00b - Level 0 01b - Level 1 10b - Level 2 11b - Level 3
3-2 —	Reserved
1 CMP_NPMD	CMP Nano Power Mode Select Enables Nano Power mode for the comparator. 0b - Disables (CCR2[CMP_HPMD] determines the mode). 1b - Enables
0 CMP_HPMD	CMP High Power Mode Select Selects Low or High Power(Speed) mode for the comparator. **NOTE** Valid only when not in Nano Power mode (CMP_NPMD = 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Low power (speed) comparison mode 1b - High power (speed) comparison mode

41.12.7 DAC Control (DCR)

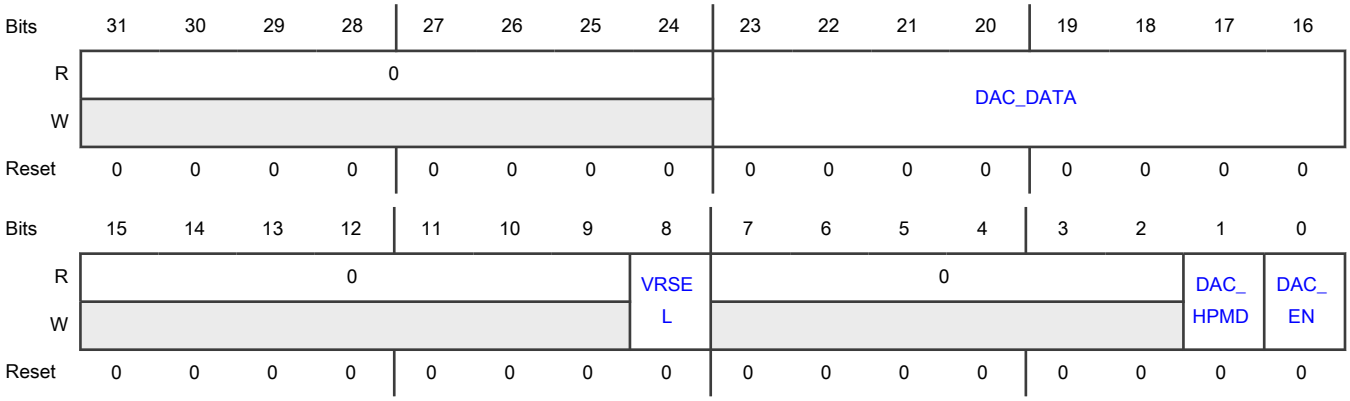
Offset

Register	Offset
DCR	18h

Function

Contains the configuration options to enable the DAC.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 DAC_DATA	DAC Output Voltage Select Selects the DAC output (DACO) voltage from one of 256 distinct levels. The DACO ranges from Vin/256 to Vin: DACO = (Vin/256) * (DAC_DATA + 1)
15-9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 VRSEL	DAC Reference High Voltage Source Select Selects the high voltage reference source for the Vin supply of the DAC's resistor ladder network. See the chip-specific LPCMP information for the source of vrefh0 and vrefh1. 0b - vrefh0 1b - vrefh1
7-2 —	Reserved
1 DAC_HPMD	DAC High Power Mode Select Enables the DAC high power mode. 0b - Disables 1b - Enables
0 DAC_EN	DAC Enable Enables the DAC. When disabled, power-down the DAC to conserve power. 0b - Disables 1b - Enables

41.12.8 Interrupt Enable (IER)

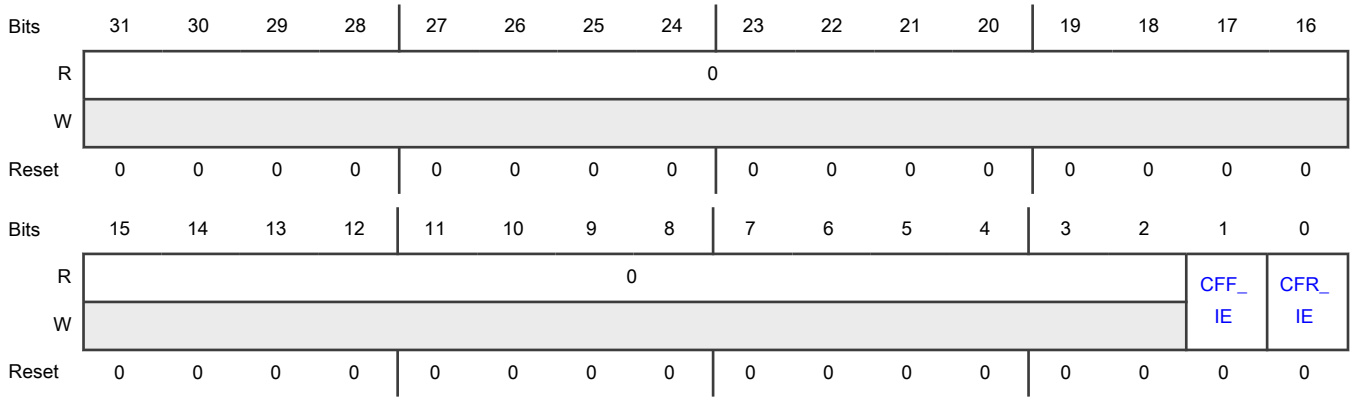
Offset

Register	Offset
IER	1Ch

Function

Provides enable fields for the comparator and round-robin flag interrupts.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 CFF_IE	Comparator Flag Falling Interrupt Enable Enables the comparator flag falling interrupt. 0b - Disables 1b - Enables: Assert an interrupt when CFF sets.
0 CFR_IE	Comparator Flag Rising Interrupt Enable Enables the comparator flag rising interrupt. 0b - Disables 1b - Enables: Assert an interrupt when CFR sets.

41.12.9 Comparator Status (CSR)

Offset

Register	Offset
CSR	20h

Function
Provides COUT, CFF, CFR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							COUT	0						CFF	CFR
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 COUT	Analog Comparator Output Returns the current value of the analog comparator output when read. This field resets to 0 and reads as CCR1[COUT_INV] after the analog comparator module disables when CCR0[CMP_EN] = 0. Writing to this field is ignored.
7-2 —	Reserved
1 CFF	Analog Comparator Flag Falling Detects when a falling edge on COUT occurs. Write 1 to clear this field when CCR1[DMA_EN] is disabled. If CCR1[DMA_EN] is enabled, the flag automatically clears after DMA is done. This field clears when CCR0[CMP_EN] is not 1. 0b - Not detected 1b - Detected
0 CFR	Analog Comparator Flag Rising Detects when a rising edge on COUT occurs. Write 1 to clear this field when CCR1[DMA_EN] is disabled. If CCR1[DMA_EN] is enabled, the flag automatically clears after DMA is done. This field clears when CCR0[CMP_EN] is not 1. 0b - Not detected 1b - Detected

Chapter 42
Voltage Reference (VREF)

42.1 Chip-specific VREF information

Table 257. Reference links to related information

Topic	Related module	Reference
Full description	VREF	VREF
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

NOTE

The VREF_OUT signal introduced in this chapter is the VREFO signal in the Signal Muxing and Pinout.

42.1.1 Module instances

This device has one instance of the VREF module, VREF0.

42.2 Overview

VREF provides a buffered reference voltage that you can set to levels ranging from 1.0 V to 2.1 V for use as an external reference. When VREF is enabled, the reference voltage, VREF_OUT, is connected to a dedicated output pin. In addition, the buffered reference is available internally for use with on-chip peripherals.

You can trim the VREF voltage output in fine or coarse voltage steps:

- For fine-tuning, use UTRIM[VREFTRIM] to trim the output with (0.05% x VREF_OUT) resolution, where VREF_OUT is the value of the voltage reference output (measured in volts) after using UTRIM[TRIM2V1] to trim the output. For example, after using UTRIM[TRIM2V1] to trim the output VREF_OUT = 1 V, the fine step resolution is : (0.05% x 1) = 0.5 mV.
- For coarse-tuning, use UTRIM[TRIM2V1] to trim the output with 100 mV resolution. For details about possible configurations of the voltage reference, see Table 258

42.2.1 Features

- Programmable User Trim (UTRIM) with 0.05% x VREF_OUT steps in fine-tuning and 100 mV in coarse-tuning. When reset, User Trim (UTRIM) is automatically loaded with a factory-trimmed value.
- Programmable Buffer mode selection:
  - Off
  - Standby mode (Bandgap enabled and output buffer disabled)
  - High-Power Buffer mode (output buffer enabled in High-Power mode)
  - Low-Power Buffer mode (output buffer enabled in Low-Power mode)
- Low-power bandgap and high-accurate bandgap selection
- Support for a dedicated output pin, VREF_OUT

42.2.2 Block diagram

Figure 127 shows the VREF block diagram.

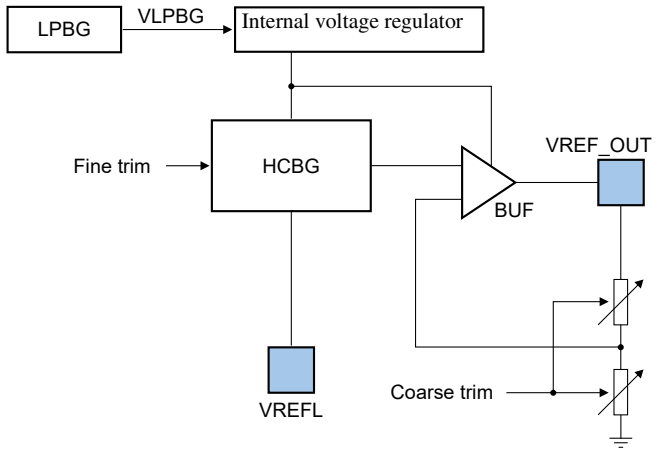


Figure 127. VREF block diagram

42.3 Functional description

VREF is a bandgap buffer system that uses unity gain amplifiers. Both internal and external peripherals use the VREF_OUT signal in Low- and High-Power Buffer modes.

NOTE

When using VREF, a 220 nF capacitor must always be connected between VREF_OUT and Analog Domain Power Supply Ground.

If CSR[HCBGEN] = 1, the voltage reference is enabled, you must set different modes by using CSR[BUF21EN] and CSR[HI_PWR_LV].

The following table shows possible configurations of the voltage reference.

Table 258. VREF_OUT values and related settings

UTRIM[TRIM2V1]	VREF_OUT (V)
0000	1.0
0001	1.1
0010	1.2
0011	1.3
0100	1.4
0101	1.5
0110	1.6
0111	1.7
1000	1.8
1001	1.9
1010	2.0
1011	2.1
11XX	1.0

42.3.1 Voltage reference disabled

When CSR[LPBGEN] = 0, CSR[HCBGEN] = 0, and CSR[BUF21EN] = 0, the voltage reference is disabled and the VREF bandgap and output buffers are disabled. VREF is in Off mode.

42.3.2 Voltage reference enabled

Enabling voltage reference results into the following main high-level configuration options:

- Write 1 to CSR[LPBGEN] and CSR[LPBG_BUF_EN]. This configuration enables the low power bandgap (LPBG) bias current and voltage output.
- Write 1 to CSR[LPBGEN], CSR[REGEN], CSR[BUF21EN], and CSR[HCBGEN], this configuration enables the VREF supply voltage on VREF_OUT.

42.3.2.1 CSR[BUF21EN] = 0, CSR[HI_PWR_LV] = X

In Standby mode, the internal VREF bandgap is enabled to generate an accurate 1.0 V output that you can trim by using UTRIM[VREFTRIM]. The bandgap requires time for startup and stabilization. Monitor CSR[VREFST] to determine whether the stabilization and startup are complete when the chop oscillator is not enabled. When chop oscillator is used, the internal bandgap reference voltage settles within the chop oscillator startup time, Tchop_osc_stup.

The output buffer is disabled in this mode, and there is no buffered voltage output. VREF is in Standby mode. If you select this mode first and Low-Power or High-Power Buffer mode is subsequently enabled, there is a delay before the buffer output settles to its final value. This is the buffer startup delay (Tstup), as specified in the appropriate chip data sheet.

42.3.2.2 CSR[BUF21EN] = 1, CSR[HI_PWR_LV] = 1

In High-Power buffer mode, the internal VREF bandgap is on, and the High-Power Buffer is enabled to generate a buffered voltage to VREF_OUT. The available voltage levels range from 1.0 to 2.1 V and can be adjusted in 0.1 V steps using UTRIM[TRIM2V1]. VREF bandgap can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from Standby mode (CSR[BUF21EN] = 0), there is a delay before the buffer output settles to its final value. This is the buffer startup delay (Tstup), as specified in the appropriate device data sheet.

If this mode is entered when VREF is enabled, you must wait longer than Tstup or until CSR[VREFST] = 1 when the chop oscillator is not enabled. If the chop oscillator is being used, wait for the time specified by Tchop_osc_stup (chop oscillator startup time) to ensure that the VREF output has stabilized.

NOTE

In this mode, a 220-nF capacitor is required to connect between the VREF_OUT pin and Analog Domain Power Supply Ground.

42.3.2.3 CSR[BUF21EN] = 1, CSR[HI_PWR_LV] = 0

In Low-Power buffer mode, the internal VREF bandgap is on, and the low-power buffer is enabled to generate a buffered voltage to VREF_OUT. The available voltage levels range from 1.0 to 2.1 V and you can adjust them in 0.1 V steps using UTRIM[TRIM2V1]. VREF bandgap can be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from Standby mode (CSR[BUF21EN] = 0), there is a delay before the buffer output settles to its final value. This is the buffer startup delay (Tstup), as specified in the appropriate chip data sheet.

If this mode is entered when the VREF module is enabled, you must wait longer than Tstup or until CSR[VREFST] = 1 when the chop oscillator is not enabled. If the chop oscillator is being used, wait for the time specified by Tchop_osc_stup (chop oscillator startup time) to ensure that the VREF output has stabilized.

NOTE

In this mode, a 220-nF capacitor connects the VREF_OUT pin and Analog Domain Power Supply Ground.

42.3.3 Internal voltage regulator

VREF contains an internal voltage regulator that you can enable to provide additional supply noise rejection. The module also contains an internal chop oscillator to generate the chop clock. When possible, enable the internal regulator and chop oscillator to attain optimum VREF performance.

When using the chop function, you must also enable the internal voltage regulator. Perform the following procedure when enabling the internal regulator:

1. Enable the internal regulator by writing 1 to CSR[REGEN].
2. Enable the chop oscillator by writing 1 to CSR[CHOPEN].
3. Write 1 to CSR[LPBGEN] and CSR[HCBGEN].

42.3.4 Low-Power modes

VREF can run when the chip is in a low-power mode. To use the VREF regulator and/or the chop oscillator in a low-power mode, the system reference voltage (also known as the bandgap voltage reference) must also be enabled in these modes. See the chip-specific VREF information on how to enable the system reference voltage.

Having the VREF regulator enabled increases the current consumption. Depending on the requirements of the system application, consider disabling the VREF regulator to minimize the current consumption in low-power modes. However, not using the VREF regulator reduces the accuracy of the output voltage by as much as several mV.

42.3.5 Clocking

See the chip-specific VREF information for the required clocking.

42.3.6 Reset

System reset, VREF convert to initial state with buffer cleared and waiting to load new register.

42.3.7 Interrupts

This module has no interrupts.

42.4 External signals

Table 259. External signals

Signal	Description	I/O
VREF_OUT	Internally generated voltage reference output. When the VREF output buffer is disabled, the VREF_OUT signal state is high-impedance.	O

42.5 Initialization

VREF_OUT requires time for startup and stabilization. After writing 1 to CSR[HCBGEN], monitor CSR[VREFST] to determine whether the stabilization and startup are complete when the chop oscillator is not enabled. If the chop oscillator is enabled, Tchop_osc_stup defines the settling time of the internal bandgap reference. You must wait for this time (Tchop_osc_stup) after the internal bandgap is enabled to ensure that the VREF internal reference voltage is stabilized.

After VREF_OUT is enabled and stabilized, changing the value of CSR[HI_PWR_LV] temporarily destabilizes the output voltage at the VREF_OUT pin, even though CSR[VREFST] remains 1. The time it takes for VREF_OUT to settle again is the same as the buffer startup delay (Tstup), as specified in the appropriate device data sheet. Also, applying a step change of the load current to the VREF_OUT pin incurs some time delay to resettle.

To enable the internal 1.75 V VREF regulator, see Internal voltage regulator for the required sequence. When the internal regulator is disabled, the VREF_OUT voltage is more sensitive to supply voltage variation. Use the internal regulator to achieve optimum VREF_OUT performance.

NOTE

You must write 1 to CSR[CHOPEN], CSR[REGEN], and CSR[ICOMPEN] to achieve the performance stated in the device data sheet.

42.6 Register descriptions

42.6.1 VREF register descriptions

42.6.1.1 VREF memory map

VREF0 base address: 4004_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0000h
8h	Control and Status (CSR)	32	RW	0000_0000h
10h	User Trim (UTRIM)	32	RW	0000_0000h

42.6.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number.

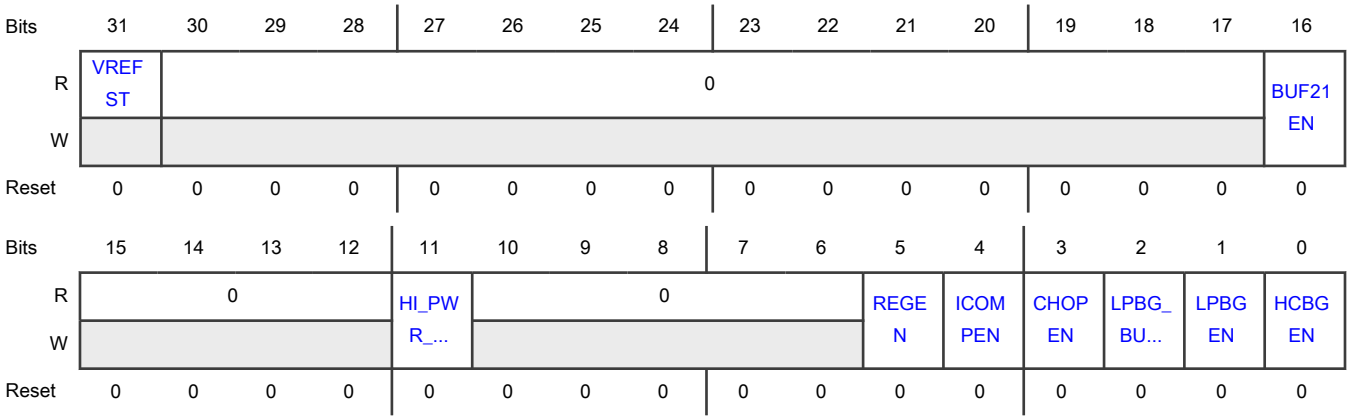
42.6.1.3 Control and Status (CSR)

Offset

Register	Offset
CSR	8h

Function
Contains the VREF control and status fields.

Diagram



Fields

Field	Function
31 VREFST	Internal HC Voltage Reference Stable Indicates whether the bandgap reference within VREF has completed its startup and stabilization. **NOTE** This field is valid only when you do not use the chop oscillator.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled and unstable 1b - Stable
30-17 —	Reserved
16 BUF21EN	Internal Buffer21 Enable Enables the Buffer21 programmable output voltage from 1.0 to 2.1 V. 0b - Disables 1b - Enables
15-12 —	Reserved
11 HI_PWR_LV	High-Power Level Controls the power mode of Buffer21. 0b - Low-power 1b - High-power
10-6 —	Reserved
5 REGEN	Regulator Enable Enables the internal 1.75 V regulator to produce a constant internal voltage supply to reduce the sensitivity to external supply noise and variation. See Internal voltage regulator for the required sequence to enable the internal regulator. To keep the regulator enabled in low-power modes, see the chip-specific VREF information for details. **NOTE** You must write 1 to this field to achieve the performance stated in the data sheet. 0b - Disables 1b - Enables
4 ICOMPEN	Current Compensation Enable Enables second-order curvature compensation. **NOTE** You must write 1 to this field to achieve the performance stated in the data sheet. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 CHOPEN	Chop Oscillator Enable Enables the chop oscillator. When set, the internal chopping operation is enabled and the internal analog offset is minimized. When using the internal voltage regulator CSR[REGEN] = 1, you must also enable the chop oscillator. If the chop oscillator is used in low-power modes, you must also enable the HC bandgap voltage reference. See the chip-specific VREF information for details. 0b - Disables 1b - Enables
2 LPBG_BUF_EN	Low-Power Bandgap Buffer Enable Enables the low-power buffer for the low-power bandgap with latch function enabled. **NOTE** You can enable this field only when CSR[LPBGEN] = 1. 0b - Disables 1b - Enables
1 LPBGEN	Low-Power Bandgap Enable Enables the low-power bandgap. 0b - Disables 1b - Enables
0 HCBGEN	HC Bandgap Enabled Enables the HC bandgap. **NOTE** You can enable this field only when CSR[LPBGEN] = 1. 0b - Disables 1b - Enables

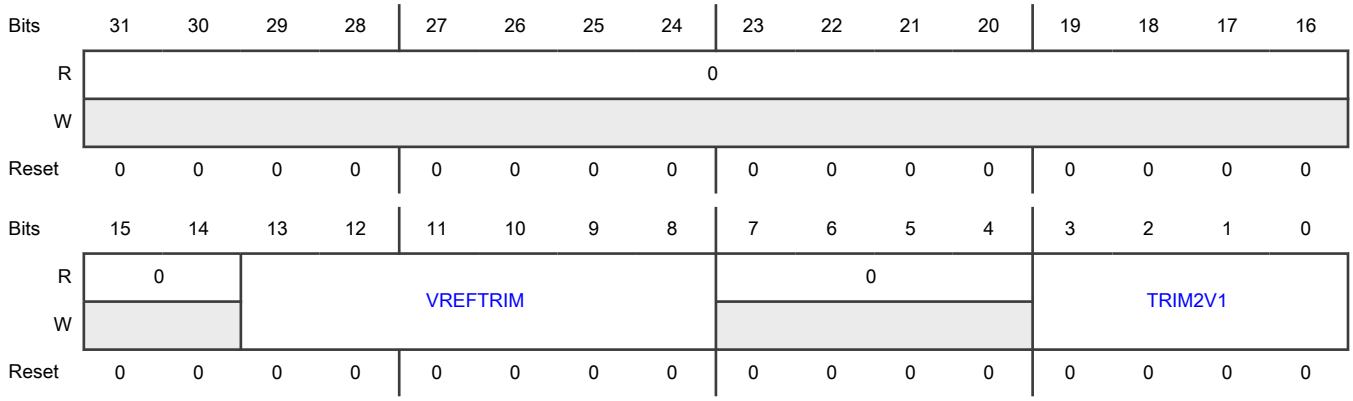
42.6.1.4 User Trim (UTRIM)

Offset

Register	Offset
UTRIM	10h

Function
Contains the trim data. You can read and write to this register in User mode. After reset, UTRIM is automatically loaded with a factory-trimmed value.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-8 VREFTRIM	VREF Trim This is an unsigned number that adjusts the VREF output by (0.5 x VREF_OUT_Nom) mV, where VREF_OUT_Nom is the target output determined by the TRIM2V1 setting. For example, if TRIM2V1 is set to output 1.0 V, VREFTRIM adjusts the output in 0.5 mV steps.
7-4 —	Reserved
3-0 TRIM2V1	VREF 2.1 V Trim Changes the resulting VREF output by approximately ± 100 mV for each step. For details about possible configurations of the voltage reference, see Table 258

Chapter 43
Real Time Clock (RTC)

43.1 Chip-specific RTC information

Table 260. Reference links to related information

Topic	Related module	Reference
Full description	RTC	RTC
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

43.1.1 Module instances

- This device has one instance of the RTC module, RTC0.
- In this device, the independent RTC supply is internally tied to VDD_IO_D.
- Not all the tamper pins are available for each package, see #unique_1319 for details.

43.1.2 RTC_CLKOUT and RTC_WAKEUP_b

This device has no direct pinout for RTC_CLKOUT or RTC_WAKEUP_b, but the TAMPER pins can be configured to get these signals:

- CR[CPE] can be used to enable RTC_CLKOUT function on TAMPER1 or TAMPER2 if available
- CR[WPE] can be used to enable RTC_WAKEUP_b function on TAMPER0 if available.

43.2 Introduction

43.2.1 Features

The RTC module features include:

- 32-bit seconds counter with roll-over protection and 32-bit alarm
- 16-bit prescaler with compensation that can correct errors between 0.12 ppm and 3906 ppm
- Register write protection
  - Lock register requires POR or software reset to enable write access
- Configurable 1, 2, 4, 8, 16, 32, 64 or 128 Hz square wave output with optional interrupt
- 64-bit monotonic counter with roll-over protection
- Up to 4 tamper input pins with optional interrupt and seconds timestamp when interrupt asserts

43.2.2 Modes of operation

The RTC remains functional in all low power modes and can generate an interrupt to exit any low power mode.

43.2.3 RTC signal descriptions

Table 261. RTC signal descriptions

Signal	Description	I/O
RTC_CLKOUT	Prescaler square-wave output or RTC 32.768 kHz clock	O
RTC_WAKEUP_b	Active low wakeup for external device	O
RTC_TAMPER[3:0]	Tamper pin input	I

43.2.3.1 RTC clock output

The RTC_CLKOUT signal can output either a square wave prescaler output (configurable to 1, 2, 4, 8, 16, 32, 64 or 128 Hz) or the RTC 32.768 kHz clock.

43.2.3.2 RTC wakeup pin

The RTC wakeup pin is an open drain, active low, output that allows the RTC to wakeup the chip via an external component. The wakeup pin asserts when the wakeup pin enable is set and either the RTC interrupt is asserted or the wakeup pin is turned on via a register bit. The wakeup pin does not assert from the RTC seconds interrupt.

NOTE

The wakeup pin is optional and may not be implemented on all devices.

43.2.3.3 RTC Tamper pins

The RTC tamper pins are input pins that can be used to generate an interrupt or invalidate the time counter. The first tamper pin to assert an interrupt will also capture a timestamp of the seconds register.

43.3 Register definition

All registers must be accessed using 32-bit writes and all register accesses incur three wait states.

Writing to a register protected by the lock register does not generate a bus error, but the writes will not be completed.

43.3.1 RTC register descriptions

43.3.1.1 RTC memory map

RTC base address: 4002_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RTC Time Seconds Register (TSR)	32	RW	0000_0000h
4h	RTC Time Prescaler Register (TPR)	32	RW	0000_0000h
8h	RTC Time Alarm Register (TAR)	32	RW	0000_0000h
Ch	RTC Time Compensation Register (TCR)	32	RW	0000_0000h
10h	RTC Control Register (CR)	32	RW	0000_0002h
14h	RTC Status Register (SR)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	RTC Lock Register (LR)	32	RW	000F_FFFFh
1Ch	RTC Interrupt Enable Register (IER)	32	RW	0000_0007h
20h	RTC Tamper Time Seconds Register (TTSR)	32	R	See section
24h	RTC Monotonic Enable Register (MER)	32	RW	0000_0000h
28h	RTC Monotonic Counter Low Register (MCLR)	32	RW	0000_0000h
2Ch	RTC Monotonic Counter High Register (MCHR)	32	RW	0000_0000h
34h	RTC Tamper Detect Register (TDR)	32	RW	0000_0001h
3Ch	RTC Tamper Interrupt Register (TIR)	32	RW	0000_0000h
40h - 4Ch	RTC Pin Configuration Register (PCR0 - PCR3)	32	RW	0000_0000h

43.3.1.2 RTC Time Seconds Register (TSR)

Offset

Register	Offset
TSR	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TSR	Time Seconds Register When the time counter is enabled, the TSR is read only and increments once a second provided SR[TOF] or SR[TIF] are not set. The time counter will read as zero when SR[TOF] or SR[TIF] are set.

Table continues on the next page...

Field	Function
	When the time counter is disabled, the TSR can be read or written. Writing to the TSR when the time counter is disabled will clear the SR[TOF] and/or the SR[TIF]. Writing to TSR with zero is supported, but not recommended because TSR will read as zero when SR[TIF] or SR[TOF] are set (indicating the time is invalid).

43.3.1.3 RTC Time Prescaler Register (TPR)

Offset

Register	Offset
TPR	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TPR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

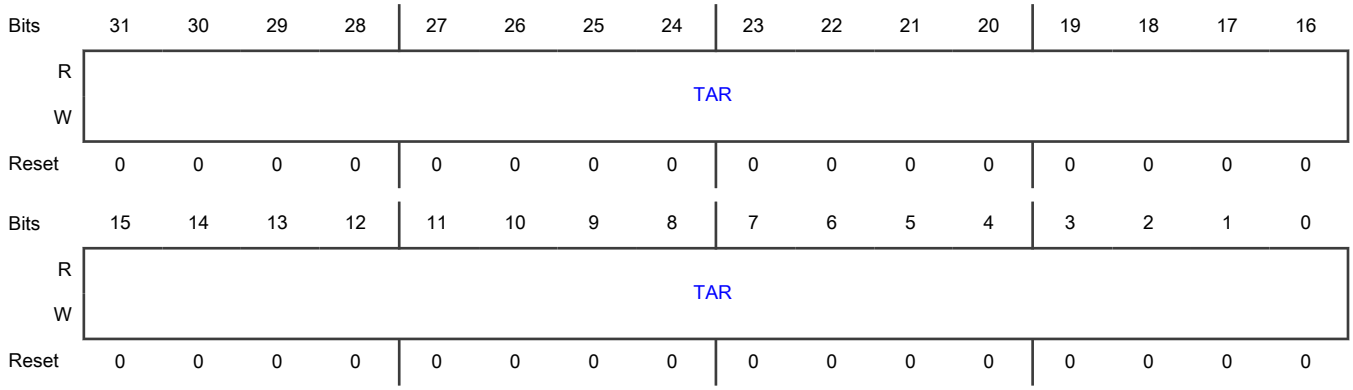
Field	Function
31-16 —	Reserved
15-0 TPR	Time Prescaler Register When the time counter is enabled, the TPR is read only and increments every 32.768 kHz clock cycle. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TPR can be read or written. The TSR[TSR] increments when bit 14 of the TPR transitions from a logic one to a logic zero.

43.3.1.4 RTC Time Alarm Register (TAR)

Offset

Register	Offset
TAR	8h

Diagram



Fields

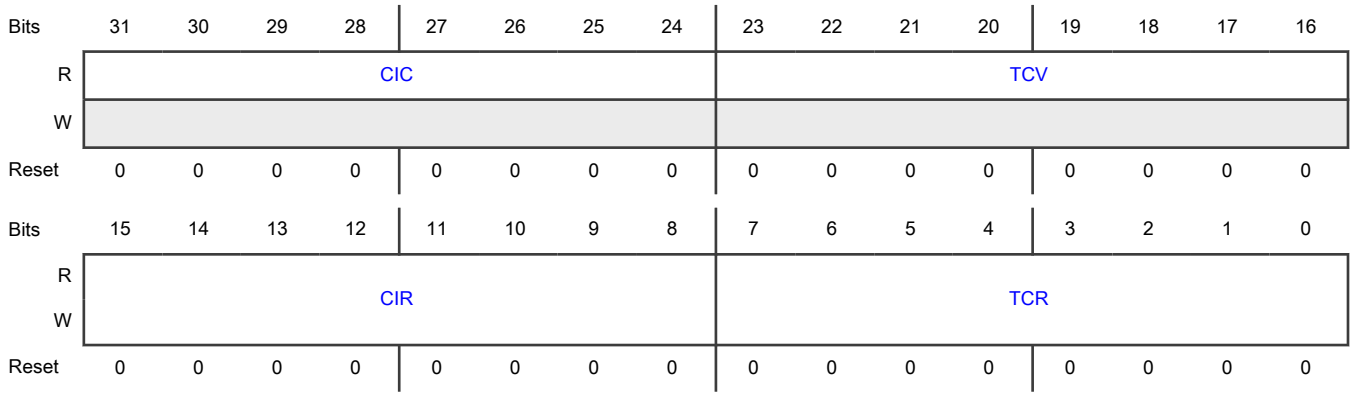
Field	Function
31-0 TAR	Time Alarm Register When the time counter is enabled, the SR[TAF] is set whenever the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. Writing to the TAR clears the SR[TAF].

43.3.1.5 RTC Time Compensation Register (TCR)

Offset

Register	Offset
TCR	Ch

Diagram



Fields

Field	Function
31-24	Compensation Interval Counter

Table continues on the next page...

Table continued from the previous page...

Field	Function
CIC	Current value of the compensation interval counter. If the compensation interval counter equals zero then it is loaded with the contents of the CIR. If the CIC does not equal zero then it is decremented once a second. Reading this register at the same time as the seconds counter is incrementing can result in an incorrect value being read.
23-16 TCV	Time Compensation Value Current value used by the compensation logic for the present second interval. Updated once a second if the CIC equals 0 with the contents of the TCR field. If the CIC does not equal zero then it is loaded with zero (compensation is not enabled for that second increment). Reading this register at the same time as the seconds counter is incrementing can result in an incorrect value being read.
15-8 CIR	Compensation Interval Register Configures the compensation interval in seconds from 1 to 256 to control how frequently the TCR should adjust the number of 32.768 kHz cycles in each second. The value written should be one less than the number of seconds. For example, write zero to configure for a compensation interval of one second. This register is double buffered and writes do not take affect until the end of the current compensation interval.
7-0 TCR	Time Compensation Register Configures the number of 32.768 kHz clock cycles in each second, equal to 32,768 - TCR (where TCR is a twos complement sign extended value). This register is double buffered and writes do not take affect until the end of the current compensation interval. Some example values are show below. 0000_0000b - Time Prescaler Register overflows every 32768 clock cycles. 0000_0001b - Time Prescaler Register overflows every 32767 clock cycles. 0111_1110b - Time Prescaler Register overflows every 32642 clock cycles. 0111_1111b - Time Prescaler Register overflows every 32641 clock cycles. 1000_0000b - Time Prescaler Register overflows every 32896 clock cycles. 1000_0001b - Time Prescaler Register overflows every 32895 clock cycles. 1111_1111b - Time Prescaler Register overflows every 32769 clock cycles.

43.3.1.6 RTC Control Register (CR)

Offset

Register	Offset
CR	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						CPE		0						Reserved	
W															0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	CLKO	Reserved	Reserved	0	CPS	Reserved	UM	Reserved	WPE	SWR
W	0	0	0	0	0	0		0	0			0		0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-26 —	Reserved
25-24 CPE	Clock Pin Enable 00b - The RTC_CLKOUT function is disabled. 01b - Enable RTC_CLKOUT function on RTC_TAMPER[1]. 10b - Enable RTC_CLKOUT function on RTC_TAMPER[2]. 11b - Enable RTC_CLKOUT function on RTC_TAMPER[3].
23-18 —	Reserved
17-16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10 —	Reserved
9 CLKO	Clock Output 0b - The 32 kHz clock is output to other peripherals. 1b - The 32 kHz clock is not output to other peripherals.
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 CPS	Clock Pin Select 0b - The prescaler output clock (as configured by TSIC) is output on RTC_CLKOUT. 1b - The RTC 32.768 kHz clock is output on RTC_CLKOUT, provided it is output to other peripherals.
4 —	Reserved
3 UM	Update Mode Allows SR[TCE] to be written even when the Status Register is locked. When set, the SR[TCE] can be written if the SR[TIF] or SR[TOF] are set or if the SR[TCE] is clear. Whenever both SR[TCE] and CR[UM] are set, then SR[TCE] should only be written once either SR[TIF] or SR[TOF] are set. Allows the monotonic enable register to be written when it is locked. When set, the monotonic enable register can always be written if the SR[TIF] or SR[MOF] are set or if the montonic counter enable is clear. Allows the tamper detect register to be written when it is locked. When set, the tamper detect register can always be written if the SR[TIF] is clear. 0b - Registers cannot be written when locked. 1b - Registers can be written when locked under limited conditions.
2 —	Reserved
1 WPE	Wakeup Pin Enable The RTC_WAKEUP pin is optional and not available on all devices.

Table continues on the next page...

Table continued from the previous page...

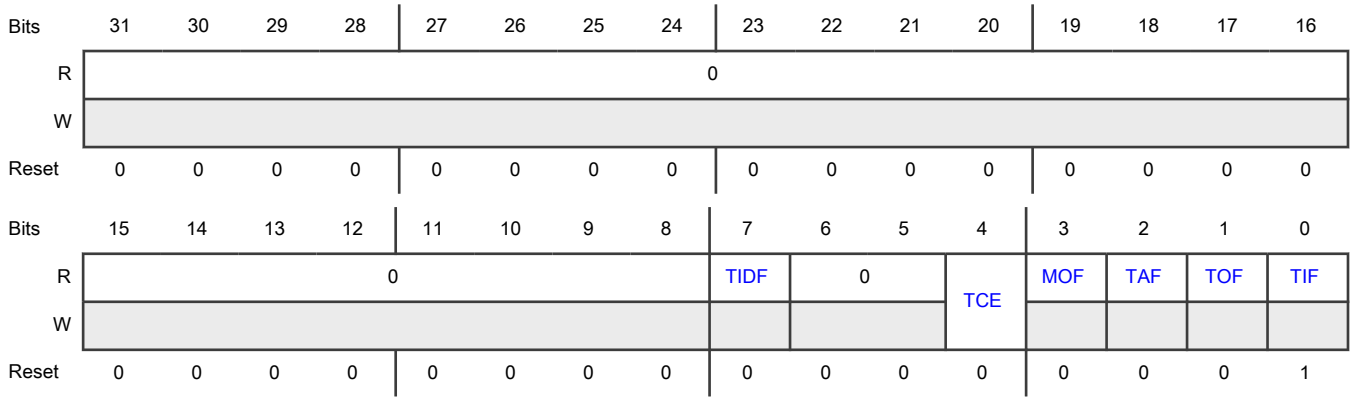
Field	Function
	0b - RTC_WAKEUP pin is disabled. 1b - RTC_WAKEUP pin is enabled and asserts if the RTC interrupt asserts or if the wakeup pin is forced on.
0 SWR	Software Reset 0b - No effect. 1b - Resets all RTC registers except for the SWR bit . The SWR bit is cleared by POR and by software explicitly clearing it.

43.3.1.7 RTC Status Register (SR)

Offset

Register	Offset
SR	14h

Diagram



Fields

Field	Function
31-8 —	Reserved
7 TIDF	Tamper Interrupt Detect Flag The Tamper Interrupt Detect Flag is set whenever the Tamper interrupt is asserted, as configured by the Tamper Status Register and Tamper Interrupt Enable Register. 0b - Tamper interrupt has not asserted.

Table continues on the next page...

Table continued from the previous page...

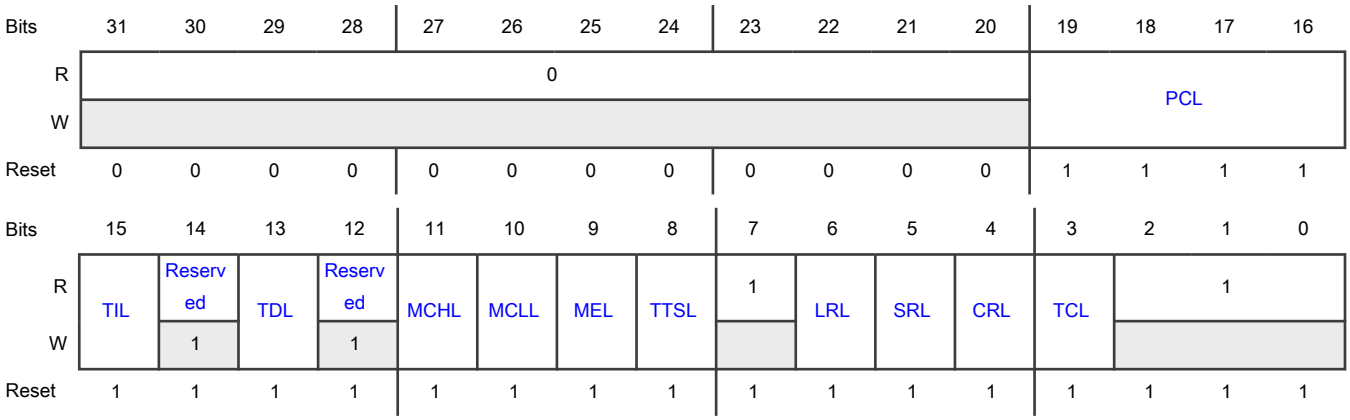
Field	Function
	1b - Tamper interrupt has asserted.
6-5 —	Reserved
4 TCE	Time Counter Enable When time counter is disabled the TSR register and TPR register are writeable, but do not increment. When time counter is enabled the TSR register and TPR register are not writeable, but increment. 0b - Time counter is disabled. 1b - Time counter is enabled.
3 MOF	Monotonic Overflow Flag Monotonic overflow flag is set when the monotonic counter is enabled and the monotonic counter high overflows. The monotonic counter does not increment and will read as zero when this bit is set. This bit is cleared by writing the monotonic counter high register when the monotonic counter is disabled. 0b - Monotonic counter overflow has not occurred. 1b - Monotonic counter overflow has occurred and monotonic counter is read as zero.
2 TAF	Time Alarm Flag Time alarm flag is set when the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. This bit is cleared by writing the TAR register. 0b - Time alarm has not occurred. 1b - Time alarm has occurred.
1 TOF	Time Overflow Flag Time overflow flag is set when the time counter is enabled and overflows. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. 0b - Time overflow has not occurred. 1b - Time overflow has occurred and time counter is read as zero.
0 TIF	Time Invalid Flag The time invalid flag is set on POR or software reset. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. The monotonic counter register is held in reset whenever the time invalid flag is set. 0b - Time is valid. 1b - Time is invalid and time counter is read as zero.

43.3.1.8 RTC Lock Register (LR)

Offset

Register	Offset
LR	18h

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 PCL	Pin Configuration Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Pin Configuration Register is locked and writes are ignored. 1b - Pin Configuration Register is not locked and writes complete as normal.
15 TIL	Tamper Interrupt Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Tamper Interrupt Register is locked and writes are ignored. 1b - Tamper Interrupt Register is not locked and writes complete as normal.
14 —	Reserved
13 TDL	Tamper Detect Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Tamper Detect Register is locked and writes are ignored.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Tamper Detect Register is not locked and writes complete as normal.
12 —	Reserved
11 MCHL	Monotonic Counter High Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Monotonic Counter High Register is locked and writes are ignored. 1b - Monotonic Counter High Register is not locked and writes complete as normal.
10 MCLL	Monotonic Counter Low Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Monotonic Counter Low Register is locked and writes are ignored. 1b - Monotonic Counter Low Register is not locked and writes complete as normal.
9 MEL	Monotonic Enable Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Monotonic Enable Register is locked and writes are ignored. 1b - Monotonic Enable Register is not locked and writes complete as normal.
8 TTSL	Tamper Time Seconds Lock After being cleared, this bit can be set only by VBAT POR or software reset. 0b - Tamper Time Seconds Register is locked and writes are ignored. 1b - Tamper Time Seconds Register is not locked and writes complete as normal.
7 —	Reserved
6 LRL	Lock Register Lock After being cleared, this bit can be set only by POR or software reset. 0b - Lock Register is locked and writes are ignored. 1b - Lock Register is not locked and writes complete as normal.
5 SRL	Status Register Lock After being cleared, this bit can be set only by POR or software reset. 0b - Status Register is locked and writes are ignored. 1b - Status Register is not locked and writes complete as normal.
4	Control Register Lock After being cleared, this bit can only be set by POR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CRL	0b - Control Register is locked and writes are ignored. 1b - Control Register is not locked and writes complete as normal.
3 TCL	Time Compensation Lock After being cleared, this bit can be set only by POR or software reset. 0b - Time Compensation Register is locked and writes are ignored. 1b - Time Compensation Register is not locked and writes complete as normal.
2-0 —	Reserved

43.3.1.9 RTC Interrupt Enable Register (IER)

Offset

Register	Offset
IER	1Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													TSIC		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								WPON	Reserved		TSIE	MOIE	TAIE	TOIE	TIIE
W										0						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-19 —	Reserved
18-16 TSIC	Timer Seconds Interrupt Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Configures the frequency of the RTC Seconds interrupt and the RTC_CLKOUT prescaler output. This field should only be altered when TSIE is clear. 000b - 1 Hz. 001b - 2 Hz. 010b - 4 Hz. 011b - 8 Hz. 100b - 16 Hz. 101b - 32 Hz. 110b - 64 Hz. 111b - 128 Hz.
15-8 —	Reserved
7 WPON	Wakeup Pin On The RTC_WAKEUP pin is optional and not available on all devices. Whenever the RTC_WAKEUP pin is enabled and this bit is set, it will assert. 0b - No effect. 1b - If the RTC_WAKEUP pin is enabled, then the pin will assert.
6-5 —	Reserved
4 TSIE	Time Seconds Interrupt Enable The seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector. It is generated at least once a second and requires no software overhead (there is no corresponding status flag to clear). The frequency of the seconds interrupt is configured by TSIC. 0b - Seconds interrupt is disabled. 1b - Seconds interrupt is enabled.
3 MOIE	Monotonic Overflow Interrupt Enable 0b - Monotonic overflow flag does not generate an interrupt. 1b - Monotonic overflow flag does generate an interrupt.
2 TAIE	Time Alarm Interrupt Enable 0b - Time alarm flag does not generate an interrupt. 1b - Time alarm flag does generate an interrupt.
1 TOIE	Time Overflow Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

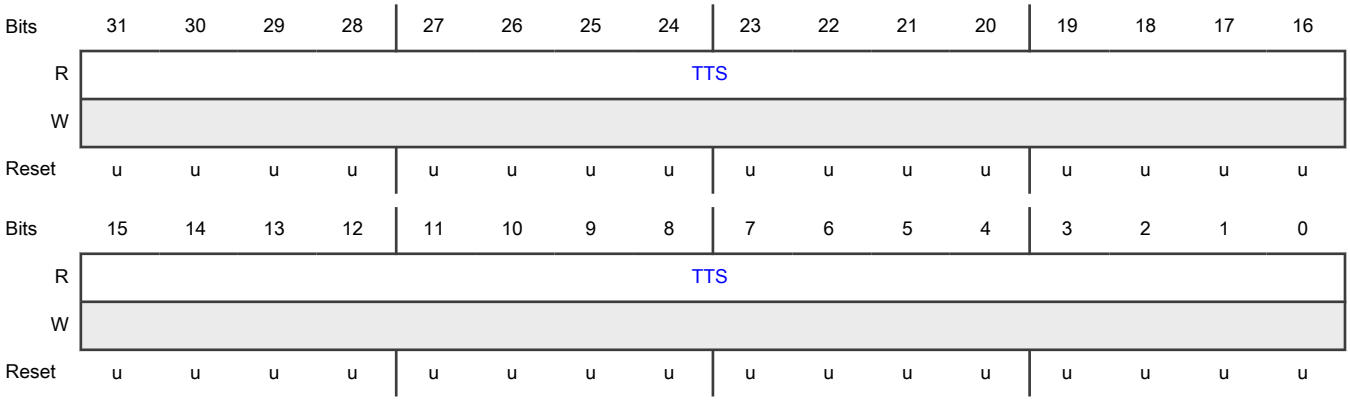
Field	Function
	0b - Time overflow flag does not generate an interrupt. 1b - Time overflow flag does generate an interrupt.
0 TIIE	Time Invalid Interrupt Enable 0b - Time invalid flag does not generate an interrupt. 1b - Time invalid flag does generate an interrupt.

43.3.1.10 RTC Tamper Time Seconds Register (TTSR)

Offset

Register	Offset
TTSR	20h

Diagram



Fields

Field	Function
31-0 TTS	Tamper Time Seconds If the time invalid flag is set then reading this register returns the contents of the time seconds register at the point at which the time invalid flag was set. If the time invalid flag is clear, but the tamper interrupt detect flag is set then reading this register returns the point at which the tamper interrupt was set. If both the time invalid flag and tamper detect interrupt are clear, then reading this register returns zero. Writing the tamper time seconds register with any value will set the time invalid flag, allowing software to invalidate the time.

43.3.1.11 RTC Monotonic Enable Register (MER)

Offset

Register	Offset
MER	24h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											MCE	0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4 MCE	Monotonic Counter Enable 0b - Writes to the monotonic counter load the counter with the value written. 1b - Writes to the monotonic counter increment the counter.
3-0 —	Reserved

43.3.1.12 RTC Monotonic Counter Low Register (MCLR)

Offset

Register	Offset
MCLR	28h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MCL	Monotonic Counter Low When the time invalid flag is set, the monotonic counter is held in reset. When the monotonic counter enable is clear, a write to this register will load the counter with the value written. When the monotonic counter enable is set, a write to this register will cause it to increment. A write to monotonic counter low that causes it to overflow will also increment monotonic counter high.

43.3.1.13 RTC Monotonic Counter High Register (MCHR)

Offset

Register	Offset
MCHR	2Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MCH	Monotonic Counter High When the time invalid flag is set, the monotonic counter is held in reset. When the monotonic counter enable is clear, a write to this register will load the counter with the value written. When the monotonic counter enable is set, a write to this register will cause it to increment. A write to monotonic counter low that causes it to overflow will also increment monotonic counter high.

43.3.1.14 RTC Tamper Detect Register (TDR)

Offset

Register	Offset
TDR	34h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												TPF			
W													W1C			
Reset [1]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TMF	FSF	STF	LCTF	0	0	0	1
W									W1C	W1C	W1C	W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

1. A software reset will clear all tamper flags, but flags can set again once software reset is negated.

Fields

Field	Function
31-20 —	Reserved
19-16 TPF	Tamper Pin Flag These flags are set whenever the corresponding tamper pin asserts. To clear, write logic one to the corresponding flag after that tamper pin negates. 0b - Tamper not detected. 1b - Tamper pin tamper detected.

Table continues on the next page...

Table continued from the previous page...

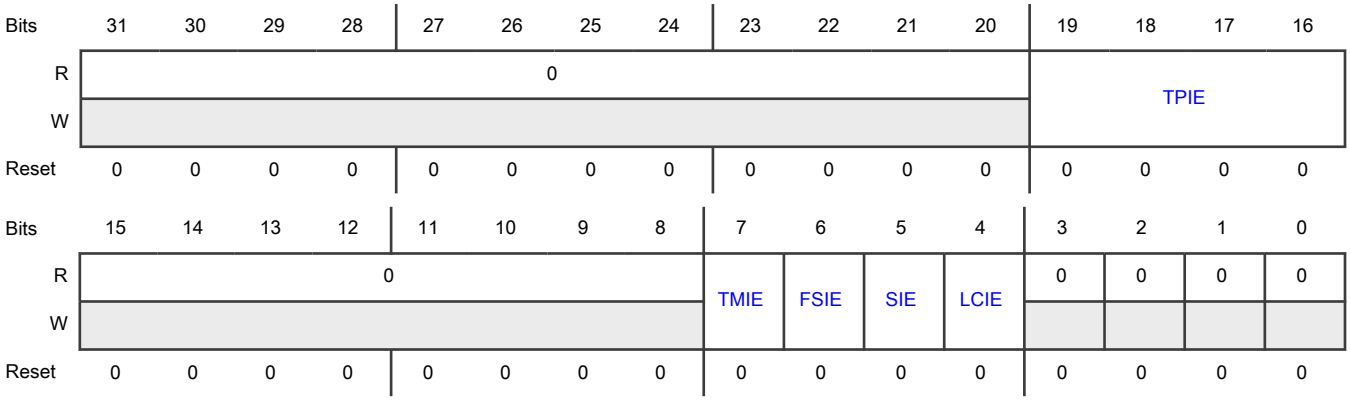
Field	Function
15-8 —	Reserved
7 TMF	Test Mode Flag This flag is set whenever any test mode is entered. To clear, write logic one to this flag after exiting from all test modes. 0b - Tamper not detected. 1b - Test mode tamper detected.
6 FSF	Flash Security Flag This flag is set whenever flash security is disabled. To clear, write logic one to this flag after flash security is enabled. 0b - Tamper not detected. 1b - Flash security tamper detected.
5 STF	Security Tamper Flag This flag is set whenever security module asserts its tamper detect. To clear, write logic one to this flag after security module has negated its tamper detect. 0b - Tamper not detected. 1b - Security module tamper detected.
4 LCTF	Loss of Clock Tamper Flag This flag is set whenever loss of clock is detected. To clear, write logic one to this flag after loss of clock negates. 0b - Tamper not detected. 1b - Loss of Clock tamper detected.
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

43.3.1.15 RTC Tamper Interrupt Register (TIR)

Offset

Register	Offset
TIR	3Ch

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 TPIE	Tamper Pin Interrupt Enable 0b - Interrupt disabled. 1b - An interrupt is generated when the corresponding tamper pin flag is set.
15-8 —	Reserved
7 TMIE	Test Mode Interrupt Enable 0b - Interrupt disabled. 1b - An interrupt is generated when the test mode flag is set.
6 FSIE	Flash Security Interrupt Enable 0b - Interrupt disabled. 1b - An interrupt is generated when the flash security flag is set.
5 SIE	Security Module Interrupt Enable 0b - Interrupt disabled. 1b - An interrupt is generated when the security module flag is set.

Table continues on the next page...

Table continued from the previous page...

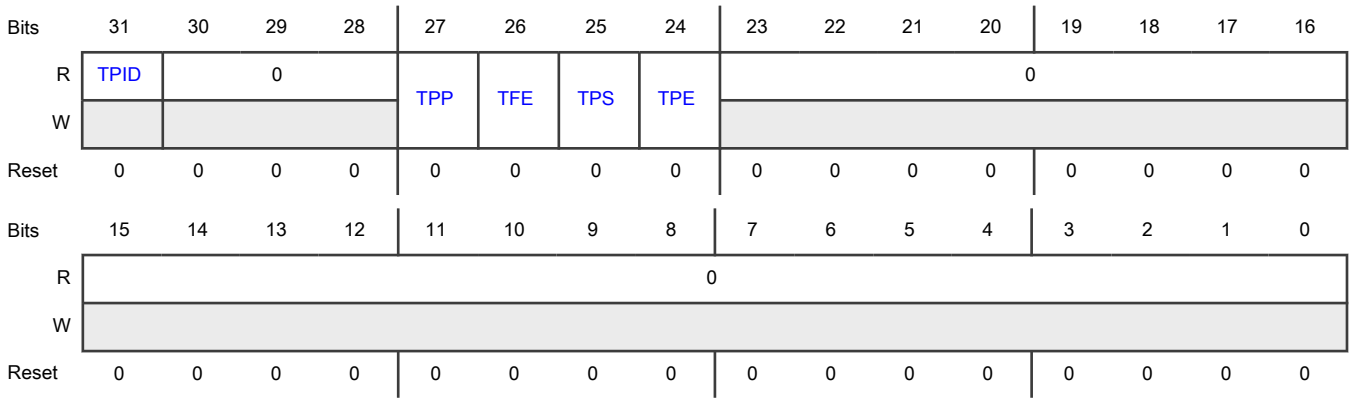
Field	Function
4 LCIE	Loss of Clock Interrupt Enable 0b - Interrupt disabled. 1b - An interrupt is generated when the loss of clock flag is set.
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

43.3.1.16 RTC Pin Configuration Register (PCR0 - PCR3)

Offset

Register	Offset
PCR0	40h
PCR1	44h
PCR2	48h
PCR3	4Ch

Diagram



Fields

Field	Function
31 TPID	Tamper Pin Input Data 0b - Tamper pin input data is logic zero. 1b - Tamper pin input data is logic one.
30-28 —	Reserved
27 TPP	Tamper Pin Polarity Configures the polarity of the tamper pin. 0b - Tamper pin is active high. 1b - Tamper pin is active low.
26 TFE	Tamper Filter Enable Enables the passive input filter on the tamper pin. 0b - Input filter is disabled on the tamper pin. 1b - Input filter is enabled on the tamper pin.
25 TPS	Tamper Pull Select Configures the direction of the tamper pin pull resistor. 0b - Tamper pin pull resistor direction will assert the tamper pin. 1b - Tamper pin pull resistor direction will negate the tamper pin.
24 TPE	Tamper Pull Enable Enables the pull resistor on the tamper pin. 0b - Pull resistor is disabled on tamper pin. 1b - Pull resistor is enabled on tamper pin.
23-0 —	Reserved

43.4 Functional description

43.4.1 Power, clocking, and reset

The RTC is an always powered block that remains active in all low power modes.

The time counter within the RTC is clocked by default from a 32.768 kHz clock.

The power-on-reset signal initializes all RTC registers to their default state. A software reset bit can also initialize all RTC registers.

43.4.1.1 Software reset

Writing 1 to CR[SWR] forces the equivalent of a POR to the rest of the RTC module. CR[SWR] is not affected by the software reset and must be cleared by software.

43.4.2 Time counter

The time counter consists of a 32-bit seconds counter that increments once every second and a 16-bit prescaler register that increments once every 32.768 kHz clock cycle.

Reading the time counter (either seconds or prescaler) while it is incrementing may return invalid data due to synchronization of the read data bus. If it is necessary for software to read the prescaler or seconds counter when they could be incrementing, it is recommended that two read accesses are performed and that software verifies that the same data was returned for both reads.

The time seconds register and time prescaler register can be written only when SR[TCE] is clear. Always write to the prescaler register before writing to the seconds register, because the seconds register increments on the falling edge of bit 14 of the prescaler register.

The time prescaler register increments provided SR[TCE] is set, SR[TIF] is clear, SR[TOF] is clear, and the 32.768 kHz (or 1 kHz) clock source is present. After enabling the oscillator, wait the oscillator startup time before setting SR[TCE] to allow time for the oscillator clock output to stabilize.

If the time seconds register overflows then the SR[TOF] will set and the time prescaler register will stop incrementing. Clear SR[TOF] by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TOF] is set.

SR[TIF] is set on VBAT POR, software reset, and when an enabled tamper source is detected. It is cleared by clearing the enabled tamper detect flag and then initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TIF] is set.

43.4.3 Compensation

The compensation logic provides an accurate and wide compensation range and can correct errors as high as 3906 ppm and as low as 0.12 ppm. The compensation factor must be calculated externally to the RTC and supplied by software to the compensation register. The RTC itself does not calculate the amount of compensation that is required, although the 1 Hz clock is output to an external pin in support of external calibration logic.

Crystal compensation can be supported by using firmware and crystal characteristics to determine the compensation amount. Temperature compensation can be supported by firmware that periodically measures the external temperature via ADC and updates the compensation register based on a look-up table that specifies the change in crystal frequency over temperature.

The compensation logic alters the number of 32.768 kHz clock cycles it takes for the prescaler register to overflow and increment the time seconds counter. The time compensation value is used to adjust the number of clock cycles between -127 and +128. Cycles are added or subtracted from the prescaler register when the prescaler register equals 0x3FFF and then increments. The compensation interval is used to adjust the frequency at which the time compensation value is used, that is, from once a second to once every 256 seconds.

Updates to the time compensation register will not take effect until the next time the time seconds register increments and provided the previous compensation interval has expired. When the compensation interval is set to other than once a second then the compensation is applied in the first second interval and the remaining second intervals receive no compensation.

Compensation is disabled by configuring the time compensation register to zero.

43.4.4 Time alarm

The Time Alarm register (TAR), SR[TAF], and IER[TAIE] allow the RTC to generate an interrupt at a predefined time. The 32-bit TAR is compared with the 32-bit Time Seconds register (TSR) each time it increments. SR[TAF] will set when TAR equals TSR and TSR increments.

SR[TAF] is cleared by writing TAR. This will usually be the next alarm value, although writing a value that is less than TSR, such as 0, will prevent SR[TAF] from setting again. SR[TAF] cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

43.4.5 Update mode

The Update Mode field in the Control register (CR[UM]) configures software write access to the Time Counter Enable (SR[TCE]) field. When CR[UM] is clear, SR[TCE] can be written only when LR[SRL] is set. When CR[UM] is set, SR[TCE] can also be written when SR[TCE] is clear or when SR[TIF] or SR[TOF] are set. This allows the time seconds and prescaler registers to be initialized whenever time is invalidated, while preventing the time seconds and prescaler registers from being changed on the fly. When LR[SRL] is set, CR[UM] has no effect on SR[TCE].

CR[UM] also configures software write access to the Monotonic Counter Enable (MER[MCE]) bit. When CR[UM] is clear, MER[MCE] can be written only when LR[MEL] is set. When CR[UM] is set, MER[MCE] can also be written when MER[MCE] is clear or when SR[TIF] or SR[MOF] are set. This allows the monotonic counter register to be initialized whenever the monotonic counter is invalid, while preventing the monotonic counter from being changed on the fly. When LR[MEL] is set, CR[UM] has no effect on MCR[MCE].

43.4.6 Monotonic counter

The 64-bit Monotonic Counter is a counter that cannot be exhausted or return to any previous value, once it has been initialized. If the monotonic overflow flag is set, the monotonic counter returns zero and does not increment.

Depending on the value of the monotonic counter enable bit, writing to the monotonic counter either initializes the register with the value written, or increments the register by one (and the value written is ignored).

When the monotonic counter is enabled, the monotonic counter high increments on either a write to the monotonic counter high register or if the monotonic counter low register overflows (due to a write to the monotonic counter low register). The monotonic overflow flag sets when the monotonic counter high register overflows and is cleared by writing the monotonic counter high register when the monotonic counter is disabled.

The monotonic counter is held in reset whenever the time invalid flag is set. Always clear the time invalid flag before initializing the monotonic counter.

43.4.7 Tamper detect

Each external tamper pin and internal tamper source can be configured to generate a tamper interrupt. The contents of the seconds register are latched in the tamper time seconds register when the tamper interrupt asserts. Read the tamper time seconds register before clearing the flag in the tamper detect register.

Writing the tamper time seconds register at any time will set the time invalid flag. Since this is a software initiated tamper, there is no status flag to indicate this tamper source. To disable the software initiated tamper, lock the tamper time seconds register to prevent write accesses to that register.

43.4.8 Register lock

The Lock register (LR) can be used to block write accesses to certain registers until the next POR or software reset. Locking the Control register (CR) will disable the software reset. Locking LR will block future updates to LR.

Write accesses to a locked register are ignored and do not generate a bus error.

43.4.9 Interrupt

The RTC interrupt is asserted whenever a status flag and the corresponding interrupt enable bit are both set. It is always asserted on POR, and software reset. The RTC interrupt is enabled at the chip level by enabling the chip-specific RTC clock gate control bit. The RTC interrupt can be used to wakeup the chip from any low-power mode. If the RTC wakeup pin is enabled and the chip is powered down, the RTC interrupt will cause the wakeup pin to assert.

The RTC interrupt is also asserted when a configured tamper flag asserts and the corresponding tamper interrupt enable bit is set. Both internal tamper events and external tamper pins can cause the tamper interrupt to assert. The contents of the seconds register are recorded in the tamper time seconds register when the tamper interrupt asserts.

The RTC seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector that is generated once a second and requires no software overhead (there is no corresponding status flag to clear). It is enabled in the RTC by the time seconds

interrupt enable bit and enabled at the chip level by setting the chip-specific RTC clock gate control bit. The frequency of the seconds interrupt defaults to 1 Hz, but can instead be configured to trigger every 2, 4, 8, 16, 32, 64 or 128 Hz. The RTC seconds interrupt does not cause the RTC wakeup pin to assert.

Chapter 44
Low-Power Timer (LPTMR)

44.1 Chip-specific LPTMR information

Table 262. Reference links to related information

Topic	Related module	Reference
Full description	LPTMR	LPTMR
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

44.1.1 Module instances

This device has two instances of the LPTMR module, LPTMR0, and LPTMR1.

44.1.2 LPTMR clocks

The prescaler and glitch filter of the LPTMR module (the module functional clock) can be clocked from one of four sources determined by LPTMRx_PSR[PCS]. The following table shows the clock assignments for this field.

NOTE

The chosen clock must remain enabled if the LPTMR is to continue operating in all required low-power modes.

The LPTMR allows the maximum clock frequency of 24 MHz.

Table 263. LPTMRn prescaler/glitch filter clocking options

LPTMRn_PSR[PCS]	Prescaler/glitch filter clock number	Chip clock
00	0	FRO-6MHz - Free Running Oscillator - 6MHz clock source.
01	1	Reserved
10	2	32K_CLK - Selectable clock output from the Battery Backed domain produced by either the OSC-32K or the FRO-32K.
11	3	Reserved

44.1.3 LPTMR pulse counter input options

LPTMRn_CSR[TPS] configures the input source used in pulse counter mode. The following table shows the input assignments for this field.

Table 264. LPTMRn pulse counter input options

LPTMRn_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0_OUT
01	1	CMP1_OUT

Table continues on the next page...

Table 264. LPTMRn pulse counter input options (continued)

LPTMRn_CSR[TPS]	Pulse counter input number	Chip input
10	2	LPTMRn_Alt2
11	3	LPTMRn_Alt3

44.2 Overview

The low-power timer (LPTMR) can be configured to operate as a time counter with optional prescaler, or as a pulse counter with optional glitch filter, across all power modes, including the low-power modes. It is reset only on Power on Reset (POR), allowing it to be used as a time-of-day counter.

44.2.1 Block diagram

The following figure shows a detailed block diagram for an LPTMR module.

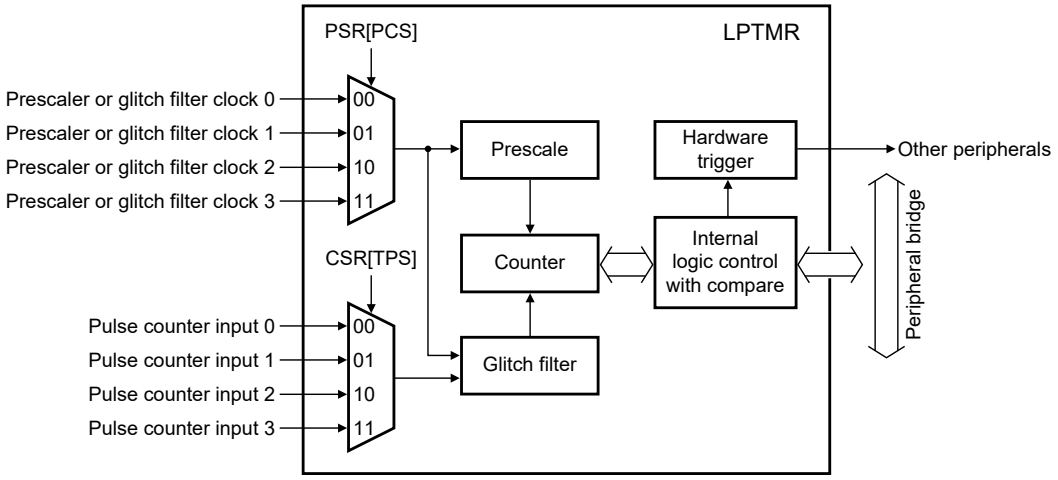


Figure 128. Functional block diagram

44.2.2 Features

- 32-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wake-up from any low-power mode.
  - Hardware trigger output.
  - Counter supports free-running mode or reset on compare.
- Configurable clock source for prescaler/glitch filter
- Configurable input source for pulse counter
  - Rising-edge or falling-edge

44.3 Functional description

44.3.1 Low-power modes

In low-power modes, LPTMR continues to operate normally. You can configure LPTMR to exit the low-power mode by generating either an interrupt or a DMA request.

44.3.2 Clocks

The LPTMR prescaler/glitch filter can be clocked by one of the clocks configured by PSR[PCS]. You must enable the clock source before you enable LPTMR.

NOTE

You may need to configure the clock source selected by PSR[PCS] for it to remain enabled in low-power modes. Otherwise LPTMR will not operate in low-power modes.

In Pulse Counter mode, with the glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the glitch filter clock source for a clock that is disabled.

NOTE

The clock source or pulse input source selected for the LPTMR must not exceed the maximum frequency of f_{LPTMR} defined in the chip data sheet.

44.3.3 Reset

LPTMR is reset only on Power on Reset (POR). When configuring the LPTMR registers, you must initially write CSR with LPTMR disabled, before configuring PSR and CMR. Then, you must set CSR[TIE] as the last step in the initialization. Doing so ensures that LPTMR is configured correctly and the LPTMR counter is reset to zero following a Power on Reset (POR).

44.3.4 Prescaler and glitch filter

The LPTMR prescaler and glitch filter share the same logic, which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

NOTE

The prescaler/glitch filter configuration must not be altered when LPTMR is enabled.

44.3.4.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks the CNR. When LPTMR is enabled, the CNR increments every 2^1 to 2^{16} prescaler clock cycles. After LPTMR is enabled, the first increment of the CNR takes an additional one or two prescaler clock cycles due to synchronization logic.

44.3.4.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments the CNR on every clock cycle. When LPTMR is enabled, the first increment takes an additional one or two prescaler clock cycles due to synchronization logic.

44.3.4.3 Glitch filter

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks the CNR. When LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

Table 265. Glitch filter output with the selected input source

If	Then
The selected input source remains deasserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output also deasserts.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output also asserts.

NOTE

The input is only sampled on the rising clock edge.

The CNR increments each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which the CNR can increment is once every 2^2 to 2^{16} glitch filter clock edges. When first enabled, the glitch filter waits an additional one or two glitch filter clock edges due to synchronization logic.

44.3.4.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the CNR every time it asserts. Before LPTMR is first enabled, the selected input source is forced to be asserted. This prevents the CNR from incrementing if the selected input source is already asserted when LPTMR is first enabled.

44.3.5 Counter

The counter (CNR) increments by one on every:

- Prescaler clock in Time Counter mode with prescaler bypassed
- Prescaler output in Time Counter mode with prescaler enabled
- Input source assertion in Pulse Counter mode with glitch filter bypassed
- Glitch filter output in Pulse Counter mode with glitch filter enabled

The CNR is reset when LPTMR is disabled or if the counter register overflows. If CSR[TFC] = 0, then the CNR is also reset whenever CSR[TCF] = 1.

When the core is halted in Debug mode:

- If configured for Pulse Counter mode, the CNR continues incrementing.
- If configured for Time Counter mode, the CNR stops incrementing.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This synchronizes and registers the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

44.3.6 Compare

On the next CNR increment when the CNR is equal to the value of the CMR, the following events occur:

- CSR[TCF] is read as 1b.
- LPTMR generates interrupt if CSR[TIE] = 1 also.
- LPTMR generates hardware trigger.
- LPTMR writes 0 to CNR if CSR[TFC] = 0.

When LPTMR is enabled, the CMR can be altered only when CSR[TCF] = 1. When updating the CMR, the CMR must be written and CSR[TCF] must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

NOTE

When LPTMR is enabled in Time Counter mode, the first increment will take an additional one or two clock cycles due to synchronization logic. This will result in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster prescaler clock or larger prescaler value will minimize this impact.

44.3.7 Interrupt

LPTMR generates an interrupt whenever CSR[TIE] and CSR[TCF] = 1. CSR[TCF] is cleared by disabling LPTMR or by writing a logic 1 to it.

You can alter CSR[TIE] and write 1 to CSR[TCF] when LPTMR is enabled.

LPTMR generates an interrupt asynchronously to the system clock. The interrupt can be used to generate a wake-up from any low-power mode, provided LPTMR is enabled as a wake-up source.

44.3.8 Hardware trigger

The LPTMR hardware trigger asserts at the same time CSR[TCF] is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

Table 266.

When	Then
CMR = 0 and CSR[TFC] = 0.	The LPTMR hardware trigger asserts on the first compare and does not deassert.
CMR is set to a nonzero value, or if CSR[TFC] = 1.	The LPTMR hardware trigger asserts on each compare and deasserts on the following increment of CNR.

44.4 External signals

Table 267. External signals

Signal	Description		Direction
LPTMR_ALT*n*	Pulse Counter Input LPTMR can select one of the input pins to be used in Pulse Counter mode.		I
	State meaning	Assertion—If configured for pulse counter mode with active-high input, then assertion causes the CNR to increment. Deassertion—If configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment.	
	Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.	

44.5 Initialization

Execute these steps to initialize LPTMR:

1. Configure CSR for the selected mode and pin configuration but with CSR[TEN]=0. A value of CSR[TEN]=0 resets the counter and clears the flag.
2. Configure PSR with the selected clock source and prescale/glitch filter configuration.
3. Configure CMR with the selected compare point.
4. Write 1 to CSR[TEN] to enable LPTMR.

44.6 Application information

Application 1: Generate an interrupt every 100 ms using 32.768 kHz clock source

1. Disable LPTMR by writing 0 to CSR[TEN].
2. Select 32.768 kHz clock source by configuring PSR[PCS].
3. Bypass the prescaler/glitch filter by writing 1 to PSR[PBYP].
4. Assert interrupt every 3277 cycles by configuring CMR[COMPARE]=0CCCh.
5. Enable LPTMR by writing 1 to CSR[TEN].
6. Enable the LPTMR interrupt by writing 1 to CSR[TIE].

Application 2: Generate an interrupt once a minute using 32.768 kHz clock source

1. Disable LPTMR by writing 0 to CSR[TEN].
2. Select 32.768 kHz clock source by configuring PSR[PCS].
3. Select the prescaler to divide the prescaler clock by 32768 to increment the counter once a second by configuring PSR[PRESCALE] = 00h.
4. Assert an interrupt every 60 seconds by configuring CMR[COMPARE]=003Bh.
5. Enable LPTMR by writing 1 to CSR[TEN].
6. Enable the LPTMR interrupt by writing 1 to CSR[TIE].

44.7 Memory map and register definition

NOTE

The LPTMR registers are reset only on Power on Reset (POR). See Reset for more details.

44.7.1 LPTMR register descriptions

44.7.1.1 LPTMR memory map

LPTMR0 base address: 4002_D000h

LPTMR1 base address: 4002_E000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Status (CSR)	32	RW	0000_0000h
4h	Prescale and Glitch Filter (PSR)	32	RW	0000_0000h
8h	Compare (CMR)	32	RW	0000_0000h
Ch	Counter (CNR)	32	RW	0000_0000h

44.7.1.2 Control Status (CSR)

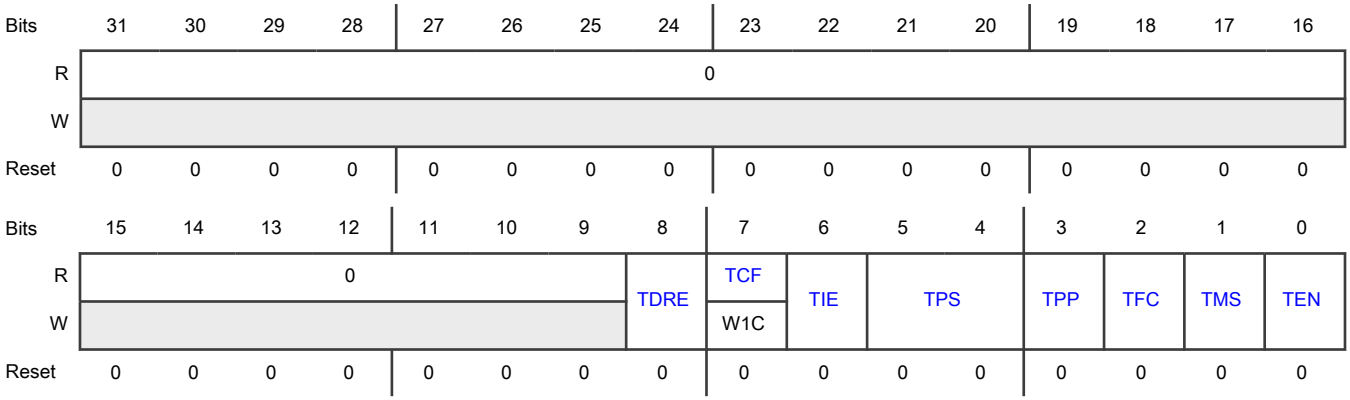
Offset

Register	Offset
CSR	0h

Function

Controls various features of LPTMR.TDRE

Diagram



Fields

Field	Function
31-9 —	Reserved
8 TDRE	Timer DMA Request Enable Enables timer DMA request. When TDRE is set, the LPTMR DMA Request is generated whenever TCF is also set. Then the TCF is cleared after the DMA Controller completes. 0b - Disable 1b - Enable
7 TCF	Timer Compare Flag Sets on the next CNR increment when LPTMR is enabled and CNR equals CMR. TCF is cleared when LPTMR is disabled or a logic 1 is written to it. **NOTE** Clear this field before enabling the Timer interrupt or DMA request. 0b - CNR ≠ (CMR + 1) 1b - CNR = (CMR + 1)

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 TIE	Timer Interrupt Enable If TIE = 1, then LPTMR generates an interrupt if TCF = 1 also. 0b - Disable 1b - Enable
5-4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. You must alter TPS only when LPTMR is disabled. The input connections vary by device. For details see the chip configuration information about connections to these inputs. 00b - Input 0 01b - Input 1 10b - Input 2 11b - Input 3
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. You must change TPP only when LPTMR is disabled. If TPP = 0, then the pulse counter input source is active-high, and the CNR increments on the rising edge. If TPP = 1, then the pulse counter input source is active-low, and the CNR increments on the falling edge. 0b - Active-high 1b - Active-low
2 TFC	Timer Free-Running Counter You must alter TFC only when LPTMR is disabled. If TFC = 0, then LPTMR resets CNR whenever TCF = 1. If TFC = 1, then LPTMR resets CNR on overflow. 0b - Reset if TCF set 1b - Reset on overflow
1 TMS	Timer Mode Select Configures the mode of LPTMR. You must alter TMS only when LPTMR is disabled. 0b - Time Counter mode 1b - Pulse Counter mode
0 TEN	Timer Enable Enables LPTMR timer. When TEN is clear, it resets the LPTMR internal logic, including the CNR and TCF. When TEN is set, LPTMR is enabled. When writing 1 to this field, CSR[5:1] must not be altered. 0b - Disable 1b - Enable

44.7.1.3 Prescale and Glitch Filter (PSR)

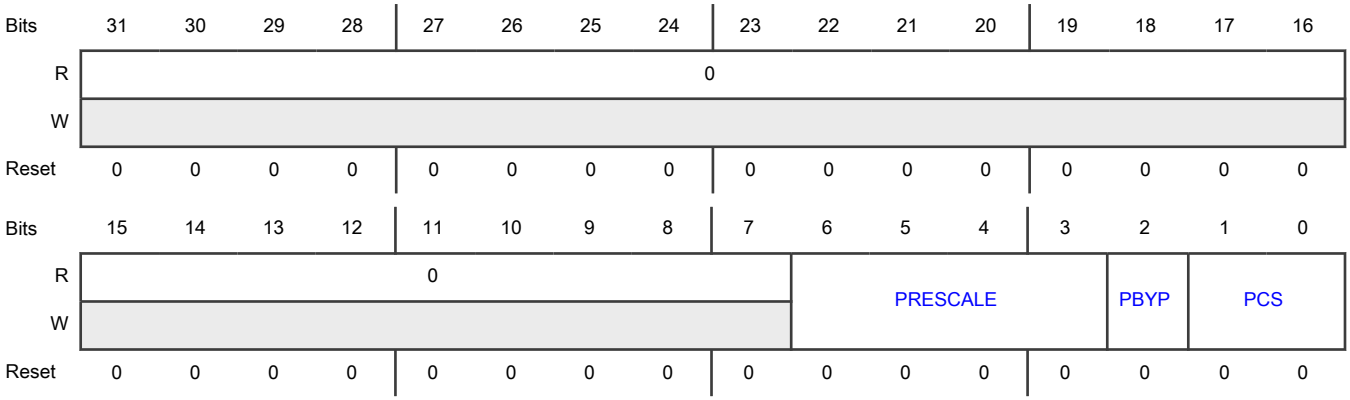
Offset

Register	Offset
PSR	4h

Function

Configures features related to the prescaler/glitch filter.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-3 PRESCALE	Prescale/Glitch Filter Value Configures the size of the prescaler in Time Counter mode or the width of the glitch filter in Pulse Counter mode. The width of the glitch filter can vary by 1 cycle due to synchronization of the pulse counter input. PRESCALE must be altered only when LPTMR is disabled. 0000b - Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration. 0001b - Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges. 0010b - Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges. 0011b - Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges. 0100b - Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges. 0110b - Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges. 0111b - Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges. 1000b - Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges. 1001b - Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges. 1010b - Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges. 1011b - Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges. 1100b - Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges. 1101b - Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges. 1110b - Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges. 1111b - Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges.
2 PBYP	Prescaler/Glitch Filter Bypass Controls clocking of CNR. When PBYP = 1, the selected prescaler clock in Time Counter mode, or else the selected input source in Pulse Counter mode, directly clocks the CNR. When PBYP = 0, the output of the prescaler/glitch filter clocks the CNR. You must change PBYP only when LPTMR is disabled. 0b - Prescaler/glitch filter enable 1b - Prescaler/glitch filter bypass
1-0 PCS	Prescaler/Glitch Filter Clock Select Selects the clock to be used by the LPTMR prescaler/glitch filter. You must alter PCS only when LPTMR is disabled. See the chip configuration details for information on the connections to these inputs. • In Time Counter mode, PCS selects the input clock to the prescaler. • In Pulse Counter mode, PCS selects the input clock to the glitch filter. 00b - Clock 0 01b - Clock 1 10b - Clock 2 11b - Clock 3

44.7.1.4 Compare (CMR)

Offset

Register	Offset
CMR	8h

Function
Configures the compare values to the CNR (see Compare for details).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMPARE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMPARE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 COMPARE	Compare Value On the next CNR increment, if LPTMR is enabled and CNR equals CMR, then: 1. LPTMR writes 1 to TCF. 2. The hardware trigger asserts until the next time CNR increments. If CMR = 0, the hardware trigger remains asserted until LPTMR is disabled. If LPTMR is enabled, you must alter CMR only if TCF = 1.

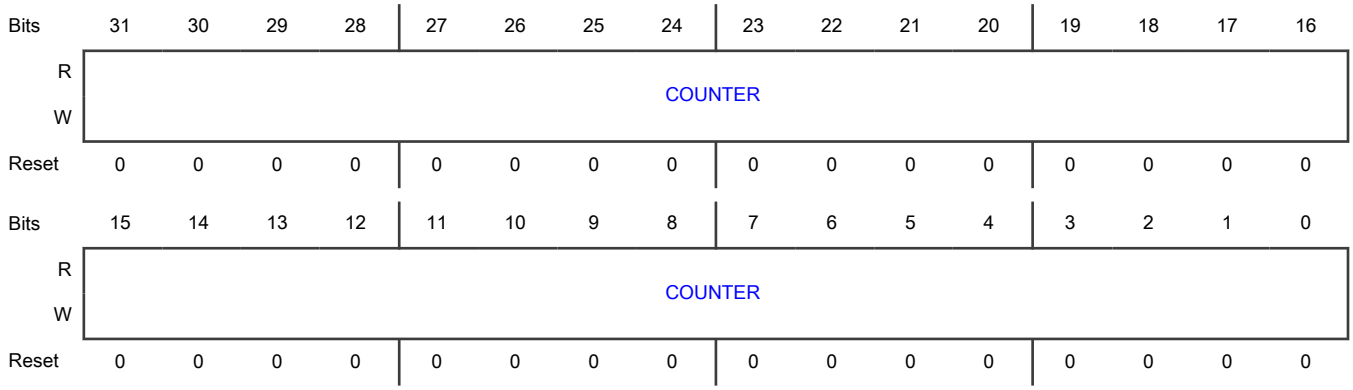
44.7.1.5 Counter (CNR)

Offset

Register	Offset
CNR	Ch

Function
Indicates counter values. See Counter for more details.

Diagram



Fields

Field	Function
31-0 COUNTER	Counter Value Contains the current value of the LPTMR counter at the time this register was last written.

Chapter 45
Low Power Periodic Interrupt Timer (LPIT)

45.1 Chip-specific LPIT information

Table 268. Reference links to related information

Topic	Related module	Reference
Full description	LPIT	LPIT
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

45.1.1 Module instances

This device has one instance of the LPIT module, LPIT0. The LPIT0 contains 4 channels.

45.1.2 External trigger source

If the TCTRLx[TRG_SRC] is set to 1, it selects to use external trigger source from TRGMUX, refer to TRGMUX0 outputs for more details.

45.2 Overview

LPIT is a low power periodic interrupt timer with multiple timer channels. After a timer reaches a programmed count, the respective timer channel generates pre-trigger and trigger output signals, and these pre-trigger and trigger outputs can be used to trigger other modules on the device.

45.2.1 Block Diagram

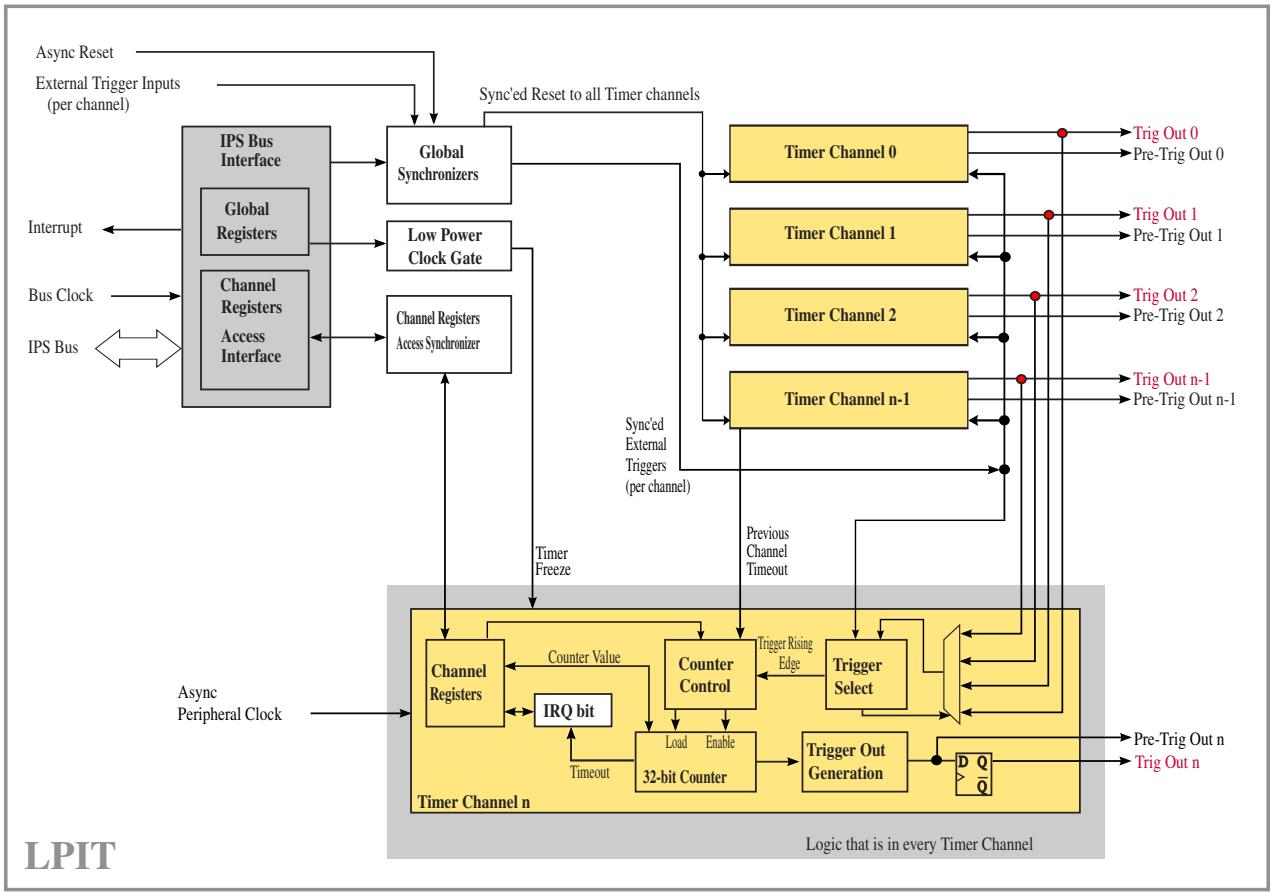


Figure 129. Detailed Block Diagram

45.2.2 Features

- Timers can be configured to be controlled using:
  - external triggers (triggers from outside the LPIT module)
  - or internal triggers (triggers from other timer channels inside the LPIT module).
- The timer channels can be chained together, to form a larger width timer.
- Depending on the timer modes, the timer channels may reload and count again, or stop after reaching the programmed count.

45.3 Functional description

45.3.1 LPIT programming model

The programming model comprises a global register set (common to all timer channels), and registers for each timer channel (that control their respective timer channels). Access to these registers gets synchronized to the asynchronous peripheral clock, then affects the timer channel registers.

- Each timer channel contains a 32-bit counter that loads the starting value and down counts on every peripheral clock's positive edge.
- After reaching a zero value (a channel timer timeout), a trigger output is generated.
- The counter enable is controlled using a timer enable register control bit, external or internal triggers or via previous channel timeout (when using timer chaining).
- After a channel timer timeout, an interrupt bit is also set, to tell the CPU about the timer timeout.

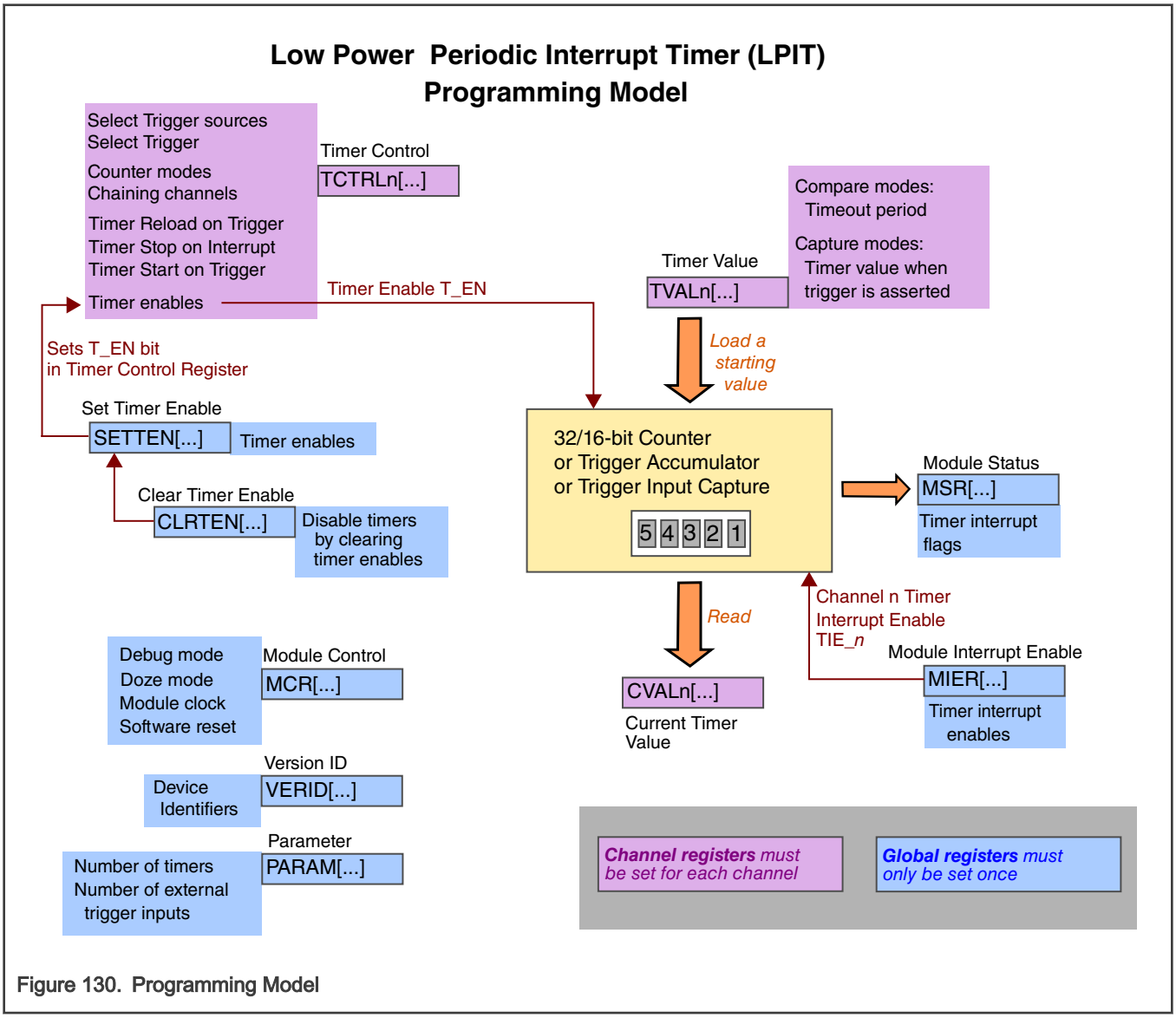


Figure 130. Programming Model

Global registers need only be set once:

- Version ID Register (VERID)
- Parameter Register (PARAM)
- Module Control Register (MCR)
- Module Status Register (MSR)
- Module Interrupt Enable Register (MIER)
- Set Timer Enable Register (SETTEN)

- Clear Timer Enable Register (CLRTEN)

Channel registers must be set for each channel:

- Timer Value Register (TVAL0 - TVAL3)
- Current Timer Value (CVAL0 - CVAL3)
- Timer Control Register (TCTRL0 - TCTRL3)

45.3.2 LPIT interface with other modules on the SoC

The next figure shows the interface of an LPIT module to the other modules on the SoC.

- The CPU interface provides the clock, reset, register read/write bus interface and handles interrupts from an LPIT.
- The trigger output signals from an LPIT may trigger other modules on the SoC, like the DMA, ADC, and other modules.
- Similarly, other timer modules may provide trigger inputs to an LPIT module, to control when an LPIT timer channel starts.

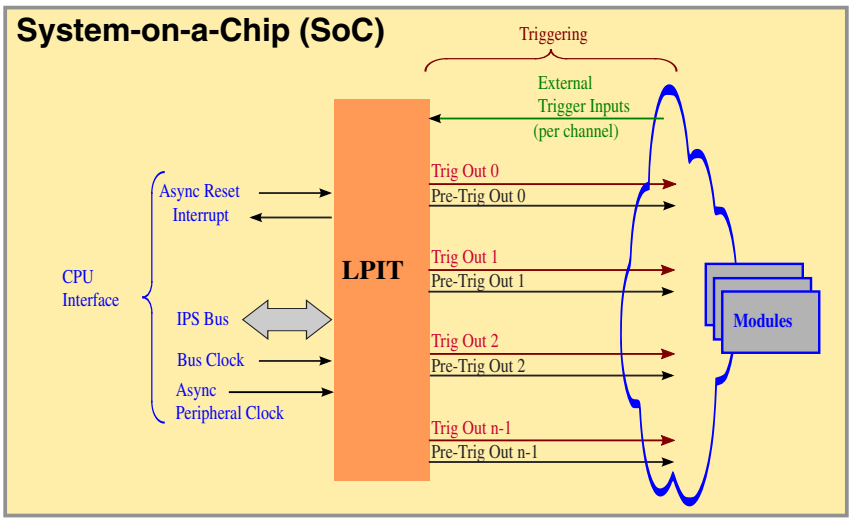


Figure 131. LPIT Interface in SoC

45.3.3 Chip power modes

The LPIT module supports the following chip power modes.

Table 269. Chip modes supported by the LPIT module

Chip mode	LPIT Operation
Run	Normal operation
Sleep	Can continue operating, if the Doze Enable bit (MCR[DOZE_EN]) is set and the LPIT is using an external or internal clock source that remains operating during Sleep mode.
Low Power Stop (also called Deep Sleep)	The Doze Enable (MCR[DOZE_EN]) bit is ignored and the LPIT will be disabled for the duration of Low Power mode.

Table continues on the next page...

Table 269. Chip modes supported by the LPIT module (continued)

Chip mode	LPIT Operation
Debug	Can continue operating provided the Debug Enable bit (MCR[DBG_EN]) is set.

45.3.4 Timer modes

The timer mode is configured by setting an appropriate value in the MODE bits in TCTRLn register.

Table 270. Timer modes that are supported

Timer mode	Operation
32-bit Periodic Counter	• The counter will load, decrement down to 0 • which will set the timer interrupt flag and assert the output pre-trigger.
Dual 16-bit Periodic Counter	• The counter will load, then the lower 16-bits will decrement down to 0 • which will assert the output pre-trigger. • The upper 16-bits will then decrement down to 0 • which will set the timer interrupt flag and then negate the output pre-trigger.
32-bit Trigger Accumulator	• The counter will load on the 1st trigger rising edge and then decrement down to 0 on each trigger rising edge • which will set the timer interrupt flag and assert the output pre-trigger.
32-bit Trigger Input Capture	• The counter will load with 0xFFFF_FFFF and then decrement down to 0. • If a trigger rising edge is detected, — then it will store the inverse of the current counter value in the timer value register — which will set the timer interrupt flag and assert the output pre-trigger.

The timer operation is controlled by Trigger Control bits (TSOT, TSOI, TROT) which control the timer load, reload, start, and restart of the timers.

NOTE

- The trigger output is asserted one Peripheral Timer Clock cycle later than pre-trigger output. The trigger output and the pre-trigger output de-assert at the same time.
- The pre-trigger output is asserted for 2 clock cycles and the trigger output is asserted for 1 clock cycle (except in 16-bit Periodic Counter mode, where both pre-trigger and trigger are asserted for many cycles depending on TMR_VAL[31:16]).
- Timer changes *that are based on external triggers* take effect after 4 peripheral clocks after the actual external trigger assertion (due to clock synchronization, rise edge detection and timer update).

45.3.5 Timer channel modes

Each timer channel in LPIT can be configured to work in either compare modes or capture modes.

- **In compare mode:** the timers decrement when enabled and generate an output pre-trigger and trigger output. The trigger output is 1 clock cycle delayed of the pre-trigger pulse. Each timer channel start, reload and restart can be controlled via control bits. The timer can be configured to always decrement from a programmed start value, or decrement on selected

trigger inputs or previous channel timeout (when channels are chained). By chaining timer channels, applications can achieve larger timeout durations.

- **In capture mode:** the timer can be used to perform measurements as the timer value is captured (in the timer value register) when a selected trigger input is asserted. The timer can support once-off or multiple measurements (like frequency measurements).

The timer channels operate on an asynchronous clock, which is independent from the register read/write access clock. Clock synchronization between the clock domains ensures normal operations.

45.3.6 Trigger control for timers

Various programmable bits control how the trigger inputs and the timer operate. TRG_SEL and TRG_SRC select the timer triggers:

- Trigger Select (TRG_SEL) selects the input trigger for the channel from all of the other channel's trigger outputs.
- Trigger Source (TRG_SRC) selects between the internal trigger and the external trigger inputs to the channel.

The selected trigger affects how the timer operates, using the configuration of the TROT, TSOI and TSOT bits.

Table 271. How bits control timer operations

If	=	Then
TSOI *Timer Stop On Interrupt*	1	then the counter stops on a Timer Interrupt flag (MSR[TIFn]) assertion. To reload and decrement, it requires: • a trigger (if TSOT = 1) • a T_EN rising edge (if TSOT = 0)
	0	then the counter does not stop after timeout
TROT *Timer Reload On Trigger*	1	then the counter is loaded on each trigger
	0	then the counter is loaded on every T_EN rising edge or timeout rising edge (timeout not used in Capture modes)
TSOT *Timer Start On Trigger*	1	then the counter will start to decrement on a trigger. Subsequent triggers are ignored until the counter times out.
	0	then the counter decrements immediately on the next clock edge. When channel is Chained or in Capture mode, TSOT has no effect.

In different timer modes, these programmable bits affect the timer operation differently:

Table 272. Time modes and programmable bits

Mode	Which programmable bits affect timer operations
32-bit Periodic Counter	All bits (TSOT, TSOI, TROT) affect the timer operation as described previously.
Dual 16-bit Periodic Counter	
32-bit Trigger Accumulator	• Only the TSOI bit controls the timer function. • TROT and TSOT bits have no effect on timer operations.
32-bit Input Trigger Capture	• Only TSOI and TROT bits control the timer function. • TSOT bit has no effect on timer operations.

45.3.7 Channel chaining

Individual timer channels can be chained together to achieve a larger value of timeout. Chaining the timer channel causes them to work in a *'nested loop'* manner thereby leading to an effective timeout value of $TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$.

The channels are chained by setting the CHAIN bit in corresponding channel's $TCTRL_{CHn}$ register. When a channel is chained, that channel's timer decrements on previous channel's timeout pulse, regardless of the timer mode (MODE bits). The TSOT bit does not have any effect if the channel timer (Channel 'n') is chained to previous channel's timer (Channel 'n-1').

45.3.8 Detailed timing

NOTE

The timing diagrams in these sections are not *cycle-accurate* (that is, some cycles may not be shown), but the timing diagrams do show the timer channel behavior across several clock cycles.

Table 273. Timing diagrams list

Mode		Timing Diagram	TSOT	TROT	TSOI	CHAIN
32-bit periodic counter (compare mode)	00	Figure 132	0	0	0	0
		Figure 133	0	0	1	0
		Figure 134	0	1	0	0
		Figure 135	0	1	1	0
		Figure 136	1	0	0	0
		Figure 137	1	0	1	0
		Figure 138	1	1	0	0
		Figure 139	1	1	1	0
16-bit dual periodic counter (compare mode)	01	Figure 140	0	0	0	0
32-bit trigger accumulator mode	10	Figure 141	X	X	0	0
		Figure 142	X	X	1	0
32-bit trigger capture mode	11	Figure 143	X	0	0	0
		Figure 144	X	1	0	0
		Figure 145	X	0	1	0
Timer chaining: effects on timing operations	-	Figure 146	-	-	-	-

Mode=00: 32-bit periodic counter (compare mode)

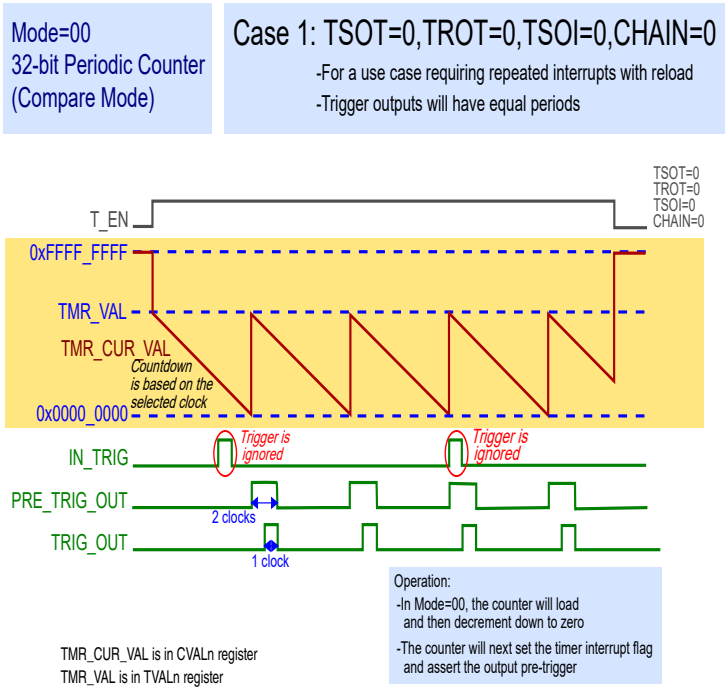


Figure 132. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

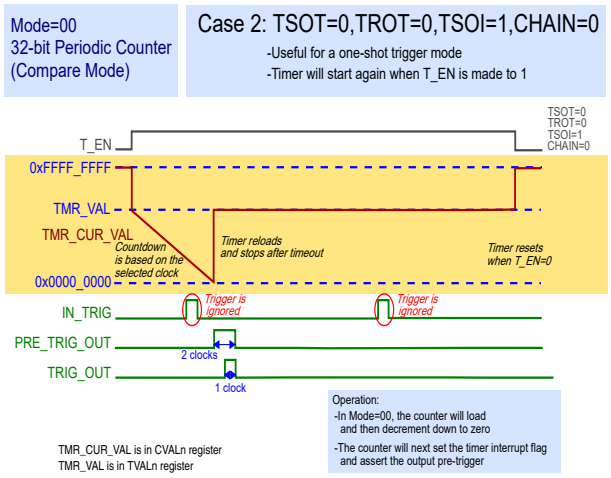


Figure 133. Case 2: TSOT = 0, TROT = 0, TSOI = 1, CHAIN = 0

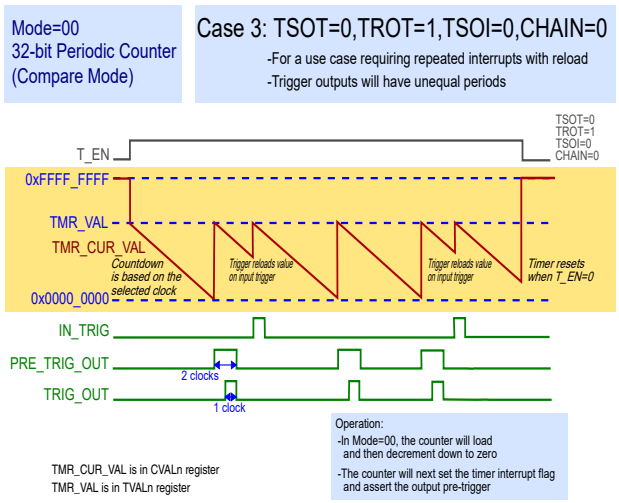


Figure 134. Case 3: TSOT = 0, TROT = 1, TSOI = 0, CHAIN = 0

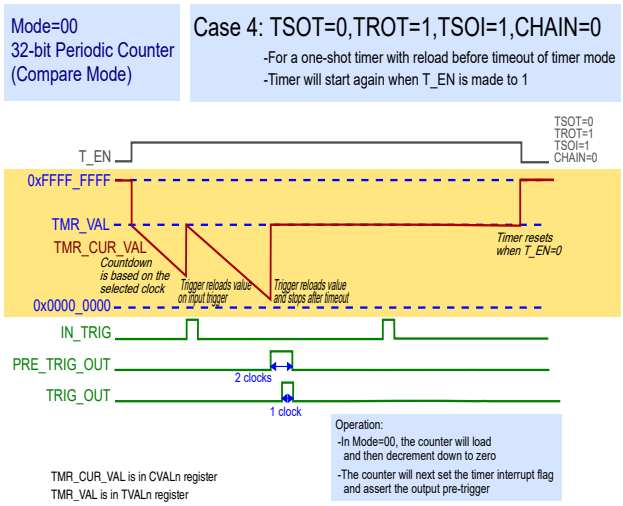


Figure 135. Case 4: TSOT = 0, TROT = 1, TSOI = 1, CHAIN = 0

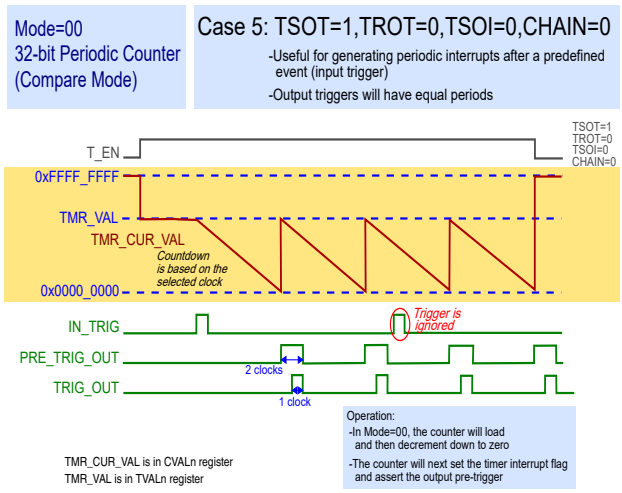


Figure 136. Case 5: TSOT = 1, TROT = 0, TSOI = 0, CHAIN = 0

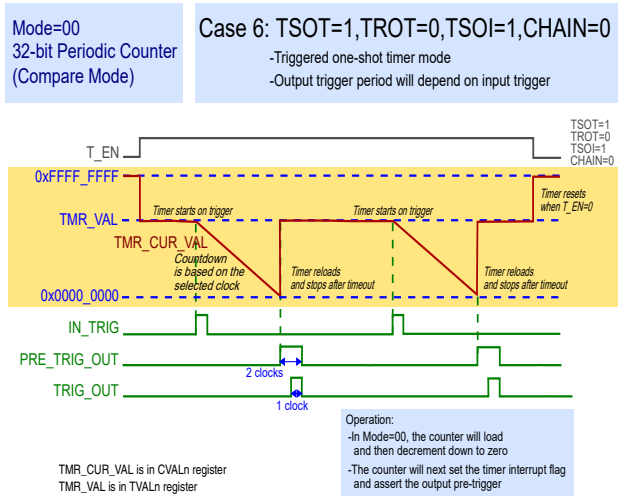


Figure 137. Case 6: TSOT = 1, TROT = 0, TSOI = 1, CHAIN = 0

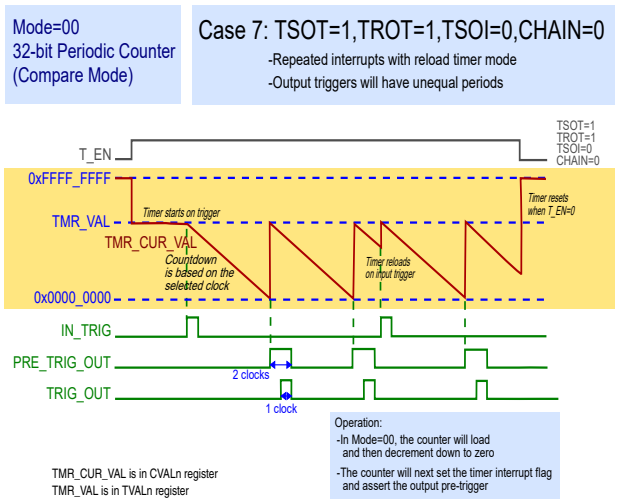


Figure 138. Case 7: TSOT = 1, TROT = 1, TSOI = 0, CHAIN = 0

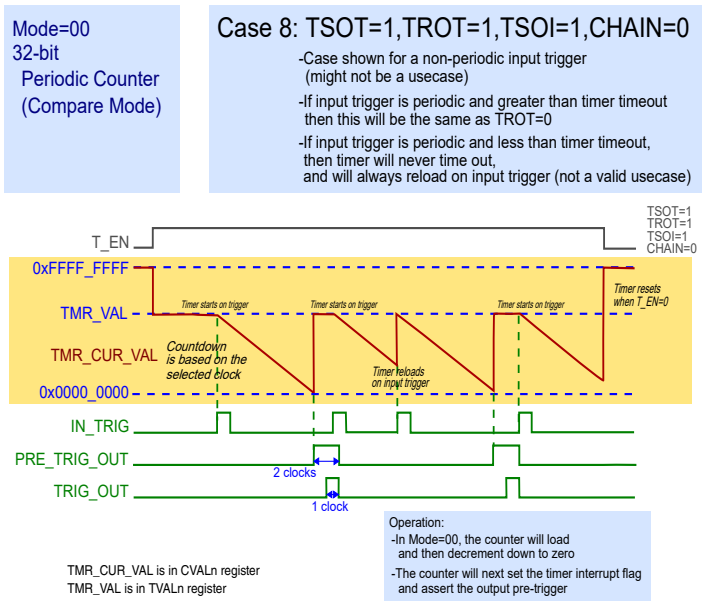


Figure 139. Case 8: TSOT = 1, TROT = 1, TSOI = 1, CHAIN = 0

Mode=01: 16-bit dual periodic counter (compare mode)

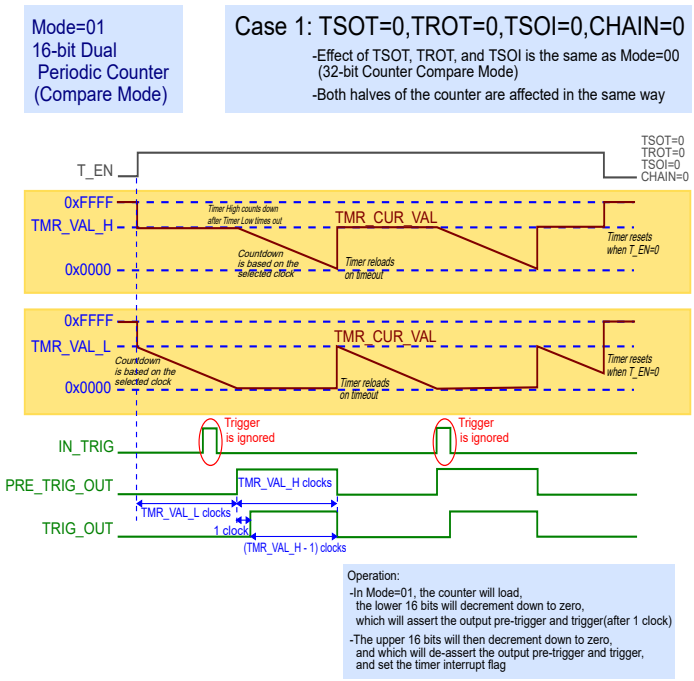


Figure 140. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

Effect of Timer Control Bits in Mode=01:

- Effect of timer control bits are the same as for Mode=00. Refer to the individual figures of Mode=00.
- The Timer Interrupt (timeout) asserts when {TMR_H,TMR_L} = 0x0000_0000.
- Behavior for Mode = 01 is explained in the next table.

Table 274. Mode=01 timer control bits

TSOT	TROT	TSOI	Function	Effect on Timer
0	0	0	• For repeated interrupts with reload • Trigger outputs will have equal periods	Similar to Figure 140.
0	0	1	One-shot mode	• Both timers stop after first count down and then time out • Timers will not count again until T_EN is made 1 again • Similar to Figure 133.
0	1	0	• For repeated interrupts with reload • Trigger outputs will have unequal periods	• Both timers will reload TMR_VAL on trigger rise edge

Table continues on the next page...

Table 274. Mode=01 timer control bits (continued)

TSOT	TROT	TSOI	Function	Effect on Timer
				• Output triggers will clear on reload, if asserted • Similar to Figure 134.
0	1	1	Reloadable one-shot mode	• If a trigger occurs before timeout, then both timers reload and count down (as shown); the timers stop after timeout • A trigger assertion after timeout simply reloads TMR_VAL into the timers • The timers will not count again until T_EN is set to 1 again • Similar to Figure 135.
1	0	0	• For generating periodic interrupts after a predefined event (input trigger) • Output triggers will have equal periods	After T_EN rises, the timers do not start until the first trigger's rising edge • Subsequent triggers will have no effect • Similar to Figure 136.
1	0	1	• Triggered one-shot timer mode • Output trigger period will depend on input trigger	After T_EN rises, the timers do not start until the first trigger's rising edge. • The timer stops counting after a timeout assertion (a timeout is asserted) • The timer does not start counting again until a new trigger's rising edge is detected • Similar to Figure 137.
1	1	0	• For repeated interrupts with reload timer mode • Output triggers will have unequal periods	After T_EN rises, the timers do not start until the first trigger's rising edge • Subsequent triggers will cause the timer to reload TMR_VAL into both counters • The output triggers will clear after a reload, if asserted • Similar to Figure 138.
1	1	1	• For a non-periodic input trigger • If input trigger is periodic and greater than timer timeout, then this will be the same as TROT=0 (which is triggered one-shot timer mode; output trigger period will depend on input trigger)	After T_EN rises, the timers do not start until the first trigger's rising edge • The timers stops counting after a timeout assertion (a timeout is asserted) • A trigger's rising edge will cause the timers to reload and then count down • Similar to Figure 139.

Mode=10: 32-bit trigger accumulator mode

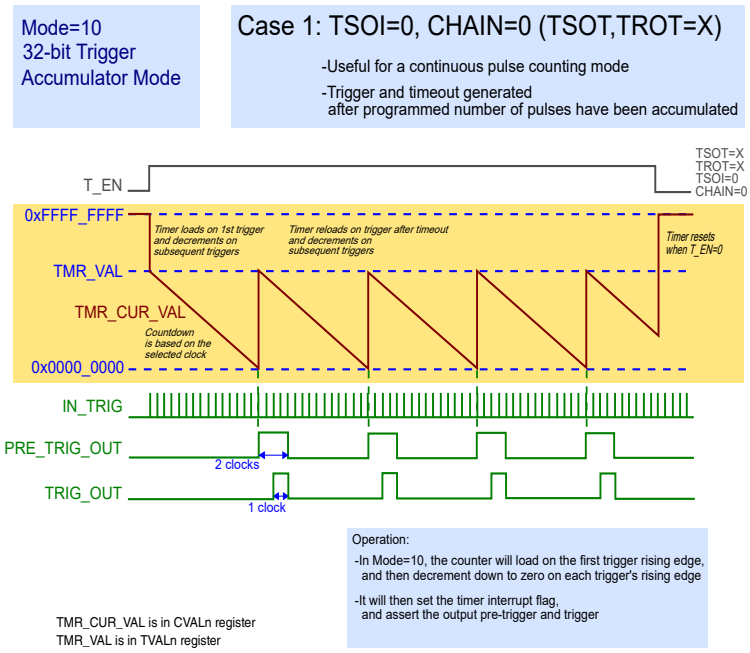


Figure 141. Case 1: TSOT = X, TROT = X, TSOI = 0, CHAIN = 0

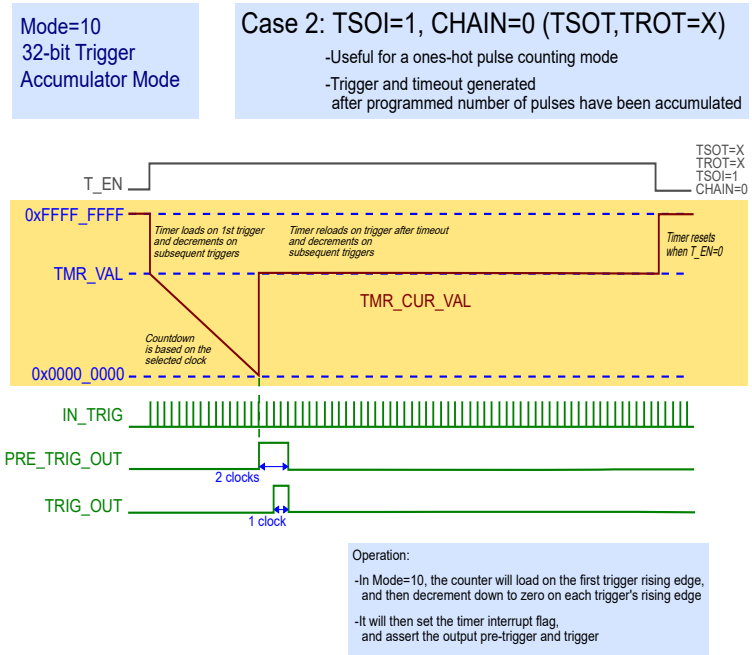


Figure 142. Case 2: TSOT = X, TROT = X, TSOI = 1, CHAIN = 0

Mode=11: 32-bit trigger capture mode

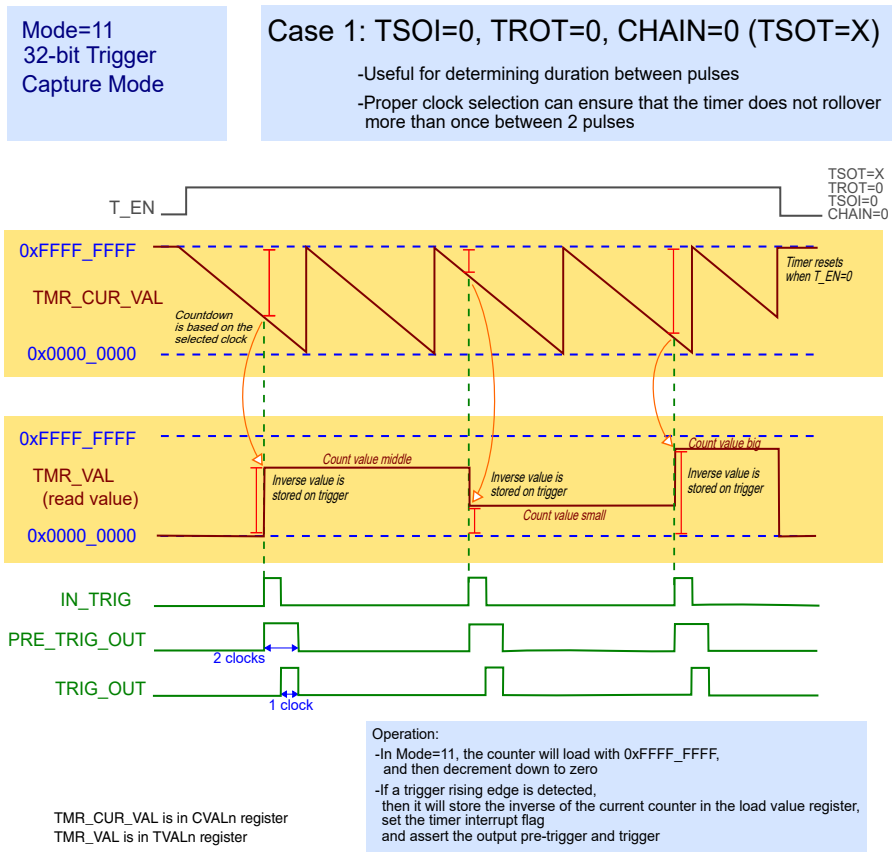


Figure 143. Case 1: TSOT = X, TROT = 0, TSOI = 0, CHAIN = 0

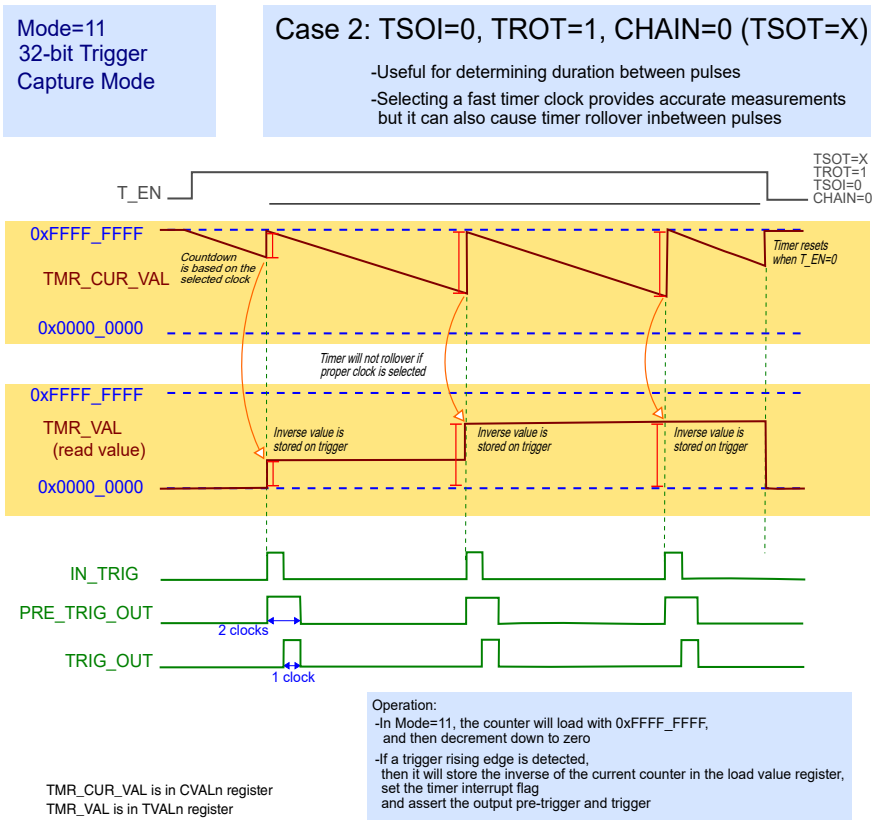


Figure 144. Case 2: TSOT = X, TROT = 1, TSOI = 0, CHAIN = 0

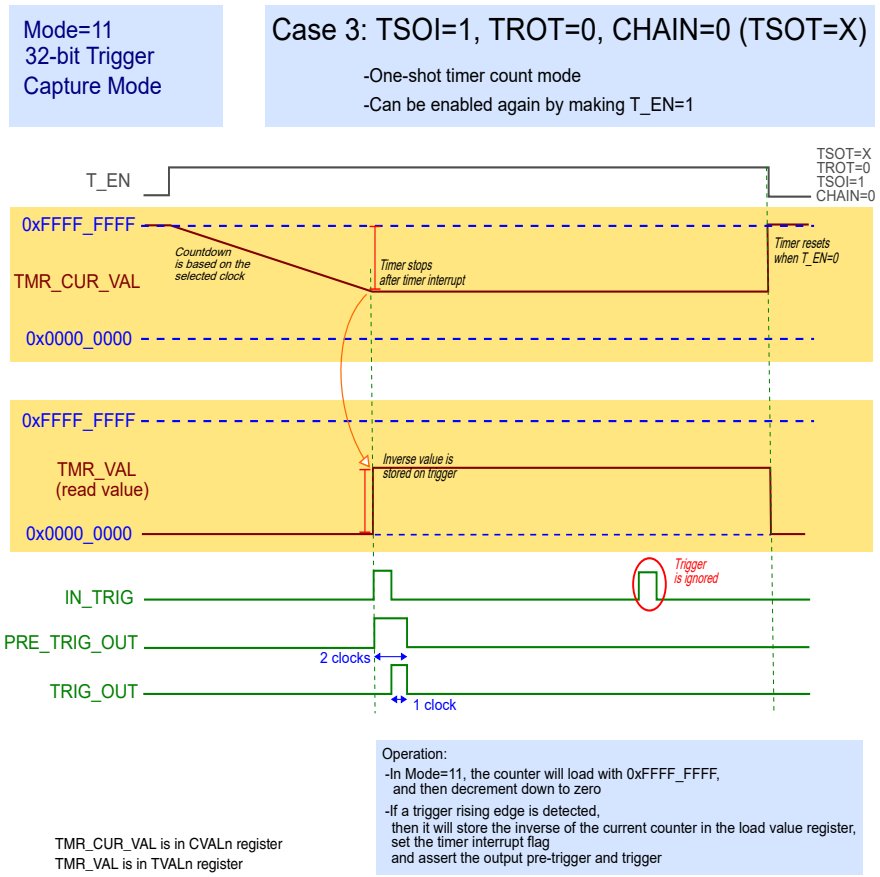


Figure 145. Case 3: TSOT = X, TROT = 0, TSOI = 1, CHAIN = 0

Case 4: TSOT = X, TROT = 1, TSOI = 1, CHAIN = 0

Same as previous case (Case 3), except that the timer reloads to 0xFFFF_FFFF and then stops. The timer does not start until T_EN is made 1 again. Case 4 is not a very useful case, because Case 3 covers this.

Timer chaining

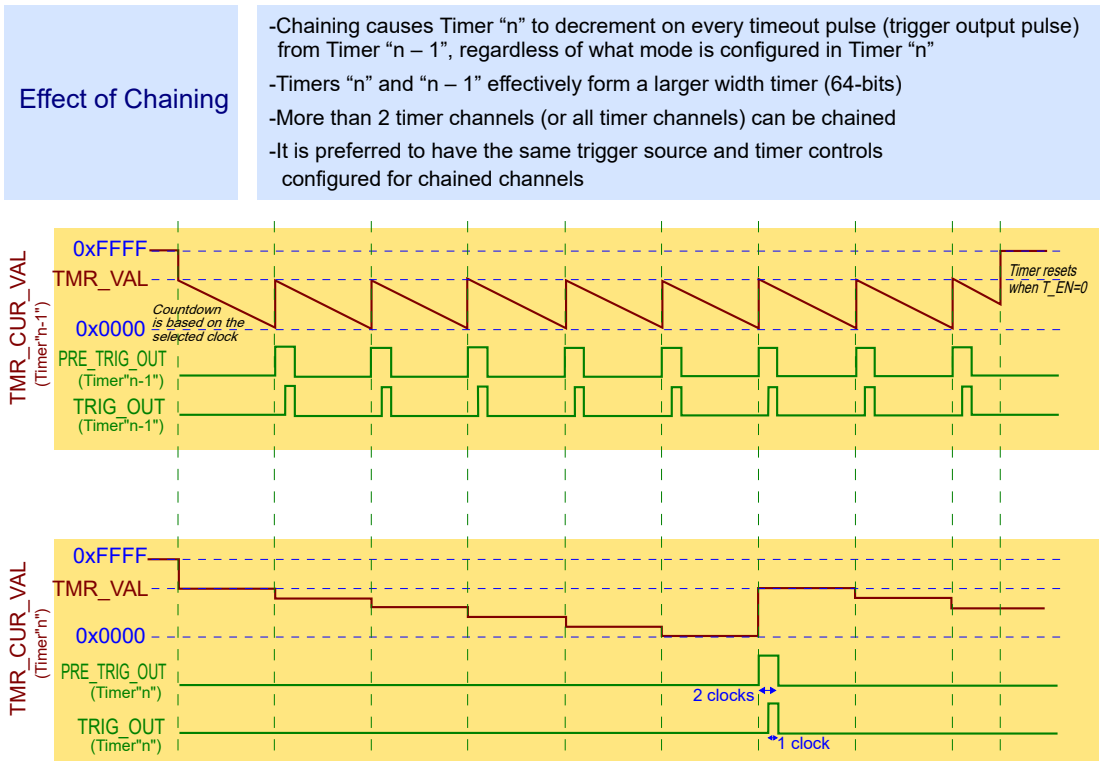


Figure 146. Chaining Effects

45.4 Initialization

Table 275. Initializing the LPIT module

Step		How? Why?
1	Enable the peripheral clock	by setting the Module Clock Enable bit (M_CEN) in the Module Control Register (MCR) register. **NOTE** • Accessing certain registers (MSR, SETTEN, CLRTEN, TVAL, CVAL, and TCTRL) while M_CEN = 0 will lead to assertion of a transfer error for that bus access. — Writing to CVAL and Reserved registers will generate a transfer error. • There might be additional clock gating bits in the device that gate the peripheral clock to this module. When enabling the clock to this module, ensure that software is configuring those additional clock gating bits (in addition to M_CEN bit).
2	Wait for 4 peripheral clock cycles	to allow time for clock synchronization and reset de-assertion.

Table continues on the next page...

Table 275. Initializing the LPIT module (continued)

Step		How? Why?
3	Configure timer control bits	for each timer channel that is to be enabled: • timer mode of operation bits TCTRLn[MODE] • trigger source selection bits TCTRLn[TRG_SEL, TRG_SRC] • trigger control bits TCTRLn[TROT, TSOT, TSOI] **NOTE** Timer controls should only be updated when the timer is disabled. There are 2 ways to disable a timer: write 1 to the specific timer's Clear Timer Enable register bit (CLRTEN[CLR_T_EN_n]) or set the timer enable bit (TCTRLn[T_EN]) = 0 for that channel.
4	Configure the channels that are to be chained	by setting the CHAIN bit in the corresponding channel's TCTRLn register.
5	Set the timer timeout value	for channels configured in Compare Mode, by programming the appropriate value in TVAL register for those channels.
6	Configure TIEn bits in MIER register	for those channels that are required to generate interrupts on timer timeouts.
7	Configure the low power modes of the module	by setting the DBG_EN and DOZE_EN bits in the MCR register. This is common to all timer channels.
8	Enable the channel timers	by setting the corresponding T_EN bit in the corresponding channel's TCRTLn register.

NOTE

When the Timer Channel is enabled in Compare Mode, the first decrement takes an additional one or two clock cycles due to synchronization logic. This results in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster counter clock minimizes this impact.

Also:

- For channels configured in Capture Mode, the timer value can be read from TVALn register when a channel timeout occurs.
- At any time, the current value of the timer for any channel can be read by reading the corresponding channel's CVALn register. The M_CEN bit must be '1' when doing so.
- The timer interrupt flag bits (TIFn) in MSR register get asserted on timer timeout. To clear these timer interrupt flag bits, write '1' to them.

45.5 Memory Map and Registers

45.5.1 LPIT register descriptions

The memory map comprises 32-bit aligned registers, which can be accessed via 8-bit, 16-bit, or 32-bit accesses.

- Write access to reserved locations will generate a transfer error.
- Read access to reserved locations will also generate a transfer error, and the read data bus will show all 0s.

NOTE

- The Memory Map and complete module is in Big Endian format.
- The LPIT module will not check if programmed values in the registers are correct; software must ensure that correct programmed values are being written.

45.5.1.1 LPIT memory map

LPIT0 base address: 4002_F000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0000h
4h	Parameter (PARAM)	32	R	0000_0404h
8h	Module Control (MCR)	32	RW	0000_0000h
Ch	Module Status (MSR)	32	RW	0000_0000h
10h	Module Interrupt Enable (MIER)	32	RW	0000_0000h
14h	Set Timer Enable (SETTEN)	32	RW	0000_0000h
18h	Clear Timer Enable (CLRTEN)	32	RW	0000_0000h
20h	Timer Value (TVAL0)	32	RW	0000_0000h
24h	Current Timer Value (CVAL0)	32	R	FFFF_FFFFh
28h	Timer Control (TCTRL0)	32	RW	0000_0000h
30h	Timer Value (TVAL1)	32	RW	0000_0000h
34h	Current Timer Value (CVAL1)	32	R	FFFF_FFFFh
38h	Timer Control (TCTRL1)	32	RW	0000_0000h
40h	Timer Value (TVAL2)	32	RW	0000_0000h
44h	Current Timer Value (CVAL2)	32	R	FFFF_FFFFh
48h	Timer Control (TCTRL2)	32	RW	0000_0000h
50h	Timer Value (TVAL3)	32	RW	0000_0000h
54h	Current Timer Value (CVAL3)	32	R	FFFF_FFFFh
58h	Timer Control (TCTRL3)	32	RW	0000_0000h

45.5.1.2 Version ID (VERID)

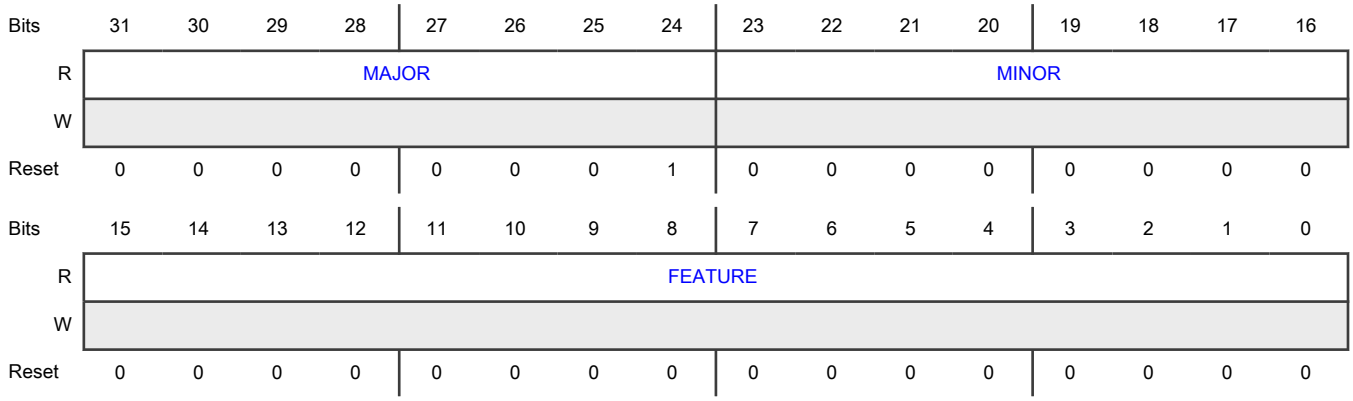
Offset

Register	Offset
VERID	0h

Function

Contains design version specification numbers.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification. Read-only field.
15-0 FEATURE	Feature Number Returns the feature set number. Read-only field.

45.5.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Provides details on the parameter settings that were used while incorporating this module into the device.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXT_TRIG								CHANNEL							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-16 —	This read-only field is reserved and always has the value 0
15-8 EXT_TRIG	Number of External Trigger Inputs Number of external triggers implemented in this device
7-0 CHANNEL	Number of Timer Channels Number of timer channels implemented in this device

45.5.1.4 Module Control (MCR)

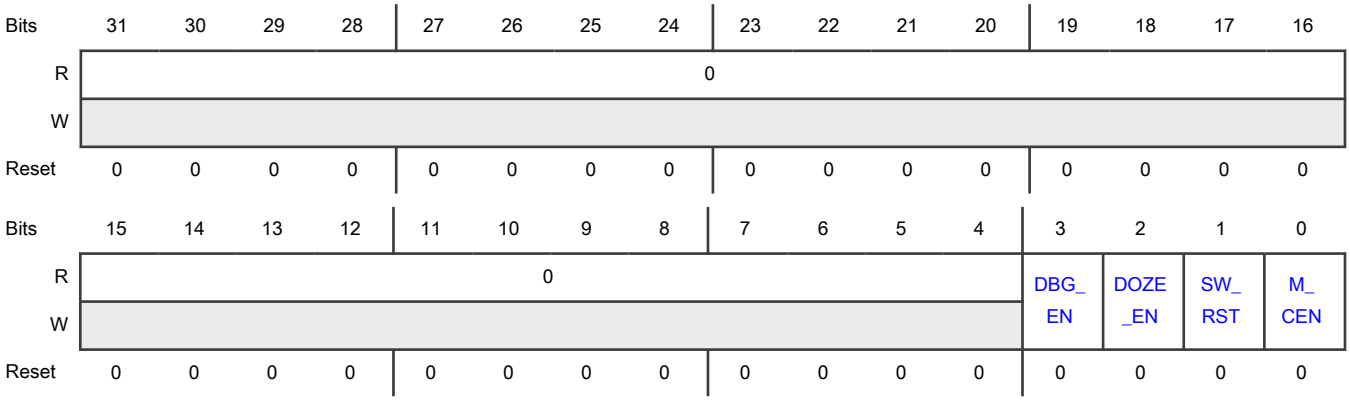
Offset

Register	Offset
MCR	8h

Function

Contains Software Reset, Clock Enable, and mode enable fields.

Diagram



Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3 DBG_EN	Debug Mode Enable Stops the timer channels when the device enters Debug mode 0b - Stop timer channels in Debug mode 1b - Allow timer channels to continue to run in Debug mode
2 DOZE_EN	DOZE Mode Enable Stops the timer channels when the device enters DOZE mode 0b - Stop timer channels in DOZE mode 1b - Allow timer channels to continue to run in DOZE mode
1 SW_RST	Software Reset Resets all channels and registers, except the Module Control Register. The Software Reset bit remains set until cleared by software. **NOTE** Before clearing the Software Reset bit, software must wait for 4 peripheral clocks (for clock synchronization and reset propagation). 0b - Timer channels and registers are not reset 1b - Reset timer channels and registers
0 M_CEN	Module Clock Enable Enables the peripheral clock to the module timers. • The Module Clock Enable bit must be asserted when accessing these registers:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— Module Status Register (MSR) — Set Timer Enable Register (SETTEN) — Clear Timer Enable Register (CLRTEN) — Timer Value Registers (TVAL*n*) — Current Timer Value Registers (CVAL*n*) — Timer Control Registers (TCTRL*n*) • Both the bus clock and peripheral clock must be enabled to allow for clock synchronization and updating the above registers. • Accessing the above mentioned registers while the Module Clock Enable bit (M_CEN) = '0', will assert a transfer error for that bus cycle. • Writing to Current Timer Value (CVAL*n*) registers and Reserved registers will always generate a transfer error. **NOTE** There may be additional clock gating bits in your device that gate the peripheral clock to the LPIT module. Ensure that those additional clock gating bits are configured appropriately, to enable the peripheral clock to the LPIT module. 0b - Disable peripheral clock to timers 1b - Enable peripheral clock to timers

45.5.1.5 Module Status (MSR)

Offset

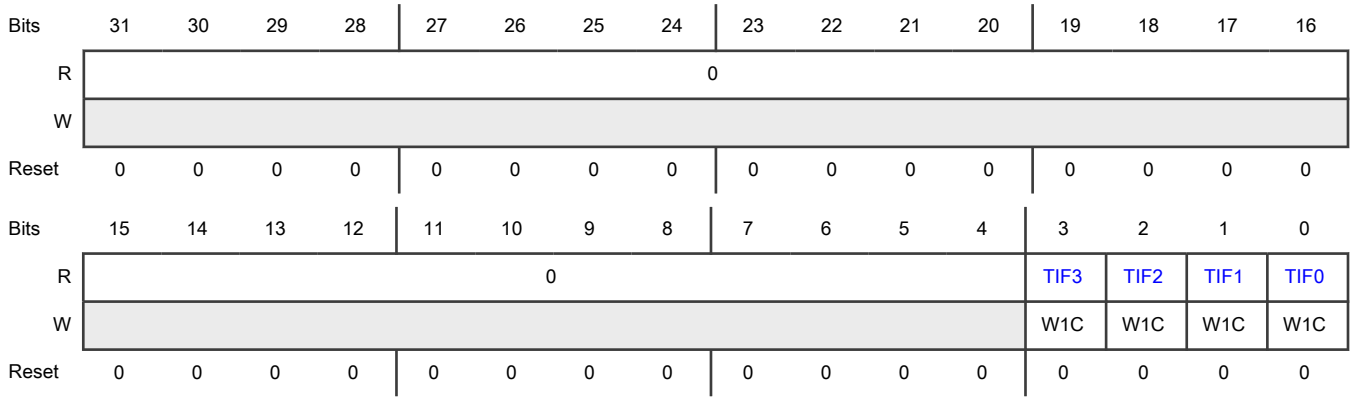
Register	Offset
MSR	Ch

Function

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled), reading or writing the Module Status register will generate a transfer error.

Diagram



Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3 TIF3	Channel 3 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)
2 TIF2	Channel 2 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)
1 TIF1	Channel 1 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Timeout has occurred (timer has timed out)
0 TIF0	Channel 0 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)

45.5.1.6 Module Interrupt Enable (MIER)

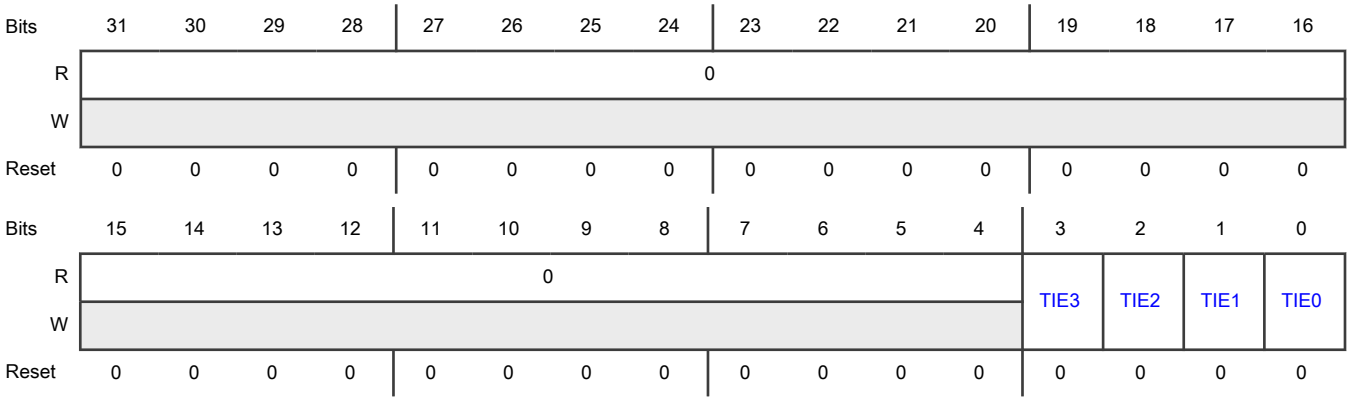
Offset

Register	Offset
MIER	10h

Function

Contains Channel Timer Interrupt Enable fields.

Diagram



Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 TIE3	Channel 3 Timer Interrupt Enable Enables interrupt generation when: • the Channel 3 Timer Interrupt Enable bit (TIE3) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF3]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled
2 TIE2	Channel 2 Timer Interrupt Enable Enables interrupt generation when: • the Channel 2 Timer Interrupt Enable bit (TIE2) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF2]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled
1 TIE1	Channel 1 Timer Interrupt Enable Enables interrupt generation when: • the Channel 1 Timer Interrupt Enable bit (TIE1) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF1]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled
0 TIE0	Channel 0 Timer Interrupt Enable Enables interrupt generation when: • the Channel 0 Timer Interrupt Enable bit (TIE0) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF0]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled

45.5.1.7 Set Timer Enable (SETTEN)

Offset

Register	Offset
SETTEN	14h

Function

The Set Timer Enable register allows the simultaneous enabling of timer channels.

- Timer channels can be enabled by

— either by writing '1' to Timer Enable bit (T_EN) in the respective TCTRLn register,

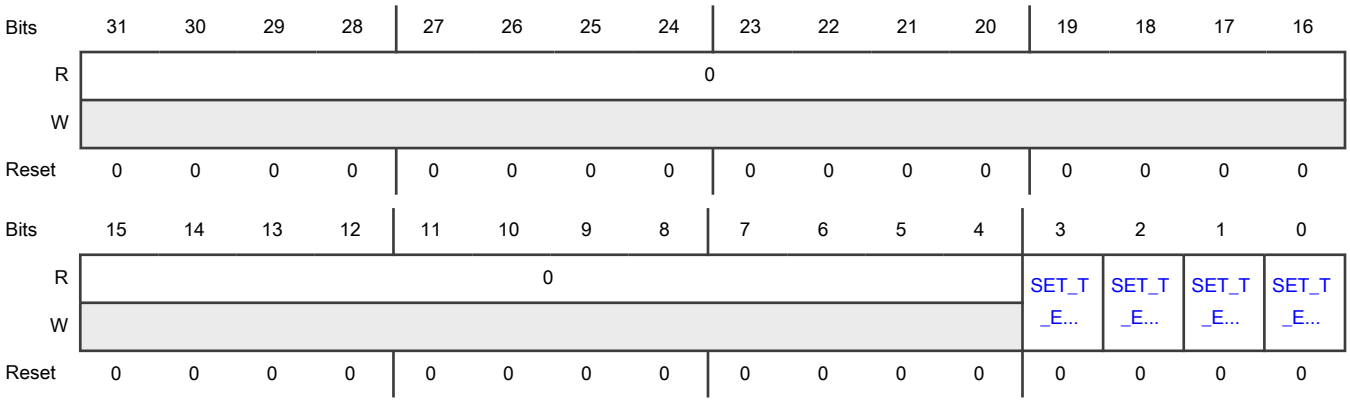
— or by setting the corresponding Set Timer Enable bit (SET_T_EN_*n*) in the Set Timer Enable register (SETTEN).

- To disable timer channels simultaneously, use the Clear Timer Enable register (CLRTEN).
- Writing a '0' to the Set Timer Enable register has no effect.

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled)), reading or writing the Set Timer Enable register will generate a transfer error.

Diagram



Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3 SET_T_EN_3	Set Timer 3 Enable To enable timer channel 3, write '1' to Set Timer 3 Enable bit. • The Set Timer 3 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL3 register. • Writing a 0 to Set Timer 3 Enable bit will not disable the counter. • The Set Timer 3 Enable bit will be cleared — when Timer Enable bit (TCTRL3[T_EN]) is set to 0 — or when '1' is written to the Clear Timer 3 Enable bit (CLRTEN[CLR_T_EN_3]). 0b - No effect 1b - Enables Timer Channel 3
2 SET_T_EN_2	Set Timer 2 Enable To enable timer channel 2, write '1' to Set Timer 2 Enable bit. • The Set Timer 2 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL2 register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Writing a 0 to Set Timer 2 Enable bit will not disable the counter. • The Set Timer 2 Enable bit will be cleared — when Timer Enable bit (TCTRL2[T_EN]) is set to 0 — or when '1' is written to the Clear Timer 2 Enable bit (CLRTEN[CLR_T_EN_2]). 0b - No Effect 1b - Enables Timer Channel 2
1 SET_T_EN_1	Set Timer 1 Enable To enable timer channel 1, write '1' to Set Timer 1 Enable bit. • The Set Timer 1 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL1 register. • Writing a 0 to Set Timer 1 Enable bit will not disable the counter. • The Set Timer 1 Enable bit will be cleared — when Timer Enable bit (TCTRL1[T_EN]) is set to 0 — or when '1' is written to the Clear Timer 1 Enable bit (CLRTEN[CLR_T_EN_1]). 0b - No Effect 1b - Enables Timer Channel 1
0 SET_T_EN_0	Set Timer 0 Enable To enable timer channel 0, write '1' to Set Timer 0 Enable bit. • The Set Timer 0 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL0 register. • Writing a 0 to Set Timer 0 Enable bit will not disable the counter. • The Set Timer 0 Enable bit will be cleared — when Timer Enable bit (TCTRL0[T_EN]) is set to 0 — or when '1' is written to the Clear Timer 0 Enable bit (CLRTEN[CLR_T_EN_0]). 0b - No effect 1b - Enables Timer Channel 0

45.5.1.8 Clear Timer Enable (CLRTEN)

Offset

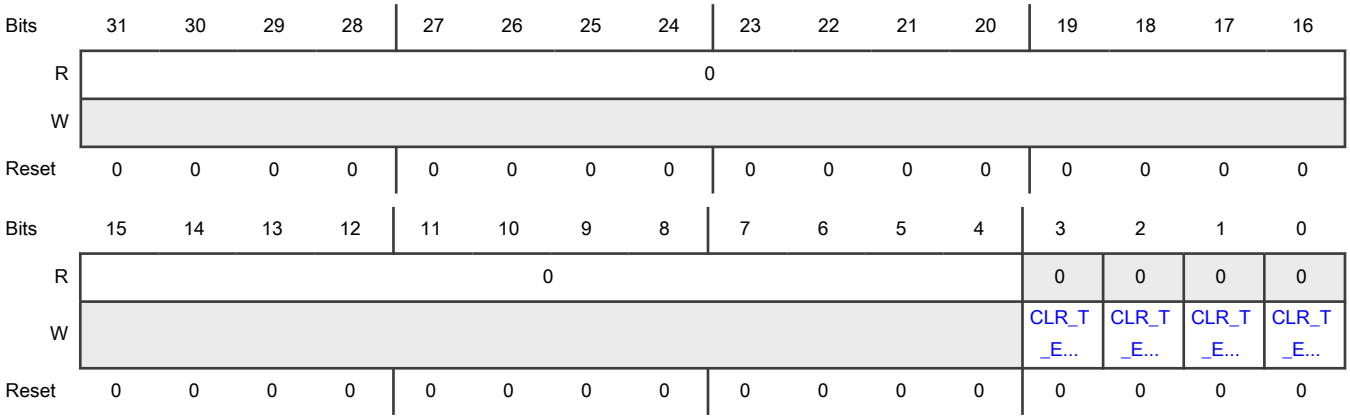
Register	Offset
CLRTEN	18h

Function

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled), writing the Clear Timer Enable register will generate a transfer error.

Diagram



Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3 CLR_T_EN_3	Clear Timer 3 Enable • To disable timer channel 3, write '1' to Clear Timer 3 Enable bit. • The Clear Timer 3 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL3 register. • Writing a 1 to Clear Timer 3 Enable bit will not enable the counter. • The Clear Timer 3 Enable bit is self-clearing, which means that the Clear Timer 3 Enable bit will always read 0. 0b - No Action 1b - Clear the Timer Enable bit (TCTRL3[T_EN]) for Timer Channel 3
2 CLR_T_EN_2	Clear Timer 2 Enable • To disable timer channel 2, write '1' to Clear Timer 2 Enable bit. • The Clear Timer 2 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL2 register. • Writing a 1 to Clear Timer 2 Enable bit will not enable the counter. • The Clear Timer 2 Enable bit is self-clearing, which means that the Clear Timer 2 Enable bit will always read 0. 0b - No Action 1b - Clear the Timer Enable bit (TCTRL2[T_EN]) for Timer Channel 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 CLR_T_EN_1	Clear Timer 1 Enable • To disable timer channel 1, write '1' to Clear Timer 1 Enable bit. • The Clear Timer 1 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL1 register. • Writing a 1 to Clear Timer 1 Enable bit will not enable the counter. • The Clear Timer 1 Enable bit is self-clearing, which means that the Clear Timer 1 Enable bit will always read 0. 0b - No Action 1b - Clear the Timer Enable bit (TCTRL1[T_EN]) for Timer Channel 1
0 CLR_T_EN_0	Clear Timer 0 Enable • To disable timer channel 0, write '1' to Clear Timer 0 Enable bit. • The Clear Timer 0 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL0 register. • Writing a 1 to Clear Timer 0 Enable bit will not enable the counter. • The Clear Timer 0 Enable bit is self-clearing, which means that the Clear Timer 0 Enable bit will always read 0. 0b - No action 1b - Clear the Timer Enable bit (TCTRL0[T_EN]) for Timer Channel 0

45.5.1.9 Timer Value (TVAL0 - TVAL3)

Offset

Register	Offset
TVAL0	20h
TVAL1	30h
TVAL2	40h
TVAL3	50h

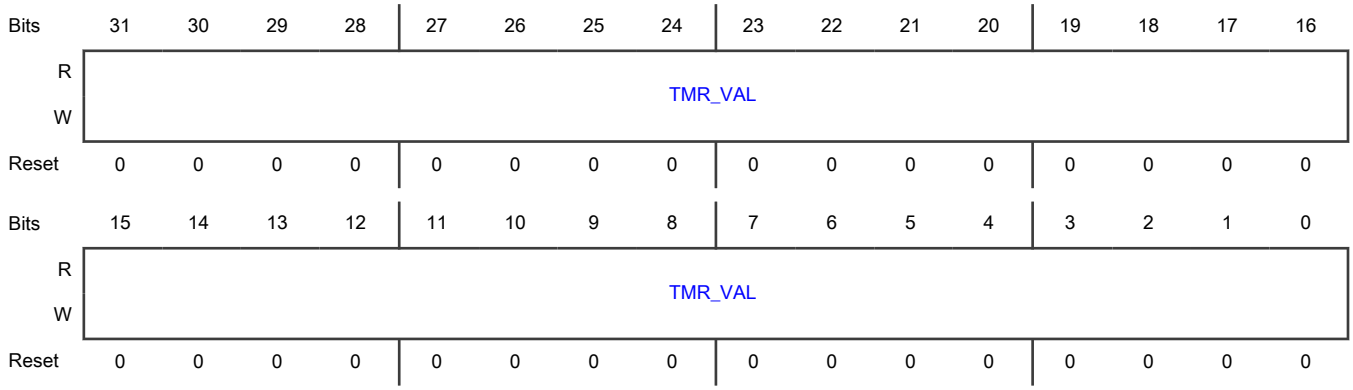
Function

- In compare modes: the Timer Value Registers (TVAL*n*) select the timeout period for the timer channels.
- In capture modes: the Timer Value Registers (TVAL*n*) are loaded with the value of the counter when the trigger asserts.

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled), then reading or writing the TVALn register will generate a transfer error.

Diagram



Fields

Field	Function
31-0 TMR_VAL	Timer Value • In compare mode: Timer Value (TMR_VAL) is the timer channel start value. — The timer counts down for (TVALn + 1) cycles until the timer reaches 0, then the timer generates an interrupt and loads the Timer Value register (TVAL*n*) value again. — Writing a new value to the Timer Value register (TVAL*n*) will not restart the timer channel; instead the new value will be loaded *after the timer expires*. — To abort the current timer cycle and start a timer period with a new value, the timer channel must be disabled and enabled *again*. • In capture mode: whenever the trigger asserts, the Timer Value register stores the inverse of the counter value. 0000_0000_0000_0000_0000_0000_0000_0000b,0000_0000_0000_0000_0000_0000_0000_0001b - Invalid load value in compare mode 0000_0000_0000_0000_0000_0000_0000_0010b-1111_1111_1111_1111_1111_1111_1111_1111b - In compare mode: the value to be loaded; in capture mode, the value of the timer

45.5.1.10 Current Timer Value (CVAL0 - CVAL3)

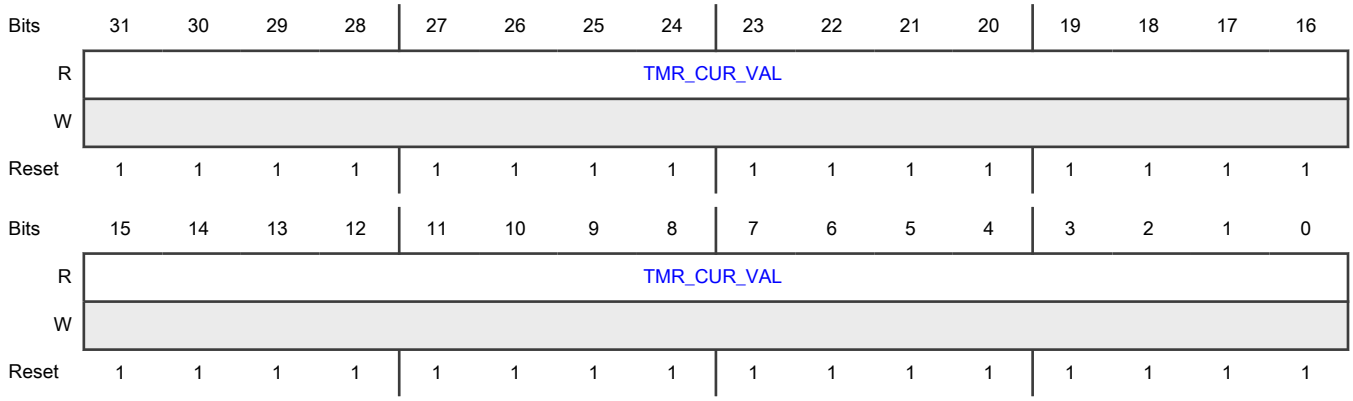
Offset

Register	Offset
CVAL0	24h
CVAL1	34h
CVAL2	44h
CVAL3	54h

Function

The Current Timer Value registers indicate the current timer counter value.

Diagram



Fields

Field	Function
31-0 TMR_CUR_VAL	Current Timer Value Represents the current timer value, if the timer is enabled.

45.5.1.11 Timer Control (TCTRL0 - TCTRL3)

Offset

Register	Offset
TCTRL0	28h
TCTRL1	38h
TCTRL2	48h
TCTRL3	58h

Function

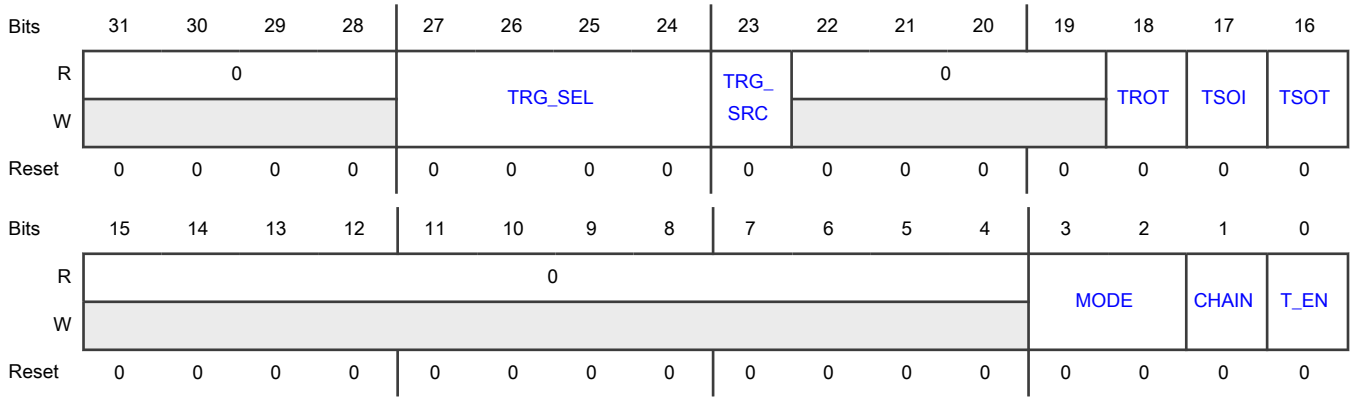
The Timer Control registers contain the control bits for each timer channel.

NOTE

- Unless the peripheral clock to the timers is enabled (Module Clock Enable bit (MCR[M_CEN]) is set (=1)), then reading or writing the Timer Control register will generate a transfer error.
- Timer controls should only be updated when the timer is disabled. In the Timer Control Register TCTRLn, these timer controls include:
  - Trigger Select TRG_SEL
  - Trigger Source TRG_SRC
  - Timer Reload On Trigger TROT
  - Timer Stop On Interrupt TSOI
  - Timer Start On Trigger TSOT
  - Timer Operation Mode MODE
  - Chain Channel CHAIN

  There are 2 ways to disable a timer: write 1 to the specific timer's Clear Timer Enable register bit (CLRTEN[CLR_T_EN_n]) or set the timer enable bit (TCTRLn[T_EN]) = 0 for that channel.

Diagram



Fields

Field	Function
31-28 —	This read-only field is reserved and always has the value 0
27-24 TRG_SEL	Trigger Select Selects the trigger to use for starting and/or reloading the LPIT timer. • The TRG_SEL field selects one trigger from the set of internal or external triggers that are selected by the Trigger Source bit (TRG_SRC) • Recall that the TRG_SRC bit selects between internal and external trigger signals for each channel **NOTE** The Trigger Select field should only be changed when the LPIT timer channel is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000b-0011b - Timer channel 0 - 3 trigger source is selected 0100b-1111b - Reserved
23 TRG_SRC	Trigger Source Selects between internal or external trigger sources. The trigger to be used is selected using the TRG_SRC and TRG_SEL bits. See LPIT chip-specific information for available external trigger options. If a channel does not have an associated external trigger, then set the Trigger Source bit (TRG_SRC) = 1. 0b - Selects external triggers 1b - Selects internal triggers
22-19 —	This read-only field is reserved and always has the value 0
18 TROT	Timer Reload On Trigger When set, the LPIT timer will reload when a rising edge is detected on the selected trigger input. The trigger input is ignored if the LPIT is disabled during debug mode (DBGEN = 0) or DOZE mode (DOZE_EN = 0) 0b - Timer will not reload on the selected trigger 1b - Timer will reload on the selected trigger
17 TSOI	Timer Stop On Interrupt Controls whether the channel timer will stop after it (the channel timer) times out or not. 0b - The channel timer does not stop after timeout 1b - The channel timer will stop after a timeout, and the channel timer will restart based on Timer Start On Trigger bit (TSOT). When TSOT = 0, the channel timer will restart after a rising edge on the Timer Enable bit (T_EN) is detected (which means that the timer channel is disabled and then enabled). When TSOT = 1, the channel timer will restart after a rising edge on the selected trigger is detected.
16 TSOT	Timer Start On Trigger Controls when the timer starts decrementing. 0b - Timer starts to decrement immediately based on the restart condition (controlled by the Timer Stop On Interrupt bit (TSOI)) 1b - Timer starts to decrement when a rising edge on a selected trigger is detected
15-4 —	This read-only field is reserved and always has the value 0
3-2 MODE	Timer Operation Mode Configures the channel timer's mode of operation. The MODE bits control how the timer decrements.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 32-bit Periodic Counter 01b - Dual 16-bit Periodic Counter 10b - 32-bit Trigger Accumulator 11b - 32-bit Trigger Input Capture
1 CHAIN	Chain Channel When enabled, the timer channel will decrement when timer channel N-1 trigger asserts. Timer channel 0 cannot be chained. 0b - Channel Chaining is disabled. The channel timer runs independently. 1b - Channel Chaining is enabled. The timer decrements on the previous channel's timeout.
0 T_EN	Timer Enable Enables or disables the Timer Channel 0b - Timer Channel is disabled 1b - Timer Channel is enabled

Chapter 46
Time Stamp Timer (TSTMR)

46.1 Chip-specific TSTMR information

Table 276. Reference links to related information

Topic	Related module	Reference
Full description	TSTMR	TSTMR
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

46.1.1 Module instances

This device has one instance of the TSTMR module, TSTMR0.

46.2 Overview

The Time Stamp Timer (TSTMR) is a 56-bit clock cycle counter, reset by system reset.

46.2.1 Features

- Free-running Time Stamp Timer

46.3 Functional description

The TSTMR module is a free running incrementing counter that starts running after system reset de-assertion and can be read at any time by the software for determining the software ticks. However, the software must follow the read sequence as mentioned in the TSTMR register descriptions for correctly reading the TSTMR value. The TSTMR is a 56-bit counter and hence requires two 32-bit reads to read the full value. The TSTMR runs off the 1 MHz clock and resets on every system reset. The counter only stops when the clock to the TSTMR is disabled.

See the chip-specific TSTMR information for implementation details of this module's instances.

46.4 TSTMR Memory map and register definition

The TSTMR Memory Map/Register Definition can be found here.

NOTE

TSTMR registers can be read with 32-bit accesses only.

46.4.1 TSTMR register descriptions

This section contains the detailed register descriptions for the TSTMR registers.

46.4.1.1 TSTMR memory map

TSTMR0.TSTMR0 base address: 4003_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Time Stamp Timer Register Low (LOW)	32	R	0000_0000h
4h	Time Stamp Timer Register High (HIGH)	32	R	0000_0000h

46.4.1.2 Time Stamp Timer Register Low (LOW)

Offset

Register	Offset
LOW	0h

Function

The Time Stamp Timer is a 56-bit clock cycle counter, reset by system reset. It is readable by way of the TSTMR HIGH and LOW registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Note the following:

- A complete read operation should include both TSTMR LOW and HIGH reads. If a HIGH read does not follow a LOW read, then any other Time Stamp value read will be locked at a fixed value.
- The TSTMR LOW read should occur first, followed by the TSTMR HIGH read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 VALUE	Time Stamp Timer Low Lower 32 bits of the 56-bit time stamp value. The software must read the LOW first, followed by a read to HIGH to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

46.4.1.3 Time Stamp Timer Register High (HIGH)

Offset

Register	Offset
HIGH	4h

Function

The Time Stamp Timer is a 56-bit counters clock cycle counter, reset by system reset. It is readable by way of the TSTMR HIGH and LOW registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Note the following:

- A complete read operation should include both TSTMR LOW and HIGH reads. If a HIGH read does not follow a LOW read, then any other Time Stamp value read will be locked at a fixed value.
- The TSTMR LOW read should occur first, followed by the TSTMR HIGH read.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								VALUE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 VALUE	Time Stamp Timer High Upper 24 bits of the 56-bit time stamp value. The software must read the LOW first, followed by a read to HIGH to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

Chapter 47
Timer/PWM Module (TPM)

47.1 Chip-specific TPM information

Table 277. Reference links to related information

Topic	Related module	Reference
Full description	TPM	TPM
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

47.1.1 Module instances

This device has three instances of the TPM module, TPM0, TPM1, and TPM2.

47.2 Overview

The TPM (Timer/PWM Module) is a 6-channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low power modes.

47.2.1 Block diagram

The TPM uses one input/output (I/O) pin per channel, CHn (TPM channel (n)) where n is the channel number.

The following figure shows the TPM structure. The central component of the TPM is the 32-bit counter with programmable final value and its counting can be up or up-down.

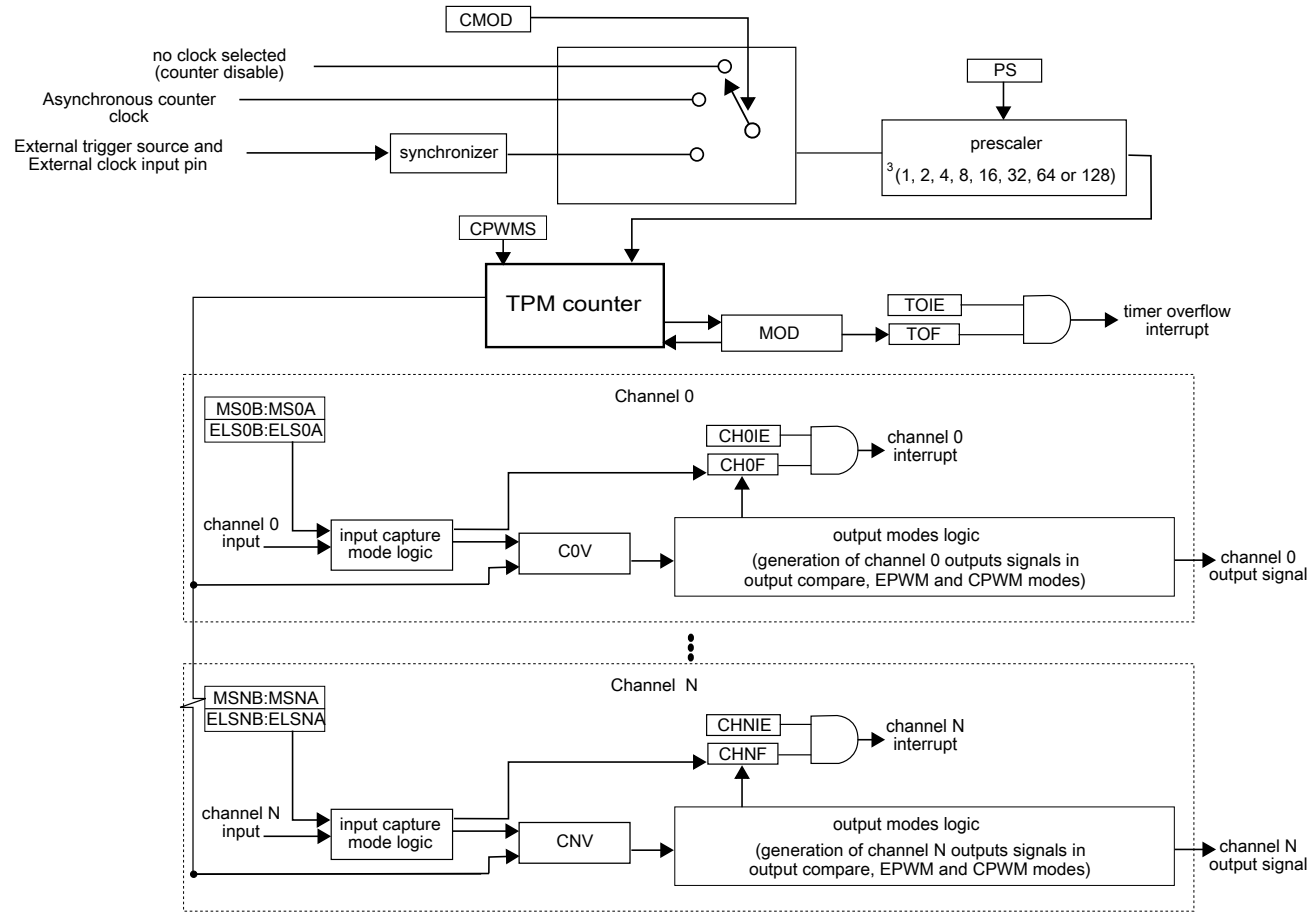


Figure 147. TPM block diagram

47.2.2 Features

The TPM features include:

- TPM clock mode is selectable
  - Can increment on every edge of the asynchronous counter clock
  - Can increment on rising edge of an external clock input synchronized to the asynchronous counter clock
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- TPM includes a 32-bit TPM counter
  - It can be a free-running counter or modulo counter
  - The counting can be up or up-down
- Includes 6 channels that can be configured as follows:
  - Input capture mode: the capture can occur on rising edges, falling edges or both edges
  - Output compare mode: the output signal can be set, cleared, pulsed, or toggled on match
  - Edge-aligned or center-aligned PWM mode for all channels
- Support the generation of an interrupt and/or DMA request per channel
- Support the generation of an interrupt and/or DMA request when the counter overflows

- Support selectable trigger input to optionally reset or cause the counter to start incrementing.
  - The counter can also optionally stop incrementing on counter overflow
- Support the generation of hardware triggers when the counter overflows and per channel

47.3 Functional description

The following sections describe the TPM features.

47.3.1 Clock domains

TPM supports two clock domains.

The bus clock domain is used by the register interface and for synchronizing interrupts and DMA requests.

The TPM counter clock domain is used to clock the counter and prescaler along with the output compare and input capture logic. The TPM counter clock is considered asynchronous to the bus clock, can be a higher or lower frequency than the bus clock and can optionally remain operational in low power modes. When the global timebase feature is enabled, any TPM instance that is sharing the global timebase must be clocked by the same TPM counter clock.

47.3.1.1 Counter Clock Mode

The CMOD[1:0] bits in the SC register either disable the TPM counter or select one of three possible clock sources for the TPM counter. After any reset, CMOD[1:0] = 0:0 so the TPM counter is disabled. The CMOD[1:0] can configure one of the following counter clock sources.

- Asynchronous counter clock
- External clock input pin
- External trigger source

The CMOD[1:0] bits may be read or written at any time. Disabling the TPM counter by writing zero to the CMOD[1:0] bits does not affect the TPM counter value or other registers, but must be acknowledged by the TPM counter clock domain before they read as zero.

The external clock input and external trigger source pass through a synchronizer clocked by the TPM counter clock to ensure that counter transitions are properly aligned to counter clock transitions. Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must be less than half of the counter clock frequency.

47.3.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and TPM counter.

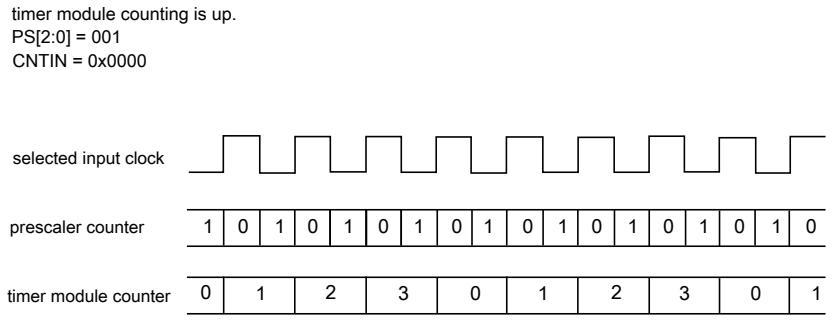


Figure 148. Example of the Prescaler Counter

47.3.3 Counter

TPM has a 32-bit counter that is used by the channels for either input or output modes. The counter updates from the selected clock divided by the prescaler. The TPM counter has these modes of operation:

- Up counting
- Up-down counting

47.3.3.1 Up counting

Up counting is selected when SC[CPWMS] = 0.

The value of zero is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with zero.

The TPM period when using up counting is (MOD + 0x0001) × period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to zero.

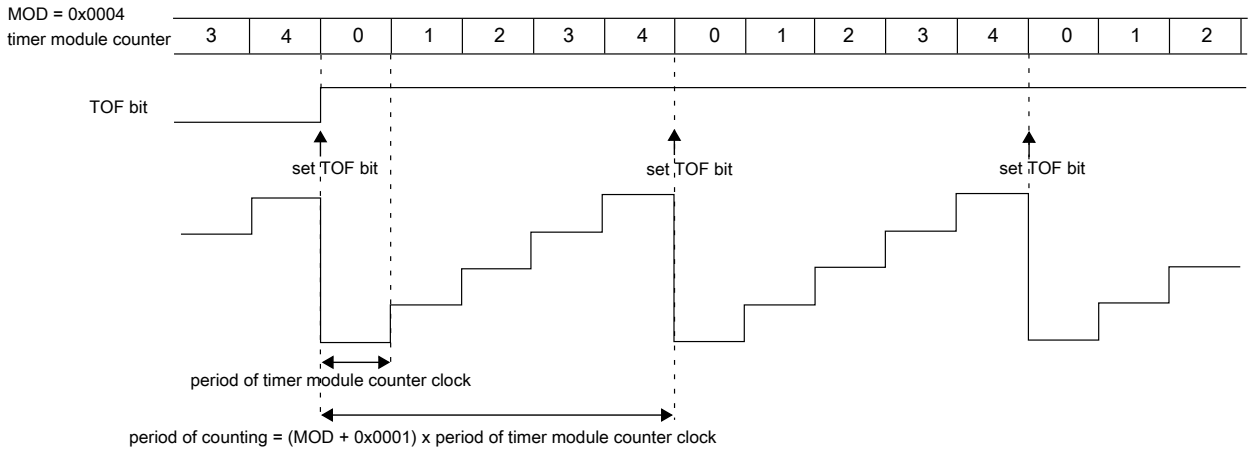


Figure 149. Example of TPM Up Counting

NOTE

- Configuring MOD = 0 is a redundant condition. In this case, the TPM counter is always equal to MOD and the TOF bit is set in each rising edge of the TPM counter clock.
- Configuring MOD = 1 and PS = 0 will attempt to set the TOF bit every second TPM counter clock. Due to synchronization delays between the TPM counter clock and the bus clock, the TOF must be cleared by software two counter clock cycles before TOF can be set again.

47.3.3.2 Up-down counting

Up-down counting is selected when SC[CPWMS] = 1. When configured for up-down counting, configuring MOD to less than 2 is not supported.

The value of 0 is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to zero and the up-down counting restarts.

The TPM period when using up-down counting is 2 × MOD × period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to (MOD – 1).

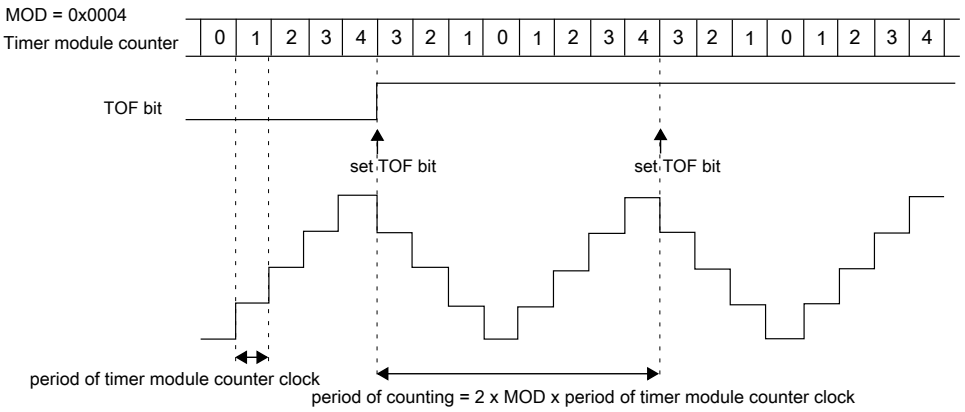


Figure 150. Example of up-down counting

47.3.3.3 Counter Reset

Any write to CNT resets the TPM counter and the channel outputs to their initial values (except for channels in output compare mode).

47.3.3.4 Global time base (GTB)

The global time base (GTB) is a TPM function that allows multiple TPM modules to share the same timebase. When the global time base is enabled (CONF[GTBEEN] = 1), the local TPM channels use the counter value, counter enable and overflow indication from the TPM generating the global time base. If the local TPM counter is not generating the global time base, then it can be used as an independent counter or pulse accumulator.

The local TPM counter can also be configured to synchronize to the global time base, by configuring (GTBSYNC = 1). When synchronized to the global time base, the local counter will use the counter enable and counter overflow indication from the TPM generating the global time base. This enables multiple TPM to be configured with the same phase, but with different periods (although the global time base must be configured with the longest period).

47.3.3.5 Counter trigger

The TPM counter can be configured to start, stop or reset in response to a hardware trigger input. The trigger input is synchronized to the asynchronous counter clock, so there is a 3 counter clock delay between the trigger assertion and the counter responding.

- When (CSOT = 1), the counter will not start incrementing until a rising edge is detected on the trigger input.
- When (CSOO= 1), the counter will stop incrementing whenever the TOF flag is set. The counter does not increment again unless it is disabled, or if CSOT = 1 and a rising edge is detected on the trigger input.
- When (CROT= 1), the counter will reset to zero as if an overflow occurred whenever a rising edge is detected on the trigger input. Any PWM channels will update to their reload state.
- When (CPOT = 1), the counter will pause incrementing whenever the trigger input is asserted. The counter will continue incrementing when the trigger input negates. PWM output channels are forced to their default state.

The polarity of the external input trigger can be configured by the TRGPOL register bit.

When an internal trigger source is selected, the trigger input is selected from one or more channel input capture events. The input capture filters are used with the internal trigger sources and the POLn bits can be used to invert the polarity of the input channels. Note that following restrictions apply with input capture channel sources.

- When (CSOT = 1), the counter will only start incrementing on a rising edge on the channel input, provided ELSnA = 1.
- When (CROT= 1), the counter will reset to zero on either edge of the channel input, as configured by ELSnB:ELSnA.

- When (CPOT = 1), the counter will pause incrementing whenever the channel input is asserted. PWM output channels are forced to their default state.

47.3.4 Input Capture Mode

The input capture mode is selected when (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

When a selected edge occurs on the channel input, the current value of the TPM counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1 (see the following figure).

When a channel is configured for input capture, the CHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is counter clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register are ignored in input capture mode.

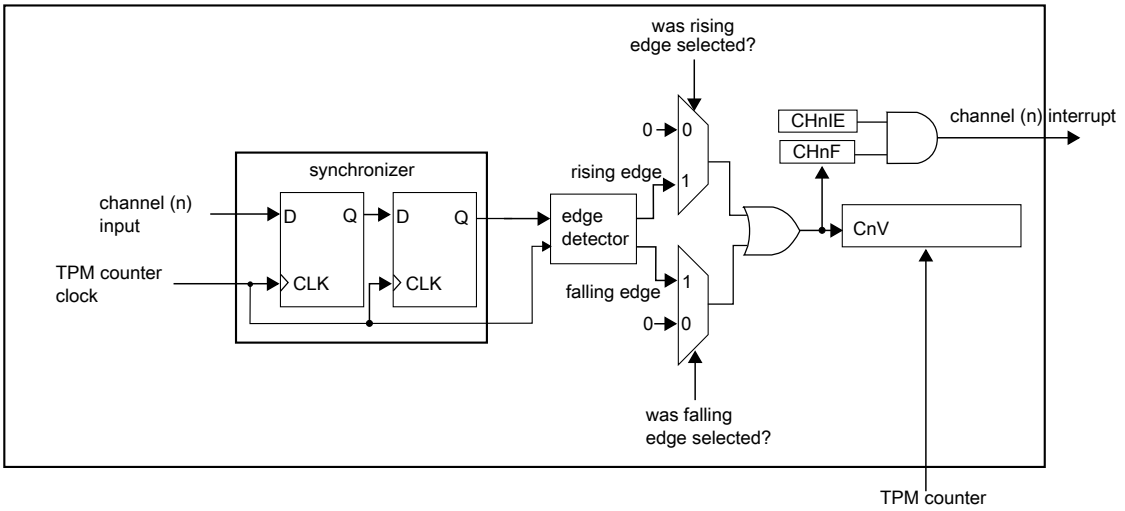


Figure 151. Input capture mode

The CHnF bit is set on the third rising edge of the TPM counter clock after a valid edge occurs on the channel input.

47.3.5 Output Compare Mode

The output compare mode is selected when (CPWMS = 0), and (MSnB:MSnA = X:1).

In output compare mode, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the TPM counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared or toggled if MSnB is clear. If MSnB is set then the channel (n) output is pulsed high or low for as long as the TPM counter matches the value in the CnV register.

When a channel is initially configured to output compare mode, the channel output updates with its negated value (logic 0 for set/toggle/pulse high and logic one for clear/pulse low).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV).

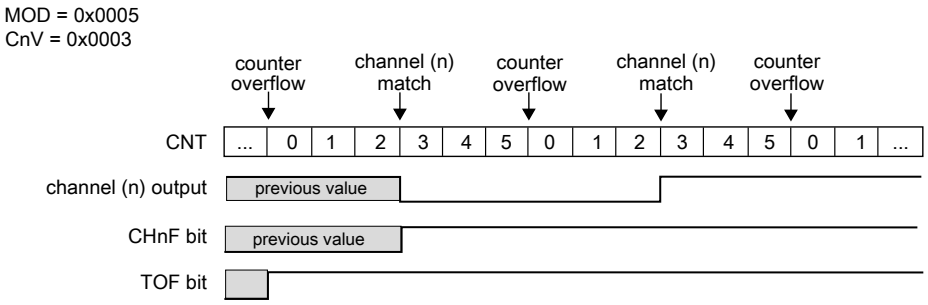


Figure 152. Example of the output compare mode when the match toggles the channel output

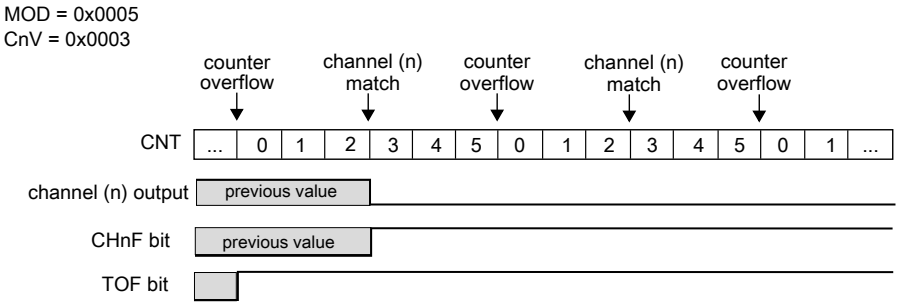


Figure 153. Example of the output compare mode when the match clears the channel output

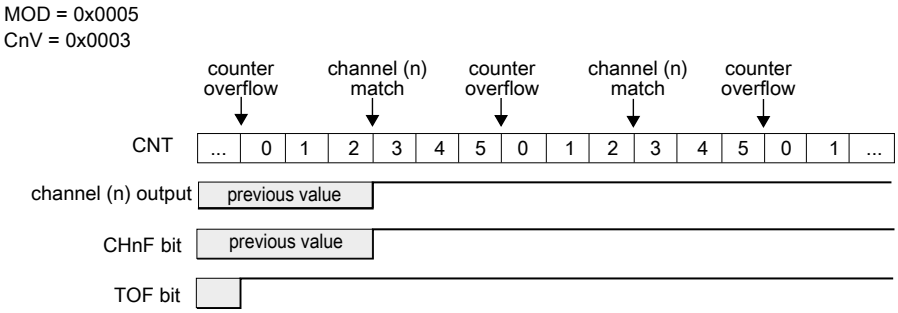


Figure 154. Example of the output compare mode when the match sets the channel output

Software can also use the output compare mode with (MSnB:MSnA = 0:1) and (ELSnB:ELSnA = 0:0). In this case (known as software compare mode), when the TPM counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1); however, the channel (n) output is not modified or controlled by TPM.

47.3.6 Edge-Aligned PWM (EPWM) Mode

The edge-aligned mode is selected when (CPWMS = 0), and (MSnB:MSnA = 1:0). The EPWM period is determined by (MOD + 0x0001) and the pulse width (duty cycle) is determined by CnV.

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an TPM.

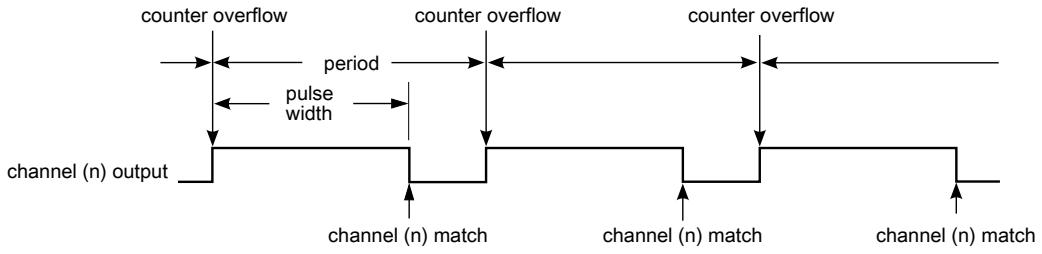


Figure 155. EPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow or reload (when the zero is loaded into the TPM counter), and it is forced low at the channel (n) match (TPM counter = CnV). When first enabled or whenever the TPM counter is paused, the channel (n) output is forced high.

If (ELSnB:ELSnA = 0:0), then the channel (n) output is forced high at the counter overflow or reload (when the zero is loaded into the TPM counter), and it is forced low at the channel (n) match (TPM counter = CnV). When first enabled or whenever the TPM counter is paused, the channel (n) output is forced low.

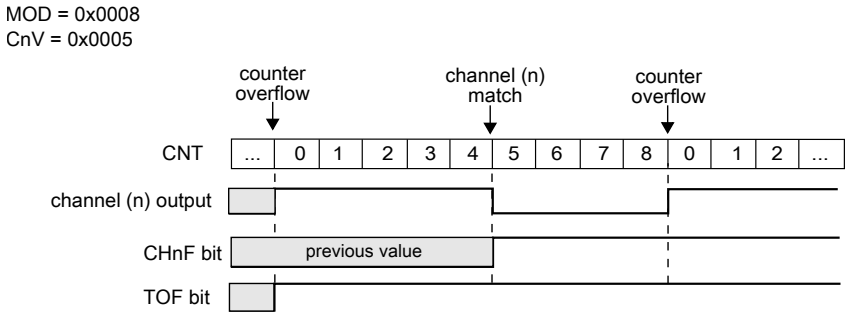


Figure 156. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:1), then the channel (n) output is forced low at the counter overflow or reload (when zero is loaded into the TPM counter), and it is forced high at the channel (n) match (TPM counter = CnV). When first enabled or whenever the TPM counter is paused, the channel (n) output is forced low.

If (ELSnB:ELSnA = 1:1), then the channel (n) output is forced low at the counter overflow or reload (when zero is loaded into the TPM counter), and it is forced high at the channel (n) match (TPM counter = CnV). When first enabled or whenever the TPM counter is paused, the channel (n) output is forced high.

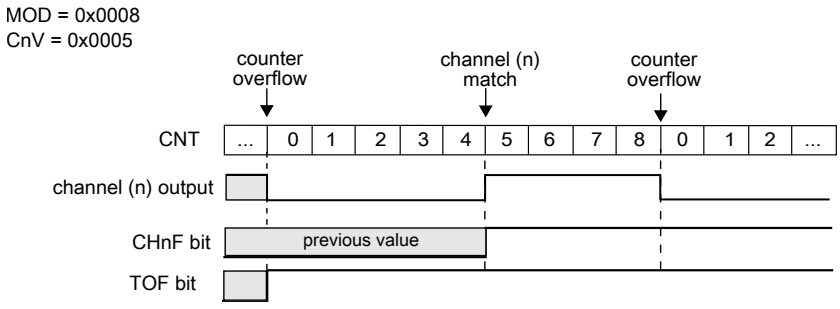


Figure 157. EPWM signal with ELSnB:ELSnA = 0:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal. If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set since there is never a channel (n) match. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

47.3.7 Center-Aligned PWM (CPWM) Mode

The center-aligned mode is selected when (CPWMS = 1) and (MSnB:MSnA = 1:0).

The CPWM pulse width (duty cycle) is determined by 2 × CnV and the period is determined by 2 × MOD (see the following figure). MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the TPM counter counts up until it reaches MOD and then counts down until it reaches zero.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV) when the TPM counting is down (at the begin of the pulse width) and when the TPM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are when the TPM counter is zero.

The other channel modes are not designed to be used with the up-down counter (CPWMS = 1). Therefore, all TPM channels should be used in CPWM mode when (CPWMS = 1).

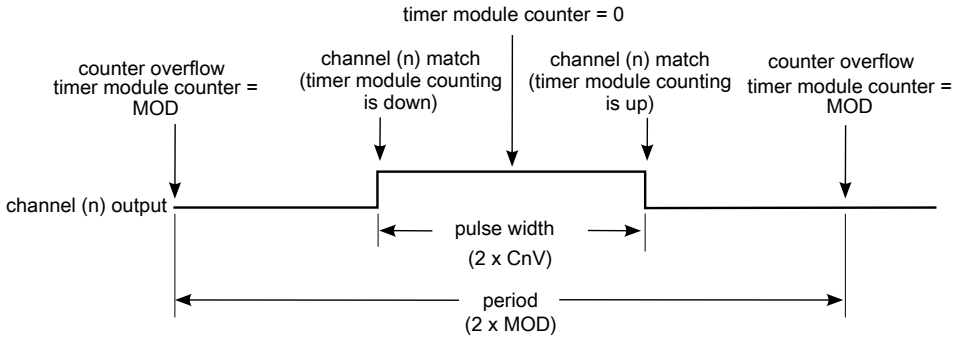


Figure 158. CPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (TPM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. When first enabled, when the TPM counter is reloaded, or whenever the TPM counter is paused, the channel (n) output is forced high.

If (ELSnB:ELSnA = 0:0), then the channel (n) output is forced high at the channel (n) match (TPM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. When first enabled, when the TPM counter is reloaded, or whenever the TPM counter is paused, the channel (n) output is forced low.

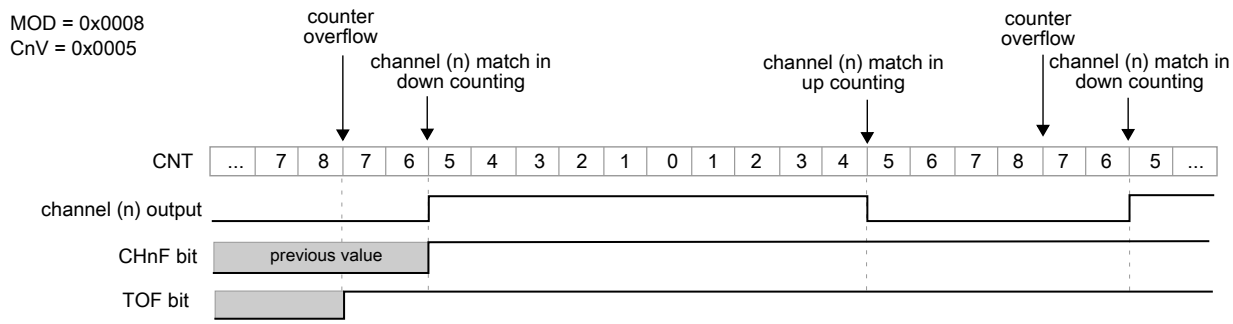


Figure 159. CPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:1), then the channel (n) output is forced low at the channel (n) match (TPM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. When first enabled, when the TPM counter is reloaded, or whenever the TPM counter is paused, the channel (n) output is forced low.

If (ELSnB:ELSnA = 1:1), then the channel (n) output is forced low at the channel (n) match (TPM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. When first enabled, when the TPM counter is reloaded, or whenever the TPM counter is paused, the channel (n) output is forced high.

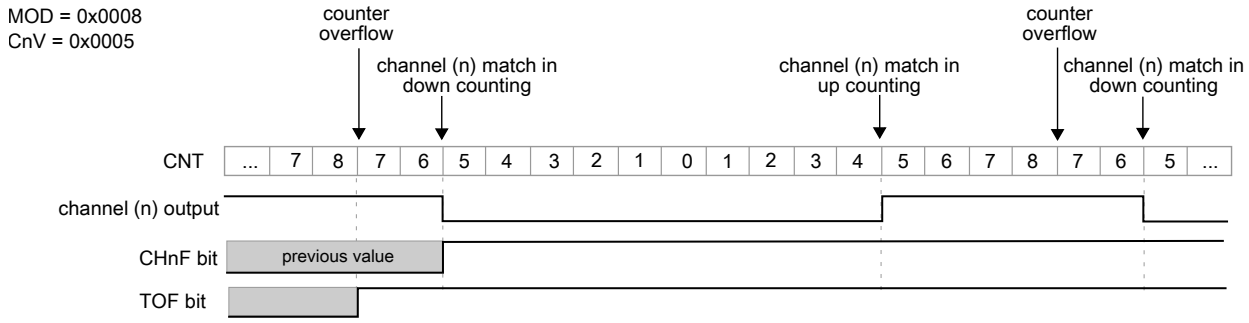


Figure 160. CPWM signal with ELSnB:ELSnA = 0:1

If (CnV = 0x0000) then the channel (n) output is a 0% duty cycle CPWM signal.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle CPWM signal, although the CHnF bit is set when the counter changes from incrementing to decrementing. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle CPWM signal.

47.3.8 Combine PWM mode

The Combine PWM mode is selected when:

- MSnB:MSnA = 10
- COMBINEn = 1
- QUADEN = 0, and
- CPWMS = 0

In Combine PWM mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

The PWM period is determined by (MOD + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|). If [C(n)V = C(n+1)V = 0x0000], then the channel (n) output has a 0% duty cycle. If [C(n)V = C(n+1)V > MOD], then the channel (n) output has a 100% duty cycle, and CHnF is not set since there is never a channel (n) match. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle signal.

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = C(n)V). The CH(n+1)F bit is set and the channel (n+1) interrupt is generated, if CH(n+1)IE = 1, at the channel (n+1) match (TPM counter = C(n+1)V).

If channel (n) (ELSnB:ELSnA = 0:1), then the channel (n) output is forced low at the beginning of the period (TPM counter is zero) and at the channel (n+1) match (TPM counter = C(n+1)V). It is forced high at the channel (n) match (TPM counter = C(n)V).

If channel (n) (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the beginning of the period (TPM counter is zero) and at the channel (n+1) match (TPM counter = C(n+1)V). It is forced low at the channel (n) match (TPM counter = C(n)V).

When (COMSWAPn = 1), then the channel (n) output is forced low or high at the beginning of the period (TPM counter is zero) and at the channel (n) match (TPM counter = C(n)V). It is forced high or low at the channel (n+1) match (TPM counter = C(n+1)V).

The channel (n+1) output is generated the same as the channel (n) output, but the output polarity is controlled by the channel (n+1) ELSnB:ELSnA configuration.

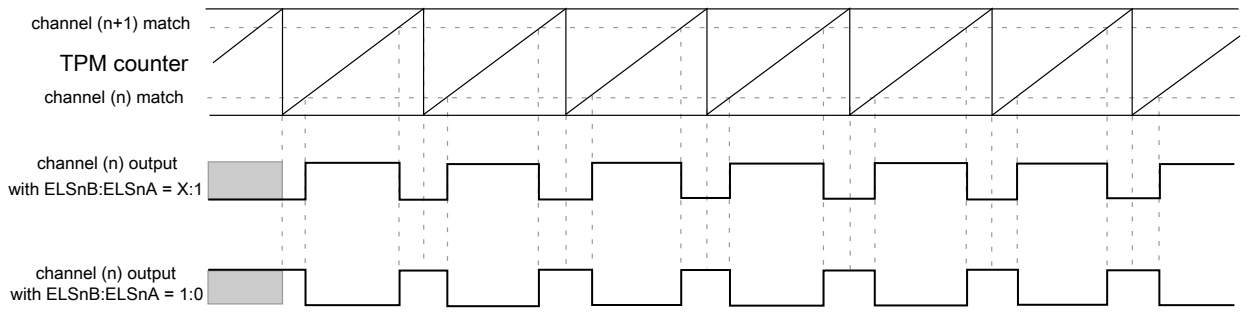


Figure 161. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

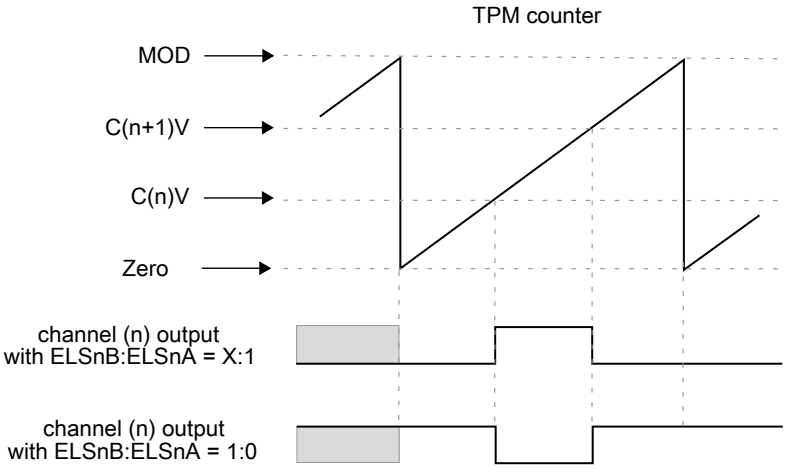


Figure 162. Channel (n) output if (C(n)V < MOD) and (C(n+1)V < MOD) and (C(n)V < C(n+1)V)

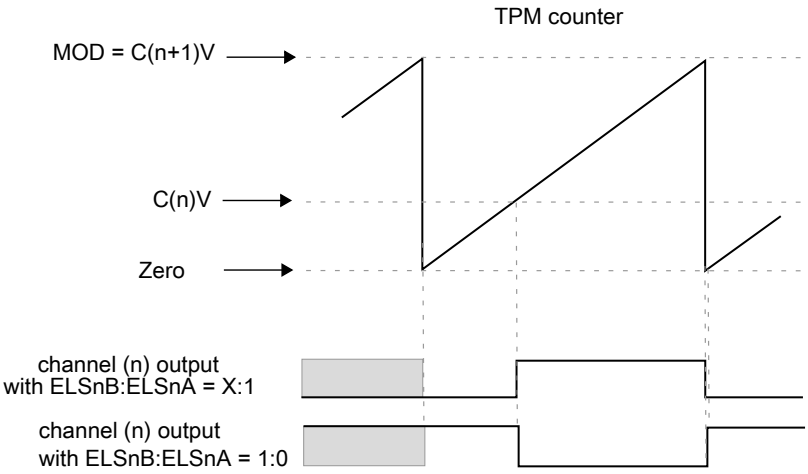


Figure 163. Channel (n) output if (C(n)V < MOD) and (C(n+1)V = MOD)

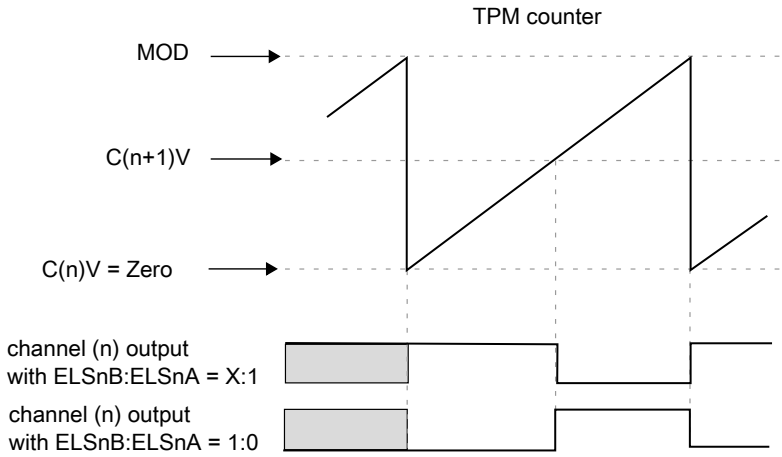


Figure 164. Channel (n) output if (C(n)V = zero) and (C(n+1)V < MOD)

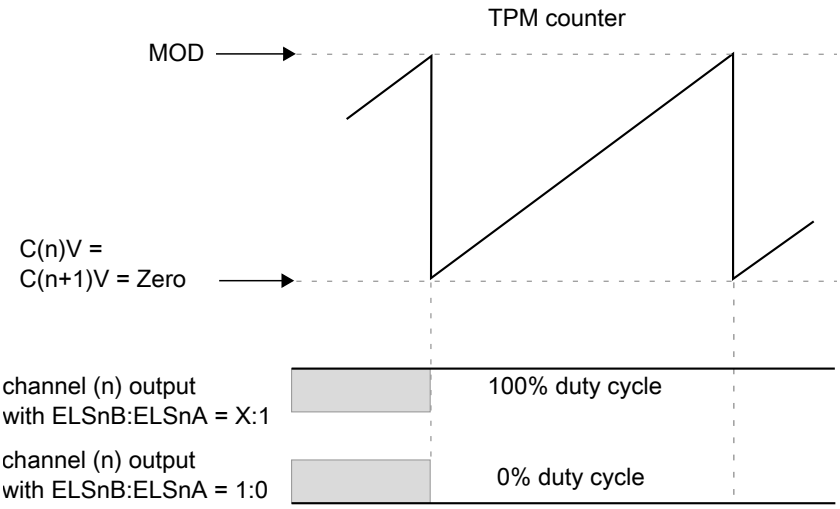


Figure 165. Channel (n) output if (C(n)V = C(n+1)V = zero)

47.3.9 Combine Input Capture mode

The Combine Input Capture mode is selected if COMBINEn= 1 and MSnB:MSnA = 00 and ELSnB:ELSnA ≠ 00. This mode allows to measure a pulse width of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode.

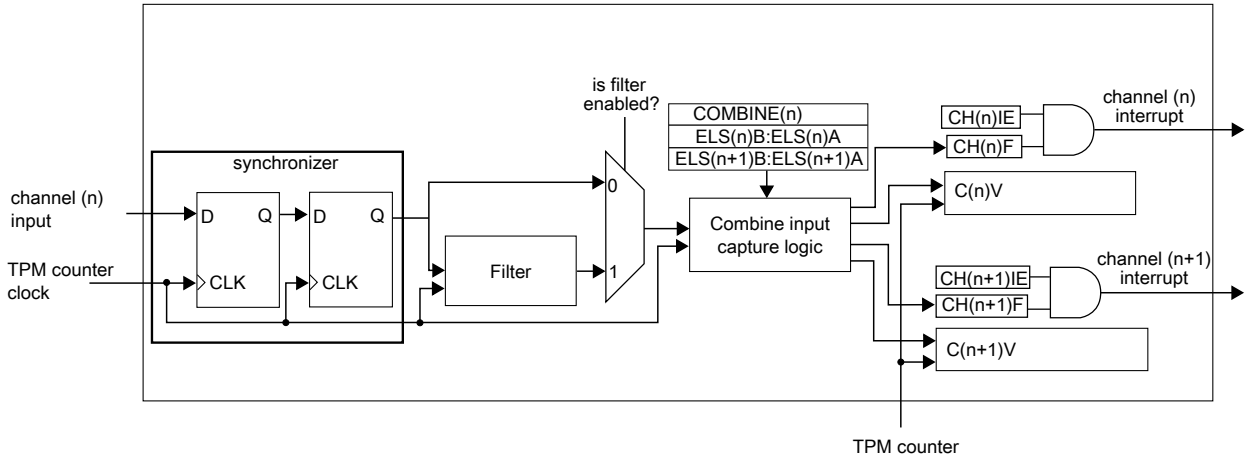


Figure 166. Combine Input Capture mode block diagram

The ELS(n)B:ELS(n)A bits select the edge that is captured by channel (n), and ELS(n+1)B:ELS(n+1)A bits select the edge that is captured by channel (n+1).

In the Combine Input Capture mode, only channel (n) input is used and channel (n+1) input is ignored, when COMSWAPn=1 then only channel (n+1) input is used and channel (n) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then CH(n)F bit is set and the channel (n) interrupt is generated (if CH(n)IE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input, then CH(n+1)F bit is set and the channel (n+1) interrupt is generated (if CH(n+1)IE = 1).

The C(n)V register stores the value of TPM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of TPM counter when the selected edge by channel (n+1) is detected at channel (n) input.

NOTE

- The CH(n)F, CH(n)IE, MS(n)A, ELS(n)B, and ELS(n)A bits are channel (n) bits.
- The CH(n+1)F, CH(n+1)IE, MS(n+1)A, ELS(n+1)B, and ELS(n+1)A bits are channel (n+1) bits.
- The Combine Input Capture mode must be used with ELS(n)B:ELS(n)A = 0:1 or 1:0, ELS(n+1)B:ELS(n+1)A = 0:1 or 1:0.

47.3.10 Input Capture Filter

The input capture filter function is only in input capture mode, or in software compare mode when quadrature decoder mode is enabled.

First, the input signal is synchronized by the TPM counter clock. Following synchronization, the input signal enters the filter block. See the following figure.

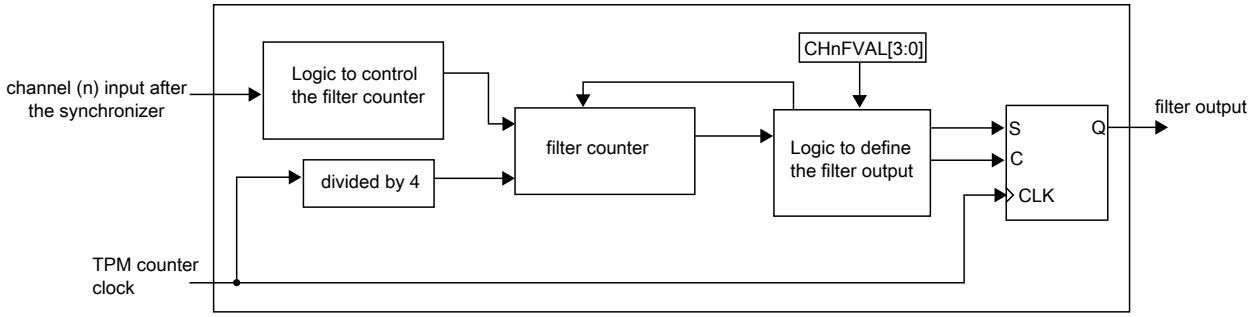


Figure 167. Channel input filter

When there is a state change in the input signal, the counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to (CHnFVAL[3:0] × 4), the state change of the input signal is validated.

If the opposite edge appears on the input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by (CHnFVAL[3:0] × 4 counter clocks) is regarded as a glitch and is not passed through the filter. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed by 2 rising edges of the counter clock. If (CHnFVAL[3:0] ≠ 0000), then the input signal is delayed by the minimum pulse width (CHnFVAL[3:0] × 4 system clocks) plus a further 3 rising edges of the system clock: two rising edges to the synchronizer, plus one more to the edge detector. In other words, CHnF is set (3 + 4 × CHnFVAL[3:0]) counter clock periods after a valid edge occurs on the channel input.

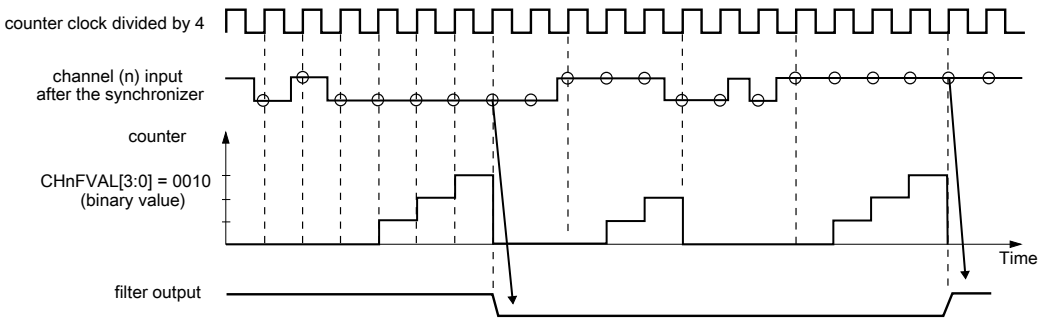


Figure 168. Channel input filter example

47.3.11 Deadtime insertion

The deadtime insertion is enabled in PWM combine modes for each pair of channels when CH(n)FVAL and CH(n+1)FVAL are non-zero. The deadtime delay that is used for each TPM channel is defined as (CH(n+1)FVAL[3:0] x 64) + (CH(n)FVAL[3:0] x 4).

The deadtime delay insertion ensures that no two complementary signals (even channel (n) and odd channel (n+1)) drive the active state at the same time.

If ELSnB:ELSnA = 0:1, ELS(n+1)B:ELS(n+1)A = 1:0, COMSWAPn = 0, and the deadtime is enabled, then when the channel (n) match (TPM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (TPM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If ELSnB:ELSnA = 1:0, ELS(n+1)B:ELS(n+1)A = 0:1, COMSWAPn = 1,and the deadtime is enabled, then when the channel (n+1) match (TPM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. Similarly, when the channel (n) match (TPM counter = CnV) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set.

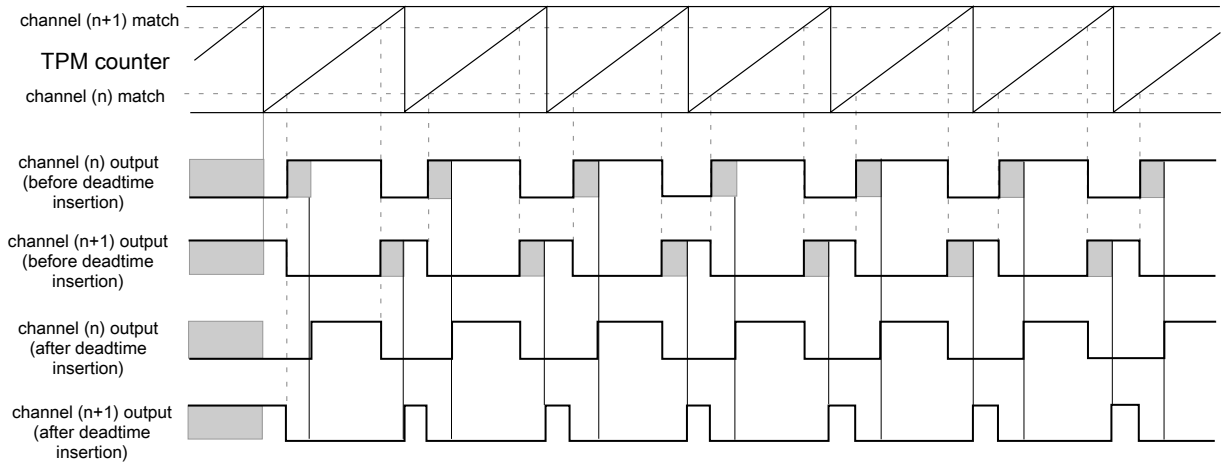


Figure 169. Deadtime insertion with ELSnB:ELSnA = 0:1, ELS(n+1)B:ELS(n+1)A = 1:0, COMSWAPn = 0

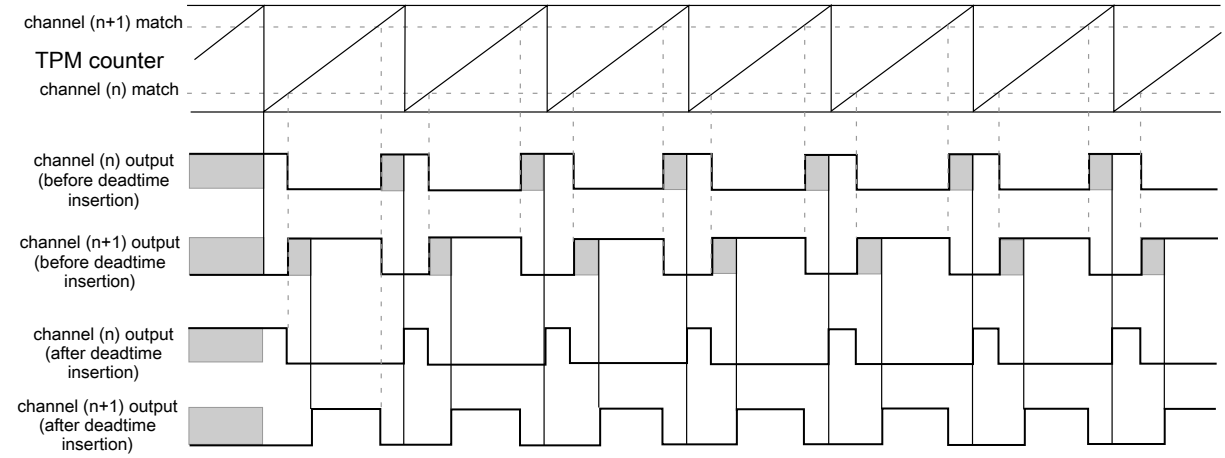


Figure 170. Deadtime insertion with ELSnB:ELSnA = 1:0, ELS(n+1)B:ELS(n+1)A = 0:1, COMSWAPn = 1

47.3.12 Quadrature Decoder mode

The Quadrature Decoder mode is selected if (QUADEN = 1). The Quadrature Decoder mode uses the channel 0 (phase A) and channel 1 (phase B) input signals to control the TPM counter increment and decrement. The following figure shows the quadrature decoder block diagram.

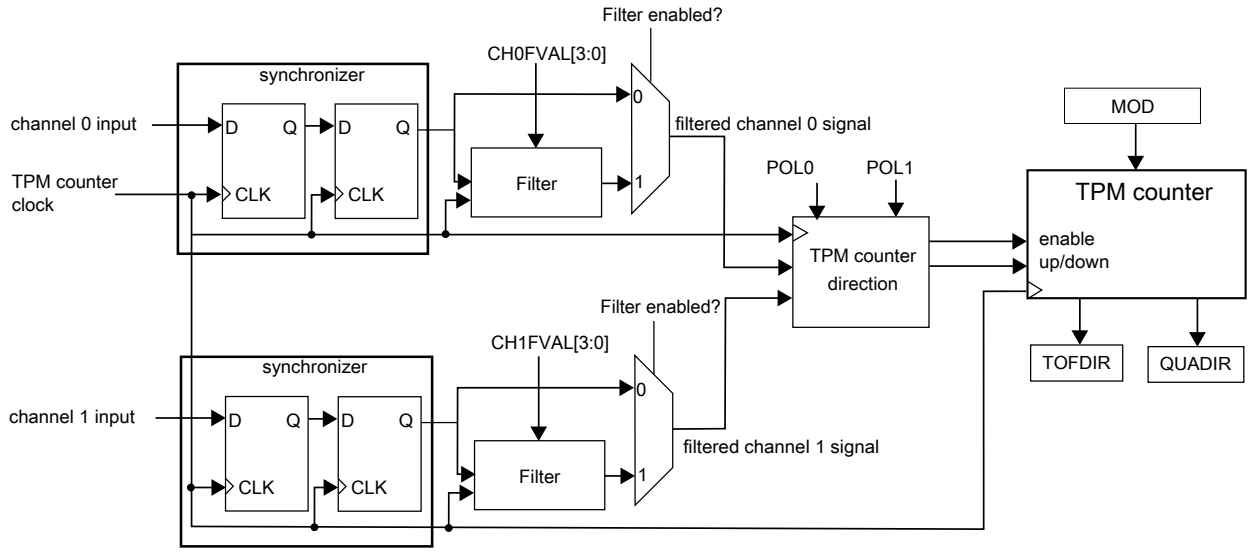


Figure 171. Quadrature Decoder block diagram

The input capture filter and channel polarity registers are used to configure the input filter and polarity for the channel 0 and channel 1 inputs in quadrature decode mode.

NOTE

Notice that the TPM counter is clocked by the channel 0 and channel 1 input signals when quadrature decoder mode is selected. Therefore In quadrature decoder mode, channel 0 and channel 1 can only be used in software compare mode and other TPM channels can only be used in input capture or output compare modes.

The QUADMODE selects the encoding mode used in the Quadrature Decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the channel 1 input value indicates the counting direction, and the channel 0 input defines the counting rate. The TPM counter is updated when there is a rising edge at channel 0 input signal.

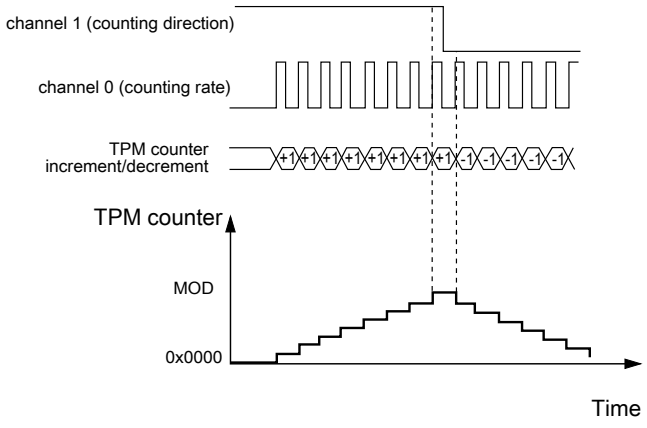


Figure 172. Quadrature Decoder – Count and Direction Encoding mode

If QUADMODE = 0, then the Phase Encoding mode is enabled; see the following figure. In this mode, the relationship between channel 0 and channel 1 signals indicates the counting direction, and channel 0 and channel 1 signals define the counting rate. The TPM counter is updated when there is an edge either at the channel 0 or channel 1 signals.

If CH0POL= 0 and CH1POL = 0, then the TPM counter increment happens when:

- there is a rising edge at channel 0 signal and channel 1 signal is at logic zero;

- there is a rising edge at channel 1 signal and channel 0 signal is at logic one;
- there is a falling edge at channel 1 signal and channel 0 signal is at logic zero;
- there is a falling edge at channel 0 signal and channel 1 signal is at logic one;

and the TPM counter decrement happens when:

- there is a falling edge at channel 0 signal and channel 1 signal is at logic zero;
- there is a falling edge at channel 1 signal and channel 0 signal is at logic one;
- there is a rising edge at channel 1 signal and channel 0 signal is at logic zero;
- there is a rising edge at channel 0 signal and channel 1 signal is at logic one.

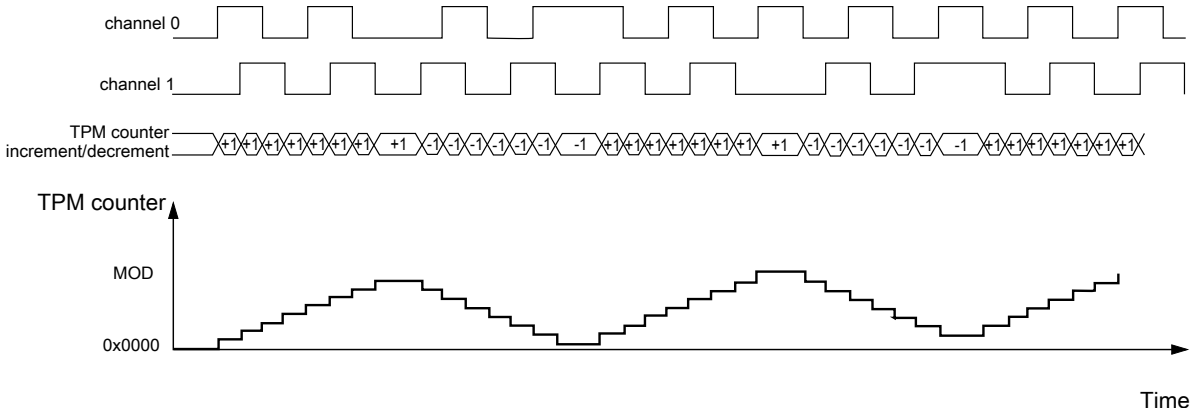


Figure 173. Quadrature Decoder – Phase Encoding mode

The following figure shows the TPM counter overflow in up counting. In this case, when the TPM counter changes from MOD to zero, TOF and TOFDIR bits are set. TOF bit indicates the TPM counter overflow occurred. TOFDIR indicates the counting was up when the TPM counter overflow occurred.

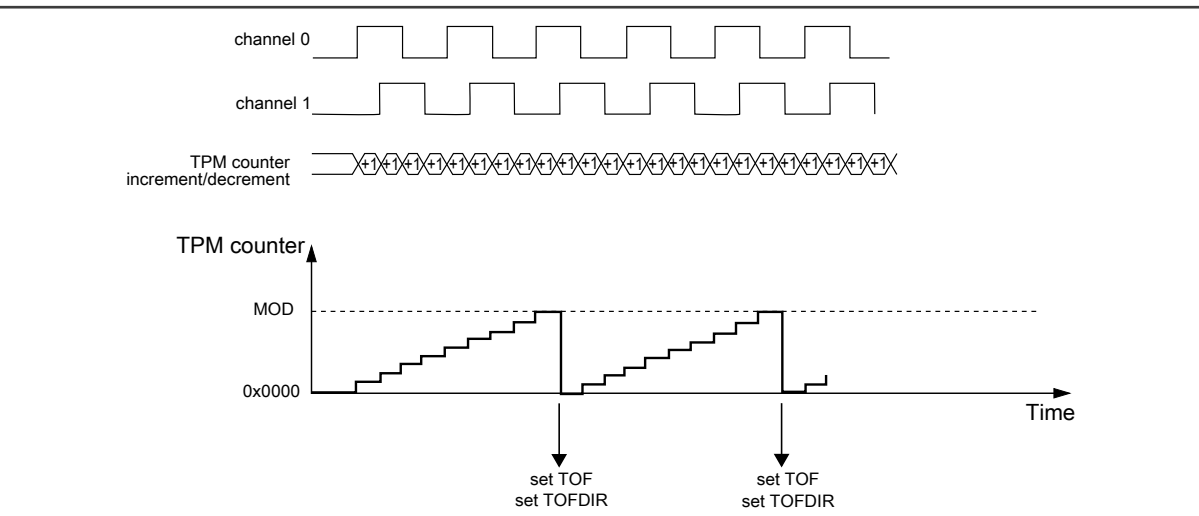


Figure 174. TPM Counter overflow in up counting for Quadrature Decoder mode

The following figure shows the TPM counter overflow in down counting. In this case, when the TPM counter changes from zero to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the TPM counter overflow occurred. TOFDIR indicates the counting was down when the TPM counter overflow occurred.

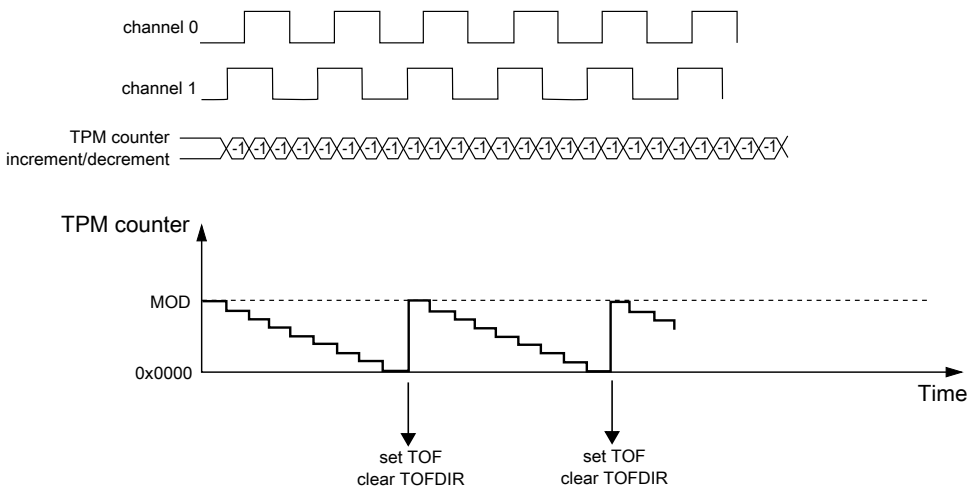


Figure 175. TPM counter overflow in down counting for Quadrature Decoder mode

47.3.13 Registers Updated from Write Buffers

47.3.13.1 MOD Register Update

If (CMOD[1:0] = 0:0) then MOD register is updated when MOD register is written.

If (CMOD[1:0] ≠ 0:0), then MOD register is updated according to the CPWMS bit, that is:

- If the selected mode is not CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to (MOD – 1).

47.3.13.2 CnV Register Update

If (CMOD[1:0] = 0:0) then CnV register is updated when CnV register is written.

If (CMOD[1:0] ≠ 0:0), then CnV register is updated according to the selected mode, that is:

- If the selected mode is output compare then CnV register is updated on the next TPM counter increment (end of the prescaler counting) after CnV register was written.
- If the selected mode is EPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to (MOD – 1).

47.3.14 DMA

The channel and overflow flags generate a DMA transfer request according to DMA and CHnIE/TOIE bits. See the following table for more information.

Table 278. DMA Transfer Request

DMA	CHnIE/ TOIE	Channel/Overflow DMA Transfer Request	Channel/Overflow Interrupt
0	0	The channel/overflow DMA transfer request is not generated.	The channel/overflow interrupt is not generated.
0	1	The channel/overflow DMA transfer request is not generated.	The channel/overflow interrupt is generated if (CHnF/TOF = 1).
1	0	The channel/overflow DMA transfer request is generated if (CHnF/TOF = 1).	The channel/overflow interrupt is not generated.
1	1	The channel/overflow DMA transfer request is generated if (CHnF/TOF = 1).	The channel/overflow interrupt is generated if (CHnF/TOF = 1).

If DMA = 1, the CHnF/TOF bit can be cleared either by DMA transfer done or writing a one to CHnF/TOF bit (see the following table).

Table 279. Clear CHnF/TOF Bit

DMA	How CHnF/TOF Bit Can Be Cleared
0	CHnF/TOF bit is cleared by writing a 1 to CHnF/TOF bit.
1	CHnF/TOF bit is cleared either when the DMA transfer is done or by writing a 1 to CHnF/TOF bit.

47.3.15 Peripheral Triggers

The connection of the TPM peripheral triggers with other peripherals are device specific.

Output Triggers

The TPM generates output triggers for the counter and each channel that can be used to trigger events in other peripherals. The counter trigger asserts whenever the TOF is set and remains asserted until the next increment.

Each TPM channel generates both a pre-trigger output and a trigger output. The pre-trigger output asserts whenever the CHnF is set, the trigger output asserts on the first counter increment after the pre-trigger asserts, and then both the trigger and pre-trigger negate on the first counter increment after the trigger asserts.

When (COMBINEn = 1) in output compare modes, the pre-trigger output for both channel (n) and channel (n+1) will assert when CH(n)F is set and will negate when CH(n+1)F is set. The trigger continues to assert on the first counter increment after the pre-trigger asserts and negates at the same time as the pre-trigger negation.

Input Trigger

The TPM input trigger can be configured to perform the following operations with configurable polarity:

- Increment the counter on trigger rising edge
- Start incrementing the counter on trigger rising edge
- Reset/reload the counter on trigger rising edge
- Pause the counter while trigger high

The TPM input trigger is synchronized and must assert for at least two counter clock cycles so that the input trigger can be detected.

47.3.16 Reset Overview

The TPM is reset whenever any chip reset occurs.

When the TPM exits from reset:

- the TPM counter and the prescaler counter are zero and are stopped (CMOD[1:0] = 0:0);
- the timer overflow interrupt is zero;
- the channels interrupts are zero;
- the channels are in input capture mode;
- the channels outputs are zero;
- the channels pins are not controlled by TPM (ELS(n)B:ELS(n)A = 0:0).

47.3.17 TPM Interrupts

This section describes TPM interrupts.

47.3.17.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

47.3.17.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

47.3.18 Low power modes

During a chip low-power mode, TPM can be configured to operate normally or to pause all counting.

Chip mode	TPM operation
Low power mode	• When CONF[DOZEEN] = 0 and TPM uses an external or internal clock source that remains functional, counting continues. TPM can generate an asynchronous interrupt to exit the chip from the low power mode. • When CONF[DOZEEN] = 1, counting pauses for the duration of the low power mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their initial state.

47.3.19 Debug mode

During chip debug mode, TPM can be configured to operate normally or to temporarily pause all counting.

Chip mode	TPM operation
Debug (core is in Debug/Halted mode)	• When CONF[DBGMODE] = 0b00, counting pauses until the core returns to normal user operating mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their initial state. • When CONF[DBGMODE] = 0b11, counting continues.

47.4 External signals

Table 280 shows the user-accessible signals for the TPM.

Table 280. TPM signal descriptions

Signal	Description	Direction
EXTCLK	External Clock If selected in SC[CMOD], EXTCLK increments the TPM counter on every rising edge synchronized to the TPM counter clock. EXTCLK must be less than half of the TPM counter clock frequency. The TPM counter prescaler selection and settings are also used when EXTCLK is selected.	I
CH*n*	Channel *n* I/O Pin (where *n* = 5 to 0) Each CH*n* can be configured to operate either as input or output. CH*n* is an output when configured in an output compare or PWM mode and the TPM counter is enabled; otherwise, the CH*n* is an input	I/O

47.5 Application information

An example of using the TPM with the asynchronous DMA is described in AN4631:Using the Asynchronous DMA features of the Kinetis L Series.

47.6 Memory Map and Register Definition

This section provides a detailed description of all TPM registers.

Attempting to access a reserved register location in the TPM memory map will generate a bus error.

47.6.1 TPM register descriptions

47.6.1.1 TPM memory map

TPM0 base address: 4003_1000h

TPM1 base address: 4003_2000h

TPM2 base address: 4898_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	See section
4h	Parameter (PARAM)	32	R	See section
8h	TPM Global (GLOBAL)	32	RW	0000_0000h
10h	Status and Control (SC)	32	RW	0000_0000h
14h	Counter (CNT)	32	RW	0000_0000h
18h	Modulo (MOD)	32	RW	0000_FFFFh
1Ch	Capture and Compare Status (STATUS)	32	RW	0000_0000h
20h	Channel (n) Status and Control (C0SC)	32	RW	0000_0000h
24h	Channel (n) Value (C0V)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
28h	Channel (n) Status and Control (C1SC)	32	RW	0000_0000h
2Ch	Channel (n) Value (C1V)	32	RW	0000_0000h
30h	Channel (n) Status and Control (C2SC)	32	RW	0000_0000h
34h	Channel (n) Value (C2V)	32	RW	0000_0000h
38h	Channel (n) Status and Control (C3SC)	32	RW	0000_0000h
3Ch	Channel (n) Value (C3V)	32	RW	0000_0000h
40h	Channel (n) Status and Control (C4SC)	32	RW	0000_0000h
44h	Channel (n) Value (C4V)	32	RW	0000_0000h
48h	Channel (n) Status and Control (C5SC)	32	RW	0000_0000h
4Ch	Channel (n) Value (C5V)	32	RW	0000_0000h
64h	Combine Channel Register (COMBINE)	32	RW	0000_0000h
6Ch	Channel Trigger (TRIG)	32	RW	0000_0000h
70h	Channel Polarity (POL)	32	RW	0000_0000h
78h	Filter Control (FILTER)	32	RW	0000_0000h
80h	Quadrature Decoder Control and Status (QDCTRL)	32	RW	0000_0000h
84h	Configuration (CONF)	32	RW	0000_0000h

47.6.1.2 Version ID (VERID)

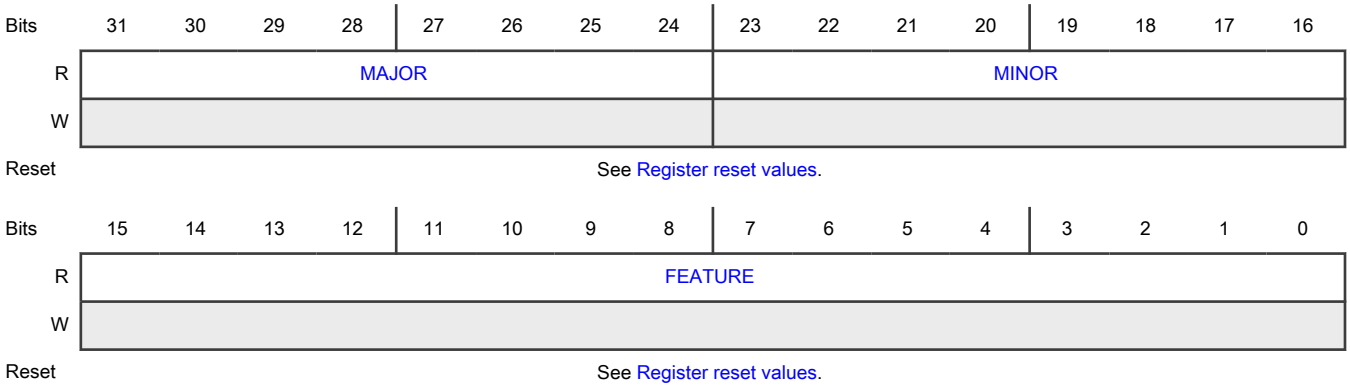
Offset

Register	Offset
VERID	0h

Function

Contains the module version ID and feature information.

Diagram



Register reset values

Register	Reset value
VERID	TPM0,TPM1: 0600_0007h TPM2: 0600_0003h

Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number 0000_0000_0000_0001b - Standard feature set. 0000_0000_0000_0011b - Standard feature set with Filter and Combine registers implemented. 0000_0000_0000_0101b - Standard feature set with Quadrature registers implemented. 0000_0000_0000_0111b - Standard feature set with Filter, Combine and Quadrature registers implemented.

47.6.1.3 Parameter (PARAM)

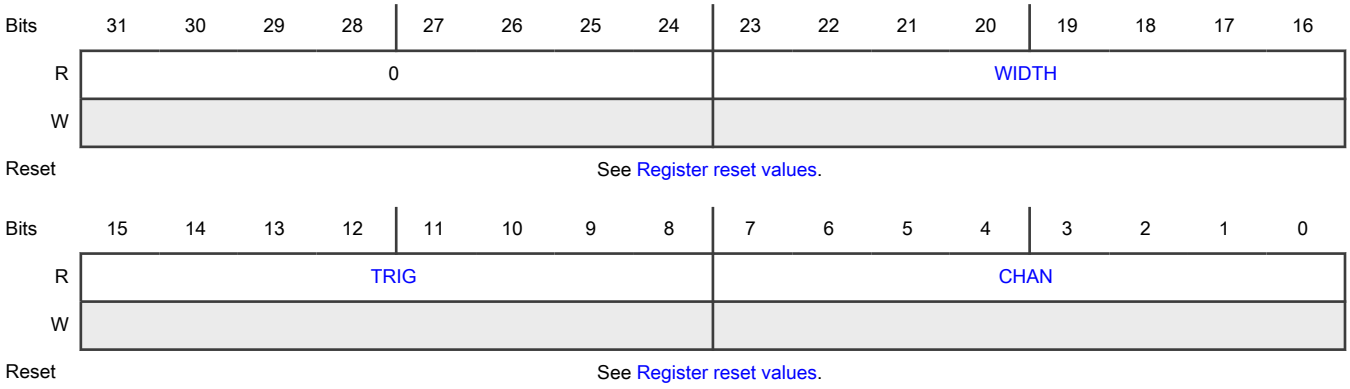
Offset

Register	Offset
PARAM	4h

Function

Contains values of module parameters.

Diagram



Register reset values

Register	Reset value
PARAM	TPM0,TPM1: 0020_0406h TPM2: 0020_0402h

Fields

Field	Function
31-24 —	Reserved
23-16 WIDTH	Counter Width Width of the counter and timer channels.
15-8 TRIG	Trigger Count Number of trigger inputs implemented.
7-0 CHAN	Channel Count Number of timer channels implemented.

47.6.1.4 TPM Global (GLOBAL)

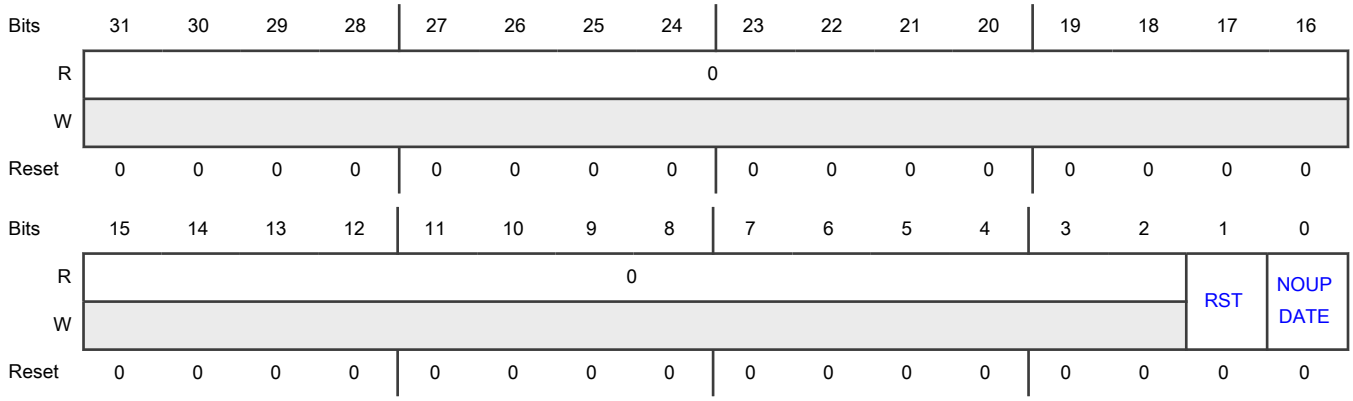
Offset

Register	Offset
GLOBAL	8h

Function

Includes the software reset control bit.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Reset all internal logic and registers, except the Global Register. Remains set until cleared by software. 0b - Module is not reset. 1b - Module is reset.
0 NOUPDATE	No Update Blocks updates to all internal double buffered registers while it remains set. Software can still update the software visible registers, but changes cannot take effect until this bit is cleared at which point they update as normal. 0b - Internal double buffered registers update as normal. 1b - Internal double buffered registers do not update.

47.6.1.5 Status and Control (SC)

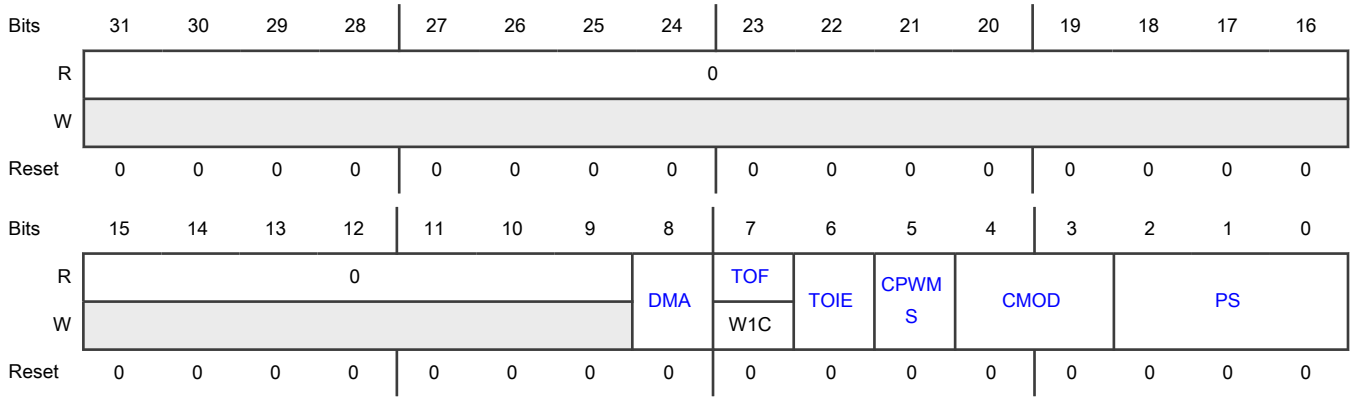
Offset

Register	Offset
SC	10h

Function

Contains the overflow status flag and control bits used to configure the interrupt enable, module configuration and prescaler factor. These controls relate to all channels within this module.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 DMA	DMA Enable Enables DMA transfers for the overflow flag. 0b - Disables DMA transfers. 1b - Enables DMA transfers.
7 TOF	Timer Overflow Flag Set by hardware when the TPM counter equals the value in the MOD register and increments. Writing a 1 to TOF clears it. Writing a 0 to TOF has no effect. If another TPM overflow occurs between the flag setting and the flag clearing, the write operation has no effect; therefore, TOF remains set indicating another overflow has occurred. In this case a TOF interrupt request is not lost due to a delay in clearing the previous TOF. 0b - TPM counter has not overflowed. 1b - TPM counter has overflowed.
6 TOIE	Timer Overflow Interrupt Enable Enables TPM overflow interrupts. 0b - Disable TOF interrupts. Use software polling or DMA request. 1b - Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the TPM to operate in up-down counting mode. This field is write protected. It can be written only when the counter is disabled. 0b - TPM counter operates in up counting mode. 1b - TPM counter operates in up-down counting mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-3 CMOD	Clock Mode Selection Selects the TPM counter clock modes. When disabling the counter, this field remains set until acknolwedged in the TPM clock domain. 00b - TPM counter is disabled 01b - TPM counter increments on every TPM counter clock 10b - TPM counter increments on rising edge of EXTCLK synchronized to the TPM counter clock 11b - TPM counter increments on rising edge of the selected external input trigger.
2-0 PS	Prescale Factor Selection Selects one of 8 division factors for the clock mode selected by CMOD. This field is write protected. It can be written only when the counter is disabled. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

47.6.1.6 Counter (CNT)

Offset

Register	Offset
CNT	14h

Function

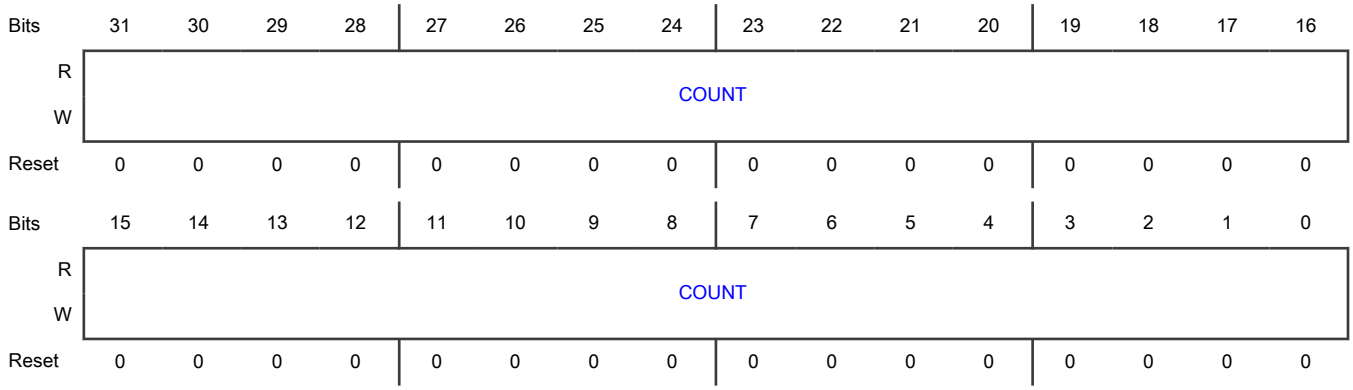
Contains the TPM counter value.

Reset clears the CNT register. Writing any value to COUNT triggers a reload of the counter, forcing PWM outputs to their reload state.

When debug is active, the TPM counter does not increment unless configured otherwise.

Reading the CNT register adds two wait states to the register access due to synchronization delays.

Diagram



Fields

Field	Function
31-0 COUNT	Counter value

47.6.1.7 Modulo (MOD)

Offset

Register	Offset
MOD	18h

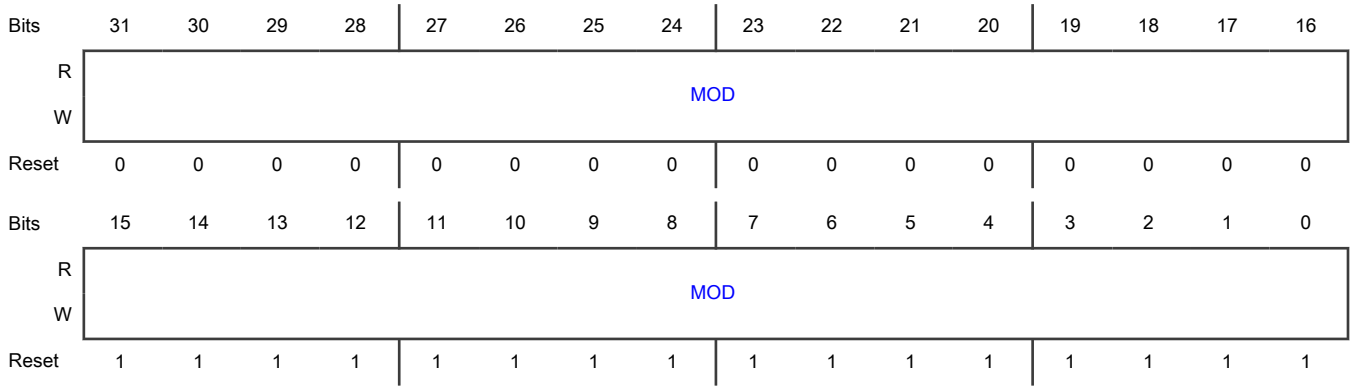
Function

Contains the modulo value for the TPM counter. When the TPM counter reaches the modulo value and increments, the overflow flag (TOF) is set and the next value of TPM counter depends on the selected counting method (see Counter).

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to MOD Register Update. Additional writes to the MOD write buffer are ignored until the register has been updated.

Initialize the TPM counter (write to CNT) before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Diagram



Fields

Field	Function
31-0 MOD	Modulo value This field must be written with single 16-bit or 32-bit access.

47.6.1.8 Capture and Compare Status (STATUS)

Offset

Register	Offset
STATUS	1Ch

Function

Contains a copy of the status flag, CnSC[CHnF] for each TPM channel, as well as SC[TOF], for software convenience. Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by writing all ones to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. Writing a 1 to CHF clears it. Writing a 0 to CHF has no effect.

If another event occurs between the flag setting and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							TOF	0		CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W								W1C			W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 TOF	Timer Overflow Flag See register description 0b - TPM counter has not overflowed. 1b - TPM counter has overflowed.
7-6 —	Reserved
5 CH5F	Channel 5 Flag See the register description. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
TPM0	STATUS	—
TPM1	STATUS	—
TPM2	—	STATUS

0b - No channel event has occurred.

1b - A channel event has occurred.

4	Channel 4 Flag See the register description.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>CH4F</td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>STATUS</td><td>—</td></tr>
<tr><td>TPM1</td><td>STATUS</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>STATUS</td></tr>
</table>
0b - No channel event has occurred.
1b - A channel event has occurred.
</td></tr>
<tr><td>3
CH3F</td><td>
Channel 3 Flag
See the register description.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>STATUS</td><td>—</td></tr>
<tr><td>TPM1</td><td>STATUS</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>STATUS</td></tr>
</table>
0b - No channel event has occurred.
1b - A channel event has occurred.
</td></tr>
<tr><td>2
CH2F</td><td>
Channel 2 Flag
See the register description.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>STATUS</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM1</td><td>STATUS</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>STATUS</td></tr>
</table>
0b - No channel event has occurred.
1b - A channel event has occurred.</td></tr>
<tr><td>1
CH1F</td><td>Channel 1 Flag
See the register description.
0b - No channel event has occurred.
1b - A channel event has occurred.</td></tr>
<tr><td>0
CH0F</td><td>Channel 0 Flag
See the register description.
0b - No channel event has occurred.
1b - A channel event has occurred.</td></tr>
</table>

47.6.1.9 Channel (n) Status and Control (C0SC - C5SC)

Offset

Register	Offset
C0SC	20h
C1SC	28h
C2SC	30h
C3SC	38h
C4SC	40h
C5SC	48h

Function

Contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

Table 281. Mode, Edge, and Level Selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
0	00	00	None	Channel disabled
0	01	00	Software compare	Pin not used for TPM
0	00	01	Input capture	Capture on Rising Edge Only
		10		Capture on Falling Edge Only
		11		Capture on Rising or Falling Edge
	01	01	Output compare	Toggle Output on match
		10		Clear Output on match
		11		Set Output on match
	10	10	Edge-aligned PWM	High-true pulses (clear Output on counter match, set Output on counter reload, set Output when counter first enabled or paused)
		00		High-true pulses (clear Output on counter match, set Output on counter reload, clear Output when counter first enabled or paused)
		01		Low-true pulses (set Output on counter match, clear Output on counter reload, clear Output when counter first enabled or paused)
		11		Low-true pulses (set Output on counter match, clear Output on counter reload, set Output when counter first enabled or paused)
	11	10	Output compare	Pulse Output low on match
		01		Pulse Output high on match
1	10	10	Center-aligned PWM	High-true pulses (clear Output on counter match-up, set Output on counter match-down, set Output on counter reload or when counter first enabled or paused)
		00		High-true pulses (clear Output on counter match-up, set Output on counter match-down, clear Output on counter reload

Table continues on the next page...

Table 281. Mode, Edge, and Level Selection (continued)

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
				or when counter first enabled or paused))
		01		Low-true pulses (set Output on match-up, clear Output on match-down, clear Output on counter reload or when counter first enabled or paused)
		11		Low-true pulses (set Output on match-up, clear Output on match-down, set Output on counter reload or when counter first enabled or paused))

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
TPM0	C0SC–C5SC	—
TPM1	C0SC–C5SC	—
TPM2	C0SC–C1SC	C2SC–C5SC

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CHF	CHIE	MSB	MSA	ELSB	ELSA	0	DMA
W									W1C							
Reset	See Register reset values.															

Register reset values

Register	Reset value
C0SC–C1SC	TPM0–TPM2: 0000_0000h
C2SC–C5SC	TPM0,TPM1: 0000_0000h TPM2: Register not supported

Fields

Field	Function
31-8 —	Reserved
7 CHF	Channel Flag Set by hardware when an event occurs on the channel. CHF is cleared by writing a 1 to the CHF bit. Writing a 0 to CHF has no effect. If another event occurs between the CHF sets and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the delay in clearing the previous CHF. 0b - No channel event has occurred. 1b - A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0b - Disable channel interrupts. 1b - Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode.
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode.
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode.
1 —	Reserved
0 DMA	DMA Enable Enables DMA transfers for the channel. 0b - Disable DMA transfers. 1b - Enable DMA transfers.

47.6.1.10 Channel (n) Value (C0V - C5V)

Offset

Register	Offset
C0V	24h
C1V	2Ch
C2V	34h
C3V	3Ch
C4V	44h
C5V	4Ch

Function

Contain the captured TPM counter value for the input modes or the match value for the output modes.

In input capture mode, any write to a CnV register is ignored.

In compare modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to CnV Register Update . Additional writes to the CnV write buffer are ignored until the register has been updated.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
TPM0	C0V–C5V	—
TPM1	C0V–C5V	—
TPM2	C0V–C1V	C2V–C5V

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VAL															
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VAL															
Reset	See Register reset values.															

Register reset values

Register	Reset value
C0V–C1V	TPM0–TPM2: 0000_0000h
C2V–C5V	TPM0,TPM1: 0000_0000h TPM2: Register not supported

Fields

Field	Function
31-0 VAL	Channel Value Captured TPM counter value of the input modes or the match value for the output modes. This field must be written with single 16-bit or 32-bit access.

47.6.1.11 Combine Channel Register (COMBINE)

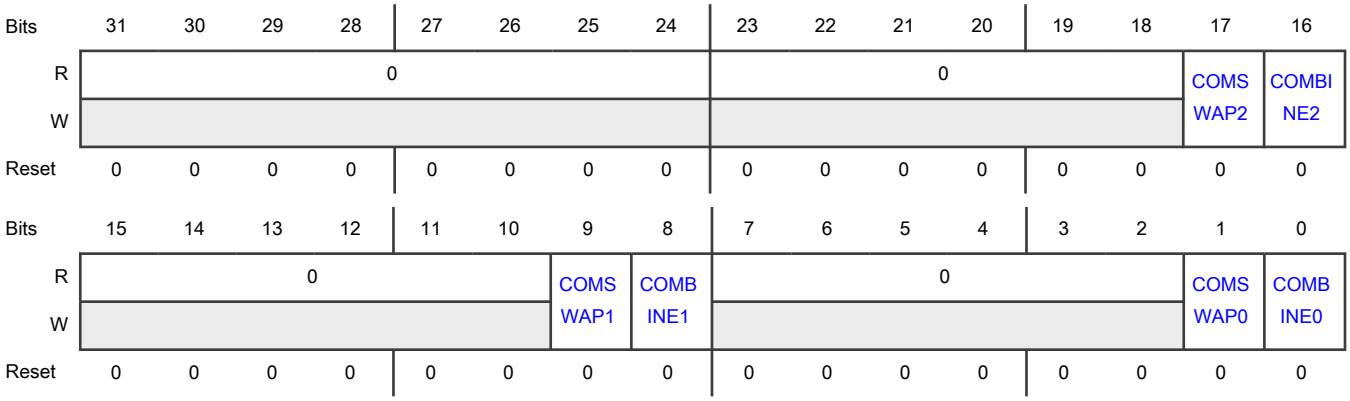
Offset

Register	Offset
COMBINE	64h

Function

Contains the control bits used to configure the combine channel modes for each pair of channels (n) and (n+1), where n is all the even numbered channels.

Diagram



Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>23-18
—</td><td>Reserved</td></tr>
<tr><td>17
COMSWAP2</td><td>Combine Channels 4 and 5 Swap
When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM1</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM2</td><td>—</td><td>COMBINE</td></tr></table>0b - Even channel is used for input capture and 1st compare.
1b - Odd channel is used for input capture and 1st compare.</td></tr>
<tr><td>16
COMBINE2</td><td>Combine Channels 4 and 5
Enables the combine feature for channels 2 and 3. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM1</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM2</td><td>—</td><td>COMBINE</td></tr></table>0b - Channels 4 and 5 are independent.
1b - Channels 4 and 5 are combined.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>15-10
—</td><td>Reserved</td></tr>
<tr><td>9
COMSWAP1</td><td>Combine Channels 2 and 3 Swap
When set in combine mode, the odd channel is used for the input capture and 1st compare, the even channel is used for the 2nd compare.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM1</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM2</td><td>—</td><td>COMBINE</td></tr></table>0b - Even channel is used for input capture and 1st compare.
1b - Odd channel is used for input capture and 1st compare.</td></tr>
<tr><td>8
COMBINE1</td><td>Combine Channels 2 and 3
Enables the combine feature for channels 2 and 3. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM1</td><td>COMBINE</td><td>—</td></tr><tr><td>TPM2</td><td>—</td><td>COMBINE</td></tr></table>0b - Channels 2 and 3 are independent.
1b - Channels 2 and 3 are combined.</td></tr>
<tr><td>7-2</td><td>Reserved</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 COMSWAP0	Combine Channel 0 and 1 Swap When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare. 0b - Even channel is used for input capture and 1st compare. 1b - Odd channel is used for input capture and 1st compare.
0 COMBINE0	Combine Channels 0 and 1 Enables the combine feature for channels 0 and 1. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0b - Channels 0 and 1 are independent. 1b - Channels 0 and 1 are combined.

47.6.1.12 Channel Trigger (TRIG)

Offset

Register	Offset
TRIG	6Ch

Function

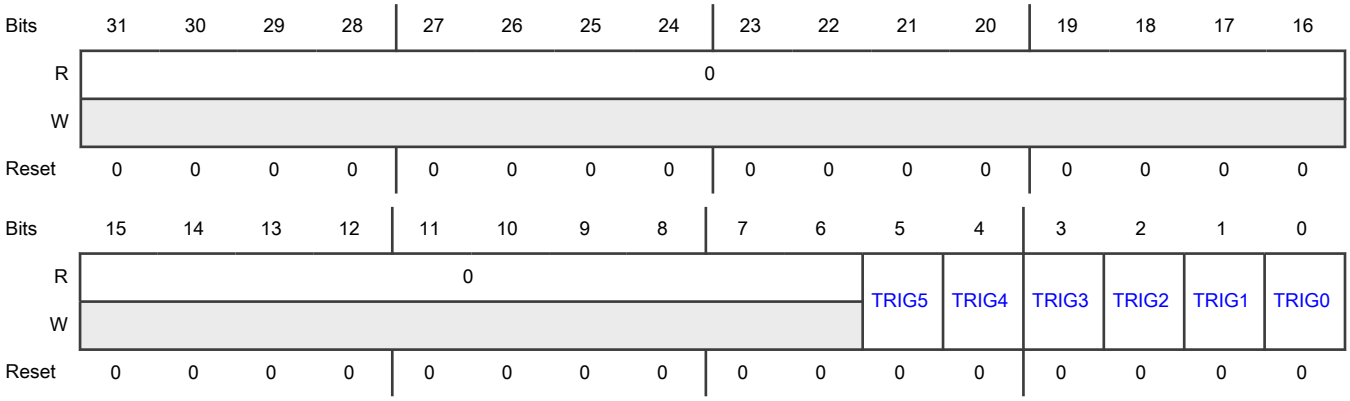
In input capture mode, configures the trigger input that is used by the channel to capture the counter value. In output compare or PWM mode, configures the trigger input used to modulate the channel output. When modulating the output, the output is forced to the channel initial value whenever the trigger is not asserted. Note that the even numbered channels share the first input trigger source and the odd numbered channels share the second input trigger source.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
TPM0	TRIG	—
TPM1	TRIG	—
TPM2	—	TRIG

Diagram



Fields

Field	Function
31-6 —	Reserved
5 TRIG5	Channel 5 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 5.
4 TRIG4	Channel 4 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 4.
3 TRIG3	Channel 3 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 3.
2 TRIG2	Channel 2 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 2.
1 TRIG1	Channel 1 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 1.
0 TRIG0	Channel 0 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 0.

47.6.1.13 Channel Polarity (POL)

Offset

Register	Offset
POL	70h

Function

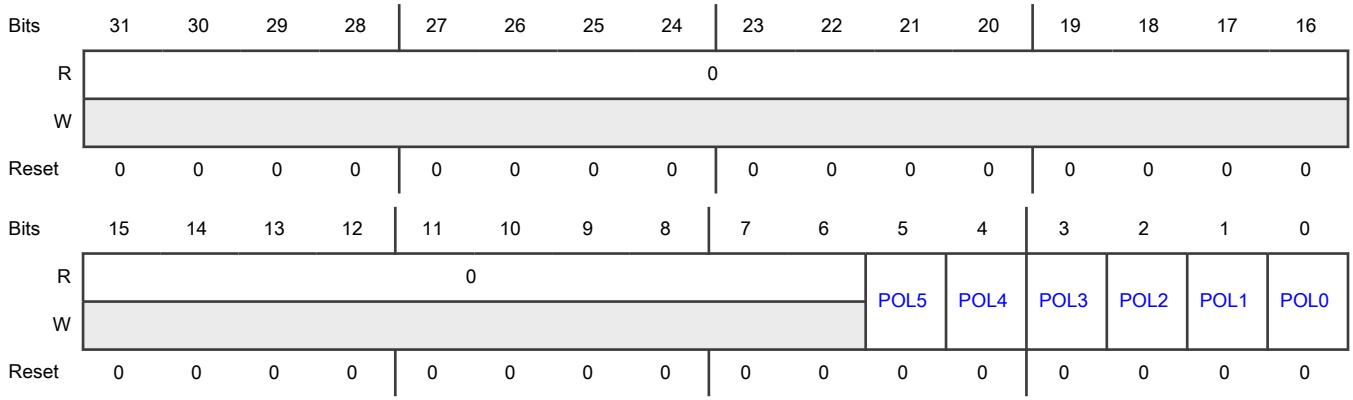
Defines the input and output polarity of each of the channels.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
TPM0	POL	—
TPM1	POL	—
TPM2	—	POL

Diagram



Fields

Field	Function
31-6 —	Reserved
5 POL5	Channel 5 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
4	Channel 4 Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
POL4	0b - The channel polarity is active high 1b - The channel polarity is active low.
3 POL3	Channel 3 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
2 POL2	Channel 2 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
1 POL1	Channel 1 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
0 POL0	Channel 0 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.

47.6.1.14 Filter Control (FILTER)

Offset

Register	Offset
FILTER	78h

Function

Selects the filter value of the channel inputs, and an additional output delay value for the channel outputs. In PWM combine modes, the filter can be used to implement deadtime insertion.

NOTE

To write values to the FILTER[CHnFVAL] bits, ensure that the corresponding channel-interrupt-status flag and control (CnSC[5:2]) bits are set to zero.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								CH5FVAL				CH4FVAL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH3FVAL				CH2FVAL				CH1FVAL				CH0FVAL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 CH5FVAL	Channel 5 Filter Value Sets the filter value for the channel 5 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH5FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 5 in PWM modes. Deadtime insertion is disabled when CH5FVAL is zero, otherwise deadtime insertion for channel 5 is configured as (CH5FVAL * 4) clock cycles. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
19-16 CH4FVAL	Channel 4 Filter Value Sets the filter value for the channel 4 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH4FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 4 in PWM modes. Deadtime insertion is disabled when CH4FVAL is zero, otherwise deadtime insertion for channel 4 is configured as (CH4FVAL * 4) clock cycles. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
TPM0	FILTER	—
TPM1	FILTER	—
TPM2	—	FILTER

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>FILTER</td><td>—</td></tr><tr><td>TPM1</td><td>FILTER</td><td>—</td></tr><tr><td>TPM2</td><td>—</td><td>FILTER</td></tr></table>
15-12 CH3FVAL	Channel 3 Filter Value Sets the filter value for the channel 3 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH3FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 3 in PWM modes. Deadtime insertion is disabled when CH3FVAL is zero, otherwise deadtime insertion for channel 3 is configured as (CH3FVAL * 4) clock cycles. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>FILTER</td><td>—</td></tr><tr><td>TPM1</td><td>FILTER</td><td>—</td></tr><tr><td>TPM2</td><td>—</td><td>FILTER</td></tr></table>
11-8 CH2FVAL	Channel 2 Filter Value Sets the filter value for the channel 2 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH2FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 2 in PWM modes. Deadtime insertion is disabled when CH2FVAL is zero, otherwise deadtime insertion for channel 2 is configured as (CH2FVAL * 4) clock cycles. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM0</td><td>FILTER</td><td>—</td></tr><tr><td>TPM1</td><td>FILTER</td><td>—</td></tr></table>

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>TPM2</td><td>—</td><td>FILTER</td></tr></table>
7-4 CH1FVAL	Channel 1 Filter Value Sets the filter value for the channel 1 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH1FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 1 in PWM modes. Deadtime insertion is disabled when CH1FVAL is zero, otherwise deadtime insertion for channel 1 is configured as (CH1FVAL * 4) clock cycles.
3-0 CH0FVAL	Channel 0 Filter Value Sets the filter value for the channel 0 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH0FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 0 in PWM modes. Deadtime insertion is disabled when CH0FVAL is zero, otherwise deadtime insertion for channel 0 is configured as (CH0FVAL * 4) clock cycles.

47.6.1.15 Quadrature Decoder Control and Status (QDCTRL)

Offset

Register	Offset
QDCTRL	80h

Function

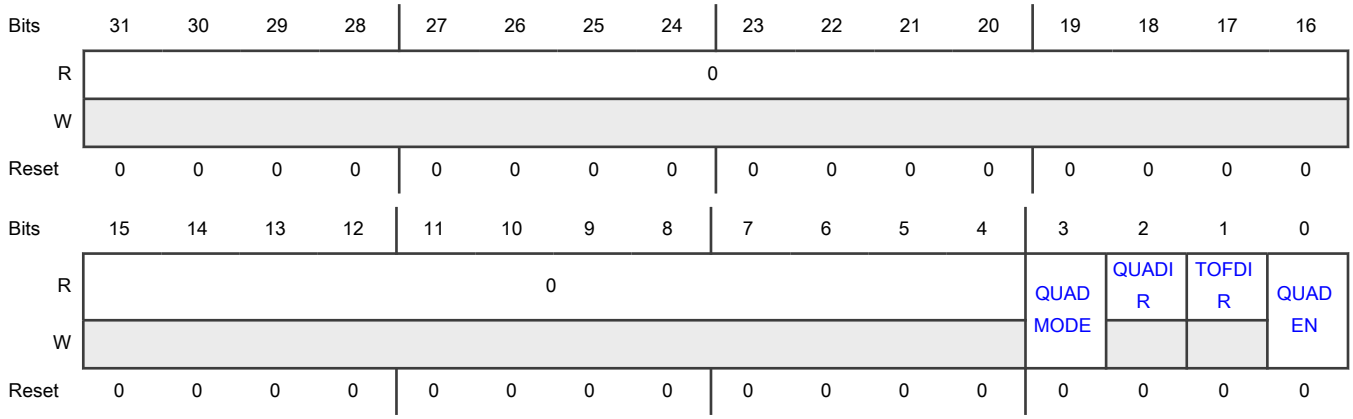
Contains the control and status bits for the quadrature decoder mode.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
TPM0	QDCTRL	—
TPM1	QDCTRL	—
TPM2	—	QDCTRL

Diagram



Fields

Field	Function
31-4 —	Reserved
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the quadrature decoder mode. 0b - Phase encoding mode. 1b - Count and direction encoding mode.
2 QUADIR	Counter Direction in Quadrature Decode Mode Indicates the counting direction. 0b - Counter direction is decreasing (counter decrement). 1b - Counter direction is increasing (counter increment).
1 TOFDIR	TOFDIR Indicates if the TOF bit was set on the top or the bottom of counting. 0b - TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (zero) to its maximum value (MOD register). 1b - TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (zero).
0 QUADEN	QUADEN Enables the quadrature decoder mode. In this mode, the channel 0 and channel 1 inputs control the TPM counter direction and can only be used for software compare. The quadrature decoder mode has precedence over the other modes. 0b - Quadrature decoder mode is disabled. 1b - Quadrature decoder mode is enabled.

47.6.1.16 Configuration (CONF)

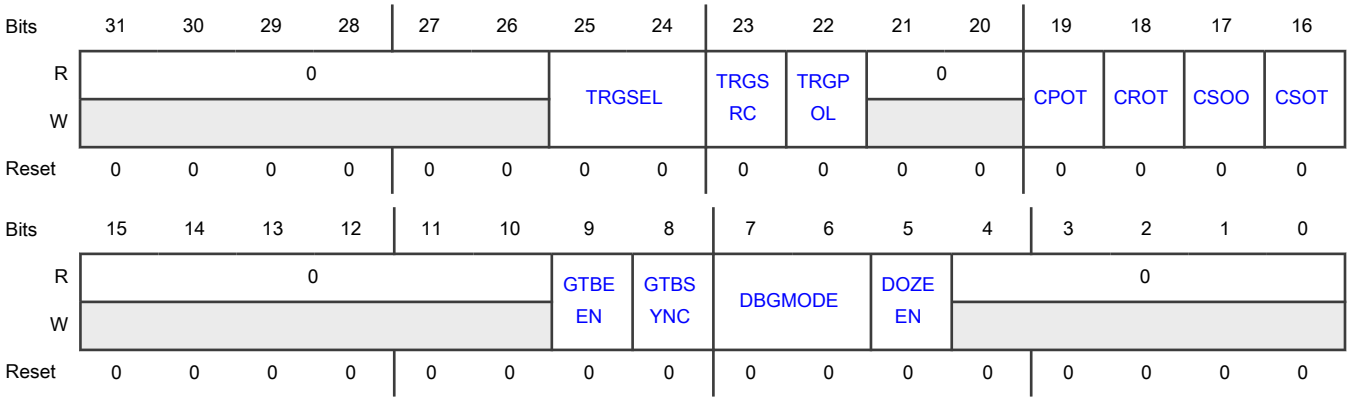
Offset

Register	Offset
CONF	84h

Function

Selects the behavior in debug and low power modes and the use of an external global time base.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 TRGSEL	Trigger Select Selects the input trigger to use for starting, reloading and/or pausing the counter. The source of the trigger (external or internal to the TPM) is configured by the TRGSRC field. This field should only be changed when the TPM counter is disabled. See the chip-specific TPM information for available external trigger options, if available. The available internal trigger sources are listed below. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
TPM0	CONF	—
TPM1	CONF	—

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
01b - Channel 0 pin input capture
10b - Channel 1 pin input capture
11b - Channel 0 or Channel 1 pin input capture
</td></tr>
<tr><td>23
TRGSRC</td><td>
Trigger Source
Selects between internal (channel pin input capture) or external trigger sources.
When selecting an internal trigger, the channel selected should be configured for input capture. Only a rising edge input capture can be used to initially start the counter using the CSOT configuration; either rising edge or falling edge input capture can be used to reload the counter using the CROT configuration; and the state of the channel input pin is used to pause the counter using the CPOT configuration. The channel polarity register can be used to invert the polarity of the channel input pins.
This field should only be changed when the TPM counter is disabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
0b - Trigger source selected by TRGSEL is external.
1b - Trigger source selected by TRGSEL is internal (channel pin input capture).
</td></tr>
<tr><td>22
TRGPOL</td><td>
Trigger Polarity
Selects the polarity of the external trigger source. This field should only be changed when the TPM counter is disabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
0b - Trigger is active high.
1b - Trigger is active low.
</td></tr>
<tr><td>21-20
—</td><td>Reserved</td></tr>
<tr><td>19
CPOT</td><td>Counter Pause On Trigger
When enabled, the counter will pause incrementing while the trigger remains asserted (level sensitive). While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state.
The trigger input is ignored if the TPM counter is paused during debug mode or a low power mode. This field should only be changed when the TPM counter is disabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
0b - TPM counter continues
1b - TPM counter pauses
</td></tr>
<tr><td>18
CROT</td><td>Counter Reload On Trigger
When set, the TPM counter will reload with zero (and set PWM outputs to their reload state) when a rising edge is detected on the selected trigger input.
The trigger input is ignored if the TPM counter is paused during debug mode or a low power mode. This field should only be changed when the TPM counter is disabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
0b - Counter is not reloaded due to a rising edge on the selected input trigger
1b - Counter is reloaded when a rising edge is detected on the selected input trigger
</td></tr>
<tr><td>17
CSOO</td><td>
Counter Stop On Overflow
When set, the TPM counter will pause incrementing once the counter equals the MOD value and increments (this also sets the TOF). Reloading the counter with 0 due to writing to the counter register or due to a trigger input does not cause the counter to stop incrementing. Once the counter has stopped incrementing, the counter will not start incrementing unless it is disabled and then enabled again, or a rising edge on the selected trigger input is detected when CSOT is set. While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state.
This field should only be changed when the TPM counter is disabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
0b - TPM counter continues incrementing or decrementing after overflow
1b - TPM counter stops incrementing or decrementing after overflow.
</td></tr>
<tr><td>16
CSOT</td><td>
Counter Start on Trigger
When set, the TPM counter will pause incrementing when it is enabled until a rising edge on the selected trigger input is detected. If the TPM counter is paused due to an overflow, a rising edge on the selected trigger input will also cause the TPM counter to start incrementing again. While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state.
The trigger input is ignored if the TPM counter is paused during debug mode or a low power mode. This field should only be changed when the TPM counter is disabled.
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
0b - TPM counter starts to increment immediately, once it is enabled.
1b - TPM counter only starts to increment when it a rising edge on the selected input trigger is detected, after it has been enabled or after it has stopped due to overflow.
</td></tr>
<tr><td>15-10
—</td><td>Reserved</td></tr>
<tr><td>9
GTBEEN</td><td>
Global time base enable
Configures the TPM to use an externally generated global time base counter. When an externally generated timebase is used, the internal TPM counter is not used by the channels but can be used to generate a periodic interruptor DMA request using the Modulo register and timer overflow flag.
0b - All channels use the internally generated TPM counter as their timebase
1b - All channels use an externally generated global timebase as their timebase
</td></tr>
<tr><td>8
GTBSYNC</td><td>
Global Time Base Synchronization
When enabled, the TPM counter is synchronized to the global time base. It uses the global timebase enable, trigger, paused state and overflow to ensure the TPM counter starts incrementing at the same time as the global timebase, stops incrementing at the same time as the global timebase and is reset at the same time as the global timebase. This field should only be changed when the TPM counter is disabled.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>TPM0</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM1</td><td>CONF</td><td>—</td></tr>
<tr><td>TPM2</td><td>—</td><td>CONF</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr></table> 0b - Global timebase synchronization disabled. 1b - Global timebase synchronization enabled.
7-6 DBGMODE	Debug Mode Configures the TPM behavior in debug mode. All other configurations are reserved. 00b - TPM counter is paused and does not increment. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their default state. 11b - TPM counter continues.
5 DOZEEN	Doze Enable Configures the TPM behavior in a low power mode. 0b - Internal TPM counter continues. 1b - Internal TPM counter is paused and does not increment. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their default state.
4-0 —	Reserved

Chapter 48
Low Power Inter-Integrated Circuit (LPI2C)

48.1 Chip-specific LPI2C information

Table 282. Reference links to related information

Topic	Related module	Reference
Full description	LPI2C	LPI2C
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

48.1.1 Module instances

This device has two instances of the LPI2C module, LPI2C0, and LPI2C1.

48.1.2 Host request configuration

Both of the LPI2C0 and LPI2C1 support host request feature.

The input trigger that can be used as host request input is from TRGMUX, refer to Trigger multiplexer (TRGMUX) for more details.

48.2 Overview

LPI2C is a low-power Inter-Integrated Circuit (I2C) module that supports an efficient interface to an I2C bus as a controller and/or as a target.

- Implements logic support for Standard, Fast, Fast+ and Ultra Fast modes of operation.
- Uses little CPU overhead, with DMA offloading of FIFO register accesses.
- Continues operating in Sleep modes if an appropriate clock is available.

The LPI2C module also complies with the System Management Bus (SMBus) Specification, version 3. The SMBus is a single-ended simple two-wire bus, which is typically used for low-bandwidth communications.

NOTE

The I^2C (Inter-Integrated Circuit) serial bus is multi-controller, multi-target, packet-switched, and single-ended, and is often used to attach microcontroller ICs to lower-speed peripheral ICs.

NOTE

Terminology in this chapter has been updated to align with I^2C-bus specification, Rev. 7.0.

Table 283. Updated terms

Updated term	Deprecated term
Controller	Master
Target	Slave

48.2.1 Block diagram

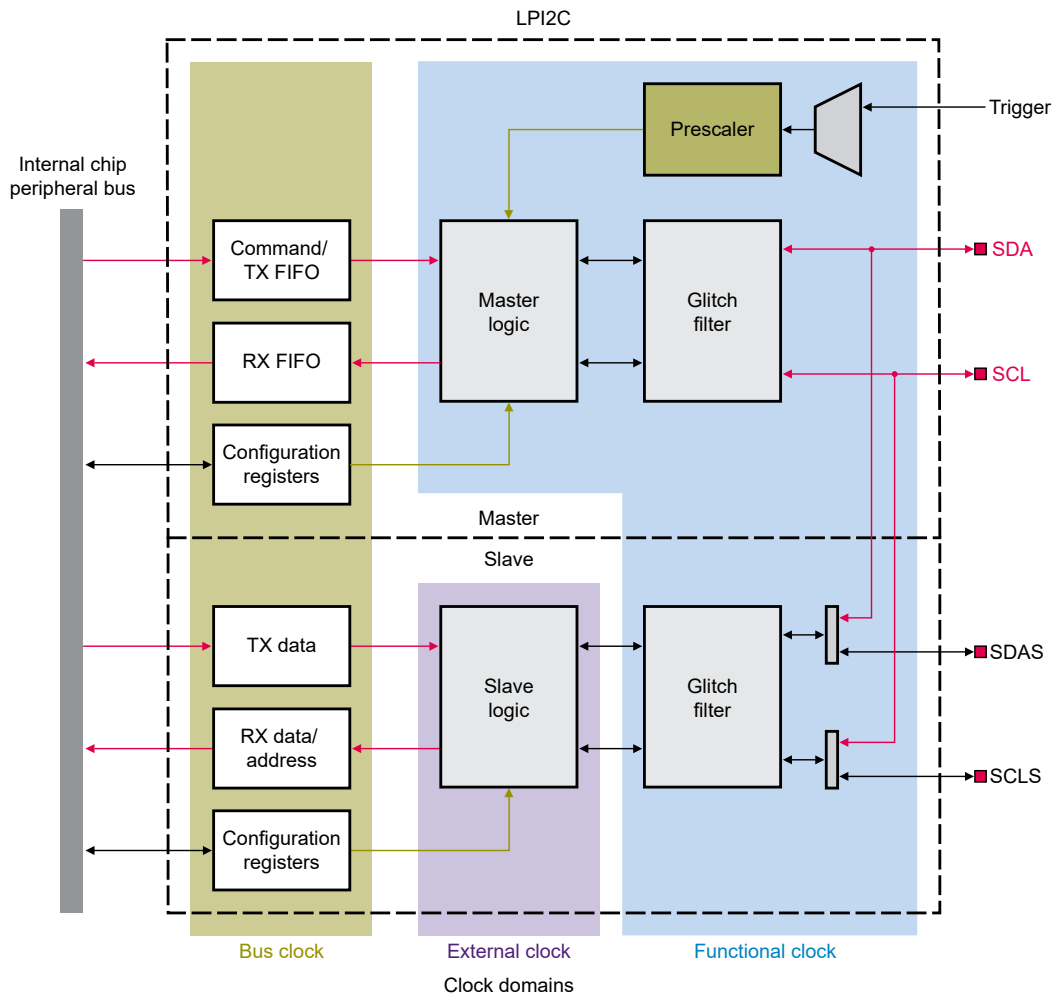


Figure 176. LPI2C block diagram

48.2.2 Features

The LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes
- High-speed mode (HS) in target mode
- High-speed mode (HS) in controller mode
- Multi-controller, including synchronization and arbitration. Multi-controller means that any number of controller nodes can be present. Additionally, controller and target roles may be changed between messages (after a STOP is sent).
- Clock stretching. Sometimes multiple I2C nodes may drive the lines at the same time. If any I2C node is driving a line low, then that line is low. I2C nodes that are starting to transmit a logical one (by letting the line float high) can detect that the line is low. In this way, the nodes can identify that another I2C node is active at the same time.
  - When node detection is used on the SCL line, it is called clock stretching. Clock stretching is used as an I2C flow control mechanism.
  - When node detection is used on the SDA line, it is called arbitration. Arbitration ensures that there is only one I2C node transmitter at a time.

- General call, seven-bit addressing, and ten-bit addressing
- Software reset, START byte, and Device ID (also require software support)

The LPI2C controller supports:

- Command/transmit FIFO of 4 words (8-bit transmit data + 3-bit command).
- Receive FIFO of 4 words (8-bit receive data).
- Command FIFO waits for idle I2C bus before initiating transfer.
- Command FIFO can initiate (repeated) START and STOP conditions and one or more controller-receiver transfers.
- Generating a STOP condition from command FIFO, or generating it automatically when the transmit FIFO is empty.
- Generating interrupt on data match and/or discard unwanted data, via flexible receive data match.
- Flags and optional interrupt signals at repeated START condition, STOP condition, loss of arbitration, unexpected NACK, and command word errors.
- Configurable bus idle timeout and pin-stuck-low timeout.

The LPI2C target supports:

- Separate I2C target registers to minimize software overhead because of controller/target switching.
- 7-bit or 10-bit addressing, address range, SMBus alert, and general call address.
- Transmit data register that supports interrupt or DMA requests.
- Receive data register that supports interrupt or DMA requests.
- Software-controllable ACK or NACK, with optional clock stretching on ACK/NACK bit.
- Configurable clock stretching, to avoid transmit-FIFO-underrun and receive-FIFO-overrun errors.
- Flags and optional interrupt at end of packet, STOP condition, or bit error detection.

48.3 Functional description

48.3.1 Controller mode

The LPI2C controller logic operates independently from the target logic to perform all controller-mode transfers on the I2C bus.

48.3.1.1 Transmit and Command FIFO commands

The transmit FIFO stores command data to initiate various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- START or repeated START condition with address byte, expecting ACK or NACK.
- Transmit data. This operation is the default for zero-extended-byte writes to the transmit FIFO.
- Receive 1-256 bytes of data. This operation can also be configured to discard received data and not to store it in the receive FIFO.
- STOP condition. This operation can also be configured to send a STOP condition when the transmit FIFO is empty.

Multiple transmit and receive commands can be inserted between the START condition and STOP condition. To comply with the I2C specification, transmit and receive commands must not be interleaved. The receive data command and the receive data and discard commands can be interleaved. This interleaving ensures that only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C controller automatically transmits a NACK on the last byte of a receive data command. It transmits the NACK unless the next command in the FIFO is also a receive data command. If the transmit FIFO is empty when a receive data command completes, a NACK is also automatically transmitted.

The LPI2C controller supports 10-bit addressing through:

1. A (repeated) START condition,
2. Followed by a transmit data byte containing the second address byte,
3. Followed by any number of data bytes with the controller transmit data.

A START or repeated START condition expecting a NACK (for example, high-speed mode controller code) must be followed by a STOP or (repeated) START condition.

48.3.1.2 Controller operations

Whenever LPI2C is enabled, it monitors the I2C bus to detect when the I2C is idle (MSR[BBF]). If either SCL or SDA are low, the I2C bus is no longer considered idle. The bus becomes idle if a STOP condition is detected or if a bus idle timeout is detected (as configured by MCFGR2[BUSIDLE]). After the bus is idle, if the transmit FIFO is not empty and the host request is asserted or disabled, the LPI2C controller initiates a transfer on the bus. This transfer involves the following steps:

1. Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler (MCFGR1[PRESCALE]).
2. Transmit a START condition and address byte using the timing configuration in Controller Clock Configuration 0 (MCCR0). If a high-speed mode transfer is configured, the timing configuration from Controller Clock Configuration 1 (MCCR1) is used instead.
3. Perform controller transmit or controller receive transfers, as configured by the transmit FIFO.
4. Transmit NACK on the last byte of a controller receive transfer. This action is done unless the next command in the transmit FIFO is also a receive data command and the transmit FIFO is not empty.
5. Transmit a Repeated START or STOP condition as configured by the transmit FIFO and/or MCFGR1[AUTOSTOP]. A repeated START can change which timing configuration register is used.

Whenever MCFGR0[RELAX] is 1, the LPI2C controller does not wait for the I2C bus to be idle before starting a transfer. It also relaxes some restrictions on the order of FIFO commands. For example, it allows a STOP command when idle to generate a START condition, followed by one SCL clock pulse and then a STOP condition.

When the LPI2C controller is disabled, LPI2C continues emptying the transmit FIFO until a STOP condition is transmitted. (The controller could be disabled due to MCR[MEN] being 0, or automatically due to mode entry.) However, LPI2C no longer stalls the I2C bus by waiting for the transmit or receive FIFO. After the transmit FIFO is empty, the LPI2C generates a STOP condition automatically.

The LPI2C controller can stall the I2C bus under certain conditions. This stalling results in SCL pulled low continuously on the first bit of a byte, until these conditions change:

- LPI2C controller is enabled and busy, the transmit FIFO is empty, and MCFGR1[AUTOSTOP] is 0. The LPI2C controller continues to stall the bus until the transmit FIFO is loaded with more data.
- LPI2C controller is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and the receive FIFO is full. The LPI2C controller continues to stall the I2C bus until the receive FIFO is emptied.

The LPI2C controller aborts the existing transfer when MCFGR0[ABORT] becomes 1. This action causes the LPI2C controller to complete the existing word and then transmit a STOP condition. If the receive FIFO is full when the receive operation is aborted, the last received data is discarded. A new transfer does not start while MCFGR0[ABORT] is 1.

48.3.1.3 Receive FIFO and data matching

The receive FIFO stores receive data during controller-receiver transfers. The LPI2C controller can be configured to discard received data instead of storing it in the receive FIFO. This option is configured by the command word in the transmit FIFO.

Received data supports a receive data match function that can match received data against one of two bytes, or against a masked data byte. The data match function can be configured to compare only the first one or two data words received since the last (repeated) START condition. Received data that is already discarded due to the command word cannot cause a data match. It delays the match on the first data word received until after the discarded data is received.

The receiver match function can be configured to discard all received data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, write 0 to MCFGR0[RDMO] before writing 0 to MSR[DMF], to allow all subsequent data to be received.

48.3.1.4 Timing parameters

The following timing parameters can be configured by the LPI2C controller. Parameters are configured separately for high-speed mode (Controller Clock Configuration 1 (MCCR1)) and other modes (Controller Clock Configuration 0 (MCCR0)). This separation allows the high-speed mode controller code to be sent using regular timing parameters. Then it allows a switch to high-speed mode timing (following a repeated START) until the next STOP condition.

The LPI2C controller timing parameters in LPI2C functional clock cycles are configured as follows. They must be configured to meet the I2C timing specification for the required mode.

Table 284. Timing Parameters

I2C Specification Timing Parameter	I2C Specification Timing Symbol	LPI2C Timing Parameter (LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) x (2 ^ PRESCALE)
hold time (repeated) START condition	tHD:STA	(SETHOLD +1) x (2 ^ PRESCALE)
LOW period of the SCL clock	tLOW	(CLKLO + 1) x (2 ^ PRESCALE)
HIGH period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) x (2 ^ PRESCALE)
setup time for a repeated START condition or STOP condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) x (2 ^ PRESCALE)
data hold time	tHD:DAT	(DATAVD + 1) x (2 ^ PRESCALE)
data setup time	tSU:DAT	(SDA_LATENCY + 1) x (2 ^ PRESCALE)
bus free time between a STOP and START condition	tBUF	(CLKLO + 1 + SDA_LATENCY) x (2 ^ PRESCALE)
data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) x (2 ^ PRESCALE)

The latency parameters are defined in the following table. These parameters assume that the risetime is less than one LPI2C functional clock cycle. The risetime depends on a number of factors, including the I/O propagation delay, the I2C bus loading, and the external pull-up resistor sizing. A larger risetime increases the number of cycles that the signal takes to propagate through the synchronizer (and glitch filter), which increases the latency.

Table 285. Synchronization Latency

Timing Parameter	Timing Definition
SCL_LATENCY	ROUNDDOWN ((2 + FILTSCL + SCL_RISETIME) / (2 ^ PRESCALE))
SDA_LATENCY	ROUNDDOWN ((2 + FILTSDA + SDA_RISETIME) / (2 ^ PRESCALE))

The following timing restrictions must be enforced to avoid unexpected START or STOP conditions on the I2C bus. These restrictions also avoid unexpected START or STOP conditions detected by the LPI2C controller. The timing restrictions can be summarized as **SDA cannot change when SCL is high outside a transmitted (repeated) START or STOP condition**.

Table 286. LPI2C Timing Parameter Restrictions

Timing Parameter	Minimum	Maximum	Comment
CLKLO	03h	-	Also: CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	01h	-	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	02h	-	Also: SETHOLD x (2 ^ PRESCALE) > SDA_LATENCY
DATAVD	01h	CLKLO - SDA_LATENCY - 1	Configure DATAVD to meet the data hold requirement in the I2C specification
FILTSCL	00h	[CLKLO x (2 ^ PRESCALE)] - 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)
FILTSDA	FILTSCL	[CLKLO x (2 ^ PRESCALE)] - 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO+SETHOLD+2) x 2	-	Must also be greater than (CLKHI+1)

The timing parameters must be configured to meet the requirements of the I2C specification. This configuration depends on the mode being supported and the LPI2C functional clock frequency. When switching between two modes using the different clock configuration registers (for example, fast and high-speed mode), the PRESCALE factor must remain constant between the modes. Example timing configurations are provided below.

Table 287. LPI2C Example Timing Configurations

I2C Mode	Clock Frequency	Baud Rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Fast	8 MHz	400 kbit/s	0h	0h/0h	04h	0Bh	05h	02h
Fast+	8 MHz	1 Mbit/s	0h	0h/0h	02h	03h	01h	01h
Fast	48 MHz	400 kbit/s	0h	1h/1h	1Dh	3Eh	35h	0Fh
Fast	48 MHz	400 kbit/s	2h	1h/1h	07h	11h	0Bh	03h
Fast+	48 MHz	1 Mbit/s	2h	1h/1h	03h	06h	04h	04h
HS-mode	48 MHz	3.2 Mbit/s	0h	0h/0h	07h	08h	03h	01h
Fast	60 MHz	400 kbit/s	1h	2h/2h	11h	28h	1Fh	08h
Fast+	60 MHz	1 Mbit/s	1h	2h/2h	07h	0Fh	0Bh	01h
HS-mode	60 MHz	3.33 Mbit/s	1h	0h/0h	04h	04h	02h	01h

48.3.1.5 Error conditions

The LPI2C controller monitors for errors while it is active. The following conditions generate an error flag and block a new START condition from being sent, until the flag is cleared by software:

- A START or STOP condition is detected and is not generated by the LPI2C controller (MSR[ALF] becomes 1).
- Transmitting data on SDA and different values are received (MSR[ALF] becomes 1). LPI2C controller stops driving SDA immediately, but continues to drive SCL for the remainder of the byte.

- NACK is detected when transmitting data, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- NACK is detected and is expecting ACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- ACK is detected and is expecting NACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- Transmit FIFO is requesting to transmit or receive data without a START condition (MSR[FEF] becomes 1). This restriction is ignored when MCFGR0[RELAX] is 1.
- SCL (or SDA if MCFGR1[TIMECFG] is 1) is low for (MCFGR2[TIMELOW] * 256) prescaler cycles without a pin transition (MSR[PLTF] becomes 1).

Software must respond to the MSR[PLTF] flag to terminate the existing command either cleanly (by writing 0 to MCR[MEN]), or abruptly (by writing 1 to MCR[RST]).

The MCFGR0[RELAX] control bit can be used to attempt to write 0 to a target with SDA stuck low. If MCFGR0[RELAX] is 1, the LPI2C controller does not wait for the I2C bus to be idle before starting a transfer. Initiating a START command with general call address 00h, then the STOP condition, should generate sufficient SCL clock edges to cause the target to release SDA.

The MCFGR2[BUSIDLE] field can be used to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE+1) prescaler cycles. The bus is considered idle when the LPI2C controller is first enabled. When BUSIDLE is configured greater than zero, then SCL and/or SDA must be high for (BUSIDLE+1) prescaler cycles before the I2C bus is considered idle.

48.3.1.6 Pin configuration

- **Open-drain support:** The LPI2C controller defaults to open-drain configuration of the SDA and SCL pins. Support for true open drain depends on the specific device, and requires the pins where LPI2C pins are muxed to support true open drain.
- **High-speed mode support:** Support for high-speed mode depends on the specific device. This mode requires the SCL pin to support the current source pull-up required in the I2C specification.
- **Ultra-fast mode support:** The LPI2C controller supports the output-only push-pull function required for I2C ultra-fast mode using the SDA and SCL pins. Support for ultra-fast mode also requires the MCFGR1[IGNACK] bit to be 1.
- **Push-pull two-wire support:** A push-pull two-wire configuration is available to the LPI2C controller. If LPI2C is the only controller and all I2C pins on the bus are at the same voltage, this configuration may support a partial high-speed mode. This configuration sets the SCL pin as push-pull for every clock except the ninth clock pulse, to allow high-speed-mode-compatible targets to perform clock stretching. In this mode, the SDA pin is tristated for controller-receive data bits and controller-transmit ACK/NACK bits, and is configured as push-pull at other times. To avoid the risk of contention when SDA is push-pull, the pin can be configured for open-drain operation, as part of the device-specific configuration.
- **Push-pull four-wire support:** The push-pull four-wire configuration separates the SCL input data and output data into separate pins. It also separates the SDA input data and output data into separate pins. The SCL/SDA pins are used for input data; the SCLS/SDAS pins are used for output data, with configurable polarity. This configuration simplifies external connections when connecting the LPI2C to the I2C bus through external level shifters or discrete components. When using this four-wire configuration, the LPI2C controller logic and LPI2C target logic are not able to connect to separate I2C buses.

48.3.2 Target mode

To perform all target mode transfers on the I2C bus, the LPI2C target logic operates independently from the LPI2C controller logic.

48.3.2.1 Address Matching

The LPI2C target can be configured:

- To match one of two addresses, using either seven-bit or ten-bit addressing modes for each address.
- To match a range of addresses in either seven-bit or ten-bit addressing modes.
- To match the General Call Address and generate appropriate flags.
- To match the SMBus Alert Address and generate appropriate flags.

- To detect the high-speed mode controller code, and to disable the digital filters and output valid delay time until the next STOP condition is detected.

After a valid address is matched, the LPI2C target automatically performs target-transmit or target-receive transfers until:

- A NACK is detected (unless SCFGR1[IGNACK] becomes 1).
- A bit error is detected (the LPI2C target is driving SDA, but a different value is sampled).
- A (repeated) START or STOP condition is detected.

When SCFGR1[RXALL] is 1, the LPI2C target receives all addresses and data on the I2C bus. Unless the address was matched, however, it never drives the SDA pin. The LPI2C target should be configured to match only on seven-bit address mode when this feature is enabled. Ten-bit address mode can be supported in software. This option is intended for debugging the contents of an I2C transfer between another controller and target. The SCFGR1[SDCFG] and SCFGR1[RSCFG] likewise configure the STOP or repeated START flags to assert on all STOP or repeated START conditions, even when there was no address match.

48.3.2.2 Transmit and Receive Data

- The Transmit and Receive Data registers are double-buffered and only update during a target-transmit and target-receive transfer, respectively.
- The target address that was received can be configured to be read from the Receive Data register (for example, when using DMA to transfer data). It can also be configured to be read from the Address Status register.
- The Transmit Data register can be configured to request data only after a target-transmit transfer is detected. It can also be configured to request new data whenever the Transmit Data register is empty.
- The Transmit Data register should only be written when the Transmit Data flag (SSR[TDF]) is 1.
- The Receive Data register should only be read when the Received Data flag (SSR[RDF]) is 1 (or the Address Valid Flag (SSR[AVF]) is 1 and SCFGR1[RXCFG] = 1).
- The Address Status register should only be read when the Address Valid Flag (SSR[AVF]) is 1.

48.3.2.3 Read request

The LPI2C target generates a read request when the I2C bus is idle and software writes 1 to the SCFGR0[RDREQ] bit. This occurrence causes the LPI2C target to pull SDA pin low until either the first SCL falling edge or the SCFGR0[RDREQ] becomes 0. Following the next STOP condition or a software timeout, software can proceed as follows:

- If the LPI2C target was accessed at the correct address, the read request is acknowledged, and software should write 0 to SCFGR0[RDREQ].
- If the SCFGR0[RDACK] is 1, then the read request was acknowledged by a START followed by one SCL pulse followed by a STOP condition. Software should write 0 to SCFGR0[RDREQ].
- If neither of the first two cases occurred, then the read request was not acknowledged. Software should write 0 and then 1 to SCFGR0[RDREQ] after an appropriate delay, provided the I2C bus is idle.
- If a software timeout occurs, the read request was not acknowledged, and software should write 0 to SCFGR0[RDREQ]. Another request can be generated after an appropriate delay.

Writing SCFGR0[RDREQ] when the I2C bus is busy results in a logic 0 being written. Software should first write 1 to SCFGR0[RDREQ] and then confirm that the register was written correctly. If the register did not update correctly, then the I2C bus is no longer idle and the read request should be attempted later.

48.3.2.4 Clock stretching

The LPI2C target supports many configurable options for clock stretching. The following conditions can be configured to perform clock stretching:

- The Address Valid flag becomes 1 during the ninth clock pulse of the address byte.
- The Transmit Data flag becomes 1 during the ninth clock pulse of a target-transmit transfer.

- The Receive Data flag becomes 1 during the ninth clock pulse of a target-receive transfer.
- The Transmit ACK flag becomes 1 during the eighth clock pulse of an address byte or a target-receive transfer. In high-speed mode, this option is disabled.
- Clock stretching can be extended for CLKHOLD cycles, to allow additional setup time to sample the SDA pin externally. In high-speed mode, this option is disabled.

When clock stretching is enabled, clock stretching extends for one peripheral bus clock cycle after SDA updates, unless extended by the CLKHOLD configuration.

48.3.2.5 Timing parameters

The LPI2C target can configure the following timing parameters:

- SDA data valid time from SCL negation to SDA update
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally
- SCL glitch filter time
- SDA glitch filter time

These parameters are disabled when SCR[FILTEN] is 0, when SCR[FILTDZ] is 1 in Doze mode, and when LPI2C target detects high-speed mode. When disabled, the LPI2C target is clocked directly from the I2C bus. In this case, the target may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

The LPI2C target places the following restrictions on the timing parameters:

- FILTSDA must be configured to greater than or equal to FILTSCL (unless compensating for board level skew between SDA and SCL).
- DATAVD must be configured less than the minimum SCL low period.

48.3.2.6 Error conditions

The LPI2C target can flag the following error conditions:

- The bit error flag becomes 1 when the LPI2C target is driving SDA but it samples a different value than what is expected.
- The FIFO error flag becomes 1 due to a transmit data underrun or a receive data overrun. To eliminate the possibility of underrun and overrun, enable clock stretching. When SCFGR1[RXNACK] is 1, the LPI2C target generates a NACK on a receive data overrun.
- The FIFO error flag also becomes 1 due to an address overrun, but only when SCFGR1[RXCFG] is 1. To eliminate the possibility of overrun, enable clock stretching. When SCFGR1[RXNACK] is 1, the LPI2C target generates a NACK on an address overrun regardless of SCFGR1[RXCFG].

The LPI2C target does not implement a timeout due to SCL and/or SDA being stuck low. If this detection is required, use the LPI2C controller logic so the software can reset the LPI2C target when this condition is detected.

48.3.3 Low-power modes

Table 288. LPI2C low-power modes

Mode	LPI2C operation
Run	Normal operations
Sleep	If MCR[DOZEN] = 0 and LPI2C uses an external or internal clock source that operates during Sleep mode, can continue operating in Sleep mode. LPI2C can generate an interrupt or DMA request to cause a wake-up from Sleep mode.

Table continues on the next page...

Table 288. LPI2C low-power modes (continued)

Mode	LPI2C operation
Deep Sleep	Before entering Deep Sleep mode, the LPI2C waits for the current transfer to finish any pending operation, while temporarily ignoring MCR[DOZEN].

48.3.4 Debug mode

Table 289. LPI2C debug mode

Mode	LPI2C operation
Debug	If MCR[DBGEN] = 1, can continue operating in debug mode.

48.3.5 Interrupts and DMA requests

Depending on the configuration, interrupts and DMA requests can be combined:

- LPI2C controller and target interrupts
- LPI2C controller and target transmit DMA requests
- LPI2C controller and target receive DMA requests

48.3.5.1 Controller mode

The table below lists the Controller mode sources that can generate LPI2C controller interrupts and LPI2C controller transmit/receive DMA requests.

Table 290. Controller interrupts and DMA requests

Controller Status (MSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit Data Flag	Data can be written to Transmit FIFO, as configured by the Transmit FIFO Watermark MFCR[TXWATER].	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the Receive FIFO, as configured by the Receive FIFO Watermark MFCR[RXWATER].	Y	RX	Y
EPF	End Packet Flag	Controller has transmitted a Repeated START or STOP condition.	Y	N	Y
SDF	STOP Detect Flag	Controller has transmitted a STOP condition and optionally the LPI2C controller is idle.	Y	N	Y
NDF	NACK Detect Flag	• During an address byte, the controller expects an ACK but detects a NACK. • During an address byte, the controller expects a NACK but detects an ACK. • During a controller-transmitter data byte, the controller detects a NACK.	Y	N	Y

Table continues on the next page...

Table 290. Controller interrupts and DMA requests (continued)

Controller Status (MSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA request?	Low-power wake-up?
ALF	Arbitration Lost Flag	• The controller lost arbitration due to a START/STOP condition detected at the wrong time, • Or the controller was transmitting data but received data different from the data that was transmitted.	Y	N	Y
FEF	FIFO Error Flag	The controller expects a START condition in the Command FIFO, but the next entry in the Command FIFO is not a START condition.	Y	N	Y
PLTF	Pin Low Timeout Flag	Pin low timeout is enabled and SCL (or SDA, if configured) is low for longer than the configured timeout.	Y	N	Y
DMF	Data Match Flag	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry.	Y	N	Y
STF	START Detect Flag	A START condition is detected on the I2C bus, and optionally, LPI2C controller is idle.	Y	N	Y
MBF	Controller Busy Flag	LPI2C controller is busy transmitting/ receiving data.	N	N	N
BBF	Bus Busy Flag	LPI2C controller is enabled and activity is detected on the I2C bus, but no STOP condition is detected and no bus idle timeout (if enabled) occurred.	N	N	N

48.3.5.2 Target mode

The table below lists the target mode sources that can generate LPI2C target interrupts and LPI2C target transmit/receive DMA requests.

Table 291. Target interrupts and DMA requests

Target Status (SSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit Data Flag	Data can be written to the Target Transmit Data (STDR).	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the Target Receive Data (SRDR).	Y	RX	Y
AVF	Address Valid Flag	Address can be read from the Target Address Status (SASR).	Y	RX	Y

Table continues on the next page...

Table 291. Target interrupts and DMA requests (continued)

Target Status (SSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TAF	Transmit ACK Flag	ACK/NACK can be written to the Target Transmit ACK (STAR).	Y	N	Y
RSF	Repeated Start Flag	Target has detected an address match followed by a repeated START condition.	Y	RX	Y
SDF	STOP Detect Flag	Target has detected an address match followed by a STOP condition.	Y	RX	Y
BEF	Bit Error Flag	Target was transmitting data, but received data different from what was transmitted.	Y	N	Y
FEF	FIFO Error Flag	• Transmit data underrun • Receive data overrun • Address status overrun (when Receive Data Configuration SCFGR1[RXCFG] = 1). FEF flag can only become 1 when clock stretching is disabled.	Y	N	Y
AM0F	Address Match 0 Flag	Target detected an address match with SAMR[ADDR0] field.	Y	N	N
AM1F	Address Match 1 Flag	Target detected an address match with SAMR[ADDR1] field or using an address range.	Y	N	N
GCF	General Call Flag	Target detected an address match with the General Call address.	Y	N	N
SARF	SMBus Alert Response Flag	Target detected an address match with the SMBus Alert address.	Y	N	N
SBF	Target Busy Flag	LPI2C target is busy receiving an address byte or is transmitting/receiving data.	N	N	N
BBF	Bus Busy Flag	LPI2C target is enabled and a START condition is detected on I2C bus, but no STOP condition has been detected.	N	N	N

48.3.5.3 End-of-packet DMA transfer

End-of-packet functionality serves serial interfaces where the transfer size may not be known by software in advance and the data is pushed by an external device. Examples include UART receive, I2C target mode, and SPI target mode. End-of-packet processing is intended to ensure that data does not become stranded in either the receive FIFO or the DMA receive buffer. Support for end-of-packet processing must be implemented in both the serial interfaces and the DMA controller.

The condition that signals the end of packet differs for each serial interface but is processed by the serial peripheral and DMA in the same way. For example:

- UART end of packet is signaled by an idle line condition.

- I2C end of packet is signaled by STOP and/or repeated START condition.
- SPI end of packet is signaled by PCS negation.

When the serial peripheral is configured to signal an end-of-packet condition to the DMA and the serial interface detects an end-of-packet condition, it asserts the DMA request for the receive FIFO regardless of the watermark configuration. For larger watermark configurations, this behavior ensures that the last few words of the transfer are first flushed from the receive FIFO.

The DMA then reads the contents of the receive FIFO, depending on the first word in the FIFO.

- If the receive FIFO is empty, the serial interface signals an end-of-packet condition to the DMA controller.
- If the receive FIFO is not empty, but the first word in the FIFO starts a new packet, data is not pulled from the receive FIFO. Also, the serial interface signals an end-of-packet condition to the DMA controller.
- If the receive FIFO is not empty, and the first word in the FIFO does not start a new packet, the DMA transfers data as normal.

Because the DMA may transfer multiple words per request, the end-of-packet condition persists until the DMA minor loop finishes and no additional data is pulled from the FIFO. The status flag that triggered the end-of-packet condition is cleared when the minor loop completes after the end of packet is signaled to the DMA controller.

When the DMA detects the end-of-packet condition, it writes all received words up to the end of packet into system memory. It also saves the destination address for the word after the last valid data. The DMA then terminates the channel as if the major loop completed, including final offsets and optional interrupts, channel linking and scatter-gather. The final destination address can be saved to system memory or is available in the destination address register.

Because the DMA terminates the major loop, the receive FIFO is not serviced until the DMA is reconfigured. This reconfiguration can be done through either software or hardware (for example, channel linking or scatter-gather). This delay should be minimized to avoid receiver FIFO overrun. Automatic DMA end-of-packet processing is not recommended when there are only a few words transferred between end-of-packet conditions. In this case, the DMA spends more time processing the end of packet than transferring the data. For example, to avoid an excessive number of idle conditions, the UART idle line length should be increased as needed.

48.3.6 Clocks

Table 292. LPI2C clocks

LPI2C functional clock	The LPI2C functional clock is asynchronous to the bus clock. It can remain enabled in low-power modes to support I2C bus transfers by the LPI2C controller. The functional clock is also used by the LPI2C target to support digital filter and data hold time configurations. The LPI2C controller divides the functional clock by a prescaler (MCFGR1[PRESCALE]) and the resulting frequency must be at least eight times faster than the I2C bus bandwidth.
External clock	The LPI2C target logic is clocked directly from the external pins SCL and SDA (or SCLS and SDAS, if controller and target are implemented on separate pins). This clocking allows the LPI2C target to remain operational, even when the LPI2C functional clock is disabled. **NOTE** If the LPI2C functional clock is disabled, the LPI2C target digital filter must be disabled. This condition can affect compliance with some of the timing parameters of the I2C specification, such as data hold time.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C controller and target registers.

For device-specific clocking information, see the Clocking chapter.

48.3.7 Resets

Table 293. LPI2C resets

Chip reset	The logic and registers for the LPI2C controller and target are reset to their default state on a chip reset.
Software reset	• The LPI2C controller implements a software reset bit in its Control Register. MCR[RST] resets all controller logic and registers to their default state, except for the MCR register itself. • The LPI2C target implements a software reset bit in its Control Register. SCR[RST] resets all target logic and registers to their default state, except for the SCR register itself.
FIFO reset	• The LPI2C controller implements write-only control bits that reset the transmit FIFO (MCR[RTF]) and receive FIFO (MCR[RRF]). After a FIFO is reset, that FIFO is empty. • The LPI2C target implements write-only control bits that reset the transmit data register (SCR[RTF] and receive data register (SCR[RRF]). After a data register is reset, that data register is empty.

48.3.8 Peripheral Triggers

The connection of the LPI2C peripheral triggers to other peripherals depend upon the specific device being used.

Table 294. LPI2C triggers

Trigger	Description
Controller Output Trigger	The LPI2C controller generates an output trigger that can be connected to other peripherals on the device. The controller output trigger asserts on either a repeated START or a STOP condition. The trigger remains asserted for one cycle of the LPI2C functional clock divided by MCFGR1[PRESCALE].
Target Output Trigger	The LPI2C target generates an output trigger that can be connected to other peripherals on the device. The target output trigger asserts on either a repeated START or a STOP condition that occurs after a target address match. The target output trigger remains asserted until the next target SCL pin negation.
Input Trigger	Controls the start of a LPI2C bus transfer. The input trigger is synchronized. To be detected, the input trigger must assert for at least two cycles of the LPI2C functional clock divided by MCFGR1[PRESCALE].

48.4 External signals

Table 295. Signals

Signal	Name	Two-Wire Scheme	Four-Wire Scheme	I/O
SCL	LPI2C clock line	SCL	In Four-wire mode, this pin is the SCL input pin.	I/O
SDA	LPI2C data line	SDA	In Four-wire mode, this pin is the SDA input pin.	I/O
SCLS	Secondary I2C clock line	Not used	In Four-wire mode, this pin is the SCLS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SCL pin.	I/O

Table continues on the next page...

Table 295. Signals (continued)

Signal	Name	Two-Wire Scheme	Four-Wire Scheme	I/O
SDAS	Secondary I2C data line	Not used	In Four-wire mode, this pin is the SDAS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SDA pin.	I/O

The I2C Two-wire serial bus diagram shows the two signal connection.

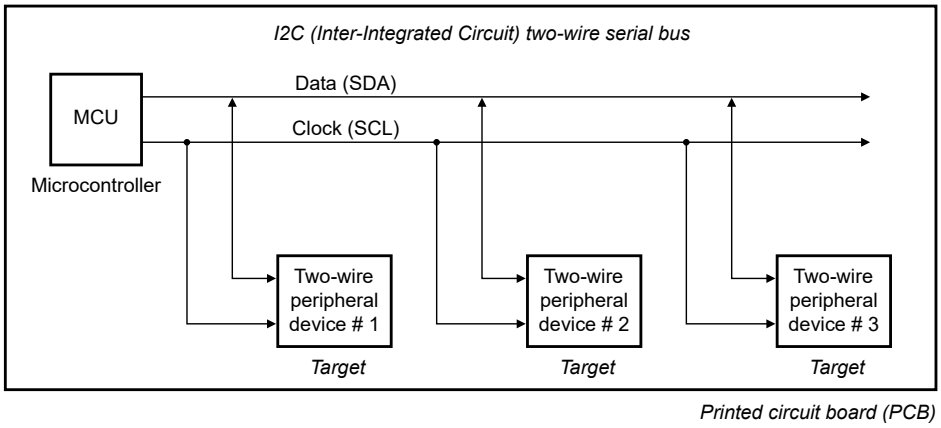


Figure 177. I2C Two-wire serial bus

The I2C Four-wire serial bus diagram shows a possible four signal connection.

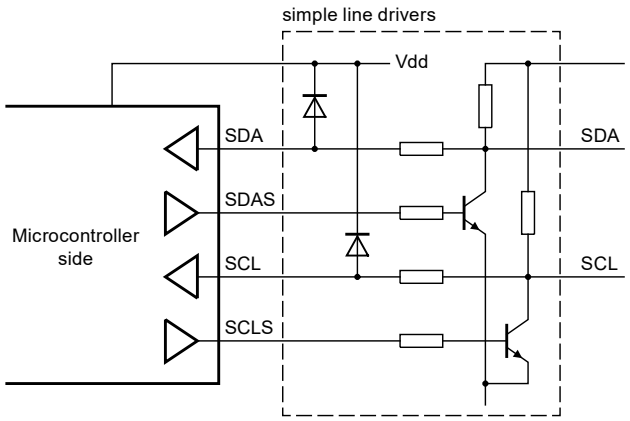


Figure 178. I2C Four-wire serial bus

48.5 Memory Map and Registers

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module will not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

48.5.1 LPI2C register descriptions

48.5.1.1 LPI2C memory map

LPI2C0 base address: 4003_3000h

LPI2C1 base address: 4003_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0103_0003h
4h	Parameter (PARAM)	32	R	0000_0202h
10h	Controller Control (MCR)	32	RW	0000_0000h
14h	Controller Status (MSR)	32	RW	0000_0001h
18h	Controller Interrupt Enable (MIER)	32	RW	0000_0000h
1Ch	Controller DMA Enable (MDER)	32	RW	0000_0000h
20h	Controller Configuration 0 (MCFGR0)	32	RW	0000_0000h
24h	Controller Configuration 1 (MCFGR1)	32	RW	0000_0000h
28h	Controller Configuration 2 (MCFGR2)	32	RW	0000_0000h
2Ch	Controller Configuration 3 (MCFGR3)	32	RW	0000_0000h
40h	Controller Data Match (MDMR)	32	RW	0000_0000h
48h	Controller Clock Configuration 0 (MCCR0)	32	RW	0000_0000h
50h	Controller Clock Configuration 1 (MCCR1)	32	RW	0000_0000h
58h	Controller FIFO Control (MFCR)	32	RW	0000_0000h
5Ch	Controller FIFO Status (MFSR)	32	R	0000_0000h
60h	Controller Transmit Data (MTDR)	32	W	0000_0000h
70h	Controller Receive Data (MRDR)	32	R	0000_4000h
78h	Controller Receive Data Read Only (MRDROR)	32	R	0000_4000h
110h	Target Control (SCR)	32	RW	0000_0000h
114h	Target Status (SSR)	32	RW	0000_0000h
118h	Target interrupt enable (SIER)	32	RW	0000_0000h
11Ch	Target DMA Enable (SDER)	32	RW	0000_0000h
120h	Target Configuration 0 (SCFGR0)	32	RW	0000_0000h
124h	Target Configuration 1 (SCFGR1)	32	RW	0000_0000h
128h	Target Configuration 2 (SCFGR2)	32	RW	0000_0000h
140h	Target Address Match (SAMR)	32	RW	0000_0000h
150h	Target Address Status (SASR)	32	R	0000_4000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
154h	Target Transmit ACK (STAR)	32	RW	0000_0000h
160h	Target Transmit Data (STDR)	32	W	0000_0000h
170h	Target Receive Data (SRDR)	32	R	0000_4000h
178h	Target Receive Data Read Only (SRDROR)	32	R	0000_4000h

48.5.1.2 Version ID (VERID)

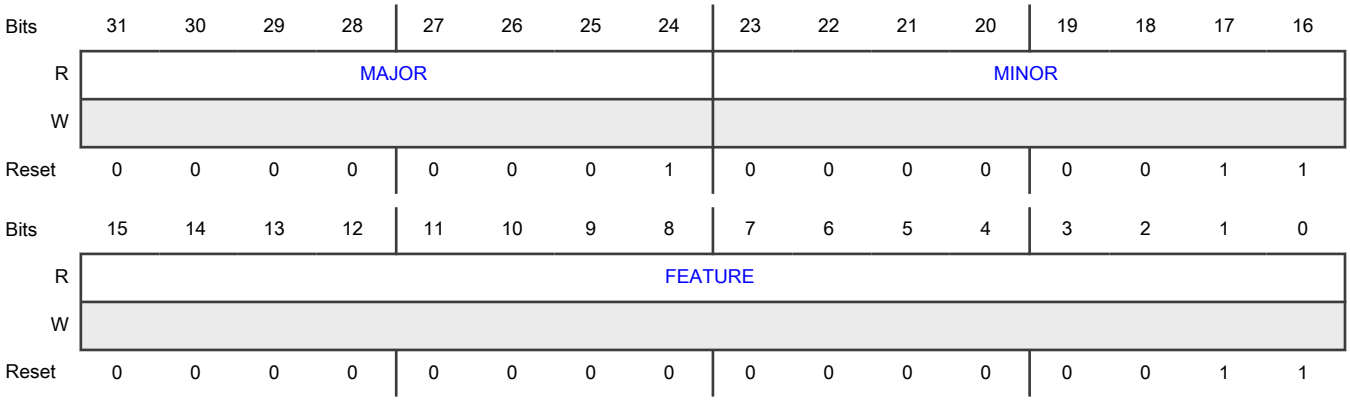
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification. Read-only field.
15-0	Feature Specification Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
FEATURE	Returns the feature set number. Read-only field. 0000_0000_0000_0010b - Controller only, with standard feature set 0000_0000_0000_0011b - Controller and target, with standard feature set

48.5.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values that were implemented in the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MRXFIFO				0				MTXFIFO			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-16 —	Reserved
15-12 —	Reserved
11-8 MRXFIFO	Controller Receive FIFO Size Configures the number of words in the controller receive FIFO to $2^{MRXFIFO}$
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 MTXFIFO	Controller Transmit FIFO Size Configures the number of words in the controller transmit FIFO to $2^{MTXFIFO}$

48.5.1.4 Controller Control (MCR)

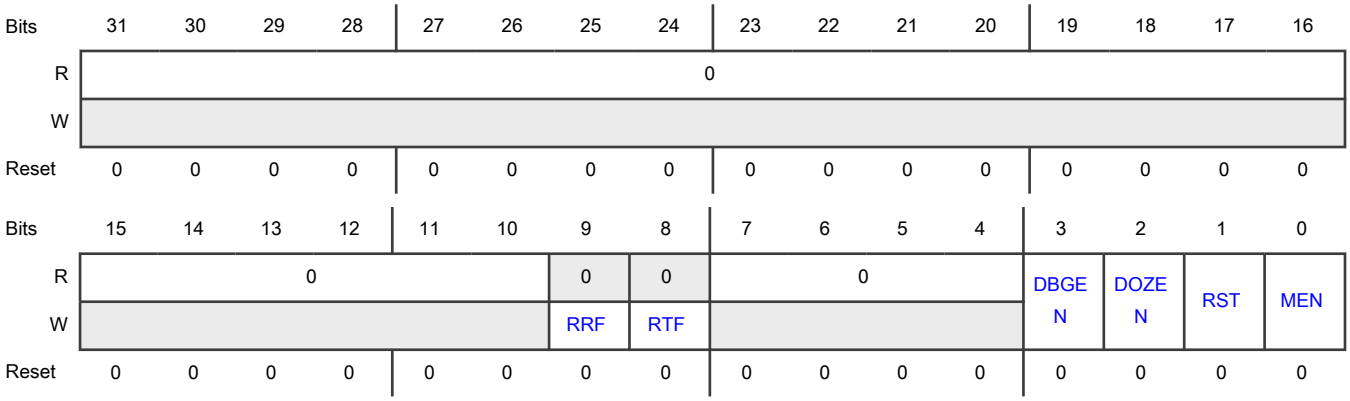
Offset

Register	Offset
MCR	10h

Function

Contains resets, debug enable, and other controller control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Resets the receive FIFO 0b - No effect 1b - Receive FIFO is reset
8	Reset Transmit FIFO

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTF	Resets the transmit FIFO 0b - No effect 1b - Transmit FIFO is reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables the controller in debug mode 0b - Controller is disabled in debug mode 1b - Controller is enabled in debug mode
2 DOZEN	Doze mode enable Enables or disables the controller in doze mode 0b - Controller is enabled in doze mode 1b - Controller is disabled in doze mode
1 RST	Software Reset Resets all internal controller logic and registers, except the Controller Control (MCR) register. The bit field remains 1 until written 0 by software. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 MEN	Controller Enable Enables the controller logic 0b - Disabled 1b - Enabled

48.5.1.5 Controller Status (MSR)

Offset

Register	Offset
MSR	14h

Function

Contains status flags for transmit and receive data, for start and stop conditions, and for bus and controller busy or idle status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STF	DMF	PLTF	FEF	ALF	NDF	SDF	EPF	0						RDF	TDF
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Indicates that the I2C bus is busy. 0b - Idle 1b - Busy
24 MBF	Controller Busy Flag Indicates that the I2C controller is busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 STF	START Flag Indicates that a START condition has been detected on the bus when the bus is idle. When MCFGR1[STARTCFG] is 0, STF only asserts if LPI2C controller is also idle. When MCFGR1[STARTCFG] is 1, STF asserts for any START condition when the I2C bus is idle. 0b - START condition not detected. 1b - START condition detected.
14 DMF	Data Match Flag Indicates that the received data has matched the MDMR[MATCH0] and/or MDMR[MATCH1] fields (as configured by MCFGR1[MATCFG]. Received data discarded due to CMD field does not cause DMF to become 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Have not received matching data 1b - Have received matching data
13 PLTF	Pin Low Timeout Flag Becomes 1 when the SCL and/or SDA input is low for more than PINLOW cycles (Pin Low Timeout, MCFGR3[PINLOW]), even when the LPI2C controller is idle. • Software is responsible for resolving the pin low condition. • PLTF cannot become 0 as long as the pin low timeout continues. • Before the LPI2C can initiate a START condition, the PLTF must be written 0. See MCFGR1[TIMECFG] for the SCL and/or SDA timeout settings. 0b - Pin low timeout has not occurred or is disabled 1b - Pin low timeout has occurred
12 FEF	FIFO Error Flag Detects an attempt to send or receive data without first generating a (repeated) START condition. This error can occur if the transmit FIFO underflows when MCFGR1[AUTOSTOP] = 1. When FEF is 1, the LPI2C controller sends a STOP condition (if busy), and does not initiate a new START condition until FEF is 0. 0b - No error 1b - Controller sending or receiving data without a START condition
11 ALF	Arbitration Lost Flag Becomes 1 if either of these conditions exist: • The LPI2C controller transmits a logic 1 and detects a logic 0 on the I2C bus • The LPI2C controller detects a START or STOP condition while the LPI2C controller is transmitting data When ALF becomes 1, the LPI2C controller releases the I2C bus (goes idle), and the LPI2C controller does not initiate a new START condition until the ALF becomes 0. 0b - Controller has not lost arbitration 1b - Controller has lost arbitration
10 NDF	NACK Detect Flag Becomes 1 if the LPI2C controller detects a NACK it was not expecting when transmitting an address or data. When 1, the controller does not initiate a new START condition until NDF has been written 0. If a NACK is expected for a given address (as configured by the command word), then the NDF is 1 if a NACK is not generated. When NDF is 1, the LPI2C controller automatically transmits a STOP condition if MCFGR1[AUTOSTOP] = 1, or the transmit FIFO is not empty. 0b - Unexpected NACK was not detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Unexpected NACK was detected
9 SDF	STOP Detect Flag Becomes 1 when the LPI2C controller generates a STOP condition. When MCFGR1[STOPCFG] = 0 the flag asserts for any STOP condition. When MCFGR1[STOPCFG] = 1 the flag only asserts if the LPI2C controller is also idle (transmit FIFO is empty). 0b - Controller has not generated a STOP condition 1b - Controller has generated a STOP condition
8 EPF	End Packet Flag Becomes 1 when the LPI2C controller generates either a repeated START condition or a STOP condition. Does not write 1 when the controller first generates a START condition. 0b - Controller has not generated a STOP or Repeated START condition 1b - Controller has generated a STOP or Repeated START condition
7-2 —	Reserved
1 RDF	Receive Data Flag Becomes 1 when the number of words in the receive FIFO is greater than MFCR[RXWATER]. 0b - Receive data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag Becomes 1 when the number of words in the transmit FIFO is equal or less than MFCR[TXWATER]. 0b - Transmit data is not requested 1b - Transmit data is requested

48.5.1.6 Controller Interrupt Enable (MIER)

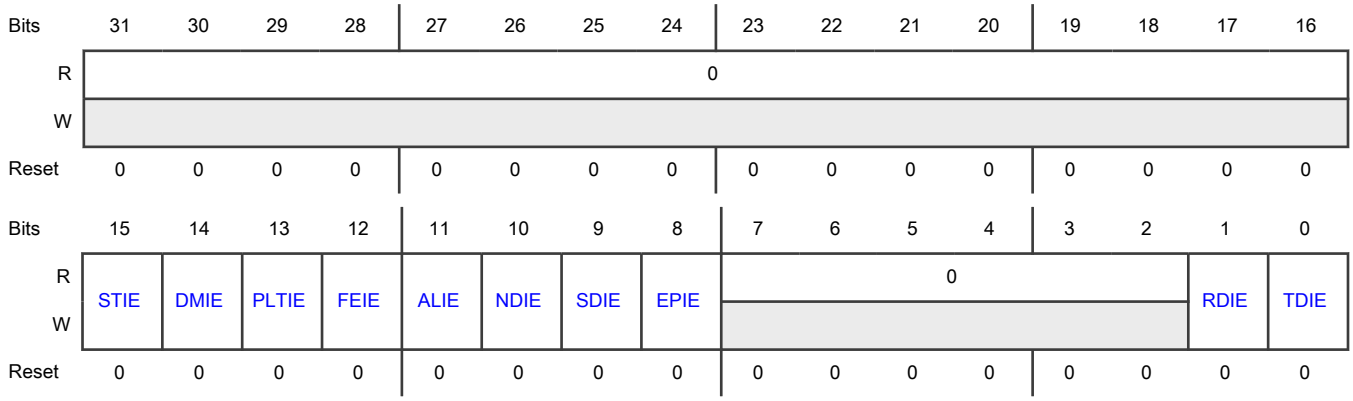
Offset

Register	Offset
MIER	18h

Function
Contains:

- Transmit and receive data interrupt enables
- START, STOP, and NACK detect interrupt enables
- DMA interrupt enables

Diagram



Fields

Field	Function
31-16 —	Reserved
15 STIE	START Interrupt Enable Enables interrupt for START. 0b - Disabled 1b - Enabled
14 DMIE	Data Match Interrupt Enable Enables interrupt for data match. 0b - Disabled 1b - Enabled
13 PLTIE	Pin Low Timeout Interrupt Enable Enables interrupt for pin-low timeout. 0b - Disabled 1b - Enabled
12 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disabled 1b - Enabled
11 ALIE	Arbitration Lost Interrupt Enable Enables interrupt for arbitration lost. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 NDIE	NACK Detect Interrupt Enable Enables interrupt for NACK detection. 0b - Disabled 1b - Enabled
9 SDIE	STOP Detect Interrupt Enable Enables interrupt for STOP detection. 0b - Disabled 1b - Enabled
8 EPIE	End Packet Interrupt Enable Enables interrupt for end packet. 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disabled 1b - Enabled

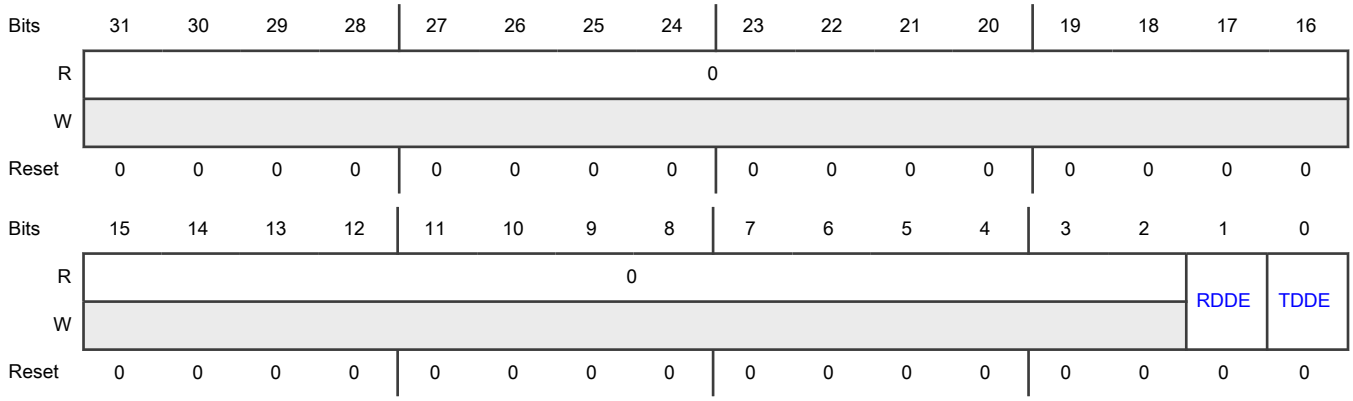
48.5.1.7 Controller DMA Enable (MDER)

Offset

Register	Offset
MDER	1Ch

Function
Contains DMA transmit, request, and receive enables.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables DMA receive data. 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit Data DMA Enable Enables DMA transmit data. 0b - DMA request is disabled 1b - DMA request is enabled

48.5.1.8 Controller Configuration 0 (MCFGR0)

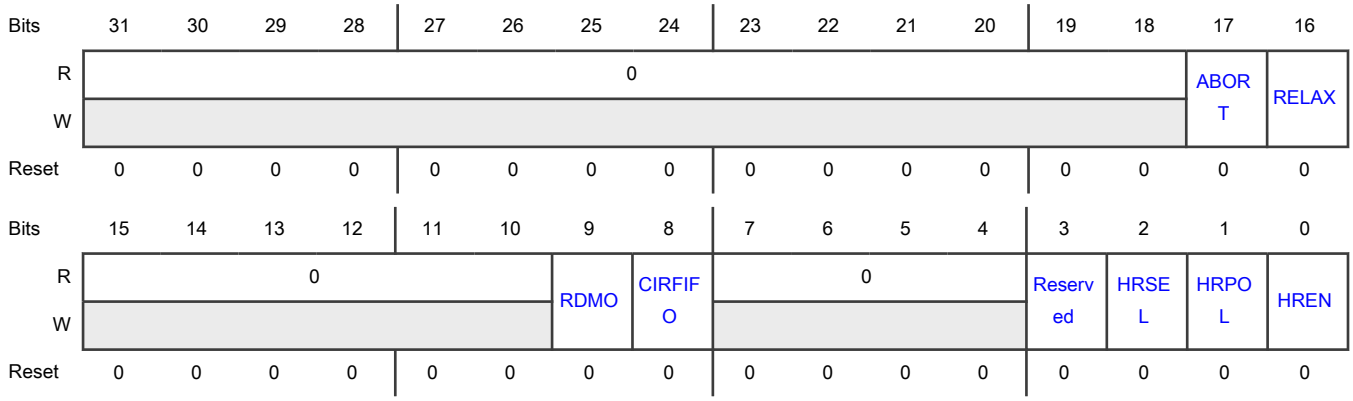
Offset

Register	Offset
MCFGR0	20h

Function

Contains host settings and other receive and transfer settings.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 ABORT	Abort Transfer When 1, the LPI2C controller completes the existing byte and then sends a STOP condition. A new transfer does not start while this bit is 1. 0b - Normal transfer 1b - Abort existing transfer and do not start new transfer
16 RELAX	Relaxed Mode When enabled, the LPI2C controller relaxes several of the transfer restrictions. Transfers start if the bus is busy and FIFO commands are not checked for errors. This mode can be used to attempt to recover a stuck I2C target by toggling SCL pin. 0b - Normal transfer 1b - Relaxed transfer
15-10 —	Reserved
9 RDMO	Receive data match only When enabled, all received data that does not cause the Data Match Flag (MSR[DMF]) to become 1 is discarded. After MSR[DMF] is 1, the RDMO configuration is ignored. When disabling RDMO, write 0 to this bit before writing 0 to MSR[DMF], to ensure that no receive data is lost. 0b - Received data is stored in the receive FIFO 1b - Received data is discarded unless the Data Match Flag (MSR[DMF]) is 1.
8 CIRFIFO	Circular FIFO enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO empties as normal, but after the LPI2C controller is idle and the transmit FIFO is empty, then the

Table continues on the next page...

Table continued from the previous page...

Field	Function
	read pointer value will be restored from the temporary register. This setting causes the contents of the transmit FIFO to be cycled through repeatedly. If MCFGR1[AUTOSTOP] is 1, then a STOP condition is sent whenever the transmit FIFO is empty and the read pointer is restored. 0b - Circular FIFO is disabled 1b - Circular FIFO is enabled
7-4 —	Reserved
3 —	Reserved
2 HRSEL	Host request select Selects the source of the host request input. When host request is enabled, this bit should not change. 0b - Reserved 1b - Host request input is input trigger
1 HRPOL	Host request polarity Configures the polarity of the host request input. When host request is enabled, this bit should not change. 0b - Active low 1b - Active high
0 HREN	Host request enable When enabled, the LPI2C controller only initiates a START condition if the host request input is asserted and the bus is idle. A repeated START is not affected by the host request. 0b - Host request input is disabled 1b - Host request input is enabled

48.5.1.9 Controller Configuration 1 (MCFGR1)

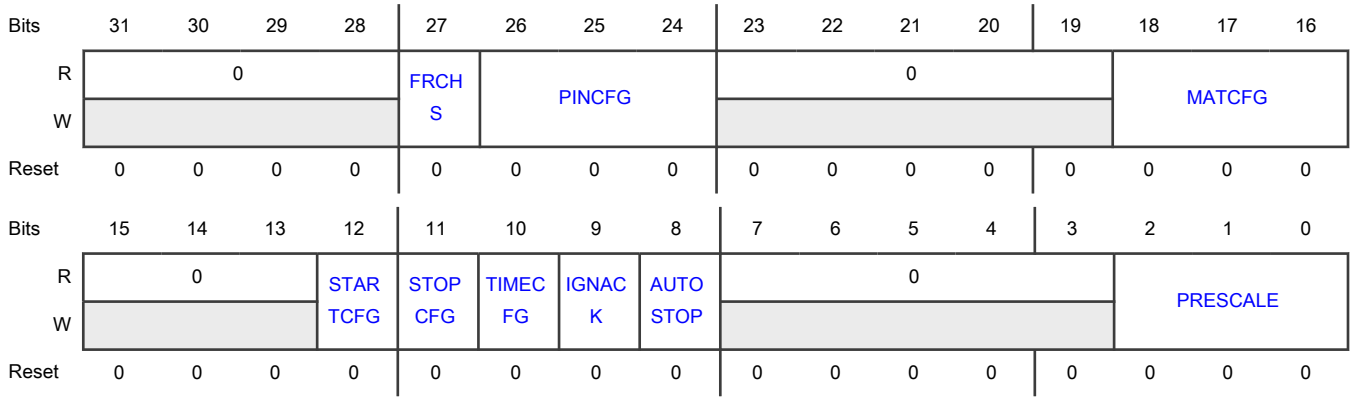
Offset

Register	Offset
MCFGR1	24h

Function

Should only be written when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 FRCHS	Force HS mode When 1, forces the LPI2C pin state into HS mode. Writing 1 does not impact the digital filter configuration or digital timing parameters. 0b - No effect 1b - LPI2C pin state forced into HS mode.
26-24 PINCFG	Pin Configuration Configures the pin mode for LPI2C. 000b - Two-pin open drain mode. SCL/SDA pins: Bidirectional open drain for controller and target. SCLS/SDAS pins: Not used. 001b - Two-pin output only mode (ultra-fast mode). SCL/SDA pins: Output-only (ultra-fast mode) open drain for controller and target. SCLS/SDAS pins: Not used. 010b - Two-pin push-pull mode. SCL/SDA pins: Bidirectional push-pull for controller and target. SCLS/SDAS pins: Not used. 011b - Four-pin push-pull mode. SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Output-only push-pull for controller and target. 100b - Two-pin open drain mode with separate LPI2C target. SCL/SDA pins: Bidirectional open drain for controller. SCLS/SDAS pins: Bidirectional open drain for target. 101b - Two-pin output only mode (ultra-fast mode) with separate LPI2C target. SCL/SDA pins: Output-only (ultra-fast mode) open drain for controller. SCLS/SDAS pins: Output-only open drain for target. 110b - Two-pin push-pull mode with separate LPI2C target. SCL/SDA pins: Bidirectional push-pull for controller. SCLS/SDAS pins: Bidirectional push-pull for target. 111b - Four-pin push-pull mode (inverted outputs). SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Inverted output-only push-pull for controller and target.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-19 —	Reserved
18-16 MATCFG	Match Configuration Configures the condition that causes MSR[DMF] to become 1. See Controller Data Match (MDMR). 000b - Match is disabled 001b - Reserved 010b - Match is enabled (first data word equals MDMR[MATCH0] OR MDMR[MATCH1]) 011b - Match is enabled (any data word equals MDMR[MATCH0] OR MDMR[MATCH1]) 100b - Match is enabled (first data word equals MDMR[MATCH0] AND second data word equals MDMR[MATCH1) 101b - Match is enabled (any data word equals MDMR[MATCH0] AND next data word equals MDMR[MATCH1) 110b - Match is enabled (first data word AND MDMR[MATCH1] equals MDMR[MATCH0] AND MDMR[MATCH1]) 111b - Match is enabled (any data word AND MDMR[MATCH1] equals MDMR[MATCH0] AND MDMR[MATCH1])
15-13 —	Reserved
12 STARTCFG	START Configuration Configures the assertion of the START Flag (MSR[STF]). 0b - MSR[STF] asserts on START condition provided both I2C bus and LPI2C controller are idle (that is, any non-repeated START condition initiated by any other controller on the bus but not the LPI2C controller). 1b - MSR[STF] asserts on START condition provided I2C bus is idle (that is, any non-repeated START condition initiated by any controller on the bus including the LPI2C controller).
11 STOPCFG	STOP Configuration Configures the assertion of the STOP Detect Flag (MSR[SDF]). 0b - MSR[SDF] asserts on any STOP condition generated by LPI2C controller. 1b - MSR[SDF] asserts on last STOP condition before LPI2C controller is idle (that is, the transmit FIFO is empty at the time of the STOP condition).
10 TIMECFG	Timeout Configuration Configures the timeout settings for the Pin Low Timeout Flag field (MSR[PLTF]). 0b - MSR[PLTF] becomes 1 if SCL is low for longer than the configured timeout. 1b - MSR[PLTF] becomes 1 if either SCL or SDA is low for longer than the configured timeout.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 IGNACK	Ignore NACK When 1, the received NACK field is ignored and assumed to be ACK. Required to be 1 in Ultra-Fast Mode. 0b - No effect 1b - LPI2C controller treats a received NACK as if it (NACK) was an ACK and the NACK Detect Flag is never written 1.
8 AUTOSTOP	Automatic STOP Generation When enabled, a STOP condition is generated whenever the LPI2C controller is busy and the transmit FIFO is empty. The STOP condition can also be generated using a transmit FIFO command. 0b - No effect 1b - STOP condition is automatically generated when the transmit FIFO is empty and the LPI2C controller is busy
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C controller logic, except for the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

48.5.1.10 Controller Configuration 2 (MCFGR2)

Offset

Register	Offset
MCFGR2	28h

Function
Should only be written when the I2C controller is disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				BUSIDLE											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C controller digital glitch filters for SDA input. • A configuration of 0 disables the glitch filter. • Glitches equal to or less than FILTSDA cycles long are filtered out and ignored. • The latency through the glitch filter is equal to FILTSDA cycles, and must be configured to be less than the minimum SCL low or high period. • The glitch filter cycle count is not affected by MCFGR1[PRESCALE], and is automatically bypassed in high-speed mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C controller digital glitch filters for SCL input. • A configuration of 0 disables the glitch filter. • Glitches equal to or less than FILTSCL cycles long are filtered out and ignored. The FILTSCL cycles are based on the functional clock. • The latency through the glitch filter is equal to FILTSCL cycles, and must be configured to be less than the minimum SCL low or high period. • The glitch filter cycle count is not affected by MCFGR1[PRESCALE], and is automatically bypassed in high-speed mode.
15-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-0 BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period in clock cycles. • If both SCL and SDA are high for longer than BUSIDLE cycles, then the I2C bus is assumed to be idle and the controller can generate a START condition • When Bus Idle Timeout is written 0, the Bus Idle Timeout is disabled

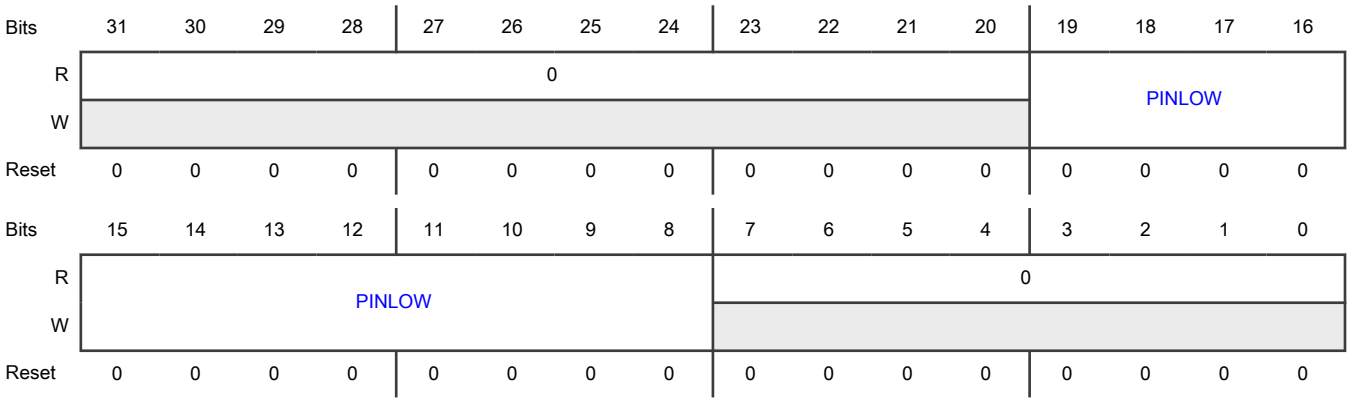
48.5.1.11 Controller Configuration 3 (MCFGR3)

Offset

Register	Offset
MCFGR3	2Ch

Function
Should only be written when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 PINLOW	Pin low timeout Configures the pin low timeout flag in clock cycles. • If SCL or, either SCL or SDA (selected by MCFGR1[TIMECFG]) is low for longer than (PINLOW * 256) cycles, then MSR[PLTF] becomes 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• When 0, the pin low timeout feature is disabled
7-0 —	Reserved

48.5.1.12 Controller Data Match (MDMR)

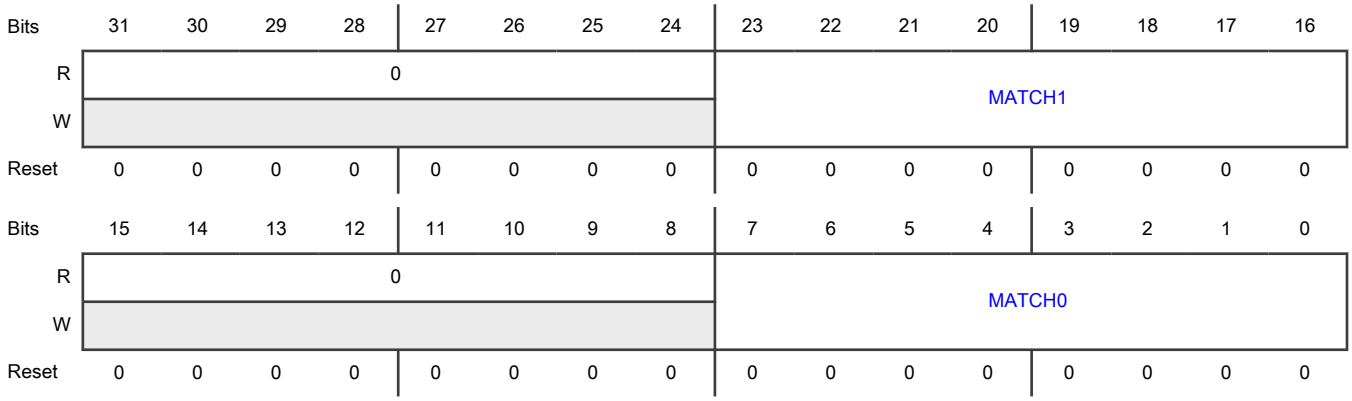
Offset

Register	Offset
MDMR	40h

Function

Should only be written when the I2C controller is disabled or idle.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Compared against the received data when receive data match is enabled.
15-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

48.5.1.13 Controller Clock Configuration 0 (MCCR0)

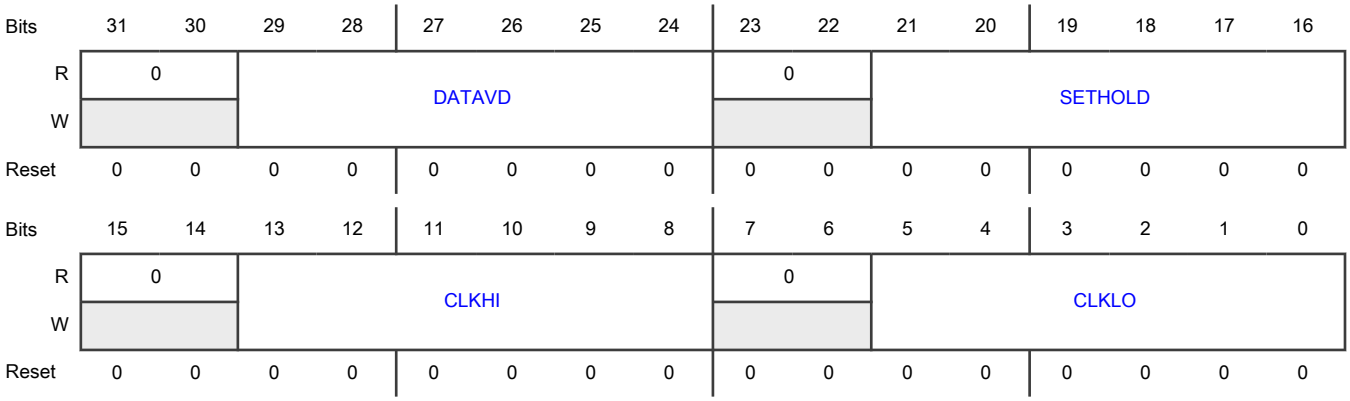
Offset

Register	Offset
MCCR0	48h

Function

Cannot be changed when the I2C controller is enabled and is used for standard, fast, fast-mode plus and ultra-fast transfers.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23-22 —	Reserved
21-16	Setup Hold Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
SETHOLD	Minimum number of cycles (minus one) that is used by the controller for these conditions: • Hold time for a START • Setup and hold time for a repeated START • Setup time for a STOP The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the controller. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the controller. The Clock Low Period value is also used for the minimum bus free time between a STOP and a START condition. This period is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

48.5.1.14 Controller Clock Configuration 1 (MCCR1)

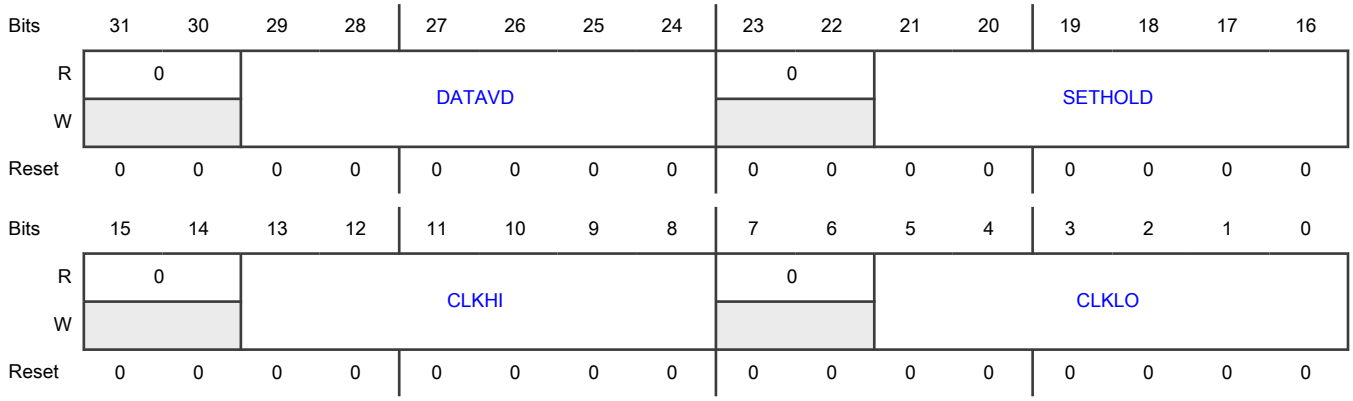
Offset

Register	Offset
MCCR1	50h

Function

Cannot be changed when the I2C controller is enabled and is used for high-speed-mode transfers. The separate clock configuration for high-speed mode allows arbitration to take place in Fast mode (with timing configured by Controller Clock Configuration 0 (MCCR0)), before switching to high-speed mode (with timing configured by MCCR1).

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured to be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the controller for these conditions: • Hold time for a START condition • Setup and hold time for a repeated START condition • Setup time for a STOP condition The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the controller. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the controller. The CLKLO value is also used for the minimum bus free time between a STOP and a START condition. This period is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

48.5.1.15 Controller FIFO Control (MFCR)

Offset

Register	Offset
MFCR	58h

Function

Used only in stop mode when the MFCR register is static (that is, the MFCR register is not changing).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														RXWATER	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														TXWATER	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark The Receive Data Flag (SSR[RDF]) becomes 1 whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal to or greater than the FIFO size truncates the value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark The Transmit Data Flag (SSR[TDF]) becomes 1 when the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal to or greater than the FIFO size truncates the value.

48.5.1.16 Controller FIFO Status (MFSR)

Offset

Register	Offset
MFSR	5Ch

Function

Gives the number of words in the transmit and receive FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													RXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													TXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15-3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2-0 TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

48.5.1.17 Controller Transmit Data (MTDR)

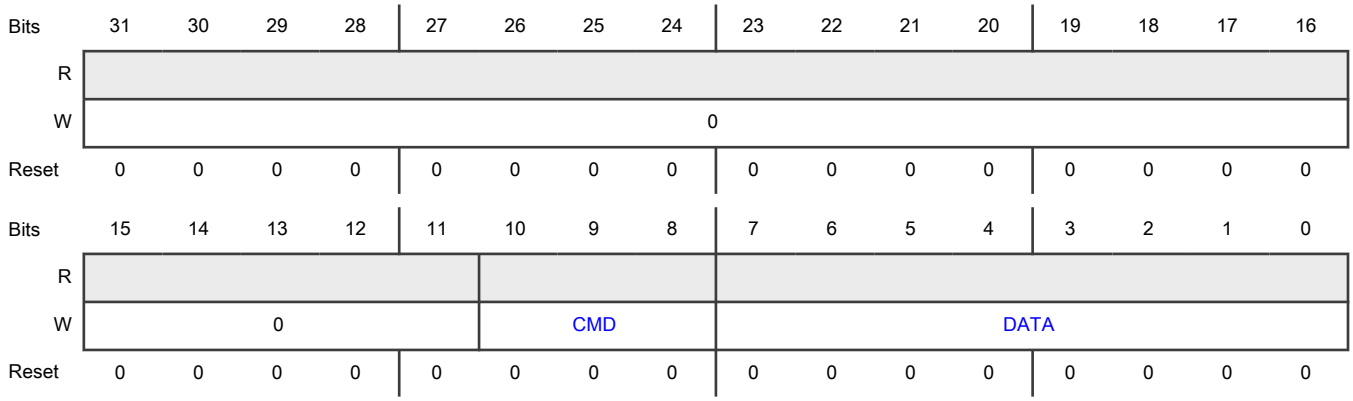
Offset

Register	Offset
MTDR	60h

Function

- An 8-bit write to the CMD field is ignored and does not increment the FIFO write pointer.
- An 8-bit write to the DATA field zero-extends the CMD field and increments the FIFO write pointer.
- A 16-bit or 32-bit writes both the CMD and DATA fields and increments the FIFO write pointer.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Data 000b - Transmit DATA[7:0]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Receive (DATA[7:0] + 1) bytes 010b - Generate STOP condition 011b - Receive and discard (DATA[7:0] + 1) bytes 100b - Generate (repeated) START and transmit address in DATA[7:0] 101b - Generate (repeated) START and transmit address in DATA[7:0]. This transfer expects a NACK to be returned. 110b - Generate (repeated) START and transmit address in DATA[7:0] using high-speed mode 111b - Generate (repeated) START and transmit address in DATA[7:0] using high-speed mode. This transfer expects a NACK to be returned.
7-0 DATA	Transmit Data Performing an 8-bit write to DATA zero-extends the CMD field.

48.5.1.18 Controller Receive Data (MRDR)

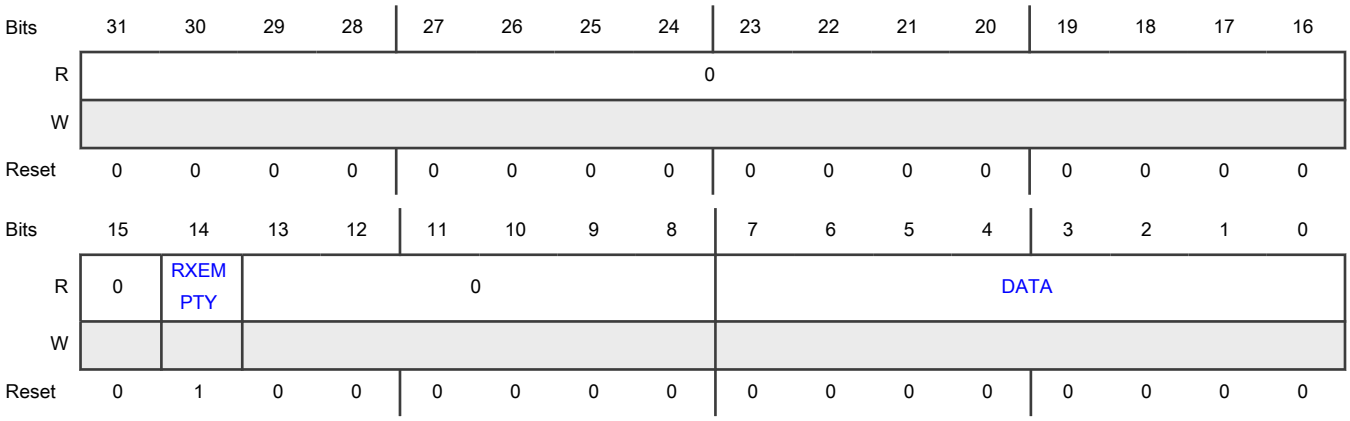
Offset

Register	Offset
MRDR	70h

Function

Reading the Receive Data register returns the data received by the I2C controller that has not been discarded.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	Receive Empty Gives the empty status of the controller receive data FIFO. 0b - Receive FIFO is not empty 1b - Receive FIFO is empty
13-8 —	Reserved
7-0 DATA	Receive Data Receive data can be discarded due to the CMD field, or the controller can be configured to discard non-matching data.

48.5.1.19 Controller Receive Data Read Only (MRDROR)

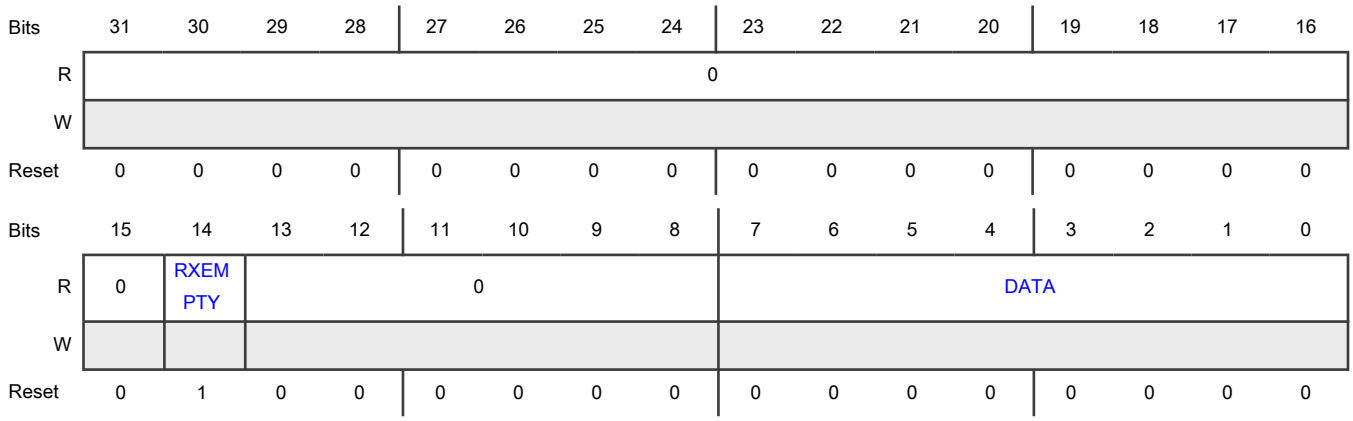
Offset

Register	Offset
MRDROR	78h

Function

Reading the Receive Data register returns the data received by the I2C controller, but does not pull the data from the FIFO.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	RX Empty Gives the empty status of the receive data FIFO. 0b - Receive FIFO is not empty 1b - Receive FIFO is empty
13-8 —	Reserved
7-0 DATA	Receive Data

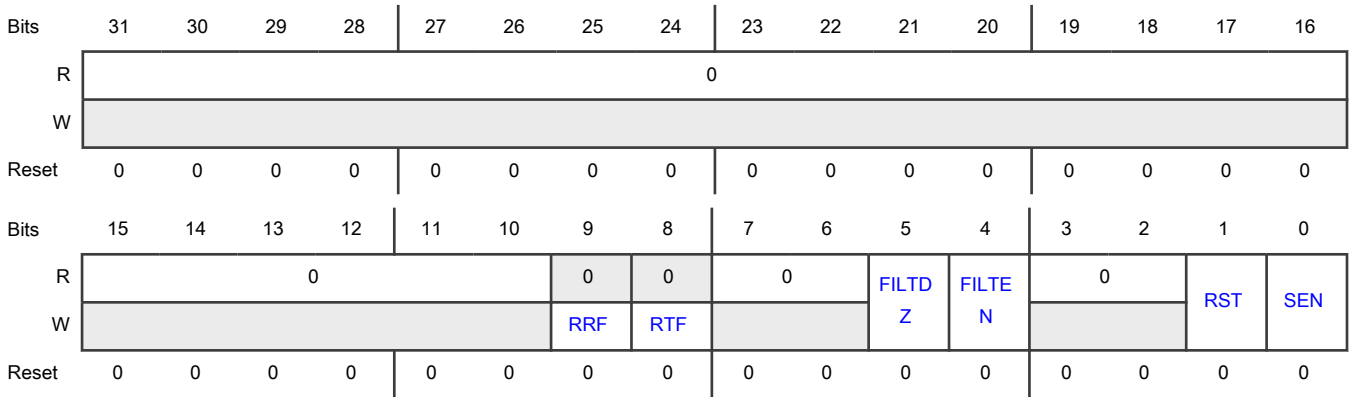
48.5.1.20 Target Control (SCR)

Offset

Register	Offset
SCR	110h

Function
Contains resets and other target control settings.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Empties the receive FIFO. 0b - No effect 1b - Receive Data Register is now empty.
8 RTF	Reset transmit FIFO Empties the transmit FIFO. 0b - No effect 1b - Transmit Data Register is now empty.
7-6 —	Reserved
5 FILTDZ	Filter doze enable FILTDZ should only be updated when the I2C target is disabled. 0b - Filter remains enabled in Doze mode 1b - Filter is disabled in Doze mode
4 FILTEN	Filter enable Should only be updated when the I2C Target is disabled. 0b - Disable digital filter and output delay counter for target mode 1b - Enable digital filter and output delay counter for target mode
3-2 —	Reserved
1 RST	Software reset Takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - Target mode logic is not reset 1b - Target mode logic is reset
0 SEN	Target Enable Enables I2C Target mode. 0b - I2C Target mode is disabled 1b - I2C Target mode is enabled

48.5.1.21 Target Status (SSR)

Offset

Register	Offset
SSR	114h

Function

Contains status flags for transmit and receive data, for error conditions, and for bus and target busy or idle status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	SBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus busy flag Indicates if an I2C bus is idle or busy. 0b - Idle 1b - Busy
24 SBF	Target busy flag Indicates if an I2C target is idle or busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15	SMBus alert response flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
SARF	• Becomes 0 by reading the Target Address Status (SASR) register. • Cannot generate an asynchronous wakeup. 0b - SMBus alert response is disabled or not detected 1b - SMBus alert response is enabled and detected
14 GCF	General call flag Indicates whether a target has detected the General Call Address. • Becomes 0 by reading the Target Address Status (SASR) register. • Cannot generate an asynchronous wakeup. 0b - Target has not detected the General Call Address or the General Call Address is disabled 1b - Target has detected the General Call Address
13 AM1F	Address match 1 flag Indicates that the received address has matched the ADDR1 field or ADDR0 to ADDR1 range as configured by SCFGR1[ADDRCFG]. • Becomes 0 by reading the Target Address Status (SASR) register. • Cannot generate an asynchronous wakeup. 0b - ADDR1 or ADDR0/ADDR1 range matching address not received 1b - ADDR1 or ADDR0/ADDR1 range matching address received
12 AM0F	Address match 0 flag Indicates that the received address has matched the ADDR0 field as configured by SCFGR1[ADDRCFG]. • Becomes 0 by reading the Target Address Status (SASR) register. • Cannot generate an asynchronous wakeup. 0b - ADDR0 matching address not received 1b - ADDR0 matching address received
11 FEF	FIFO error flag Can only become 1 when clock stretching is disabled. 0b - FIFO underflow or overflow was not detected 1b - FIFO underflow or overflow was detected
10 BEF	Bit error flag Becomes 1 if the LPI2C target transmits a logic 1 and detects a logic 0 on the I2C bus. The target ignores the rest of the transfer until the next (repeated) START condition. 0b - Target has not detected a bit error 1b - Target has detected a bit error

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 SDF	STOP detect flag Becomes 1 when the LPI2C target detects a STOP condition and if the LPI2C target matched the last address byte. When SCFGR1[SDCFG] becomes 1, this bit becomes 1 on any STOP condition. 0b - Target has not detected a STOP condition 1b - Target has detected a STOP condition
8 RSF	Repeated start flag Becomes 1 when the LPI2C target detects a repeated START condition and if the LPI2C target matched the last address byte. When SCFGR1[RSCFG] becomes 1, this bit becomes 1 on any repeated START condition. The RSF does not become 1 when the target first detects a START condition. 0b - Target has not detected a Repeated START condition 1b - Target has detected a Repeated START condition
7-4 —	Reserved
3 TAF	Transmit ACK flag Becomes 0 by writing to the Target Transmit ACK (STAR) register. 0b - Transmit ACK/NACK is not required 1b - Transmit ACK/NACK is required
2 AVF	Address valid flag Becomes 0 by reading the Target Address Status (SASR) register. When SCFGR1[RXCFG] = 1, SSR[AVF] also becomes 0 by reading the Target Receive Data (SRDR) register. 0b - Address Status Register is not valid 1b - Address Status Register is valid
1 RDF	Receive data flag Becomes 0 by reading the Target Receive Data (SRDR) register. When SCFGR1[RXCFG] = 1, this bit does not become 0 when reading the Target Receive Data (SRDR) register and if SSR[AVF] is 1. 0b - Receive data is not ready 1b - Receive data is ready
0 TDF	Transmit data flag Becomes 0 by writing the to the Target Transmit Data (STDR) register. When SCFGR1[TXCFG] = 0 and if a NACK or a Repeated START or STOP condition is detected, then this bit also becomes 0. 0b - Transmit data not requested 1b - Transmit data is requested

48.5.1.22 Target interrupt enable (SIER)

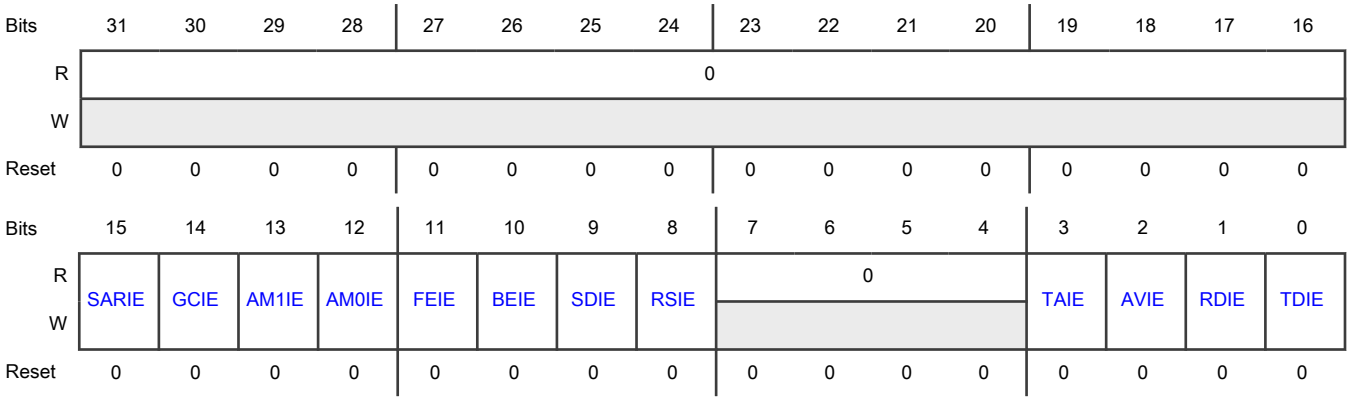
Offset

Register	Offset
SIER	118h

Function

Contains transmit and receive data interrupt enables, start and stop detect interrupt enables, and other target interrupt enables.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SARIE	SMBus alert response interrupt enable Enables interrupt for SMBus alert response. 0b - Disabled 1b - Enabled
14 GCIE	General call interrupt enable Enables interrupt for general call. 0b - Disabled 1b - Enabled
13 AM1IE	Address match 1 interrupt enable Enables interrupt for address match 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
12 AM0IE	Address match 0 interrupt enable Enables interrupt for address match 0. 0b - Disabled 1b - Enabled
11 FEIE	FIFO error interrupt enable Enables interrupt for FIFO error. 0b - Disabled 1b - Enabled
10 BEIE	Bit error interrupt enable Enables interrupt for bit error. 0b - Disabled 1b - Enabled
9 SDIE	STOP detect interrupt enable Enables interrupt for STOP detection. 0b - Disabled 1b - Enabled
8 RSIE	Repeated start interrupt enable Enables interrupt for repeated start. 0b - Disabled 1b - Enabled
7-4 —	Reserved
3 TAIE	Transmit ACK interrupt enable Enables interrupt for transmit ACK. 0b - Disabled 1b - Enabled
2 AVIE	Address valid interrupt enable Enables interrupt for valid address.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
1 RDIE	Receive data interrupt enable Enables interrupt for receive data. 0b - Disabled 1b - Enabled
0 TDIE	Transmit data interrupt enable Enables interrupt for transmit data. 0b - Disabled 1b - Enabled

48.5.1.23 Target DMA Enable (SDER)

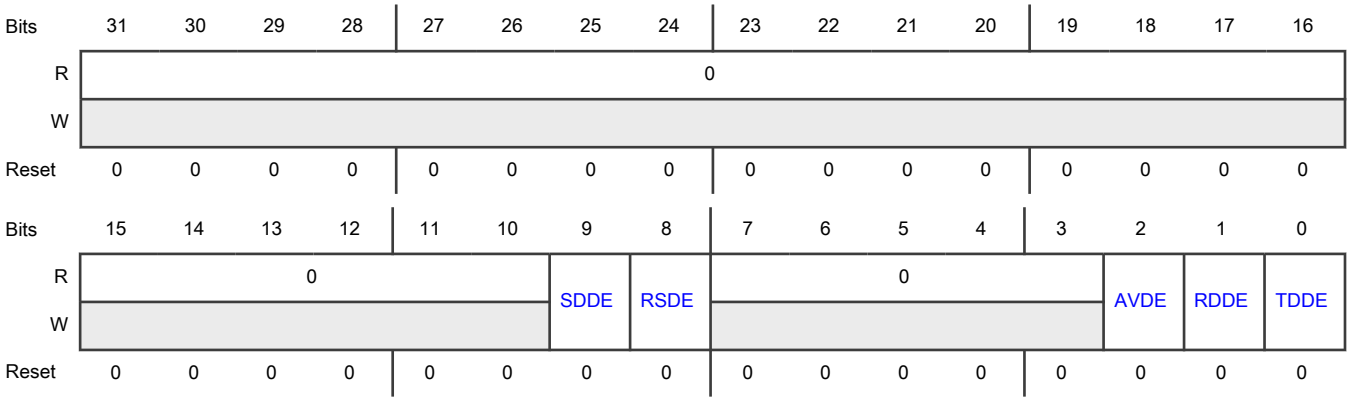
Offset

Register	Offset
SDER	11Ch

Function

Contains the transmit, request, and receive enables for DMA.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 SDDE	Stop detect DMA enable Enables DMA end-of-packet processing on stop detection. Reading the Receive Data register when the Receive Data register is empty: • Generates a DMA end-of-packet response. • Returns 0x0000_40FF. • Writes 0 to the STOP Detect Flag (MSR[SDF]) 0b - DMA request is disabled 1b - DMA request is enabled
8 RSDE	Repeated start DMA enable Enables DMA end-of-packet processing on repeated start. Reading the Receive Data register when the Receive Data register is empty: • Generates a DMA end of packet response. • Returns 0x0000_40FF. • Writes 0 to the Receive Data Flag (MSR[RSF]). 0b - DMA request is disabled 1b - DMA request is enabled
7-3 —	Reserved
2 AVDE	Address valid DMA enable The Address Valid DMA request is shared with the Receive Data DMA request. If both Address Valid DMA request and Receive Data DMA request are enabled, then set SCFGR1[RXCFG] = 1, to allow the DMA to read the address from the Target Receive Data (SRDR) register. 0b - DMA request is disabled 1b - DMA request is enabled
1 RDDE	Receive data DMA enable Enables receive data for DMA. 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit data DMA enable Enables transmit data for DMA.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - DMA request is disabled 1b - DMA request is enabled

48.5.1.24 Target Configuration 0 (SCFGR0)

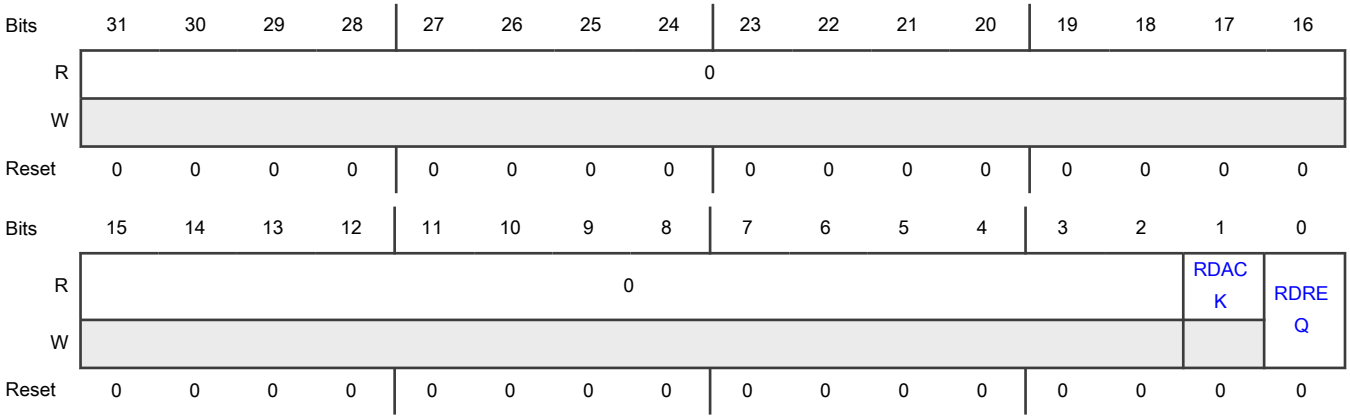
Offset

Register	Offset
SCFGR0	120h

Function

Configures the read request feature

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDACK	Read Acknowledge Indicates whether the read request was acknowledged by a START then STOP sequence with one SCL pulse in-between while RDREQ was asserted. This status flag becomes 0 by writing 0 to RDREQ. 0b - Read Request not acknowledged 1b - Read Request acknowledged
0	Read Request

Table continues on the next page...

Table continued from the previous page...

Field	Function
RDREQ	Writing 1 to this bit when the I2C bus is idle causes the LPI2C target to drive SDA low, triggering a START condition. The LPI2C target releases SDA on the next falling edge of SCL or if software writes 0 to the Read Request configuration bit. Initiating a second read request (for example following I2C bus activity that did not acknowledge the request) requires software to write 0 and then 1 to this bit. Writing this register when the I2C bus is busy always writes 0 to this bit. 0b - Disabled 1b - Enabled

48.5.1.25 Target Configuration 1 (SCFGR1)

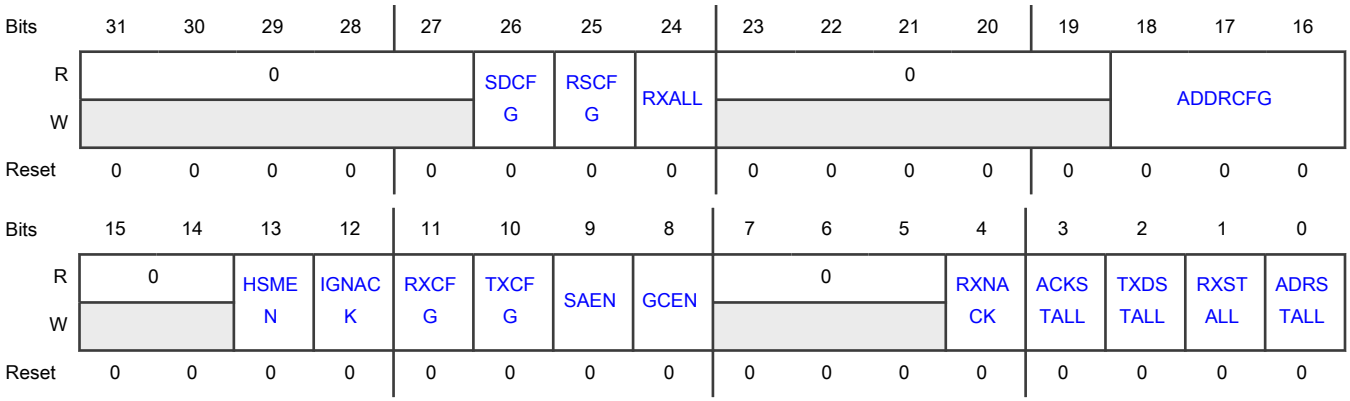
Offset

Register	Offset
SCFGR1	124h

Function

Should only be written when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26	Stop Detect Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
SDCFG	Configures the conditions under which the stop detect flag becomes 1. 0b - Any STOP condition following an address match 1b - Any STOP condition
25 RSCFG	Repeated start configuration Configures the conditions under which the repeated start flag becomes 1. 0b - Any repeated START condition following an address match 1b - Any repeated START condition
24 RXALL	Receive all When enabled, the LPI2C target stores all addresses on the I2C bus in the Address Status Register and all data on the I2C bus in the Receive Data Register. However, the LPI2C target does not drive SDA (for ACK or target-transmit transfer) unless there was an address match. The LPI2C target can drive SCL if clock stretching is enabled. When RXALL is 1, only 7-bit addressing modes are supported by the LPI2C target (ADDRCFG must be configured for 7-bit address match) although 10-bit addresses can be supported in software. The first byte of a 10-bit address is saved to the address match register and the second byte is saved as the first data byte in the receive data register. Receive all is intended to aid debugging of the I2C bus by saving all data on the bus without altering the state of the bus. When RXALL is 1, it is recommended to also write 1 to RSCFG and SDCFG so that software can track the full state of the I2C bus. 0b - Receive all disabled 1b - Receive all enabled
23-19 —	Reserved
18-16 ADDRCFG	Address configuration Configures the condition that causes an address to match. 000b - Address match 0 (7-bit) 001b - Address match 0 (10-bit) 010b - Address match 0 (7-bit) or Address match 1 (7-bit) 011b - Address match 0 (10-bit) or Address match 1 (10-bit) 100b - Address match 0 (7-bit) or Address match 1 (10-bit) 101b - Address match 0 (10-bit) or Address match 1 (7-bit) 110b - From Address match 0 (7-bit) to Address match 1 (7-bit) 111b - From Address match 0 (10-bit) to Address match 1 (10-bit)
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13 HSMEN	High-speed mode enable Enables detection of the high-speed mode controller code of target address 0000_1XX, but does not cause an address match on this code. When this field is 1 and any high-speed mode controller code is detected, SCR[FILTEN] and SCFGR1[ACKSTALL] bits are ignored until the next STOP condition is detected. 0b - Disable 1b - Enable
12 IGNACK	Ignore NACK When Ignore NACK is 1, the LPI2C target continues transfers after a NACK is detected. Ignore NACK bit is required to be 1 in Ultra-Fast mode. 0b - Target ends transfer when NACK is detected 1b - Target does not end transfer when NACK detected
11 RXCFG	Receive Data Configuration 0b - Reading the Receive Data register returns received data and writes 0 to the Receive Data flag. See MSR[RDF]. 1b - Reading the Receive Data register when the Address Valid flag (SSR[AVF]) is 1, returns the Address Status register and writes 0 to SSR[AVF]. Reading the Receive Data register when SSR[AVF] is 0, returns received data and writes 0 to the Receive Data flag (MSR[RDF]). See SSR[AVF] and MSR[RDF].
10 TXCFG	Transmit flag configuration Always becomes 1 before a NACK is detected at the end of a target-transmit transfer. This change can cause an extra word to be written to the transmit data FIFO. • When 0, the Transmit Data register is automatically emptied when a target-transmit transfer is detected. This setting causes the transmit data flag to become 1 whenever a target-transmit transfer is detected, and causes the transmit data flag to become 0 at the end of the target-transmit transfer. • When 1, MSR[TDF] becomes 1 whenever the Transmit Data register is empty, and MSR[TDF] becomes 0 when the Transmit Data register is full. This setting allows the Transmit Data register to be filled before a target-transmit transfer is detected, but can cause the Transmit Data register to be written before a NACK is detected on the last byte of a target-transmit transfer. 0b - MSR[TDF] becomes 1 only during a target-transmit transfer when the Transmit Data register is empty 1b - MSR[TDF] becomes 1 whenever the Transmit Data register is empty
9 SAEN	SMBus alert enable Enables a match on an SMBus alert. 0b - Disables match on SMBus Alert 1b - Enables match on SMBus Alert

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 GCEN	General call enable Enables a general call address. 0b - General Call address is disabled 1b - General Call address is enabled
7-5 —	Reserved
4 RXNACK	Receive NACK When 1, the LPI2C target responds with a NACK under the following conditions: • SSR[AVF] should become 1 due to matching an address, but the flag is already 1 (address overrun). • SSR[RDF] should become 1 due to receiving data, but the flag is already 1 (receive data overrun). 0b - ACK/NACK always written 1 by TXNACK 1b - NACK always generated on address overrun or receive data overrun, otherwise ACK/NACK is written 1 by TXNACK.
3 ACKSTALL	ACK SCL stall Enables SCL clock stretching during target-transmit address byte(s) and target-receiver address and data byte(s), to allow software to write the Target Transmit ACK (STAR) register before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the ninth bit, and is therefore not compatible with high-speed mode. When enabled, there is no need to write 1 to either RX SCL Stall (SCFGR1[RXSTALL]) or Address SCL Stall (SCFGR1[ADRSTALL]). When ACKSTALL is enabled and there is an address match on the first byte of a 10-bit address, SSR[AVF] becomes 1, allowing software to read the Received Address before writing to the Target Transmit ACK (STAR) register. 0b - Disabled 1b - Enabled
2 TXDSTALL	Transmit data SCL stall Enables SCL clock stretching when the SSR[TDF] = 1 during a target-transmit transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with high-speed mode. 0b - Disabled 1b - Enabled
1 RXSTALL	RX SCL stall Enables SCL clock stretching when SSR[RDF] = 1 during a target-receive transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with high-speed mode. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 ADRSTALL	Address SCL stall Enables SCL clock stretching when SSR[AVF] = 1. Clock stretching only occurs following the ninth bit, and is therefore compatible with high-speed mode. 0b - Disabled 1b - Enabled

48.5.1.26 Target Configuration 2 (SCFGR2)

Offset

Register	Offset
SCFGR2	128h

Function
Should only be written when the I2C target is disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DATAVD						0				CLKHOLD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch filter SDA Configures the I2C target digital glitch filters for SDA input. • A configuration of 0 disables the glitch filter.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Glitches equal to or less than FILTSDA cycles long are filtered out and ignored. • The latency through the glitch filter is equal to FILTSDA + 3 cycles, and must be configured to be less than the minimum SCL low or high period. • The glitch filter cycle count is not affected by MCFGR1[PRESCALE], and the glitch filter cycle count is disabled in high-speed mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch filter SCL Configures the I2C target digital glitch filters for SCL input. • A configuration of 0 disables the glitch filter. • Glitches equal to or less than FILTSCL cycles long are filtered out and ignored. • The latency through the glitch filter is equal to FILTSCL + 3 cycles, and must be configured to be less than the minimum SCL low or high period. • The glitch filter cycle count is not affected by MCFGR1[PRESCALE], and the glitch filter cycle count is disabled in high-speed mode.
15-14 —	Reserved
13-8 DATAVD	Data valid delay Configures the SDA data valid delay time for the I2C target, which is equal to FILTSCL+DATAVD+3 cycles. • The data valid delay must be configured to be less than the minimum SCL low period. • The I2C target data valid delay time is not affected by MCFGR1[PRESCALE], and the I2C target data valid delay time is disabled in high-speed mode.
7-4 —	Reserved
3-0 CLKHOLD	Clock hold time Configures the minimum clock hold time for the I2C target, when clock stretching is enabled. • The minimum hold time is equal to CLKHOLD + 3 cycles. • The I2C target clock hold time is not affected by MCFGR1[PRESCALE], and the I2C target clock hold time is disabled in high-speed mode.

48.5.1.27 Target Address Match (SAMR)

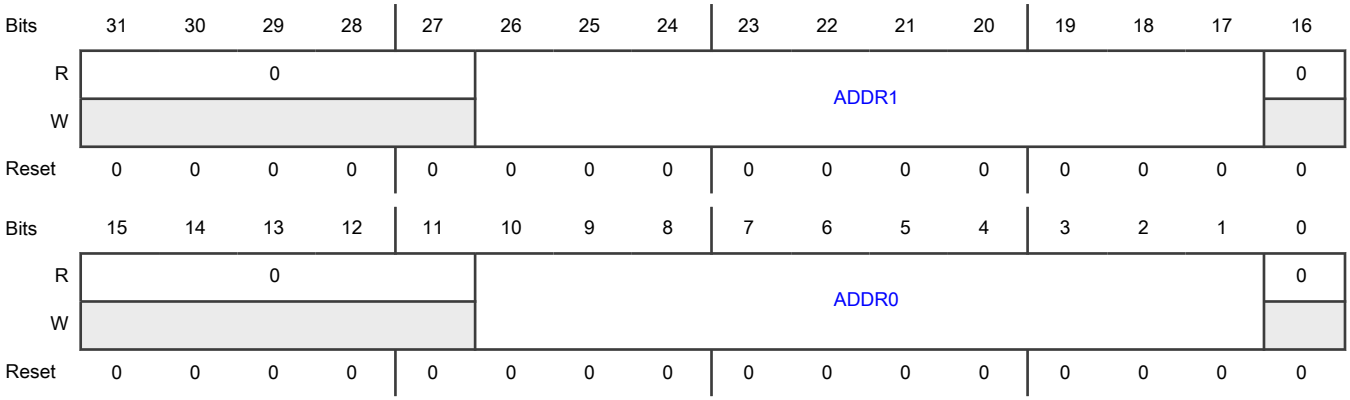
Offset

Register	Offset
SAMR	140h

Function

Should only be written when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 value Compared against the received address to detect the Target Address. • In 10-bit mode, the first address byte is compared to { 11110, ADDR1[26:25] } and the second address byte is compared to ADDR1[24:17]. • In 7-bit mode, the address is compared to ADDR1[23:17].
16-11 —	Reserved
10-1 ADDR0	Address 0 value Compared against the received address to detect the Target Address. • In 10-bit mode, the first address byte is compared to { 11110, ADDR0[10:9] } and the second address byte is compared to ADDR0[8:1]. • In 7-bit mode, the address is compared to ADDR0[7:1].

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

48.5.1.28 Target Address Status (SASR)

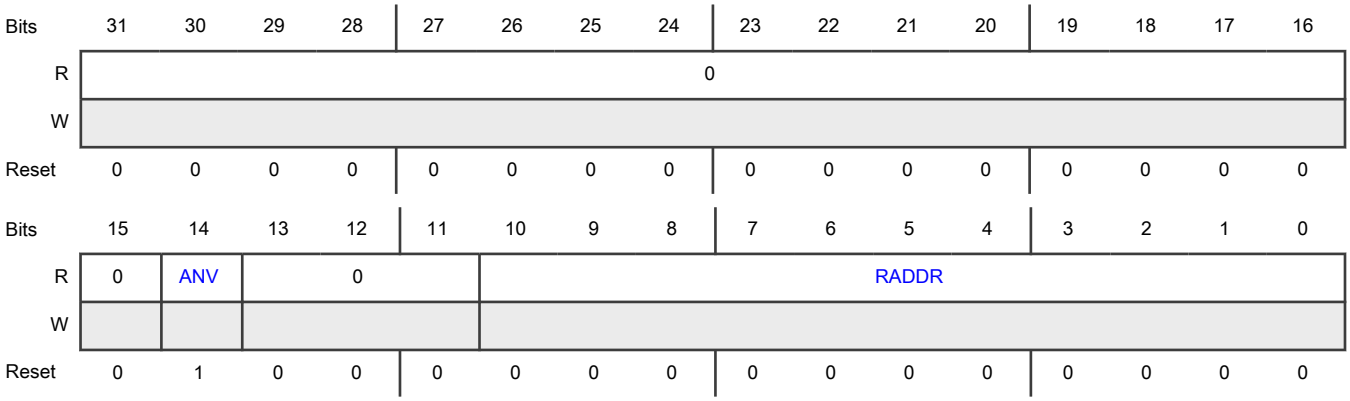
Offset

Register	Offset
SASR	150h

Function

Contains the read-only RADDR and ANV fields.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 ANV	Address not valid 0b - Received Address (RADDR) is valid 1b - Received Address (RADDR) is not valid
13-11 —	Reserved
10-0	Received Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
RADDR	Updates whenever SSR[AM0F] or SSR[AM1F] becomes 1. SSR[AM0F] or SSR[AM1F] become 0 by reading the Target Address Status register. • In 7-bit mode, the address byte is stored in RADDR[7:0]. • In 10-bit mode, the first address byte is { 11110, RADDR[10:9], RADDR[0] } and the second address byte is RADDR[8:1]. The R/W bit is therefore always stored in RADDR[0]. • When SCFGR1[ACKSTALL] is 1, if the first address byte matches in 10-bit mode, then the first address byte is stored in RADDR[7:0] so software can read the Received Address before writing the Transmit ACK. The Received Address then updates with the full 10-bit address if the second address byte matches.

48.5.1.29 Target Transmit ACK (STAR)

Offset

Register	Offset
STAR	154h

Function

Can only be written when the ACK SCL Stall bit (SCFGR1[ACKSTALL]) is 1.

- SCFGR1[ACKSTALL] enables clock stretching during the ACK/NACK bit slot, and during this time, the STAR register can be written by software.
- The logic ensures that the clock stretching continues for at least one bus clock cycle after the STAR register is updated.
- This clock stretching time can be extended using the Clock Hold Time field (SCFGR2[CLKHOLD]).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TXNACK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK After receiving each word, software can transmit either an ACK (logic 0) or a NACK (logic 1). This bit field selects which to use: ACK or NACK. • When SCFGR1[ACKSTALL] is 1, a Transmit NACK must be written once for each matching address byte and each received word. ACKSTALL must be 1, because that setting stalls the data transfer until software reads the received word (and decides whether to respond with an ACK or NACK). • To configure the default ACK/NACK, Transmit NACK can also be written when LPI2C target is disabled or idle. 0b - Write a Transmit ACK for each received word 1b - Write a Transmit NACK for each received word

48.5.1.30 Target Transmit Data (STDR)

Offset

Register	Offset
STDR	160h

Function

Clock stretching (enabled or disabled) affects when the transmit data is transferred. The SCFGR1[TXDSTALL] bit enables clock stretching during the first data bit of a target-transmit transfer.

- **If clock stretching is enabled (SCFGR1[TXDSTALL] = 1)**, then the transmit data transfer is stalled until the STDR is updated. Clock stretching is extended by at least 1 bus clock cycle after STDR is updated, and clock stretching can be delayed even more using the Clock Hold Time field (SCFGR2[CLKHOLD].
- **If clock stretching is disabled (SCFGR1[TXDSTALL] = 0)**, then the transmit data should be written before the start of the target-transmit transfer. If the transmit data is not written before the start of the target-transmit transfer, the FIFO Error Flag (SSR[FEF]) becomes 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	0								DATA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Transmit data Writing to the Target Transmit Data Register (STDR) stores I2C target transmit data in the Target Transmit Data Register

48.5.1.31 Target Receive Data (SRDR)

Offset

Register	Offset
SRDR	170h

Function

Reading the Target Receive Data Register returns the data received by the I2C target.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SOF	RXEM PTY	0		RADDR				DATA							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 SOF	Start of frame Indicates whether this data word is the first data word. 0b - Not the first data word since a (repeated) START or STOP condition 1b - Is the first data word since a (repeated) START or STOP condition
14 RXEMPTY	Receive empty Gives the empty status of the receive data register. 0b - The Receive Data Register is not empty 1b - The Receive Data Register is empty
13-11 —	Reserved
10-8 RADDR	Received address When both SCFGR1[RXCFG] and SSR[AVF] are 1, the Address Status Register (SASR[RADDR])bits [10:8] are returned. Otherwise this field returns zero.
7-0 DATA	Receive data Contains data received by the I2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, the Address Status Register (SASR[RADDR]) bits [7:0] are returned.

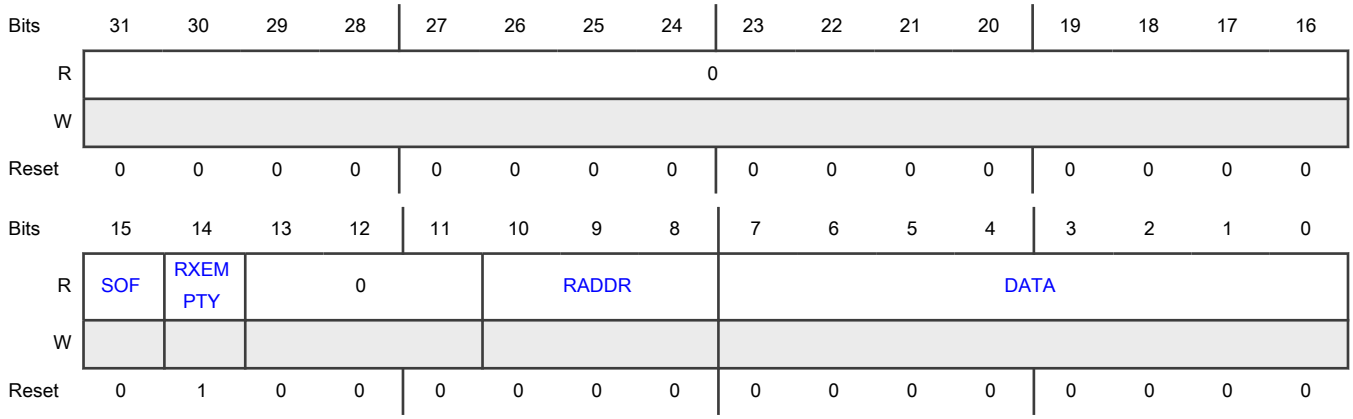
48.5.1.32 Target Receive Data Read Only (SRDROR)

Offset

Register	Offset
SRDROR	178h

Function
Reading the Target Receive Data Register returns the data received by the I2C target, but does not pull the data from the register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SOF	Start of frame 0b - Is not the first data word since a (repeated) START or STOP condition 1b - Is the first data word since a (repeated) START or STOP condition
14 RXEMPTY	Receive empty 0b - The Receive Data Register is not empty 1b - The Receive Data Register is empty
13-11 —	Reserved
10-8 RADDR	Received address When both SCFGR1[RXCFG] and SSR[AVF] are 1, the Address Status Register (SASR[RADDR]) bits [10:8] are returned. Otherwise this field returns zero.
7-0 DATA	Receive data When both SCFGR1[RXCFG] and SSR[AVF] are 1, the Address Status Register (SASR[RADDR]) bits [7:0] are returned.

Chapter 49
Improved Inter-Integrated Circuit (I3C)

49.1 Chip-specific I3C information

Table 296. Reference links to related information

Topic	Related module	Reference
Full description	I3C	I3C
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

49.1.1 Module instances

This device has one instance of the I3C module, I3C0.

49.1.2 PORT module configuration for I3C mode pull-up resistor control

In order to work in the I3C mode, the PV (pull value), PE (pull enable) and PS (pull select) values in the corresponding PORT_PCR registers should be correctly configured in advance.

The following table lists the required PORT_PCR register bit fields configuration of corresponding I3C pins to support I3C pull-up resistor control.

Table 297. PORT module configuration to support different I3C pull-up resistor control

I3C pins	Weak pull-up resistor enable	Strong pull-up resistor enable
I3C0_SDA	Corresponding PCRx[PE] = 1, PCRx[PV] = 0, PCRx[PS] = 1	Corresponding PCRx[PE] = 1, PCRx[PV] = 1, PCRx[PS] = 1
I3C0_SCL	Corresponding PCRx[PE] = 1, PCRx[PS] = 1	–

49.1.3 DMA wake-up support in I3C mode

In order to support I3C mode DMA wake-up, the CMC_CKCTRL[CKMODE] should be configured as 0x0 (No clock gating), 0x1 (Core clock is gated), or 0x3 (Core and platform clocks are gated).

49.2 Overview

The MIPI Alliance Improved Inter-Integrated Circuit (MIPI I3C) brings major improvements in use and power over I2C, and provides an alternative to SPI for mid-speed applications. The I3C bus is designed to support future sensor interface architectures, widely expected in Internet-of-Things applications.

The I3C bus is intended to be used by microcontrollers (MCU) and application processors (AP) to connect to sensors, actuators, and other MCUs (as slaves). Connecting an MCU to other MCUs and connecting an AP to an MCU are considered to be the major use cases.

The I3C bus protocol supports:

- In-band interrupts: interrupts can go from Slave to Master without extra wires, such that the Master knows which Slave sent the interrupt.
- Common Command Codes (CCC)

- Dynamic addressing
- Multi-master / multi-drop
- Hot-Join
- I2C compatibility

49.2.1 Features

I3C module:

- 2-wire multi-drop bus capable of 12 MHz clock speeds, with up to 11 devices.
  - Uses standard pads (I2C uses special pads) with 4 mA drive
  - Slave addresses are dynamically assigned, and slaves do not require a static address. However, slaves may have an I2C static address assigned at start-up, so that the slave can operate naturally on an I2C bus.
  - Slaves may use the inbound SCL clock as the peripheral clock, which means that devices can have slow or inaccurate clocks internally.
  - Simple slaves can have no internal clock; for example, like a temperature sensor.
  - For reads from a slave, the slave normally ends the read, but the I3C master can also terminate the read. For I2C and SPI, the master must know the length of the data that is being read, but for I3C the master does not have to know the length of the data being read.
- In-Band interrupts (IBI) are allowed, which allow slaves to notify a master.
  - Can be both equivalent to a separate GPIO, but can also be directly data-bearing.
  - In-Band interrupts are prioritized, so that if multiple Slaves want to interrupt a master at the same time, then the order is resolved. dynamic addresses are used to establish the priority of the slaves, which means that the master controls the priority of the slaves.
  - Interrupts can be started even when the master is not active on the bus, and yet no free-running clock is needed (for that situation).
  - A time-stamping option allows the resolution of an initial event, not just when an interrupt gets through
- Built-in commands are in separate space, so that these commands do not collide with normal Master to Slave messages.
  - Controls bus behavior, modes and states, low-power state, inquiries, and more
  - Has additional room for new built-in commands to be used by other groups
- Organized forms of multi-master modes:
  - Secondary masters that can use clean handoffs between different Masters
- Hot-join onto I3C bus allows devices to connect to the bus later than when the bus starts.
  - Enables a device or module to get onto the I3C bus, because it woke up after power-up, or was physically inserted onto the I3C bus.
  - Also provides a clean method for notification when new devices or modules get onto the I3C bus.
- Capable of using both I2C and I3C buses:
  - I3C supports specific legacy I2C devices on the bus
  - I3C Slave devices can operate on I2C buses
  - Supports bridging to I2C, SPI, UART, and other buses
- Higher data rate modes are optionally available
  - Only the master and the specific slave have to support the higher data rate; the other slaves are able to ignore the higher data rate.

  - Has an HDR-DDR mode (High Data Rate - Double Data Rate), which is about 2x the data rate of SDR (so about 20 Mbps)

- Slaves can be as small as 2 K gates (or less), which allows the I3C to be fully state-machine driven, as well as using a processor (to control the I3C).
- Masters can be as small as 2.5 K gates, but rely on the processor to handle the I3C.

The I3C peripheral supports the full I3C feature set, except for the ternary data rates (HDR-TSP, HDR-TSL).

49.3 Functional description

The following sections describe functional details of the I3C module.

49.3.1 Operating modes

This section describes all functional operation modes of the I3C module.

49.3.1.1 Master vs. slave roles for I3C

The I3C protocol defines 5 roles for devices on the I3C bus:

- **Main Master**: controls the I3C bus
- **Secondary Master**: controls the I3C with permission from the Main Master, and returns control of the bus back to the Main Master when it (Secondary Master) has finished with its tasks
- **Slave**: responds to commands from any I3C master
- **Peer-to-Peer Slave**: can read or write to other peer-to-peer slaves without any help from the I3C master
- **I2C Slave**: responds to commands from any I3C master

The I3C peripheral contains both a master and slave component, and can be parameterized to be either master or slave, or both master and slave. However, if the I3C is chosen to be a master, then the I3C peripheral should generally support slave mode, to make handoffs to multiple masters easier to do.

In general, any I3C master should be able to be a master or a slave, because a master becomes a slave when giving over control to another master. The only exception to this rule would be when using a point-to-point master or when using a master that never shares mastership.

Master requirements

Master mode uses software that supports the various requirements of the master (including as a Secondary Master):

- Special handling of Enter Dynamic Address Assignments (ENTDAA): assigning dynamic addresses to each slave. This is a process that is supported by the I3C peripheral, but requires the software to make choices.
- Building up a table of slaves and their capabilities, which is used to control what kinds of actions and commands may be sent to the slaves.
- Handling requests such as In-band interrupts, peer-to-peer slave, and master request (handoff).
- Adjusting clock speed or write-to-read timing to match the slave's limitations. This can be done in hardware using dividers and uneven duty cycles, but the software must make the decisions.
- Adjusting the maximum data length.
- Being a slave after a master handoff to another master

Additionally, a master needs 3 pins normally, unless a very precise, high strength pullup is included in the SDA pad. The 3-pin model allows 1 pad/pin to enable a system-provided pullup, which is sized to system requirements.

The master needs a reasonably accurate clock, normally capable of a frequency that is a multiple of a frequency between 11 MHz and 12.5 MHz. For example:

- 24 MHz (multiple of 12 MHz)
- 48 MHz (multiple of 12 MHz)
- 25 MHz (multiple of 12.5 MHz)
- 50 MHz (multiple of 12.5 MHz)
- 33 MHz (multiple of 11 MHz)
- 66 MHz (multiple of 11 MHz)

Higher ratios can also be used.

Slave requirements

The I3C slave mode is intended to allow a much simpler module to be used, allowing the device implementation to use fewer logic circuits when space on the device is limited. For example, when the slave minimum requirements are very small (just I3C SDR mode and a few Common Command Code (CCC) commands). Conversely, slave functionality in the module logic can be increased, increasing the module's area on the device, but saving software development costs. What is important is that the Slave can be scaled for system requirements, by doing more in software or by doing more in hardware. This decision is made when the module is integrated into the device.

I3C slave acting as a normal I2C slave on I3C buses

The I3C slave, if assigned an I2C static address, acts like a normal I2C device when the I3C slave first comes up. If the I3C slave is placed on an I2C bus with an I2C master, then the I3C slave will simply stay in I2C mode and operate normally. The software knows that the I3C slave is in I2C mode, because:

- There has not been a DACHG indicating that a dynamic address was assigned
- The SDYNADDR register contains 0x0

If full I2C support for Fast Mode (FM) and Fast Mode Plus (FM+) is desired, then the pads should also support a 50ns spike filter, which must be turned off when the I3C 7'h7E broadcast address is seen (indicating an I3C master). Turning off the spike filters can be done by hardware (using the raw net indicating that the address was seen) or can be done by software. A 50ns spike filter is not needed for the I3C to operate on an I2C bus; therefore the spike filter is not a requirement for the I3C peripheral.

How a slave re-joins the I3C bus

When a slave powers up or after a hard reset, when the slave tries to rejoin the I3C bus, the slave needs a new Dynamic Address (DA). The slave provides ways to rejoin the I3C bus:

- If the dynamic address is lost, then Hot-Join is used.
- If the dynamic address is retained in the peripheral (for example, with SRFFs), then the SCONFIG.OFFLINE bit is used (see below).
- If the dynamic address is retained in separate always-on memory, then the SDYNADDR register (as a writable mechanism) can be used with the SCONFIG.OFFLINE bit.

When SLVENA is set to 1, the OFFLINE bit of the SCONFIG register may be set. This causes the peripheral to safely rejoin the bus, by ensuring that the I3C bus is not in High Data Rate (HDR) mode, using the same approach as S0/S1 exit does: waiting for the first of the HDR Exit Pattern, or waiting for 60 µs of SCL and SDA lines not changing.

After using OFFLINE, the I3C peripheral still cannot safely use IBI until it sees a STOP (SSTATUS register), which ensures that the next START is safe to use for IBI).

If the application needs to do an IBI right away, then the application should wait for the first STOP (SSTATUS register), or should wait for 200 µs of SCL and SDA lines being High. This can be done using the SSTATUS and SINTSET controls.

- If the SSTATUS indicates that the bus is not busy and the peripheral interrupts on START or STOP, then use a timer to measure 200 µs. If the timer goes off with no START or STOP, then it is safe to use IBI.
- If a START causes an interrupt, then the timer should be turned off and the application should wait for a STOP.

- If a STOP causes an interrupt, then it is safe to use IBI.

49.3.1.2 Using the I3C Master for I2C and I3C

The I3C Master operates with a set of built-in capabilities mixed with the application flow:

- Built-in Enter Dynamic Address Assignment (ENTDAA) mechanism to simplify assignment of DAs to slaves (when Set Dynamic Address from Static Address (SETDASA) is not available).
- Request for START+Address with IBI support, including both I2C and I3C modes
- SDR Write flow via FIFO, with automatic parity in I3C and ACK/NACK detect in I2C
- SDR Read flow via the FIFO, with an automatic NACK generator for I2C, and optional use of a terminate generator for I3C
- Request for repeated START+Address or STOP when finished with previous, including both I2C and I3C
- Auto-IBI mode which responds immediately to a Slave-initiated IBI; can be used when in sleep or deep sleep
- Special message mode for SDR and DDR to simplify for DMA use
- All SCL, SDA, Pull-up, and High-Keeper controls are automated
- I2C and I3C Frequency and duty cycle configurations from functional clock

Note the MCONFIG register and the MIBIRULES register should be configured before using the Master.

49.3.2 Operations

This section describes the I3C module's operations.

49.3.2.1 Reading and writing I2C messages using normal method

The normal method:

1. Set up interrupts. Normally:
    a. MCTRLDONE: optional
    b. COMPLETE: to fire when end of data
    c. RXPEND if read; this can be used with the MDATACTRL register to set the FIFO trigger. The alternative is for DMA to read out bytes.
    d. TXNOTFULL if write; this can be used with the MDATACTRL register to set the FIFO trigger. The alternative is for DMA to supply bytes.
    e. IBIWON: so the software knows if an IBI has occurred
    f. MERRWARN: in case of an error
2. Write the MCTRL register with:
    a. REQUEST set to 1 (EmitStartAddr)
    b. TYPE set to 1 (I2C)
    c. IBIRESP set depending on how you want IBIs handled
    d. DIR set to 1 if Read, or set to 0 if Write
    e. ADDR set to the static address of the I2C slave
    f. If read, then RDTERM may be set for the length, or may be set as the data is read out (for example, set to 1 (with REQUEST=0) when you want it to stop next).
3. If Write: Write the MWDATAB register for each byte before the last byte, then write the MWDATABE register for the last byte.
    a. This can be done or started before step 1 **Set up interrupts**

b. If there is more data than the FIFO can hold, use the TXNOTFULL interrupt based on the trigger level, to allow the application to provide more, or use DMA.

4. If Read: wait for RXPEND, and then read out data via the MRDATAB register. DMA may also be used
5. On COMPLETE, the message may be ended with a STOP, or a new message started with a repeated START.
   a. Write MCTRL with REQUEST of 2 (EmitStop to STOP)
   b. Write MCTRL with REQUEST of 1 (EmitStartAddr to start another message).

      Note that IBI is not possible in this case.

49.3.2.2 Reading/writing I3C messages using normal methods (SDR and HDR-DDR)

The normal method for I3C is the same as for I2C with a few differences. Note that **Option 2: From stopped and not in HDR-DDR mode** below is preferred from stopped (bus free condition) when in Single Data Rate mode (SDR) [and not in High Data Rate - Double Data Rate mode (HDR-DDR)], because it allows any slave to issue an IBI, it avoids collisions with IBI addresses, and it is faster (when MSB of dynamic addresses are always 0).

1. Set up interrupts. Normally:
   - MCTRLDONE: optional
   - COMPLETE: to fire when end of data
   - RXPEND if read; this can be used with the MDATACTRL register to set the FIFO trigger. The alternative is for DMA to read out bytes.
   - TXNOTFULL if Write; this can be used with the MDATACTRL register to set the FIFO trigger. The alternative is for DMA to supply bytes.
   - IBIWON: so the software knows if an IBI has occurred
   - MERRWARN: in case of an error
2. Write the MCTRL register:
   - **Option 1:** Write the MCTRL register with:
     a. REQUEST set to 1 (EmitStartAddr) (cannot have pre-written TX data in the FIFO)
     b. TYPE set to 0 (if I3C) or set to 2 (if DDR)
     c. IBIRESP set depending on how you want IBIs handled
     d. DIR set to 1 (if Read) or set to 0 (if Write)
     e. ADDR set to the dynamic address of the I3C slave
     f. If read, then optionally RDTERM is set for the maximum length to auto-terminate.
     g. If write, then pre-writing the data (MWDATAB or MWDATAW) is preferred, to ensure that there are no time delays waiting on the data.
   - **Option 2:** From stopped and not in HDR-DDR mode, write MCTRL register with;
     a. REQUEST set to 1 (EmitStartAddr)
     b. TYPE set to 0 (I3C)
     c. IBIRESP set depending on how you want IBIs to be handled
     d. DIR set to 0
     e. ADDR set to 0x7E
     f. Wait for MCTRLDONE (for example, by interrupt), then proceed with **Option 1:**. This Option 2 method has advantages for IBIs when 7E is emitted on START (but not on repeated STARTs).

NOTE

The Option 2 method would not allow an immediate switch to read; the normal I3C process is to write index, then read data. So, if doing a blind read (no index is written), do not use Option 2 method.

3. If Write: then write the MWDATAB register for each byte before the last byte, then write the MWDATABE register for the last byte.
   - This can be done or started before step 1 [REQUEST set to 1 (EmitStartAddr)]) of **Option 1: Write the MCTRL register with:**. It should not be before **Option 2: From stopped and not in HDR-DDR mode**.
   - If there is more data than the FIFO can hold, use the TXNOTFULL interrupt based on the trigger level, to allow the application to provide more; or use DMA.
4. If Read: wait for RXPEND, and then read out data using the MRDATAB register. DMA may also be used.
5. On COMPLETE, the message may be ended with STOP, or a new message started with a repeated START.
   - Write MCTRL with REQUEST of 2 (EmitStop to STOP)
   - Write MCTRL with REQUEST of 1 (EmitStartAddr to start another message).

     Note that IBI is not possible in this case.

49.3.2.3 Sending a CCC to one or all I3C slaves

The normal Common Command Code (CCC) method is to use an I3C Write with an address of 0x7E. The 1st byte will be the CCC.

- If Broadcast, then any remaining bytes are the bytes that go with the Common Command Code (CCC). This would end with a STOP or a repeated START and 0x7E normally.
- If Direct, then only the CCC byte is sent, followed by a repeated START and the address of the I3C Slave (for SETDASA, this is its I2C Static address). This may be repeated with more repeated START and addresses until done. Then, it may end in STOP or a repeated START and 0x7E.

NOTE

Each repeated START is just a new EmitStart request. This can be chained by interrupt or pushed by message model using DMA.

49.3.2.4 In-band interrupt (IBI) handling

An in-band interrupt occurs when a slave sends its address after a START, and the slave's address is numerically lower than the address that the master sent (and any other slaves trying to send their addresses). When the master sends a 0x7E, the master will always lose, which is the intention.

The in-band interrupt (IBI) can happen unexpectedly when any new START (vs. repeated START) is emitted, but the in-band interrupt can also be in response to a slave pulling SDA low, which can happen in one of two ways:

- The slave has set the request to AutoIBI mode, and so it happens automatically (the engine emits 0x7E to allow the slave to win).
- The application gets the SLVSTART (slave START request) interrupt and so emits 0x7E (normally).

In either of these cases, the application can decide whether to:

- Always accept (ACK) the IBI
- Always reject (NACK) the IBI
- Allow the decision to be made by the application on a case-by-case basis (manual).

How the IBI is handled is configured by the MCTRL.IBIRESP field, which can be set to ACK, NACK, or manual.

- If manual is selected, then the application rewrites it when stopped, pending on an IBI. The application can ACK or NACK based on the IBI address (which is in the MSTATUS register). Note that manual mode chooses if there is an IBI byte or not when doing ACK.

- If always ACK was selected in the IBIRESP field, then the MIBIRULES register must be configured so that the engine knows if bytes will follow.
- If IBI bytes will follow (also known as IBI Mandatory byte), then the COMPLETE bit is not set when IBIWON is set, and RXPEND is set for 1 or more bytes. When the last byte is received, then the COMPLETE status is set.

49.3.2.5 Assigning dynamic addresses to I3C devices

If Dynamic Addresses (DAs) are all assigned below 0x40 (7-bit values from 0x3F down to 0x03, except where this is not allowed), then the MSB0 bit can be set in the Master In-Band Interrupt Registry and Rules register (MIBIRULES). This optimizes the START timing. When dynamic addresses are assigned, the MIBIRULES register should be programmed based on which I3C slave uses IBI bytes (known from Bus Capabilities Register (BCR)).

Any SETDASA (Set Dynamic Address from Static Address) assignments can be done using the normal Common Command Code (CCC) model (except when using the static address for the directed part).

There is a built-in mechanism to process the Dynamic Address Assignment (DAA) mode:

1. Set up interrupts. Normally:
   - MCTRLDONE: need to know when a slave has sent its ID and BCR/DCR and a new DA is needed.
   - COMPLETE: to fire when DAA done (NACKed by all slaves).
   - RXPEND to read the IDs of the slaves. The alternative is for DMA to read out bytes.
   - IBIWON: so the software knows if an IBI happened, which should not be possible normally.
   - MERRWARN: in case of an error
2. Write the MCTRL register with:
   - REQUEST set to 4 - ProcessDAA
   - IBIRESP set to IBI response, if that is possible (if not the 1st Slave assignment)
3. Wait for the MCTRLDONE interrupt, while reading in ID using the RXPEND interrupt. If MSTATUS indicates in DAA mode (STATE=5) and BETWEEN (bit 4=1), then it is waiting for a dynamic address for the slave whose ID was read in.
   - Write the dynamic address into MWDATAB using bits 6:0, such as 0x14 for dynamic address=7'h14.
   - Write MCTRL REQUEST to 4 (ProcessDAA again). This writes the dynamic address and then move on to the next dynamic address.
   - If MSTATUS indicates all is done via the COMPLETE bit being set and STATE=0, then all slaves have been assigned.
   - If MSTATUS indicates NACK after having written a new dynamic address, then the dynamic address was not accepted by the slave. The normal next step is REQUEST set to EmitStop and start over. However, it is also OK to set REQUEST to 4 (ProcessDAA again).

49.3.2.6 Using Master Message mode

Master Message mode is really intended for use with DMA, although message mode can also be used by the processor. The message mode concept is that all writes are to the same location, including the control and then the data; the read back is from an associated location.

NOTE

The data writes for message mode work exactly in the same way as the half-word access registers MWDATAH and MRDATAH.

To send a message via Single Data Rate (SDR), the following steps are used (from DMA or from processor):

1. Write to the MWMSG_SDR register with the control request, which includes the address, I2C vs. I3C, read or write, the count of bytes to process, and how to end (on STOP or ready for repeated START).

2. Process the Data:
   - If Write: then write the rest of the data to MWMSG, using the DMA trigger (or interrupt) to keep the TX FIFO full. Will stop asking when count is down to 0.
   - If Read: then a DMA trigger (or interrupt) indicates when the RX FIFO is to be read out via the MRMSG_SDR.
3. In-band interrupt (IBI) behavior is selected in the MCTRL register.
4. Use END type STOP to exit MSG mode. This can be done with the original message, with a 0 length message, or by using the MCTRL register.

To send a message via Double Data Rate (DDR), the following steps are used (from DMA or from processor):

1. Write to the MWMSG_DDR register with the control request, which includes the count of byte-pairs and how to end (on High Data Rate (HDR) Exit or ready for HDR Restart).
2. Write the 2nd control data to MWMSG_DDR, which includes the address, read or write, and the 7-bit command value.
3. Process the Data:
   - If Write: then write the rest of the data to same register, using the DMA trigger (or interrupt) to keep the TX FIFO full. Will stop asking when count is down to 0.
   - If Read: then a DMA trigger (or interrupt) indicates when the RX FIFO is to be read out.
4. In-band interrupt (IBI) behavior is selected in the MCTRL register.
5. Use END type Exit to exit DDR MSG mode. This can be done with the original message, with a 0 length message, or by using the MCTRL register.

49.3.2.7 Handing off mastership to another slave and getting it back

To hand off mastership, the master either:

- Waits for an Master Request (MR) (an IBIWON interrupt with MSTATUS indicating that an MR is using the IBITYPE field)
- Or pushes the request manually

In either case, the master sends a GETACCMST request. This is a directed GET that tells the slave that the slave is being assigned mastership (if the slave wants it). If the slave accepts (and sends back its own dynamic address in bits 7:1 and the negative parity of the dynamic address in bit 0), then a STOP should be issued and the MCONFIG.MSTENA field should be set to 2 (switching to slave mode).

To gain mastership, the slave sends an MR using the SCTRL register.

- If the MSTENA field is 2, then the GETMSTACC CCC (Common Command Code) will accept
- If the MSTENA field is not 2, then the GETMSTACC CCC will refuse

After mastership has granted, the MSTATUS.NOWMASTER bit is set. The application must enable the NOWMATER bit in the MINTSET register, so the application will be interrupted when a mastership transfer happens.

49.4 External signals

This table describes the I3C module signals.

Signal	I/O	Description
SCL	I/O	Serial clock
SDA	I/O	Serial data

Table continues on the next page...

Table continued from the previous page...

Signal	I/O	Description
PUR	O	Pull up resistance. There is internal pull up resistance on SDA, which is contolled by the I3C master. PUR can be used to actively control an external pull up resistance on SDA, if the internal pull up is not enough.

49.5 Initialization

49.5.1 Configuration initialization

The configuration is handled once (normally) when initializing the I3C module. Configuration is done using the MCONFIG register, which controls the frequencies, duty cycle, optimizations for performance, and other parameters.

Table 298. Configuration parameters used to initialize the I3C module

Configuration Parameter		Description
MSTENA	Master Enable	Determines whether the I3C peripheral starts in Master mode (ie. the “Main Master” in I3C terms) or starts in slave mode (and will switch to Master later).
HKEEP	High Keeper	Determines how the high-keeper (weak pullup) is to be implemented, depending on the device capabilities.
PPBAUD	Push-Pull BAUD	Sets the push-pull frequency as a divider from the FCLK (alternative clock fed to the peripheral). This sets the half-clock period baseline (used at least for the high time of SCL). • If FCLK = 24 MHz, then a PPBAUD = 0 yields 12 MHz (42.67 ns per half period) • If FCLK = 50 MHz, then a PPBAUD = 1 yields 12.5 MHz (20+20= 40 ns per half period)
PPLOW	Push-Pull Low	Changes the duty cycle for push-pull. It indicates how many more FCLKs to use for low. For example, a 50 MHz FCLK, a PPBAUD of 1, and a PPLOW of 1, yields 20+20= 40 ns high and 20+20+20= 60 ns low. This would be equivalent of 10 MHz SCL timing, but maintaining the 40 ns high needed, so that I2C devices do not see the high periods.
ODBAUD	Open Drain BAUD	The number of PPBAUD periods to make up one I3C open-drain half-clock baseline. For example, if PPBAUD yields 12 MHz (40 ns per PPBAUD period), then 5 PPBAUD can be used to get 200 ns. See also ODHPP for details about short high and long low.
ODHPP	Open Drain High Push-Pull	Optional field that allows for the I3C open drain to be long low and short high. The high period of SCL will be the PPBAUD period. This leaves enough time for the pull-up resistor to pull the SDA high when SCL is low, but is quick when SCL is high and there are no changes happening.
I2CBAUD	I2C Baud	Indicates how many ODBAUD periods are needed to communicate with I2C devices. For example, if ODBAUD gives 200 ns, and the goal is FM+ (Fast Mode, 1 MHz), then the sum needs to be 1 us. If odd vs. even, then I2CBAUD acts differently. For example, if I2CBAUD= 3, it will give 3 ODBAUD periods low and 2 ODBAUD periods high. • If I2CBAUD= 4, then it will give 3 ODBAUD periods for low and 3 ODBAUD periods for high.

Table continues on the next page...

Table 298. Configuration parameters used to initialize the I3C module (continued)

Configuration Parameter		Description
		• Also, if I2CBAUD= 3, you get 200*3= 600 ns and 200*2= 400 ns, with the sum 600 + 400 = 1000ns = 1us.
SKEW	Skew	The normal SDA skew from SCL is handled by using the time for the SCL to reach its pad and come back to the design. This is normally 2 ns to 5 ns (or sometimes worse). If the SKEW is too fast, then to add more delay, the SKEW allows specifying the number of FCLKs to insert.

Additonal optimizations:

- Use MIBIRULES.MSB0 to get much faster START header times. If the master application assigns all I3C Dynamic Addresses to be less than 0x40 (it does not have MSB set), then this bit (MSB0) can be set. This means that when the master emits 0x7E (broadcast) and the 1st bit is not driven low by a slave, the rest of the header can be at push-pull speeds. This is about 2x faster (or more, depending on optimizations above).
- Auto-emit 0x7E speeds up the frame when used in conjunction with MSB0, because it allows the processor to be sleeping when the frame starts automatically (in response to a slave).

49.6 Application information

This section describes applications supported by the I3C module.

49.6.1 Using registers when the System clock is much faster than the Functional clock

The Master block has 2 basic clocks:

- System clock: which controls the access into the memory-mapped registers.
- Functional clock (FCLK): which is used to generate the SCL clock rate on the I2C/I3C bus.

Because there are two clocks, requests made to MCTRL and MWMSG_xxx registers have to cross over the clock domains. This is also true of data reads and writes, but there are certain aspects of MCTRL that require consideration.

The normal use is to write MCTRL and wait for an interrupt indicating DONE and/or COMPLETE. Once that interrupt arrives, the interrupt handler likely writes a new request (like a new message or a STOP).

- **When the System clock is less than or equal to 8x faster than the FCLK**, there are no special actions needed. The servicing of the interrupt takes long enough to ensure that all clock-crossing details are settled. For example, if the FCLK is 25 MHz and the system clock is 150 MHz, there are no problems.
- **When the System clock is more than 8x faster than FCLK (or if the code is using a polling spin loop on MSTATUS)**, then special considerations may apply:
  - The MSTATUS sticky bits for DONE and COMPLETE are set, and optionally an interrupt is triggered.
  - To complete the clock crossing, the internal state goes through a handshake process.
  - If a new request posts before 2 FCLKs have completed, then the new request may be lost.
    - For example, if the FCLK is 25 MHz and the System clock is 400 MHz, then 2 FCLKs is 16 beats of the System clock, or really 17 beats (since the clocks are not aligned). If the interrupt handler is entered in under 16 clocks and the interrupt handler tries to write the MCTRL register immediately, then the new request is lost.
    - In reality, most processors need around 12 clocks to enter an interrupt handler, and then the handler code needs to read MSTATUS and make decisions, so this will not be an issue; but if the processor is very fast, then an extra delay is needed.
    - Likewise, if using a polling spin loop, then the timing from detecting MSTATUS DONE or COMPLETE to writing the next MCTRL would need to consider that time required (2 FCLKs).

49.6.2 Master registers summary

The Master registers are integrated with the Slave registers. Other than the configuration registers, the Master registers are above all of the Slave registers, starting at 0x80. The mapping is designed to work for both types of registers, and many Master registers are just aliases of Slave registers, but with different meanings.

- When Master operation is enabled (MCONFIG.MSTENA = 1), the Master registers should be used.
- When Slave operation is enabled (MCONFIG.MSTENA = 0), the Slave registers should be used.

Offset	Type	Register		Description
0x000	RW	MCONFIG	Master Configuration	Configuration for the master
0x084	RW	MCTRL	Master Control	Master main control. Starts actions on the bus (see Using Master Message mode).
0x088	RO and R/W1C	MSTATUS	Master Status	Status for the master, including interrupt causes.
0x08C	RW	MIBIRULES	Master In-Band Interrupts Register and Rules	Rules for IBI use, and registry of Slaves which use an IBI byte.
0x09C	RW	MERRWARN	Master Errors and Warnings	Error and Warning state from the misuse of registers and detected errors on the bus. Related to MSTATUS.ERRWARN bit.
0x0A0	RW	MDMACTRL (alias)	Master DMA Control	DMA control register, if DMA is enabled in the I3C module. Alias of Slave DMACTRL (SDMACTRL).
0x0AC	RW	MDATACTRL (alias)	Master Data Control	Controls data buffering and indicates the current buffer state. Alias of Slave DATACTRL (SDATACTRL). This is also controls the FIFO.
0x0B0	WO	MWDATAB (alias)	Master Write Data Byte	Write a byte of data, possibly the last byte. Alias of Slave WDATAB (SWDATAB).
0x0B4	WO	MWDATABE (alias)	Master Write Last Data Byte	Write the last byte of data. Alias of Slave WDATABE (SWDATABE).
0x0B8	WO	MWDATAH (alias)	Master Write Data Half-Word	Write a half-word of data, possibly the last half-word of data. Alias of Slave WDATAH (SWDATAH).
0x0BC	WO	MWDATAHE (alias)	Master Write Data Half-Word as End	Write the last half-word of data. Alias of Slave WDATAHE (SWDATAHE).
0x0C0	RO	MRDATAB (alias)	Master Read Data Byte	Read a byte of data. Alias of Slave RDATAB (SRDATAB).

Table continues on the next page...

Table continued from the previous page...

Offset	Type	Register		Description
0x0C4	RO	MRDATAH (alias)	Master Read Data Half-Word	Read a half-word of data. Alias of Slave RDATAH (SRDATAH).
0xD0	WO	MWMSG_SDR (semi-alias)	Master Write SDR Message	Set up and push through a message in Single Data Rate (SDR) mode. Written with 16b words.
0xD4	RO	MRMSG_SDR (alias)	Master Read SDR Message	Read content from Slave in message mode. Read as 16b words.
0xD8	WO	MWMSG_DDR	Master Write DDR Message	Set up and push through a message in DDR mode. Written with 16b words.
0xDC	RO	MRMSG_DDR (alias)	Master Read DDR Message	Read content from a slave in Double Data Rate (DDR) message mode. Read as 16b words.
0xE4	RW	MDYNADDR	Master Writable Dynamic Address	Allows writing its own Dynamic Addresswhen switching to slave mode

49.6.3 Slave registers summary

Offset	Type	Register		Description
0x000	RW	Reserved		Reserved for the master
0x004	RW	SCONFIG	Configuration	Configuration fields to set up before a block is activated
0x008	RO	SSTATUS	Status	Status for master and slave. Not all bits used if only slave mode is used.
0x00C	RW	SCTRL	Control	Control for active use, usually event generation (interrupts to the master)
0x010	R/W1S	SINTSET	Interrupts Set	Interrupt enables
0x014	W1C	SINTCLR	Interrupts Clear	Interrupt disables
0x018	RO	SINTMASKED	Interrupts Masked	Mask of sourced interrupts (SSTATUS ANDed with SINTSET)
0x01C	RW	SERRWARN	Errors and Warnings	Error and Warning state from protocol errors and issues. Related to SERRWARN status and interrupt.
0x020	RW	SDMACTRL	DMA Control	DMA control register
0x02C	RW	SDATACTRL	Data Control	Controls data buffering and indicates the current buffer state. This register also controls the FIFO.
0x030	WO	SWDATAB	Write Data Byte	Write a byte of data, including using the 9th bit to mark the end (of the last byte)

Table continues on the next page...

Table continued from the previous page...

Offset	Type	Register		Description
0x034	WO	SWDATABE	Write Last Data Byte	Write a byte of data, which is the end (of the last byte).
0x038	WO	SWDATAH	Write Data Half-Word	Write a half-word of data, using the 16th bit to mark as the end (the last byte of a half-word is the end).
0x03C	WO	SWDATAHE	Write Data Half-Word as End	Write a half-word of data, which is the end (the last byte of a half-word is the end).
0x040	RO	SRDATAB	Read Data Byte	Read a byte of data
0x048	RO	SRDATAH	Read Data Half-Word	Read a half-word of data
0x060	RO	SCAPABILITIES	Capabilities	Indicates what is available/supported in this module, including master and/or slave, HDR modes, and others.
0x064	RO	SDYNADDR	Writable Dynamic Address	The dynamic address once assigned; otherwise this is 0.
0x068	RO	SMAXLIMITS	Maximum Limits	Indicates the maximum read and write limits set by the master (or the original requested limits).
0x06C	RW	SIDPARTNO	ID Part Number	Enables the application to write the ID part-number.
0x070	RW	SIDEXT	ID Extensions	Enables the application to write the ID extension of DCR and/or BCR and/or instance.
0x074	RW	SVENDORID	Vendor ID	Enables the application to write the Vendor ID.
0x078	RW	STCCLOCK	Time Control Clock	Enables the application to dynamically set the time control clock and accuracy information.
0x07C	RO	SMSGMAPADDR	Message Mapped Address	Returns the index of the matching Dynamic address or Static address.
0xFFC	RO	SID	Block ID	Shows the block ID (module ID) and revision.

49.7 I3C register descriptions

Table 299. Register Types

Type		
R	Read	
RO	Read-Only	
RW	Read/Write	
W1C	Write 1 to Clear	W1S and W1C registers are readable (they can be read).
W1S	Write 1 to Set	
WO	Write-Only	

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module will not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

However, the I3C peripheral will ignore writes to RO registers, and the I3C peripheral will return 0 for reads from WO or non-existent registers. It requires all 32-bit regs to be read and written as 32-bit and aligned to 32 bits; the peripheral uses the Advanced Peripheral Bus (APB), so the peripheral would not know if partial read or write operations (less than 32 bits) are used.

49.7.1 I3C memory map

I3C0.I3C base address: 4003_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Master Configuration Register (MCONFIG)	32	RW	See section
4h	Slave Configuration Register (SCONFIG)	32	RW	See section
8h	Slave Status Register (SSTATUS)	32	RW	See section
Ch	Slave Control Register (SCTRL)	32	RW	See section
10h	Slave Interrupt Set Register (SINTSET)	32	RW	See section
14h	Slave Interrupt Clear Register (SINTCLR)	32	RW	See section
18h	Slave Interrupt Mask Register (SINTMASKED)	32	R	See section
1Ch	Slave Errors and Warnings Register (SERRWARN)	32	RW	See section
20h	Slave DMA Control Register (SDMACTRL)	32	RW	See section
2Ch	Slave Data Control Register (SDATACTRL)	32	RW	See section
30h	Slave Write Data Byte Register (SWDATAB)	32	RW	See section
34h	Slave Write Data Byte End (SWDATABE)	32	RW	See section
38h	Slave Write Data Half-word Register (SWDATAH)	32	RW	See section
3Ch	Slave Write Data Half-word End Register (SWDATAHE)	32	RW	See section
40h	Slave Read Data Byte Register (SRDATAB)	32	R	See section
48h	Slave Read Data Half-word Register (SRDATAH)	32	R	See section
60h	Slave Capabilities Register (SCAPABILITIES)	32	R	E83F_FE78h
64h	Slave Dynamic Address Register (SDYNADDR)	32	RW	See section
68h	Slave Maximum Limits Register (SMAXLIMITS)	32	RW	See section
6Ch	Slave ID Part Number Register (SIDPARTNO)	32	RW	0000_0000h
70h	Slave ID Extension Register (SIDEXT)	32	RW	See section
74h	Slave Vendor ID Register (SVENDORID)	32	RW	See section
78h	Slave Time Control Clock Register (STCCLOCK)	32	RW	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7Ch	Slave Message-Mapped Address Register (SMSGMAPADDR)	32	R	See section
84h	Master Main Control Register (MCTRL)	32	RW	See section
88h	Master Status Register (MSTATUS)	32	RW	See section
8Ch	Master In-band Interrupt Registry and Rules Register (MIBIRULES)	32	RW	0000_0000h
90h	Master Interrupt Set Register (MINTSET)	32	RW	See section
94h	Master Interrupt Clear Register (MINTCLR)	32	RW	See section
98h	Master Interrupt Mask Register (MINTMASKED)	32	R	See section
9Ch	Master Errors and Warnings Register (MERRWARN)	32	RW	See section
A0h	Master DMA Control Register (MDMACTRL)	32	RW	See section
ACh	Master Data Control Register (MDATACTRL)	32	RW	See section
B0h	Master Write Data Byte Register (MWDATAB)	32	RW	See section
B4h	Master Write Data Byte End Register (MWDATABE)	32	RW	See section
B8h	Master Write Data Half-word Register (MWDATAH)	32	RW	See section
BCh	Master Write Data Byte End Register (MWDATAHE)	32	RW	See section
C0h	Master Read Data Byte Register (MRDATAB)	32	R	See section
C8h	Master Read Data Half-word Register (MRDATAH)	32	R	See section
CCh	Write Byte Data 1 (to bus) (MWDATAB1)	32	RW	0000_0000h
D0h	Master Write Message in SDR mode (MWMSG_SDR_CONTROL)	32	RW	See section
D0h	Master Write Message Data in SDR mode (MWMSG_SDR_DATA)	32	RW	0000_0000h
D4h	Master Read Message in SDR mode (MRMSG_SDR)	32	R	See section
D8h	Master Write Message in DDR mode (MWMSG_DDR_CONTROL)	32	RW	See section
D8h	Master Write Message Data in DDR mode (MWMSG_DDR_DATA)	32	RW	0000_0000h
DCh	Master Read Message in DDR mode (MRMSG_DDR)	32	RW	See section
E4h	Master Dynamic Address Register (MDYNADDR)	32	RW	See section
FFCh	Slave Module ID (SID)	32	R	See section

49.7.2 Master Configuration Register (MCONFIG)

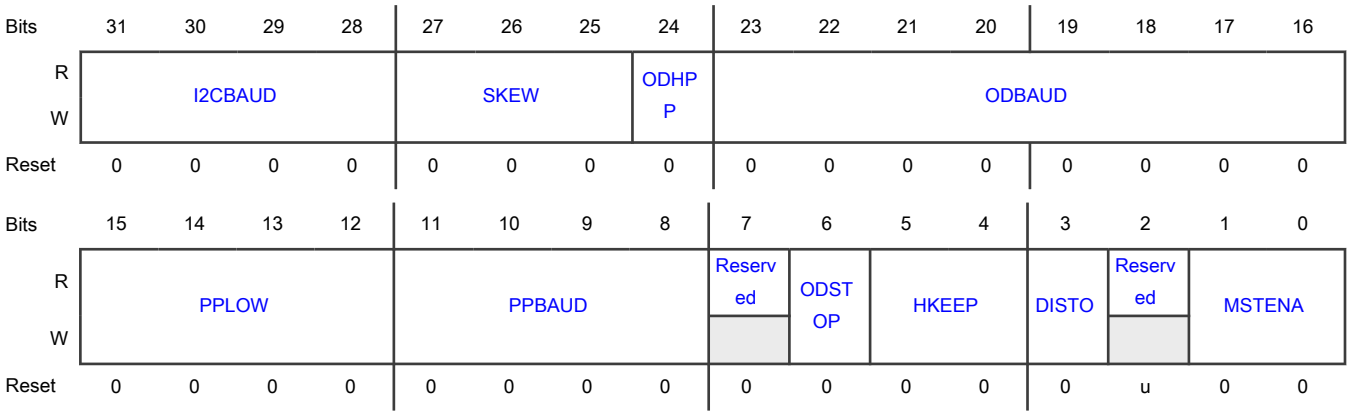
Offset

Register	Offset
MCONFIG	0h

Function

The MCONFIG register controls all Master states when Master operation is enabled. The MCONFIG register should not be changed when an active transaction is happening.

Diagram



Fields

Field	Function
31-28 I2CBAUD	I2C baud rate The I2C low and high times in ODBAUD counts. • I2CBAUD > 1 is the main count load, and it is *count-1*. Therefore, for I2CBAUD > 1: I2CBAUD==0 is 1 ODBAUD beat, and I2CBAUD==1 is 2 ODBAUD beats. • If I2CBAUD[28] is 1, then low will be 1 extra ODBAUD beat. This means that the I2CBAUD field is normally 3, where ODBAUD gives 200 ns; therefore, this means that for I2CBAUD > 1 I2CBAUD==1, which means 2 ODBAUD. Therefore, (2+1)*200=600ns low, with 2*200=400ns high for 1 us period, to match Fast Mode Plus (FM+). For Fast Mode (FM), I2CBAUD is normally 11 (giving 2.6 us) or is 6 (giving 2.4 us).
27-25 SKEW	Skew Number of FCLK counts for an SDA change after SCL for I3C push-pull; this is in addition to the roundtrip of the SCL line from the pad back to the module (3 ns to 4 ns or more). • SKEW is normally not needed, so assign SKEW=0. • SKEW is only used if SDA is not naturally skewing from an SCL change. • I2C automatically skews SDA (but not PUR) to match I2C rules.
24 ODHPP	Open drain high push-pull If ODHPP=1, then Open-Drain High should be 1 PPBAUD count for I3C messages; otherwise ODHPP should be the same value as ODBAUD. Using an ODHPP that is the same value as ODBAUD is commonly used so that I2C devices do not see high.
23-16 ODBAUD	Open drain baud rate ODBAUD= (number of PPBAUD counts) - 1. ODBAUD should not be 0 (not the same as push-pull). This would normally go for 200 ns (see I2CBAUD for I2C counts). When used with ODHPP, this gives 250 ns

Table continues on the next page...

Table continued from the previous page...

Field	Function
	per clock in I3C. Therefore, if PPBAUD gives 12 MHz, then 1 PPBAUD is ½ of 12MHz or 41.67 ns, and then to get around 200 ns, use 5-1=4 for ODBAUD.
15-12 PPLOW	Push-Pull low Adder for push-pull low, to create a duty-cycle with a longer low period, with up to 15 more FCLKs low than high. PPLOW=0 means a 50:50 duty cycle.
11-8 PPBAUD	Push-pull baud rate The FCLK Counter for each push-pull low and normally high period. So, PPBAUD=0 if run at ½ input FCLK speed (for example, a 24 MHz FCLK yields a 12 MHz SCL, because each FCLK is SCL Low or SCL High). Note the use of duty-cycle via Push-Pull Low (PPLOW). For example, 24 MHz with 50:50 duty cycle is 12 MHz, but with PPLOW adding 3 more low beats, the push-pull baud rate becomes 4.8 MHz (from 24 MHz/5 beats).
7 —	Reserved
6 ODSTOP	Open drain stop If ODSTOP=1, then STOP will be emitted at open-drain speeds even for I3C messages. This can be useful for legacy devices, to ensure that the legacy devices see the STOP.
5-4 HKEEP	High-Keeper Indicates how High-Keeper is to be supported. **NOTE** Your specific device may not support any or all of these High-Keeper methods. Refer to the chip-specific section for this module, at the beginning of this chapter. 00b - NONE. Use PUR (Pull-Up Resistor). Hold SCL High. 01b - WIRED_IN. Wired-in High Keeper controls; use pin_HK (High Keeper) controls. 10b - PASSIVE_SDA. Can Hi-Z (high impedance) for Bus Free (IDLE) and hold. 11b - PASSIVE_ON_SDA_SCL. Can Hi-Z (high impedance) both for Bus Free (IDLE), and can Hi-Z SDA for hold.
3 DISTO	Disable Timeout This timeout detects application errors. • If DISTO=1, then it disables the timeout (if the timeout is configured). • If the master has been left in a state other than STOPped for more than 100 us (because 10 KHz is the slowest allowed I3C speed), then the timeout will send an MERRWARN (interrupt). • To prevent the MERRWARN interrupt when doing development or testing, write 1 to DISTO to disable the timeout. • In systems that support timeouts, timeout will be disabled automatically during debug.
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 MSTENA	Master enable Indicates if the master is enabled and what “states” are allowed for it (the master). 00b - MASTER_OFF. Master is off (is not enabled). If MASTER_OFF is enabled, then the I3C module can only use slave mode. 01b - MASTER_ON. Master is on (is enabled). When used from start-up, this I3C module is master by default (the main master). The module will control the bus unless the master is handed off. If the master is handed off, then MSTENA must move to 2 after that happens. The handoff means emitting GETACCMST and if accepted, the module will emit a STOP and set the MSTENA bit to 2 (or 0). 10b - MASTER_CAPABLE. The I3C module is master-capable; however the module is operating as a slave now. When used from the start, the I3C module will start as a slave, but will be prepared to switch to master mode. To switch to master mode, the slave emits an Master Request (MR), or gets a GETACCMST CCC command and accepts it (to switch on the STOP). 11b - Reserved

49.7.3 Slave Configuration Register (SCONFIG)

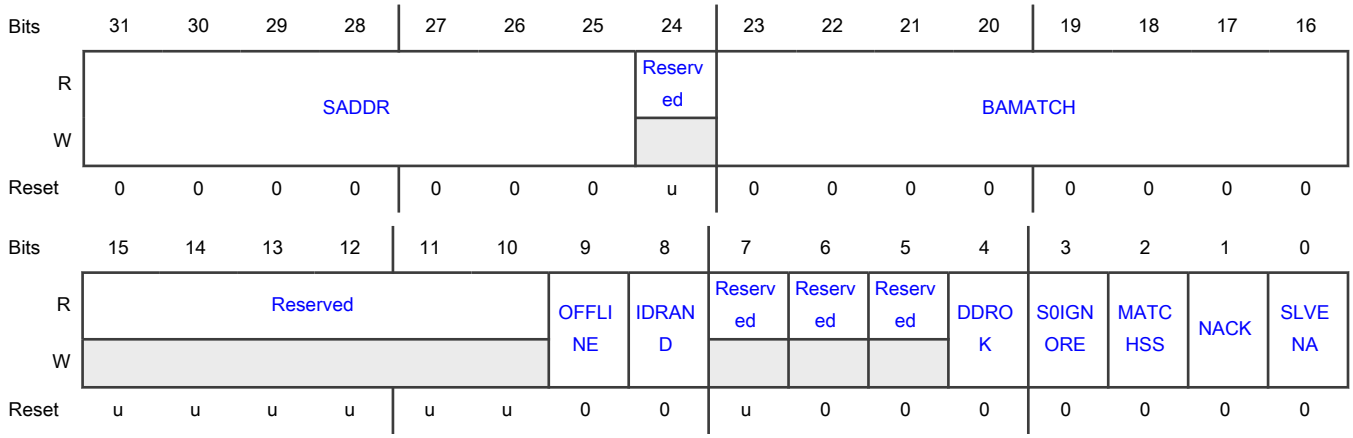
Offset

Register	Offset
SCONFIG	4h

Function

Contains fields that must be configured before the module is activated.

Diagram



Fields

Field	Function
31-25 SADDR	Static address SADDR sets the I2C 7-bit static address.
24 —	Reserved
23-16 BAMATCH	Bus available match This is the Bus Available condition match value for the current slow clock. BAMATCH provides the count of the slow clock to count out 1 us (or more), to allow an In-Band Interrupt (IBI) to drive SDA low when the master is not doing so. The maximum width and maximum values are controlled by the I3C module.
15-10 —	Reserved
9 OFFLINE	Offline If OFFLINE=1 when the slave enable (SCONFIG.SLVENA) is set to 1, then the I3C module will wait for either 60 us of bus quiet or a High Data Rate (HDR) Exit Pattern. This ensures that the bus is not in High Data Rate (HDR) mode, and so can safely track Single Data Rate (SDR) mode.
8 IDRAND	ID random • If IDRAND=1, then SIDPARTNO.PARTNO is a random value. • If IDRAND=0, then SIDPARTNO.PARTNO is a part number and an instance.
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 DDROK	Double Data Rate OK If DDROK=1, then HDR-DDR messaging is allowed; set the corresponding SIDEXT.BCR bit (Bus Characteristics Register bit in Slave ID Extension Register) to say that High Data Rate (HDR) is available, and configure the corresponding HDRCAP HDR-DDR bit to allow using Double Data Rate (DDR) mode. **NOTE** The DDROK bit must be set before the Slave can connect to the I3C bus; the Slave peripheral will indicate to the Master whether it (the slave) can support the feature during dynamic address assignment.
3	S0/S1 errors ignore

Table continues on the next page...

Table continued from the previous page...

Field	Function
S0IGNORE	If S0IGNORE=1, then the slave will not detect S0 or S1 errors, and therefore the slave will not lock up, and will wait for an exit pattern. S0IGNORE=1 should only be used when the I3C bus will not use a High Data Rate (HDR) mode.
2 MATCHSS	Match START or STOP If MATCHSS=1, then the START and STOP sticky STATUS bits will only be set if SSTATUS.MATCHED is set. This allows START and STOP to be used to detect the end of a message to/from this slave.
1 NACK	Not acknowledge If NACK=1, then the slave will NACK all requests to it, except a Common Command Code (CCC) broadcast. NACK=1 should be used with caution, because the master may decide that the slave is missing, if NACK is overused.
0 SLVENA	Slave enable • If SLVENA=1, then a slave can operate on the I2C or I3C bus. • If SLVENA=0, then a slave will ignore the I2C or I3C bus. SLVENA should not be set before registers like SCONFIG (SIDPARTNO, SIDEXT, and others) are set, because these registers affect the data to/from the master. Slave enable is normally only configured just one time before the I3C bus comes up. If slave enable is used at other times, see Hot-Join. In the case of Hot-Join, the Hot-Join bit (SCAPABILITIES.IBI_MR_HJ])should be set before setting the slave enable bit (SLVENA), so that the device does not see a START or STOP incorrectly.

49.7.4 Slave Status Register (SSTATUS)

Offset

Register	Offset
SSTATUS	8h

Function

Not all bits are used if the module only acts as a slave. The Slave Status register is used to indicate both sticky status for interrupts, as well as “states” and “modes” related to the I3C bus. The fields are divided into current activity, interrupt maskable actions, then states and modes on the bus.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TIMECTRL		ACTSTATE		HJDIS	Reserved	MRDIS	IBIDIS	Reserved		EVDET		Reserved	EVENT	CHANDLED	HDRMATCH
W																
Reset	0	0	0	0	0	u	0	0	0	0	u	u	u	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	CCC	DACHG	TXNOTFU...	RX_PEND	STOP	MATCHED	START	Reserved	STHDR	STDAA	STREQWR	STREQRD	STCCCH	STMSG	STNOTST...
W																
Reset	0	0	0	1	0	0	0	0	u	0	0	0	0	0	0	0

Fields

Field	Function
31-30 TIMECTRL	Time control Indicates if time control is currently enabled. 00b - NO_TIME_CONTROL. No time control is enabled 01b - Reserved 10b - ASYNC_MODE. Asynchronous standard mode (0) is enabled 11b - Reserved
29-28 ACTSTATE	Activity state from Common Command Codes (CCC) 00b - NO_LATENCY. Normal bus operations 01b - LATENCY_1MS. 1 ms of latency 10b - LATENCY_100MS. 100 ms of latency 11b - LATENCY_10S. 10 seconds of latency
27 HJDIS	Hot-Join is disabled HJDIS=1 if Hot-Join is disabled at this time. While Hot-Join is disabled, CTRL requests will be held off (not responded to).
26 —	Reserved
25 MRDIS	Master requests are disabled MRDIS=1 if Master Requests are disabled at this time. While Master Requests are disabled, CTRL requests will be held off (not responded to).
24 IBIDIS	In-Band Interrupts are disabled IBIDIS=1 if In-Band Interrupts are disabled at this time. While In-Band Interrupts are disabled, CTRL requests will be held off (not responded to).

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-22 —	Reserved
21-20 EVDET	Event details Current details of the last (EVENT=1) or pending event. 00b - NONE. No event or no pending event 01b - NO_REQUEST. Request not sent yet. Either there was no START yet, or is waiting for Bus-Available or Bus-Idle (HJ). 10b - NACKED. Not acknowledged(Request sent and NACKed); the module will try again. 11b - ACKED. Acknowledged (Request sent and ACKed), so Done (unless the time control data is still being sent).
19 —	Reserved
18 EVENT	Event For a Slave, a pending In-Band Interrupt (IBI), P2P (peer-to-peer), MR (Master Request), or Hot-Join (HJ) has been sent as requested. See the upper status register fields for details. Note that for an in-band interrupt, this occurs on the ACK if no IBI byte is present, and after the 1 IBIDATA byte has been sent. So, if time control is used, those time control bytes (in IBI) will go out after this EVENT is signaled.
17 CHANDLED	Common-Command-Code handled A Common Command Code (CCC) is being handled by the module. This is a notification only, but the result may be an updated SSTATUS register.
16 HDRMATCH	High Data Rate command match An HDR command matched this device's I3C Dynamic Address. The HDR command will be available as the 1st byte and RXPEND will be set (whether the command is read or write, the MSb of that command byte indicates if it is a read or a write command). If the HDR command is a read, and there are to-bus bytes waiting, then the command will be ACKed and the data sent back; otherwise the HDR command will be NACKed. Note that when HDRMATCH is set, the ERRWARN bit should be checked, because the HPAR error may have be encountered after signaling this HDR command (the parity is after the destination address and command).
15 ERRWARN	Error warning An error or warning has occurred, such as data underrun, data overrun, parity error, HDR-DDR CRC error, or other error or warning condition. See the Slave Errors and Warnings Register (SERRWARN).
14 CCC	Common Command Code A Common-Command-Code (CCC) has been received, and is not handled by the I3C module. There are 2 types of Common Command Codes: • Broadcasted CCC, which will then also correspond with RXPEND and the 1st byte will be the CCC (command).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Direct CCC, which may never be directed to this device. If Direct CCC are directed to this device, then the TXSEND or RXPEND will be triggered, and the RXPEND will contain the command.
13 DACHG	DACHG The Slave Dynamic Address has been assigned, re-assigned, or reset (lost) and is now in that state of being valid or none.
12 TXNOTFULL	Transmit buffer is not full TXNOTFULL=1 when the To-bus buffer/FIFO can accept more data to go out. For all but External FIFO, this uses SDATACTRL.TXTRIG, which defaults to not-full. If DMA is enabled for TX, then it will also be signaled to provide more data.
11 RX_PEND	Received message pending Receiving a message from master, which is not being handled by the I3C module (not a Common Command Code (CCC), so is internally processed by the module). For all but External FIFO, this uses SDATACTRL.RXTRIG, which defaults to not-empty. If DMA is enabled for RX, then DMA will be signaled as well. RX_PEND will self-clear if data is read (from FIFO and non-FIFO sources).
10 STOP	Stop The STOP bit is from a stopped state being detected, meaning that a STOP state was present on the bus since the bus was last cleared. The STNOTSTOP state will also indicate if the I3C module is in stop mode. A fast STOP/START combination may not trigger the STOP status; for that case, START will always be set.
9 MATCHED	Matched An incoming header matched this device's I3C Dynamic or I2C Static address (if any) since the bus was last cleared.
8 START	Start START=1 if a START or repeated START was seen after the START bit was last cleared. This is not usually needed, but can be used for wake events.
7 —	Reserved
6 STHDR	Status High Data Rate STHDR=1 if the I3C bus is in HDR-DDR, HDR-TSP, or HDR-TSL modes, regardless of whether HDR mode is supported by this module or not, and regardless of whether the message is to this module or to some other module.
5 STDAA	Status Dynamic Address Assignment STDAA=1 if the I3C bus is in Enter Dynamic Address Assignment (ENTDAA) mode, regardless of whether this bus slave has or does not have a Dynamic Address.
4	Status request write

Table continues on the next page...

Table continued from the previous page...

Field	Function
STREQWR	STREQWR=1 if the request (REQ) in process is SDR write data from the master to this bus slave (or all I3C slaves), but not in Enter Dynamic Address Assignment (ENTDAA) mode. See Status High Data Rate (STHDR) for Double Data Rate (DDR) handling.
3 STREQRD	Status required STREQRD=1 • if the REQ in process is an SDR read from this slave • or if an In-Band Interrupt (IBI) is being pushed out Also see Status High Data Rate (STHDR) for Double Data Rate (DDR) handling.
2 STCCCH	Status Common Command Code Handler STCCCH=1 if a Common Command Code (CCC) message is being handled automatically.
1 STMSG	Status message STMSG=1 if this bus slave is listening to the bus traffic or responding. If STNOSTOP=1, then STMSG will be 0 when a non-matching address is seen, until the next repeated START or STOP occurs.
0 STNOTSTOP	Status not stop • when the bus is busy (has activity), STNOTSTOP=1 • when the I3C module is in a STOP condition, STNOTSTOP=0 Other SSTATUS bits may also be set when busy. Note that STNOTSTOP can also be true (=1) after an S0 or S1 error, where the I3C module is waiting for an Exit Pattern.

49.7.5 Slave Control Register (SCTRL)

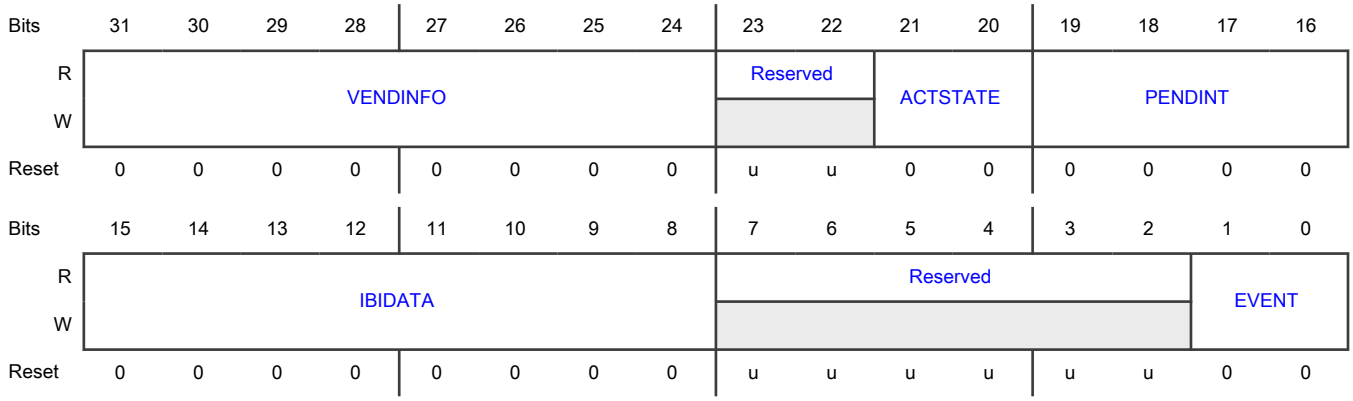
Offset

Register	Offset
SCTRL	Ch

Function

Contains controls for the active use of the I3C bus, like event generation (for example, interrupts to the master). The Slave Control register is used to activate various special operations for the Slave, but only if the module is configured to support those operations. This includes events such as IBI and GETSTATUS fields (except the Protocol error, which is automatically set).

Diagram



Fields

Field	Function
31-24 VENDINFO	Vendor information Should be set to the Vendor Reserved field that the GETSTATUS CCC will return. The vendor information should be maintained by the application, because the master will read this field. If VENDINFO is not configured, then the GETSTATUS field will always return 0.
23-22 —	Reserved
21-20 ACTSTATE	Activity state (of slave) Should be set to the slave's activity state that the GETSTATUS CCC (command code) will return as activity mode. The activity state should be maintained by the application, because the master will read this field. If the activity state is not configured, then the GETSTATUS command will always return 0.
19-16 PENDINT	Pending interrupt Should be set to the pending interrupt that the GETSTATUS CCC (command code) will return. The pending interrupt should be maintained by the application, because the master will read this field. If PENDINT=0 and • if an IBI interrupt is pending, then the GETSTATUS command will return 1. • if an IBI interrupt is not pending, then the GETSTATUS field will return 0.
15-8 IBIDATA	In-Band Interrupt Data Data byte to go with an In-Band Interrupt (IBI), if the module is enabled for in-band interrupts. If SCTRL.IBIDATA is enabled, then in-band interrupts are required.
7-2 —	Reserved
1-0 EVENT	EVENT • If EVENT is set to non-0, it will request an event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• After being requested, SSTATUS.EVENT and SSTATUS.EVDET will show the status as it progresses. • After completion, the EVENT field will automatically return to 0. • After EVENT is non-0, only 0 can be written to EVENT (to cancel), until the event processing is done (finished). 00b - NORMAL_MODE. If EVENT is set to 0 after was a non-0 value, event processing will cancel if the event processing has not yet started; if event processing has already been started, then event processing will not be be cancelled. 01b - IBI. Start an In-Band Interrupt. This will try to push an IBI interrupt onto the I3C bus. If data is associated with the IBI, then the data will be read from the SCTRL.IBIDATA field. If time control is enabled, then this data will also include any time control-related bytes; additionally, the IBIDATA byte will have bit 7 set to 1 automatically (as is required for time control). The IBI interrupt will occur after the 1st (mandatory) IBIDATA, if any. 10b - MASTER_REQUEST. Start a Master-Request. 11b - HOT_JOIN_REQUEST. Start a Hot-Join request. A Hot-Join Request should only be used when the device is powered on after the I3C bus is already powered up, or when the device is connected using hot insertion methods (the device is powered up when it is physically inserted onto the powered-up I3C bus). The hot join will wait for Bus Idle, and SCTRL.EVENT=HOT_JOIN_REQUEST must be set before the slave enable (SCONFIG.SLVENA).

49.7.6 Slave Interrupt Set Register (SINTSET)

Offset

Register	Offset
SINTSET	10h

Function

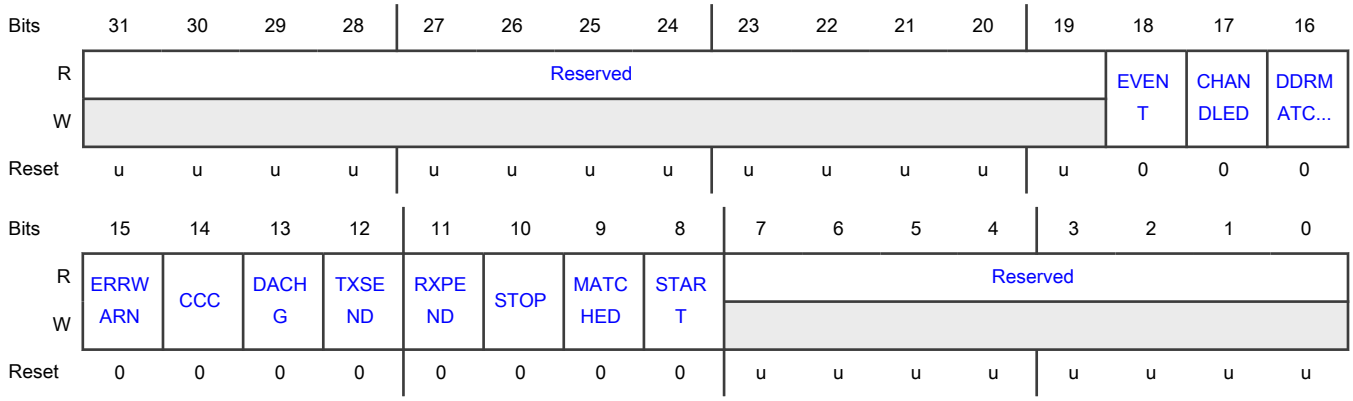
Sets interrupt enables for select STATUS bits. Reading this register (SINTSET) returns the status of the interrupt enables.

- To activate an interrupt enable, write 1 to its field in this register (SINTSET).
- To disable an interrupt enable, write 1 to the appropriate bit in the Interrupt Clear Register (SINTCLR). Writing 0 to the interrupt enable in this register (SINTSET) does not disable the interrupt.

The Interrupt registers allow masking interrupt sources, as well as checking which interrupts have activated. The normal method is to enable an interrupt, and then once the interrupt fires, the interrupt is either cleared by writing the SSTATUS register or cleared by action on the corresponding data register. The Interrupt is level held, meaning the interrupt stays set until the cause is cleared by some method. The module prevents races, so that if a new event comes in, that new event will not be lost.

- SINTSET sets interrupt enables for STATUS bits. Reading the SINTSET register returns the status of the interrupt enables.
- SINTCLR clears interrupt enables for STATUS bits.
- SINTMASKED returns the value of the STATUS bits ANDed with their interrupt enables.

Diagram



Fields

Field	Function
31-19 —	Reserved
18 EVENT	Event interrupt enable
17 CHANDLED	Common Command Code (CCC) (that was handled by I3C module) interrupt enable
16 DDRMATCHED	Double Data Rate (DDR) interrupt enable
15 ERRWARN	Error/warning interrupt enable
14 CCC	Common Command Code (CCC) (that was not handled by I3C module) interrupt enable
13 DACHG	Dynamic address change interrupt enable
12 TXSEND	Transmit interrupt enable
11 RXPEND	Receive interrupt enable
10 STOP	Stop interrupt enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 MATCHED	Match interrupt enable
8 START	Start interrupt enable
7-0 —	Reserved

49.7.7 Slave Interrupt Clear Register (SINTCLR)

Offset

Register	Offset
SINTCLR	14h

Function

Clear interrupt enables for select STATUS bits. To clear an interrupt enable, write 1 to the corresponding bit in this register (SINTCLR); writing 0 has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													EVENT	CHANDLED	DDRMATC...
W														W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	CCC	DACHG	TXSEND	RXPEND	STOP	MATCHED	START	Reserved							
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 EVENT	EVENT interrupt enable clear
17 CHANDLED	CHANDLED interrupt enable clear
16 DDRMATCHED	DDRMATCHED interrupt enable clear
15 ERRWARN	ERRWARN interrupt enable clear
14 CCC	CCC interrupt enable clear
13 DACHG	DACHG interrupt enable clear
12 TXSEND	TXSEND interrupt enable clear
11 RXPEND	RXPEND interrupt enable clear
10 STOP	STOP interrupt enable clear
9 MATCHED	MATCHED interrupt enable clear
8 START	START interrupt enable clear
7-0 —	Reserved

49.7.8 Slave Interrupt Mask Register (SINTMASKED)

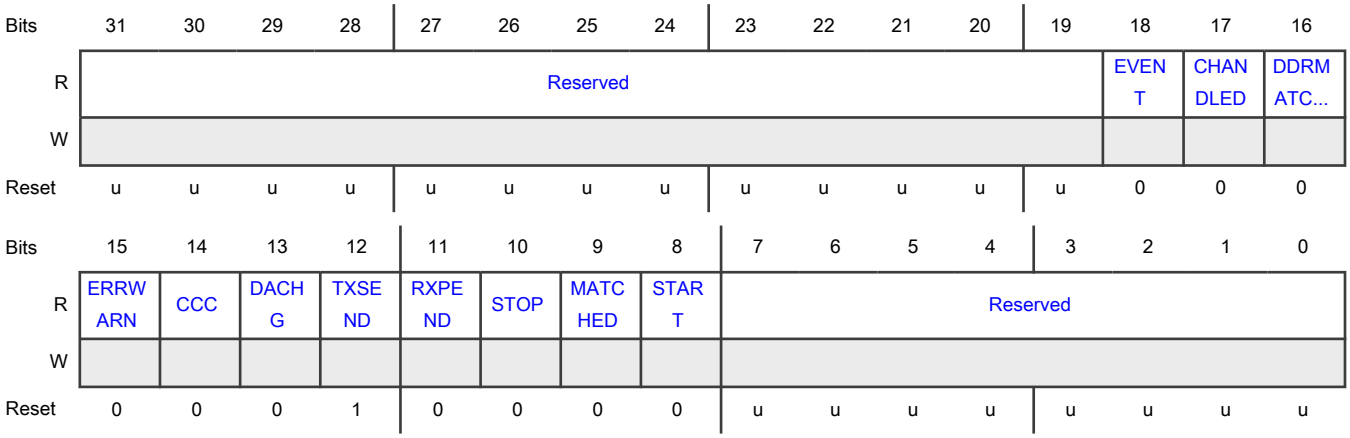
Offset

Register	Offset
SINTMASKED	18h

Function

Returns the status of enabled interrupts (the value of STATUS ANDed with INTSET).

Diagram



Fields

Field	Function
31-19 —	Reserved
18 EVENT	EVENT interrupt mask
17 CHANDLED	CHANDLED interrupt mask
16 DDRMATCHED	DDRMATCHED interrupt mask
15 ERRWARN	ERRWARN interrupt mask
14 CCC	CCC interrupt mask
13 DACHG	DACHG interrupt mask
12 TXSEND	TXSEND interrupt mask
11	RXPEND interrupt mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXPEND	
10 STOP	STOP interrupt mask
9 MATCHED	MATCHED interrupt mask
8 START	START interrupt mask
7-0 —	Reserved

49.7.9 Slave Errors and Warnings Register (SERRWARN)

Offset

Register	Offset
SERRWARN	1Ch

Function

Contains errors and warnings from I3C/I2C protocol, which includes internal issues such as overrun and underrun, detected errors and conditions like parity errors, CRC errors, and read terminations by the Master. Related to SSTATUS.ERRWARN bit and SINTSET.ERRWARN interrupt.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														OWRITE	OREAD
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				S0S1	HCRC	HPAR	SPAR	Reserved			INVSTART	TERM	URUNNACK	URUN	ORUN
W																
Reset	u	u	u	u	0	0	0	0	u	u	u	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17 OWRITE	Over-write error The SWDATAB/BE register was written when FULL.
16 OREAD	Over-read error The SRDATAB register was read for more bytes than were available by the application.
15-12 —	Reserved
11 S0S1	S0 or S1 error A S0 or S1 error has occurred and the slave is locked and waiting for an HDR Exit Pattern. Writing 1 to S0S1 will cause the module to release the lock, but that should be used with great care. Normally, S0S1 will clear automatically when an Exit Pattern is detected. Therefore, writing 1 to S0S1 should only be used under controlled circumstances, to avoid problems. Before starting to operate normally, the module will then wait for a START (or repeated START) or STOP.
10 HCRC	HDR-DDR CRC error HDR-DDR Cyclic Redundancy Check (CRC) error on a message from the master. This calls into question the data from the whole DDR command frame. Note that this includes an HDR Restart or Exit being issued before an HDR-DDR message from master has finished.
9 HPAR	HDR parity error HDR Parity error or framing error on a message from the master. Note that the corresponding command or data that had the error will normally be in the RX buffer (which can be read using the Slave Read Data Byte Register (SRDATAB)).
8 SPAR	SDR parity error SDR Parity error on a message from the master. This will also set the GETSTATUS Protocol Error sticky bit (which is cleared after a GETSTATUS read).
7-5 —	Reserved
4 INVSTART	Invalid start error Invalid start with SCL falling while SDA=1 is in a STOP condition.
3 TERM	Terminated error The master terminated a read from a slave when an END was not set (on the same or previous read).
2 URUNNACK	Underrun and Not Acknowledged (NACKed) error The internal to-bus buffer/FIFO was underrun in the read header and so the module NACKed the header.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 URUN	Underrun error The internal to-bus buffer/FIFO was underrun during data read (the application did not provide the data fast enough). The END bit or register should be used if that was the last one.
0 ORUN	Overrun error The internal from-bus buffer/FIFO was overrun (too many characters are coming in and cannot be processed by the the user application fast enough).

49.7.10 Slave DMA Control Register (SDMACTRL)

Offset

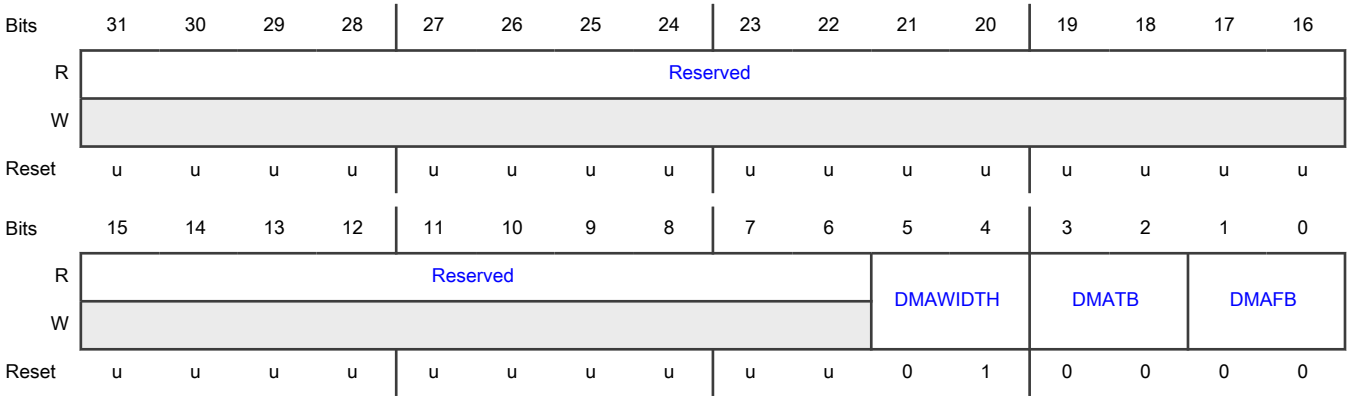
Register	Offset
SDMACTRL	20h

Function

The DMA Control register allows for DMA to be used for inbound messages and outbound messages. The DMA Control register is limited in value for Slave use, because the Slave has to be reactive to what happens. Two common use models:

- From-bus collection to avoid being overrun: The SCONFIG.MATCHSS bit is set, and then the processor enables the interrupts for START, and STOP, as well as enabling the DMA to collect the data. The START or STOP interrupt will only occur after a message is directed to the Slave (MATCHED bit set), and the DMA copied data can then be examined.
- To-bus for larger reads: Because I3C and I2C reads are preceded by a write that indicates what will be read (or in response to an IBI from the Slave), the DMA can be used to push through the data.
  - For I3C, the last value needs to be handled by the processor, unless the DMA moves wider words and is able to set the END bit (i.e., 16-bit values when in byte mode or 32-bit values when in half-word mode).
  - For I2C, the last value is determined by the Master, so the DMA may end early or may run out when the Master still wants more.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-4 DMAWIDTH	Width of DMA operations The width of DMA operations, if configured to allow half-word data access. 00b - BYTE 01b - BYTE_AGAIN 10b - HALF_WORD: Half word (16 bits). This will make sure that 2 bytes are free/available in the FIFO. 11b - Reserved
3-2 DMATB	DMA Write (To-bus) trigger If enabled with 1 or 2, DMATB will start a request DMA on a TX trigger; see the Slave Data Control Register (SDATACTRL). DMATB will request until full unless the DMA is set up as a trigger. DMATB will cancel on MSTATUS.ERRWARN. 00b - NOT_USED. DMA is not used 01b - ENABLE_ONE_FRAME. DMA is enabled for 1 Frame (ended by DMA or terminated). DMATB auto-clears on a STOP or START (see the Match START or STOP bit (SCONFIG.MATCHSS). 10b - ENABLE. DMA is enabled until turned off. Normally, ENABLE should only be used with Master Message mode.
1-0 DMAFB	DMA Read (From-bus) trigger If enabled with 1 or 2, DMAFB will request DMA on RX trigger (see Slave Data Control Register (SDATACTRL)). It will request until empty unless the DMA is set up as a trigger. DMAFB will cancel on SSTATUS.ERRWARN. 00b - DMA not used 01b - DMA is enabled for 1 frame. Auto-clears on STOP or repeated START. See the Match START or STOP bit (SCONFIG.MATCHSS). 10b - DMA enable. DMA is enabled until it is turned off.

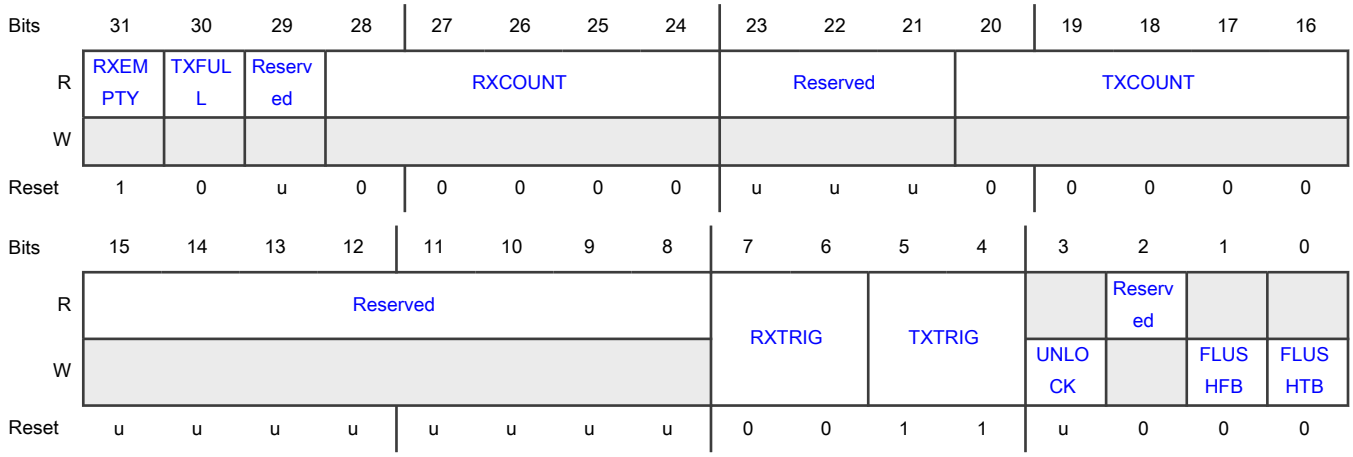
49.7.11 Slave Data Control Register (SDATACTRL)

Offset

Register	Offset
SDATACTRL	2Ch

Function
Controls data buffering and FIFO behavior.

Diagram



Fields

Field	Function
31 RXEMPTY	RX is empty 0b - RX is not empty 1b - RX is empty
30 TXFULL	TX is full 0b - TX is not full 1b - TX is full
29 —	Reserved
28-24 RXCOUNT	Count of bytes in RX
23-21 —	Reserved
20-16 TXCOUNT	Count of bytes in TX
15-8 —	Reserved
7-6 RXTRIG	Trigger level for RX FIFO fullness Affects interrupts and DMA (if enabled). 00b - Trigger on not empty

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Trigger on ¼ or more full 10b - Trigger on ½ or more full 11b - Trigger on 3/4 or more full
5-4 TXTRIG	Trigger level for TX FIFO emptiness Affects interrupts and DMA (if enabled). The default is Trigger on 1 less than full or less (=3). 00b - Trigger on empty 01b - Trigger on ¼ full or less 10b - Trigger on ½ full or less 11b - Trigger on 1 less than full or less (Default)
3 UNLOCK	Unlock • If 0 is written to UNLOCK, then the RXTRIG and TXTRIG fields cannot be changed on a write. • If 1 is written to UNLOCK, then the RXTRIG and TXTRIG fields can be changed on a write.
2 —	Reserved
1 FLUSHFB	Flushes the from-bus buffer/FIFO Not normally used.
0 FLUSHTB	Flush the to-bus buffer/FIFO Used when the master terminates a to-bus (read) message prematurely.

49.7.12 Slave Write Data Byte Register (SWDATAB)

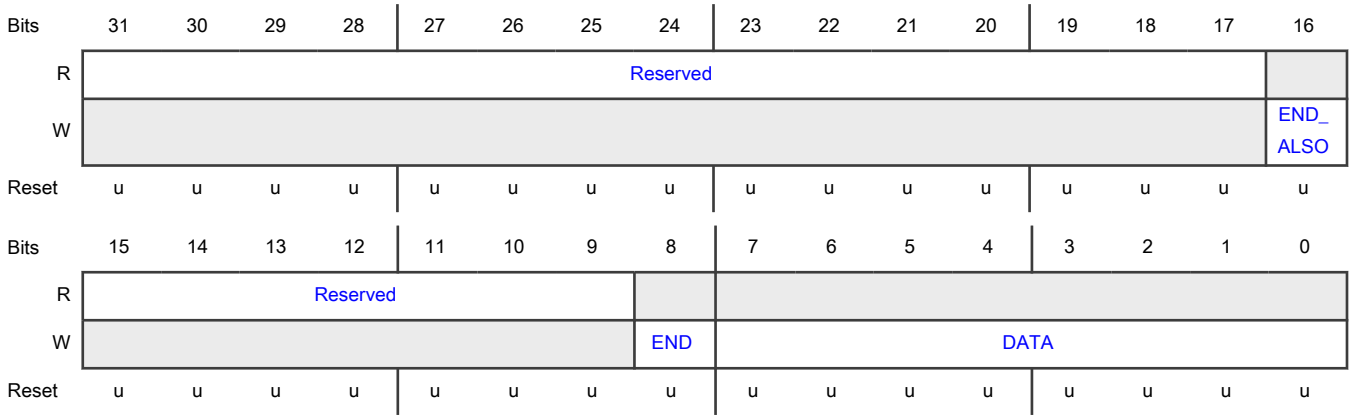
Offset

Register	Offset
SWDATAB	30h

Function

Write a byte of data, using the 9th bit to mark as the end (the last byte of the message). This register allows writing a byte to the bus (to Master) unless an external FIFO is used. This takes a byte and an end-of-data (last) marker bit. A byte should not be written unless there is room, as indicated by the TXNOTFULL bit being set in the SSTATUS register.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END_ALSO	End also • If END_ALSO=1, then this bit marks the last byte of the message • If END_ALSO=0, then there are more bytes in the message END_ALSO is required to be used in I3C, but is optional in I2C. For HDR-DDR (Double Data Rate), the byte with the END_ALSO must be an even byte (2nd, 4th, 6th, etc.) because DDR uses byte-pairs.
15-9 —	Reserved
8 END	End • If END=1, then this bit marks the last byte of the message • If END=0, then there are more bytes in the message END is required to be used in I3C, but is optional in I2C. For HDR-DDR (Double Data Rate), the byte with the END must be an even byte (2nd, 4th, 6th, etc.) because DDR uses byte-pairs.
7-0 DATA	The data byte to send to the master

49.7.13 Slave Write Data Byte End (SWDATABE)

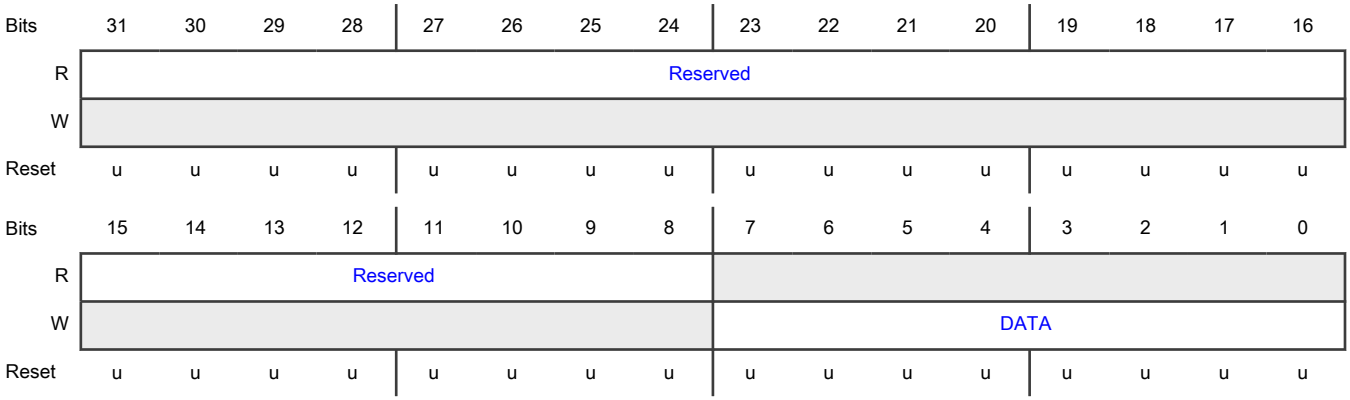
Offset

Register	Offset
SWDATABE	34h

Function

Write a byte of data, which is the end (the last byte of the message). Write a byte just like SWDATAB, but mark as end-of-data (last byte). For HDR-DDR, the byte with the END must be an even (2nd, 4th, 6th, etc) because DDR uses byte-pairs. A byte should not be written unless there is room, as indicated by the TXNOTFULL bit being set in the SSTATUS register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA	The data byte to send to the master

49.7.14 Slave Write Data Half-word Register (SWDATAH)

Offset

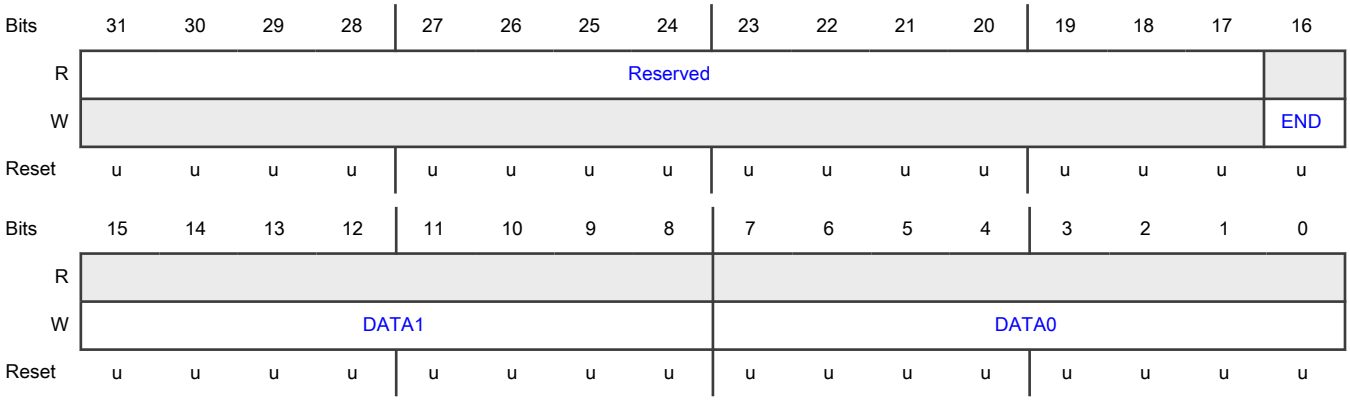
Register	Offset
SWDATAH	38h

Function

Write a half-word of data, using the 16th bit to mark as the end (the last byte of the half-word is the end). This register allows writing a half-word (pair of bytes) to the bus unless an external FIFO is used. This takes a half-word, which will send out the Low byte and then the High byte. An end-of-data (last) marker bit is allowed (or must be 0). A half-word should not be written

unless there is room for both, as indicated by use of TX FIFO level trigger or TXCOUNT available space in the SDATACTRL register.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of message • If END=1, then END bit marks the last byte of the message • If END=0, then there are more bytes in the message For this register, this always marks DATA1 as the end. END is required to be used in I3C, but is optional in I2C. For HDR-DDR (Double Data Rate), the byte with the END must be an even byte (2nd, 4th, 6th, etc.) because DDR uses byte-pairs.
15-8 DATA1	The 2nd byte to send to the master
7-0 DATA0	The 1st byte to send to the master

49.7.15 Slave Write Data Half-word End Register (SWDATAHE)

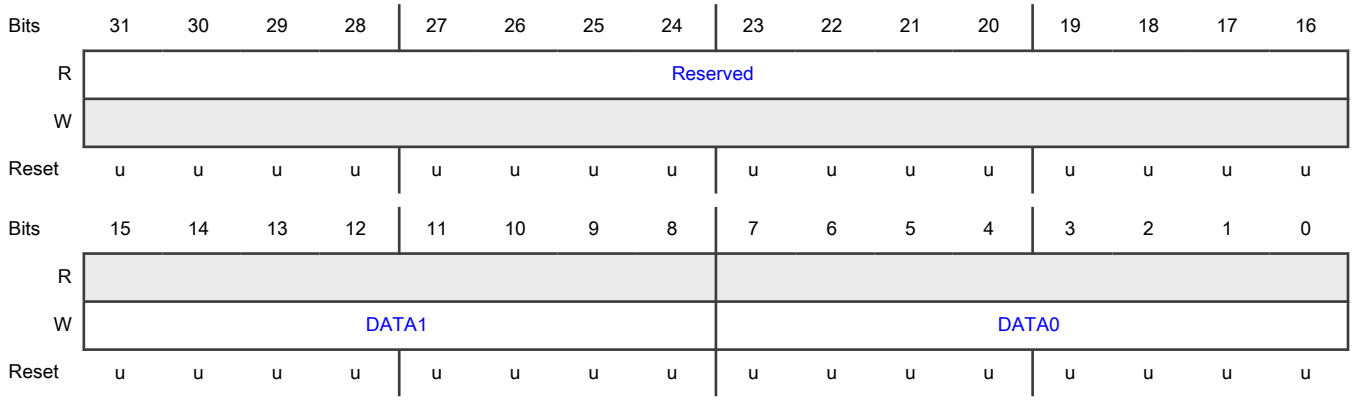
Offset

Register	Offset
SWDATAHE	3Ch

Function

Write a half-word of data, which is the end (the last byte of the half-word is the end). Write a half-word (byte pair) just like MWDATAH, but mark the 2nd bytes as end-of-data (last byte). For HDR-DDR, the byte with the END must be an even (2nd, 4th, 6th, etc) because DDR uses byte-pairs. A half-word should not be written unless there is room for both, as indicated by use of TX FIFO level trigger or TXCOUNT available space in the SDATACTRL register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DATA1	The 2nd byte to send to the master
7-0 DATA0	The 1st byte to send to the master

49.7.16 Slave Read Data Byte Register (SRDATAB)

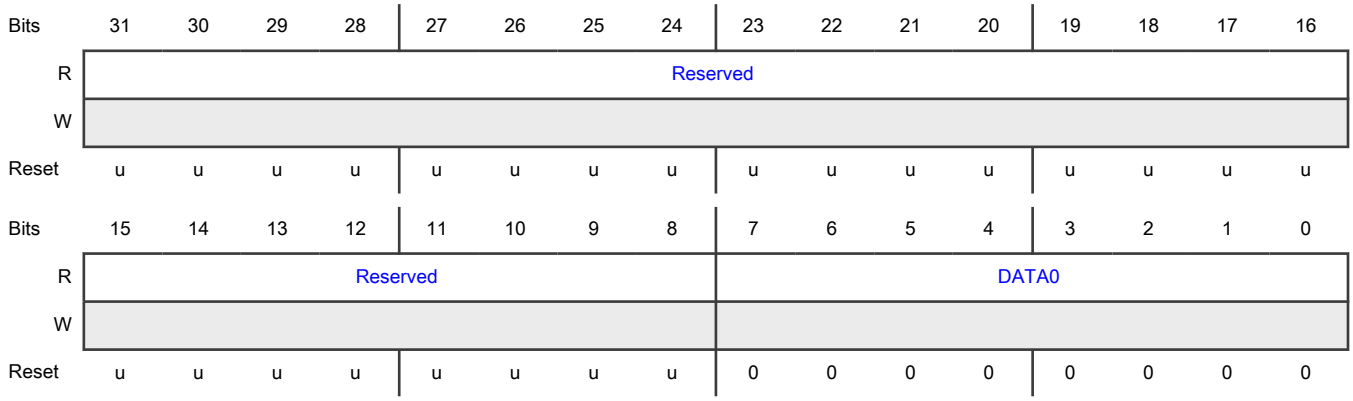
Offset

Register	Offset
SRDATAB	40h

Function

Read a byte of data. This register allows reading a byte from the bus (Master). A byte should not be read unless there is data waiting, as indicated by the RXPEND bit being set in the SSTATUS register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA0	Byte read from the master

49.7.17 Slave Read Data Half-word Register (SRDATAH)

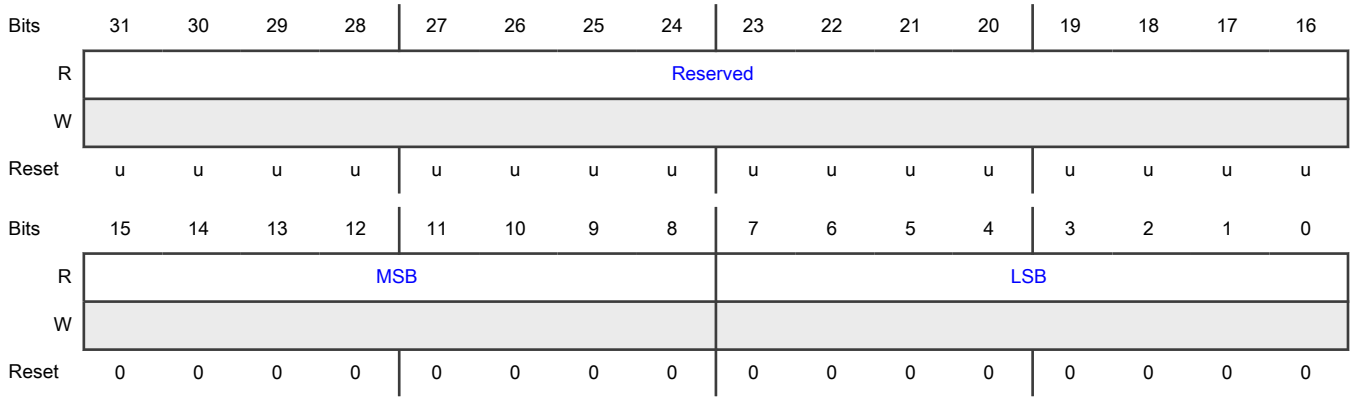
Offset

Register	Offset
SRDATAH	48h

Function

Read a half-word of data. This register allows reading a half-word (byte pair) written by the Slave on an SDR Read or DAA or DDR. This is only used when using MCTRL to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively. A half-word should not be read unless there is at least 2 bytes of data waiting, as indicated the RX FIFO level trigger or RXCOUNT available space in the SDATACTRL register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 MSB	The 2nd byte read from the slave
7-0 LSB	The 1st byte read from the slave

49.7.18 Slave Capabilities Register (SCAPABILITIES)

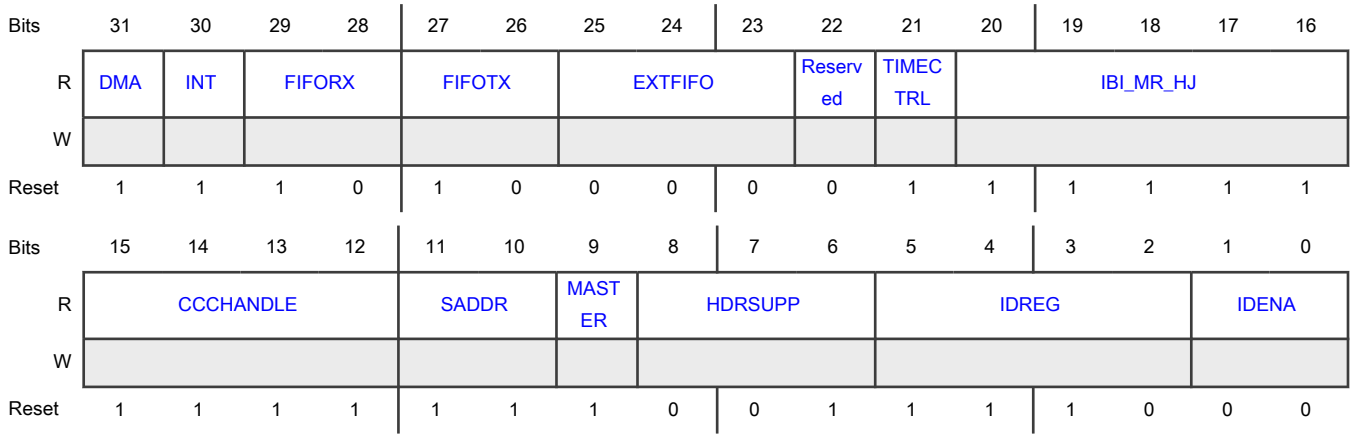
Offset

Register	Offset
SCAPABILITIES	60h

Function

Indicates which features are available and supported in this I3C module, including master and/or slave capabilities, HDR modes, and others.

Diagram



Fields

Field	Function
31 DMA	DMA 0b - DMA is not supported 1b - DMA is supported
30 INT	Interrupt 0b - Interrupts are not supported 1b - Interrupts are supported.
29-28 FIFORX	FIFO receive Indicates if RX (from-bus) is enabled and what size it is. 00b - FIFO_2BYTE. 2 (or 3)-byte RX FIFO, the default FIFO receive value (FIFORX) 01b - FIFO_4BYTE. 4-byte RX FIFO 10b - FIFO_8BYTE. 8-byte RX FIFO 11b - FIFO_16BYTE. 16-byte RX FIFO
27-26 FIFOTX	FIFO transmit Indicates if TX (to-bus) is enabled and what size it is. 00b - FIFO_2BYTE. 2-byte TX FIFO, the default FIFO transmit value (FIFOTX) 01b - FIFO_4BYTE. 4-byte TX FIFO 10b - FIFO_8BYTE. 8-byte TX FIFO 11b - FIFO_16BYTE. 16-byte TX FIFO
25-23 EXTFIFO	External FIFO Indicates whether External FIFOs are enabled. If External FIFOs are not enabled, then check FIFOTX and FIFORX for the internal FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - NO_EXT_FIFO. No external FIFO is available 001b - STD_EXT_FIFO:. Standard available/free external FIFO 010b - REQUEST_EXT_FIFO. Request track external FIFO 011b - Reserved
22 —	Reserved
21 TIMECTRL	Time control Specifies if any time-control type is supported or not. 0b - NO_TIME_CONTROL_TYPE. No time control is enabled 1b - NO_TIME_CONTROL_TYPE. ATLEAST1_TIME_CONTROL: at least one time-control type is supported
20-16 IBI_MR_HJ	In-Band Interrupts, Master Requests, Hot Join events Indicates which events (In-Band Interrupts, Master Requests, Hot Join) are to be allowed. For example, if this field is set to 00011b, it means that IBI (bit 0) and IBI_HAS_DATA (bit 1) functionality are both enabled. • Bit 0: IBI: supports the application generating an In-Band Interrupt (IBI) • Bit 1: IBI_HAS_DATA: when bit 0=1, the In-Band Interrupt (IBI) has data from the SCTRL register • Bit 2: MASTER_REQUEST: supports the application generating a Master Request for a secondary master or a Peer-to-Peer • Bit 3: HOT_JOIN: supports the application generating Hot-Join • Bit 4: BAMATCH_FOR_BUS: use the BAMATCH register for bus-available timing
15-12 CCCHANDLE	Common Command Codes (CCC) handling Indicates who handles Common Command Codes (CCC) between I3C module and the user application. • Bit 0: BLOCK_HANDLE: the block (I3C module) handles events, activities, status, HDR, and if enabled for it, ID and static address related things • Bit 1: MAX_READ_WRITE: the block handles maximum read and write lengths, and max data speed • Bit 2: PENDINT_ACTSTATE: GETSTATUS CCC (command code) will return the SCTRL register's Pending interrupt (PENDINT) and activity state (ACTSTATE) fields • Bit 3: VENDINFO: GETSTATUS CCC will return the SCTRL register's VENDINFO (vendor info) bits
11-10 SADDR	Static address Indicates how the static address is handled. 00b - NO_STATIC. No static address 01b - STATIC. Static address is fixed in hardware

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - HW_CONTROL. Hardware controls the static address dynamically (for example, from the pin strap) 11b - CONFIG. SCONFIG register supplies the static address
9 MASTER	Master Specifies if a master capability is supported or not. 0b - MASTERNOTSUPPORTED. Master capability is not supported. 1b - MASTERSUPPORTED. Master capability is supported.
8-6 HDRSUPP	HDR support Indicates which High Data Rate (HDR) modes are supported. • Bit 0: DDR: Double Data Rate • Bit 1: TSP: Ternary Symbol for pure bus (no I2C devices) - NOT SUPPORTED • Bit 2: TSL: Ternary Symbol for legacy-inclusive-bus - NOT SUPPORTED
5-2 IDREG	ID register Bits indicate what is in the registers vs. what is in the hardware. • Bit 0: ID_INSTANCE: ID Instance is a register, and is used if there is no PARTNO register. • Bit 1: IDRAND: an ID Random field is available • Bit 2: DCR: a Device Characteristic Register (DCR) is available • Bit 3: BCR: a Bus Characteristics Register (BCR) is available
1-0 IDENA	ID 48b handler Indicates who handles the ID 48b value. 00b - APPLICATION. Application handles ID 48b 01b - HW. Hardware handles ID 48b 10b - HW_BUT. In hardware but the I3C module instance handles ID 48b. 11b - PARTNO. A part number register (PARTNO) handles ID 48b

49.7.19 Slave Dynamic Address Register (SDYNADDR)

Offset

Register	Offset
SDYNADDR	64h

Function

Contains the dynamic address after being assigned; otherwise =0.

The Slave Dynamic Address register is filled in with the assigned address once the Master has assigned it via SETDASA or ENTDAA CCC commands. It will clear if the RESETDAA CCC is used. The current validity state is also indicated via the SSTATUS.DAVALID bit, which can be used to interrupt the processor

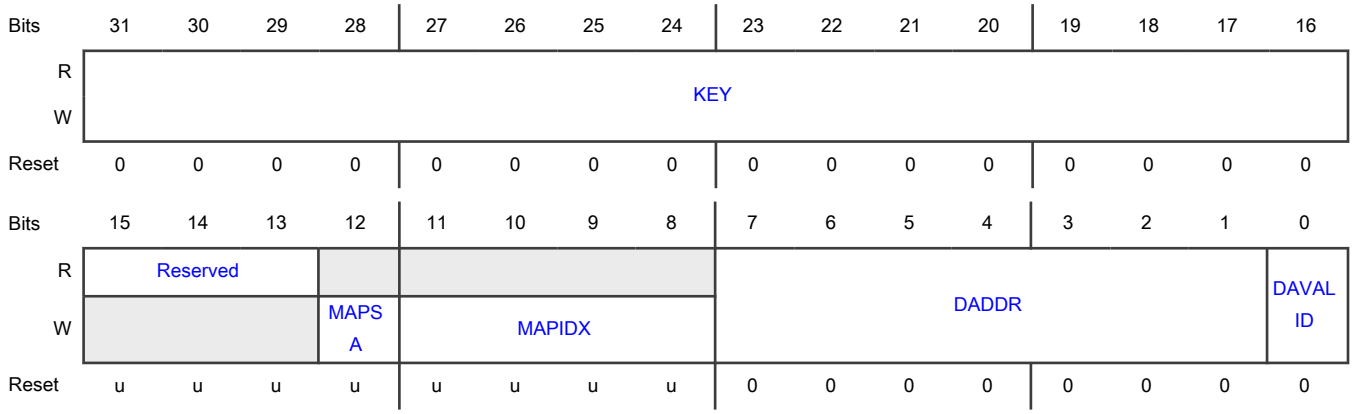
If configured to allow write, this is normally only used to restore the DA after a power-down (an ultra-low power state that loses power to peripherals but retains the DA somewhere else). This is not needed if state-retention flops are used for the DA. This mechanism only allows writes when Slave is disabled (and it will be ignored otherwise). If the Master uses RSTDAA or SETNEWDA, then it will over-ride this mechanism and cede (yield) to the master-assigned DA. Note that when when enabling the Slave, the SCONFIG.OFFLINE bit should also be set. This will wait for evidence that the bus is not in I3C HDR mode; and will exit when an HDR Exit pattern is seen or when 60 usec has expired. This makes it safe to monitor START and STOP. If the application needs to do an IBI, then the application should either wait for a STOP (see STATUS) or make sure that 200 usec have gone by with no activity (no START or STOP) before the app emits the IBI.

The MAPIDX/MAPSA model allows writing additional DAs (and SAs) into a list (the number in the list is pre-configured); any DA or SA with DAVALID=0 are never matched. The additional DAs may be based on the bridge target CCC, or done by a move (read current DA and then copy into upper map, then invalidate the 0 map) when matching ENTDAA over and over. Any mapped location can be invalidated by writing the MAPIDX and DAVALID=0. The mapped ones are not reset by the RSTDA CCC. See the SMSGMAPADDR register.

To copy the base DA to the mapped set, the configuration has to be set up to allow it, otherwise they are distinct mechanisms.

- If used for the copy model, then a special reset feature is allowed. In this case, the SDYNADDR register is written with KEY=CB19 and the rest 0. This will clear the DA and then self-clear.
- If used in I3C mode, a base DA is required, so the last one should not be cleared (or writing one with DAVALID=1 is necessary).

Diagram



Fields

Field	Function
31-16 KEY	Key Must set to 0xA4D9 to write DADDR field (and set DAVALID bit to 1). Only writable when a slave is not enabled (for restoring after power-down sleep with auto-restore). The mapped locations and base may be written when a slave is enabled, but care should be taken to not do that when there are transactions on the I3C bus. KEY reads back as 1 if overwritten, else KEY is 0 if assigned by the master, including when the master changes it. If address mapping is allowed, then writing with KEY=0xCB19 is used to clear the base DA.
15-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12 MAPSA	Map a Static Address If MAPSA=1 on a write with MAPIDX!=0, then this sets a static address into the list;, otherwise a dynamic address is used.
11-8 MAPIDX	Mapped Dynamic Address Selects which mapped DA to write to, with MAPIDX=0 meaning the DA is not mapped (which is the normal use of write SDYNADDR if allowed). This allows for a list of matching DAs or SAs (I2C static addresses). Note that the mechanism is write-only.
7-1 DADDR	Dynamic address This is the assigned Dynamic Address, when DAVALID is 1.
0 DAVALID	DAVALID Determines if a Dynamic Address is assigned. 0b - DANOTASSIGNED. A Dynamic Address is not assigned 1b - DAASSIGNED. A Dynamic Address is assigned

49.7.20 Slave Maximum Limits Register (SMAXLIMITS)

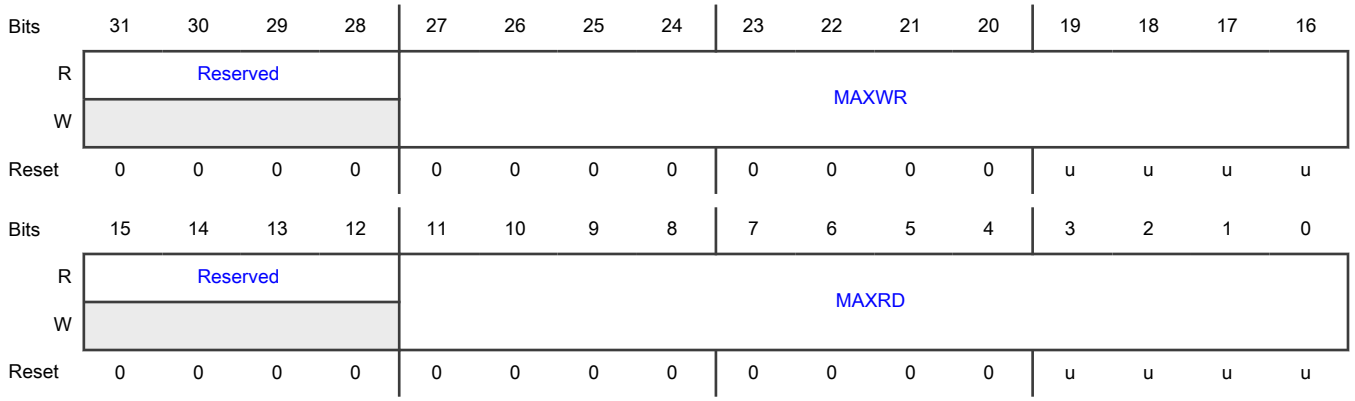
Offset

Register	Offset
SMAXLIMITS	68h

Function

Indicates the limits set by the master (or the original requested limits). The maximum limits may or may not be enabled in the hardware design, including maximum read and write lengths. If those maximum read/write lengths are enabled, then the current setting (including default request) shows up in this register (SMAXLIMITS).

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 MAXWR	Maximum write length The maximum write length must be between 8 to 4095 (saturation). The application should not set the maximum write length longer than what the master sets the maximum write length to; the application should set the maximum write length to be less than what the master sets the maximum write length to
15-12 —	Reserved
11-0 MAXRD	Maximum read length The maximum read length must be between 16 to 4095 (saturation). The application should not set the maximum read length longer than what the master sets the maximum read length to; the application should set the maximum read length to be less than what the master sets the maximum read length to.

49.7.21 Slave ID Part Number Register (SIDPARTNO)

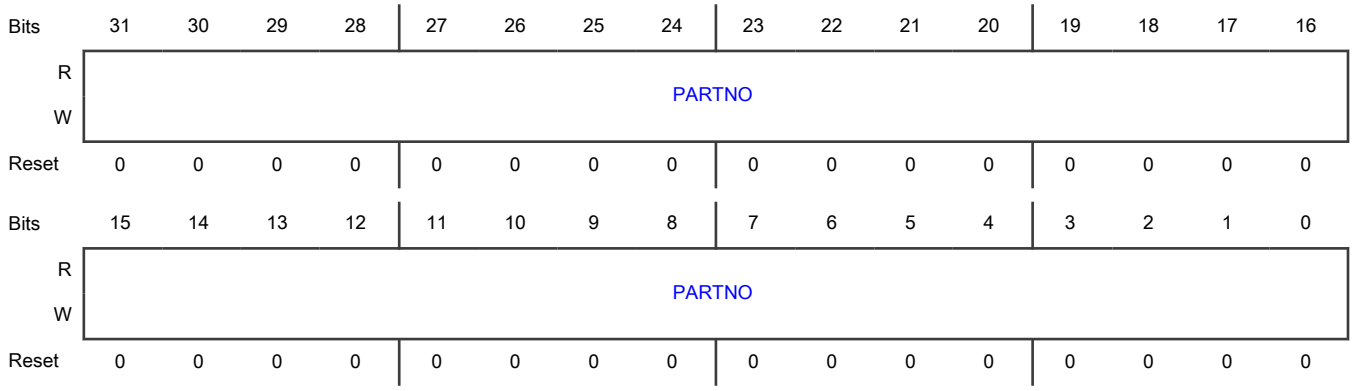
Offset

Register	Offset
SIDPARTNO	6Ch

Function

Allows an application to write the ID part-number. The application must write a value into the PARTNO field, because normally, 0 is not valid.

Diagram



Fields

Field	Function
31-0 PARTNO	Part number

49.7.22 Slave ID Extension Register (SIDEXT)

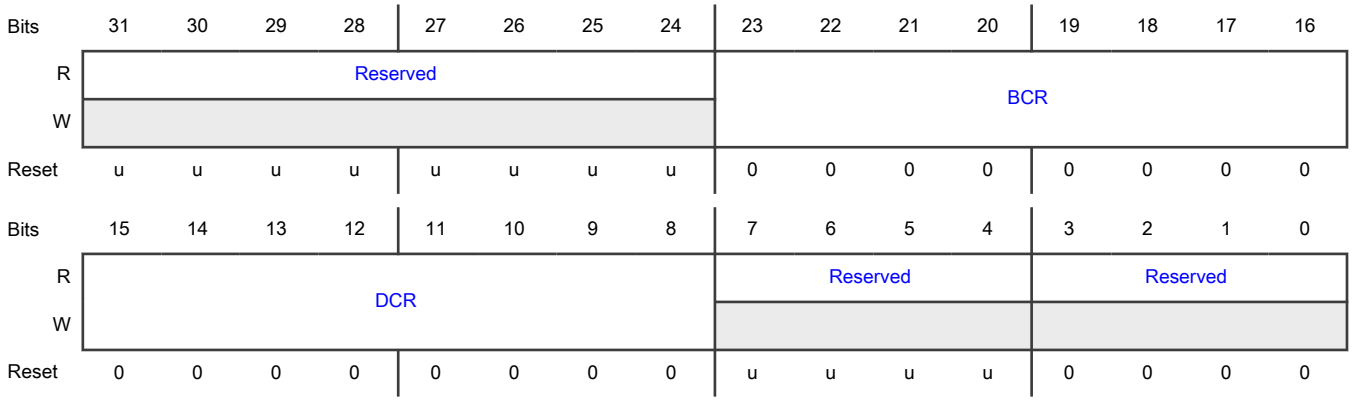
Offset

Register	Offset
SIDEXT	70h

Function

Allows an application to write the ID extension of the Device Characteristic Register (DCR) and/or the Bus Characteristics Register (BCR).

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 BCR	Bus Characteristics Register Set the Bus Characteristics Register (BCR) if configured for it. This controls features like Peer-to-Peer or Secondary Master, slow speed requirements, and others.
15-8 DCR	Device Characteristic Register Set the Device Characteristic Register (DCR) if configured for it.
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 —	Reserved

49.7.23 Slave Vendor ID Register (SVENDORID)

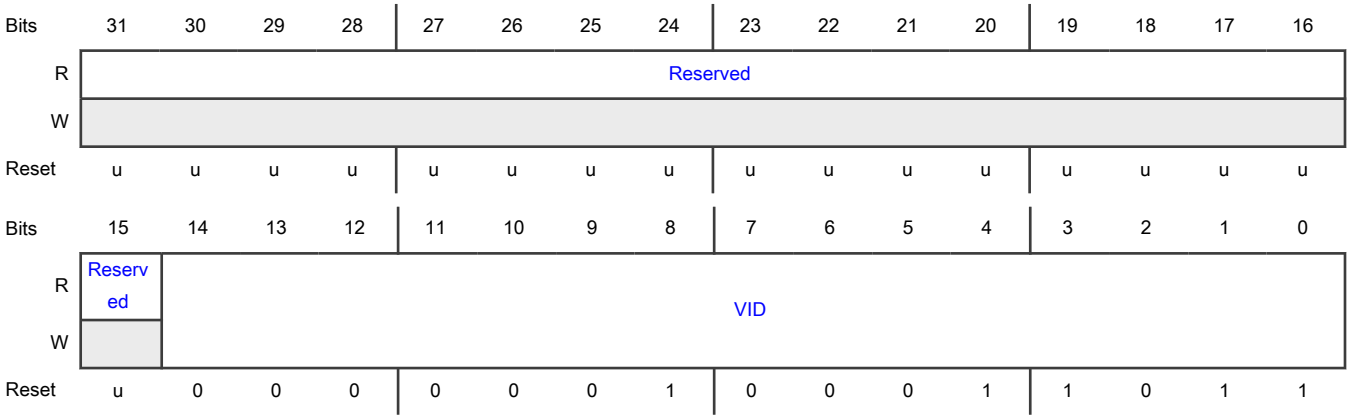
Offset

Register	Offset
SVENDORID	74h

Function

Allows an application to write the Vendor ID. The default will be set from the constant field and so usually will be the chip vendor. If using the chip vendor ID, the PARTNO must then not collide with other uses. The MIPI Vendor ID is available to all companies (MIPI membership is not required); to get a vendor ID, make a request at the mipi.org website.

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 VID	Vendor ID May be set to the 15-bit MIPI Vendor ID, if used.

49.7.24 Slave Time Control Clock Register (STCCLOCK)

Offset

Register	Offset
STCCLOCK	78h

Function

Allows an application to dynamically set the time control clock and accuracy information. The clock frequency and accuracy is normally a constant set by the hardware. But if the clock can be adjusted (i.e., divided) or if the accuracy could vary with knowable information, then the clock may be set via this register. This register should be updated whenever the clock source is changed.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FREQ								ACCURACY							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 FREQ	Clock frequency Clock frequency in 0.5-MHz steps. For example, 10 MHz is FREQ= 20. Default: set by parameters if configured.
7-0 ACCURACY	Clock accuracy Clock accuracy in 1/10ths of %. For example, 1.5% is 15. Default: set by parameters if configured.

49.7.25 Slave Message-Mapped Address Register (SMSGMAPADDR)

Offset

Register	Offset
SMSGMAPADDR	7Ch

Function

Allows an application to dynamically set the time control clock and accuracy information. When the SDYNADDR register is used to build a list of extra DAs or SAs to match, then the SMSGMAPADDR register enables the software to determine which address was matched when the SSTATUS.MATCHED bit is set. The SMSGMAPADDR register holds the last 3 matches (MAPLAST, MAPLASTM1, MAPLASTM2).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												MAPLASTM2			
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				MAPLASTM1				Reserved				MAPLAST			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 MAPLASTM2	Previous match index 2 0 for the base address.
15-12 —	Reserved
11-8 MAPLASTM1	Previous match index 1 0 for the base address.
7-4 —	Reserved
3-0 MAPLAST	Matched address index Matched address index for current or last matched message. 0 for the base address. Only valid if mapped address list is enabled.

49.7.26 Master Main Control Register (MCTRL)

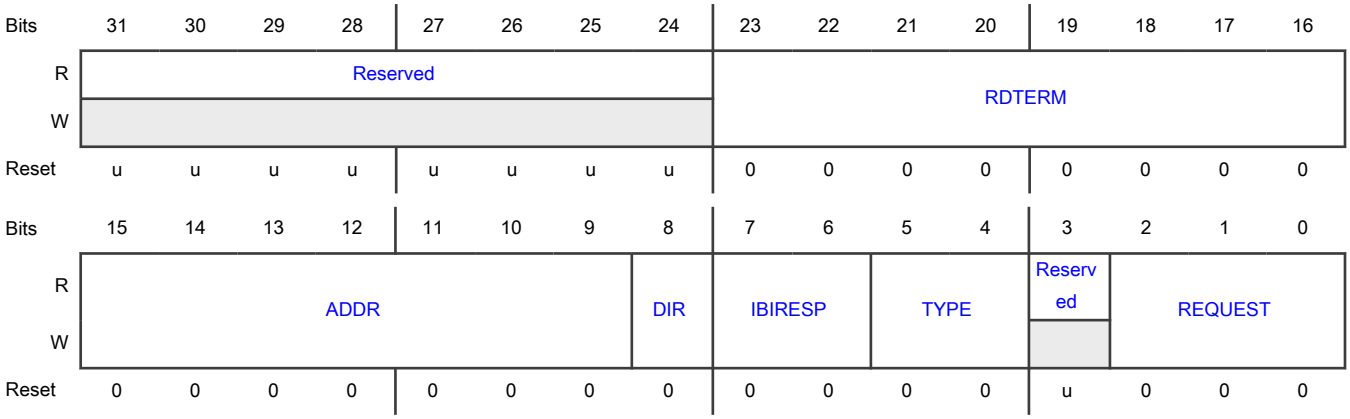
Offset

Register	Offset
MCTRL	84h

Function

Starts activities on the I3C or I2C bus.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RDTERM	Read terminate The termination counter for reads. • For I2C, RDTERM controls when to NACK the read. • For I3C, RDTERM can be use to terminate (end) a read that is too long. — RDTERM=0 has no effect — RDTERM=1 will terminate after the next character — RDTERM=2 will terminate after the next 2 characters Supports up to 255 characters. If in Double Data Rate (DDR) mode, then RDTERM will terminate the read, based on word counts (for DDR) vs. byte counts (for SDR).
15-9 ADDR	ADDR Address with START for I3C or I2C
8	DIR

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIR	0b - DIRWRITE: Write 1b - DIRREAD: Read
7-6 IBIRESP	In-Band Interrupt (IBI) response The response to use when you get an In-Band Interrupt (IBI) from START, and to force using a IbiAckNack request when completing a manual In-Band Interrupt. Completion of a manual In-Band Interrupt means that the slave Dynamic Address (DA) is known, and so the mandatory byte (or not) is specified by the application when ACKing. MIBIRESP register is also used when a message is emitted in Message Mode using the MWMSG_SDR or MWMSG_DDR registers. 00b - ACK. Acknowledge. A mandatory byte (or not) is decided by the Master In-band Interrupt Registry and Rules Register (MIBIRULES). To limit the maximum number of IBI bytes, configure the Read Termination field (MCTRL.RDTERM). 01b - NACK. Not acknowledge 10b - ACK_WITH_MANDATORY. Acknowledge with mandatory byte (ignores the MIBIRULES register). Acknowledge with mandatory byte should not be used, unless only slaves *with a mandatory byte* can cause an In-Band Interrupt. 11b - MANUAL. Stop and wait for a decision using the IBIAckNack request
5-4 TYPE	Bus type with START 00b - I3C. Normally the SDR mode of I3C. For ForceExit, the Exit pattern. 01b - I2C. Normally the Standard I2C protocol. 10b - DDR. (Double Data Rate): Normally the HDR-DDR mode of I3C. Enter DDR mode (7E and then ENTHDR0), if the module is not already in DDR mode. The 1st byte written to the TX FIFO should be a command, and should already be in the FIFO. To end DDR mode, use ForceExit. For ForceExit, the normal IBHR (In-Band Hardware Reset). 11b - For ForcedExit, this is forced IBHR.
3 —	Reserved
2-0 REQUEST	Request Emits the requested operation when doing in pieces vs. doing by message. The MSTATUS register should be checked because the state of the Master must be such that the request makes sense; for example, the system cannot do SDR from DDR (but can do DDR from SDR since it enters), cannot use an incorrect request if in DAA mode, and other similar situations. 000b - NONE. Returns to this when finished with any request. The MSTATUS register indicates the master's state. See also AutoIBI mode. NONE is only written as 0: when setting RDTERM to 1 (to stop a read in progress) or when setting IBI reponse field (IBIRESP) for MSG use. 001b - EMITSTARTADDR. Emit START with address and direction from a stopped state or in the middle of a Single Data Rate (SDR) message. If from a stopped state (IDLE), then emit start may

Table continues on the next page...

Table continued from the previous page...

Field	Function
	be prevented by an event (like IBI, MR, HJ), in which case the appropriate interrupt is signaled; note that Emit START can be resubmitted. 010b - EMITSTOP. Emit a STOP on bus. Must be in Single Data Rate (SDR) mode. If in Dynamic Address Assignment (DAA) mode, Emit stop will exit DAA mode. 011b - IBIACKNACK. Manual In-Band Interrupt (IBI) Acknowledge (ACK) or Not Acknowledge (NACK). When IBIRESP has indicated a hold on an In-Band Interrupt to allow a manual decision, this request completes it. Uses IBIRESP to provide the information. 100b - PROCESSDAA. If not in Dynamic Address Assignment (DAA) mode now, will issue START, 7E, ENTDAA, and then will emit 7E/R to process first Slave. Will stop just before the new Dynamic Address (DA) is to be emitted. The DA is written using MWDATAB bits 6:0 and then ProcessDAA is requested again to write the new address, and then starts the next; an MSTATUS indicating NACK means DA was not accepted (e.g. parity error). If Process DAA is NACKed on the 7E/R, the interrupt will indicate so and this will emit a STOP. 101b - Reserved 110b - FORCEEXIT and IBHR. Emit an Exit Pattern from any state, but end Double Data Rate (DDR) (including MSGDDR), if in DDR mode now. Includes a STOP afterward. If TYPE != 0, then it will perform an IBHR (In-Band Hardware Reset). If TYPE=2, then it does a normal reset (DEFRST can prevent the reset). If TYPE=3, it does a forced reset (will always reset). 111b - AUTOIBI. Hold in a stopped state, but auto-emit START,7E when the slave is holding down SDA to get an In-Band Interrupt (IBI). Actual In-Band Interrupt handling is defined by IBIRESP.

49.7.27 Master Status Register (MSTATUS)

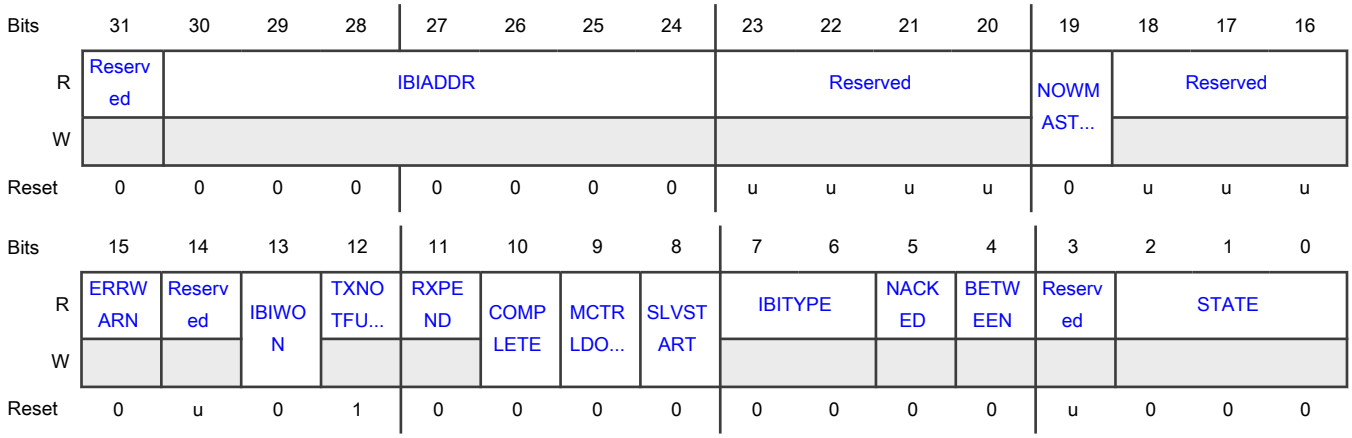
Offset

Register	Offset
MSTATUS	88h

Function

Status for the master, including which events caused the interrupts. The peripherals share the IRQ (called Parallel-to-Slave status). Because a peripheral can only be in a Master or Slave mode, but not be in both modes at the same time, then only one (Slave or Master peripheral) can be the cause of the IRQ. So, if there is an IRQ and the peripheral is a Master, then this register (MSTATUS) has the status. If there is an IRQ and the peripheral is a Slave, then the SSTATUS register has the status.

Diagram



Fields

Field	Function
31 —	Reserved
30-24 IBIADDR	IBI address The address of the In-Band Interrupt (while IBITYPE=1) or the Master Request (MR)(while IBITYPE=2) or is 7'h2 if Hot-Join (while IBITYPE=3).
23-20 —	Reserved
19 NOWMASTER	Now master (now this module is a master) NOWMASTER=1 if the module is now a master (it was previously a slave, acceptance was requested from the previous master and it was accepted). Note that the reverse operation (master becomes a slave) does not need an interrupt, because the application is granting it via the Common-Command-Code status bit MSTATUS.CCC.
18-16 —	Reserved
15 ERRWARN	Error or warning ERRWARN=1, if an error occurred (like improper register use, overrun or underrun of FIFO/buffer, invalid parity or CRC in DDR read, and others). See the Master Errors and Warnings Register (MERRWARN).
14 —	Reserved
13 IBIWON	In-Band Interrupt (IBI) won IBIWON is set to 1 if IBI or MR or HJ won the arbitration on a header address, regardless of whether it was NACKed or ACKed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 TXNOTFULL	TX buffer/FIFO not yet full TXNOTFULL=1 if TX buffer/FIFO or message register can accept another byte or half-word.
11 RXPEND	RXPEND The receiving message from a slave and byte(s) are in the input buffer/FIFO. • If using a FIFO, this is one FIFO trigger's worth. • If DMA is enabled for RX, then DMA will be signaled. RXPEND will self-clear when the data is read.
10 COMPLETE	COMPLETE A message has completed: • With MWMSG_SDR or MWMSG_DDR, this is count to 0. • With MCTRL using EmitStartAddr, this is the end of write or read by terminate or end. • With IBI (AutoIBI or EmitStartAddr), this is the end of IBI data (if any).
9 MCTRLDONE	Master control done After the module completes an MCTRL request, MCTRLDONE is set to 1 . MCTRLDONE self-clears when writing a new control; otherwise write to MCTRLDONE to clear it (MCTRLDONE). • When MCTRL.REQUEST=EmitStartAddr, MCTRLDONE fires (sets to 1) when the address went out (and was ACKed or NACKed or ended in an IBI). However if ACKed, it will then fire COMPLETE when the write or read data has completed (finished). • When MCTRL.REQUEST=ProcessDAA, this fires when the module is ready to emit the Dynamic Address (DA) for the slave, or when no more slaves are ACKing. This can be determined using the MSTATUS.BETWEEN and MSTATUS.STATE fields.
8 SLVSTART	Slave start If a slave is/was requesting a START by holding SDA low, SLVSTART is set to 1. Handling will start automatically if MCTRL.REQUEST=AutoIBI.
7-6 IBITYPE	In-Band Interrupt (IBI) type Indicates the type of In-Band Interrupt (IBI) of the last event that won the arbitration (whether the interrupt is ACKed or NACKed or pending) 00b - NONE. cleared when IBI Won bit (MSTATUS.IBIWON) is cleared 01b - IBI. In-Band Interrupt 10b - MR. Master Request 11b - HJ. Hot-Join
5 NACKED	Not acknowledged If NACKED=1, then the last Start and Address was NACKed (was not ACKed by the addressed slave).
4	Between

Table continues on the next page...

Table continued from the previous page...

Field	Function
BETWEEN	Between messages or Dynamic Address Assignments (DAA) Active if: • STATE is MSGSDR, DDR, or DAA, the state is between messages/DAAs and so is expecting a new messages/DAAs to start (or STOP or Exit). • STATE is NORMACT, the module is stuck waiting on the TX FIFO to be not-empty or the RX FIFO to be not full. 0b - Inactive. For other cases. 1b - Active. If STATE is MSGSDR, DDR, or DAA; or if STATE is NORMACT.
3 ——	Reserved
2-0 STATE	State of the master Indicates the current master state 000b - IDLE. The bus has STOPped. 001b - SLVREQ. (Slave Request state) The bus has STOPped but a slave is holding SDA low. If using auto-emit IBI (MCTRL.AutoIBI), then the master will not stay in the Slave Request state. 010b - MSGSDR. In Single Data Rate (SDR) Message state (from using MWMSG_SDR) 011b - NORMACT. Normal active Single Data Rate (SDR) state (from using MCTRL and MWDATAn and MRDATAn registers). The master will stay in the NORMACT state until a STOP is issued. 100b - MSGDDR. In Double Data Rate (DDR) Message mode (from using MWMSG_DDR or using the normal method with DDR). The master will stay in the DDR state, until the master exits using EXIT (emits the Exit pattern). 101b - DAA. In Enter Dynamic Address Assignment (ENTDAA) mode 110b - IBIACK. Waiting for an In-Band Interrupt (IBI) ACK/NACK decision 111b - IBIRCV. Receiving an In-Band Interrupt (IBI); this IBIRCV state is used after IBI/MR/HJ has won the arbitration, and IBIRCV state is also used for IBI mandatory byte (if any) and any bytes that follow.

49.7.28 Master In-band Interrupt Registry and Rules Register (MIBIRULES)

Offset

Register	Offset
MIBIRULES	8Ch

Function

Contains the rules for using In-band Interrupts, and keeps a registry of the slaves who use the IBI byte.

Concerning ADDRn fields: The address is 6 bits, and assuming that IBIRULES.MSB0=1, then each address has its most significant bit = 0. This means is that each address is 7 bits (usually written as A7 to A1), and the A7 must be 0. If the application does not use that optimal convention, then the Manual method of IBI ACK handling must be used (see MCTRL).

- The default is that the ADDRn values indicate the slaves with a mandatory byte.
- If MIBIRULES.NOBYTE is set, then the ADDRn values indicate slaves that do not have a mandatory IBI byte.

NOTE

A7=0 is only needed for Slaves that use IBI. For legacy I2C devices, A7 can use any valid value, because I2C devices cannot cause an IBI.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NOBYTE	MSB0	ADDR4						ADDR3				ADDR2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR2				ADDR1				ADDR0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NOBYTE	No IBI byte • If NOBYTE=1, then the ADDRx fields refer to slaves without a mandatory IBI byte. • If NOBYTE=0, then the ADDRx fields refer to slaves with a mandatory IBI byte.
30 MSB0	Set Most Significant address Bit to 0 If MSB0=1, then all I3C dynamic addresses will be assigned with the MSb==0 (most significant bit); this allows the START header to be optimized.
29-24 ADDR4	ADDR4 Address of slave with or without Mandatory IBI byte. See ADDR0 field description for more details.
23-18 ADDR3	ADDR3 Address of slave with or without Mandatory IBI byte. See ADDR0 field description for more details.
17-12 ADDR2	ADDR2 Address of slave with or without Mandatory IBI byte. See ADDR0 field description for more details.
11-6 ADDR1	ADDR1 Address of slave with or without Mandatory IBI byte. See ADDR0 field description for more details.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-0 ADDR0	ADDR0 Address of slave with or without Mandatory IBI byte. If 0, then the address does not apply.

49.7.29 Master Interrupt Set Register (MINTSET)

Offset

Register	Offset
MINTSET	90h

Function

Set interrupt enables for select bits in MSTATUS. Reading the MINTSET register returns the status of the interrupt enables.

- To activate an interrupt enable, write 1 to it.
- To disable an interrupt enable, write 1 to the appropriate bit in the Interrupt Clear Register (MINTCLR). Writing 0 to the interrupt enable in this register (MINTSET) does not disable the interrupt.

The Interrupt registers allow masking interrupt sources, as well as checking which interrupts have activated in the MSTATUS register. The normal method is to enable an interrupt and then once that interrupt fires, the interrupt is either cleared (by writing the MSTATUS register) or cleared by action (on the corresponding data register). The interrupt is level held, meaning that the interrupt stays set until the cause is cleared, by some method. The module prevents races, so that if a new event comes in, that new event will not be lost.

These interrupts are parallel to the Slave INTSET/INTCLR set, and only one interrupt set is active depending on state (Master or Slave operation).

- MINTSET: sets interrupt enables for MSTATUS bits. Reading MINTSET register returns the status of the interrupt enables.
- MINTCLR: clears interrupt enables for MSTATUS bits.
- MINTMASKED: returns the value of the MSTATUS bits ANDed with their interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												NOWMAST...	Reserved		
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	0	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRWARN	Reserved	IBIWON	TXNOTFU...	RXPEND	COMPLETE	MCTRLDO...	SLVSTART	Reserved							
W																
Reset	0	u	0	0	0	0	0	0	u	u	u	u	u	u	u	u

Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	Now master (now this I3C module is a master) interrupt enable
18-16 —	Reserved
15 ERRWARN	Error or warning (ERRWARN) interrupt enable
14 —	Reserved
13 IBIWON	In-Band Interrupt (IBI) won interrupt enable
12 TXNOTFULL	TX buffer/FIFO is not full interrupt enable
11 RXPEND	RX pending interrupt enable
10 COMPLETE	Completed message interrupt enable
9 MCTRLDONE	Master control done interrupt enable
8 SLVSTART	Slave start interrupt enable
7-0 —	Reserved

49.7.30 Master Interrupt Clear Register (MINTCLR)

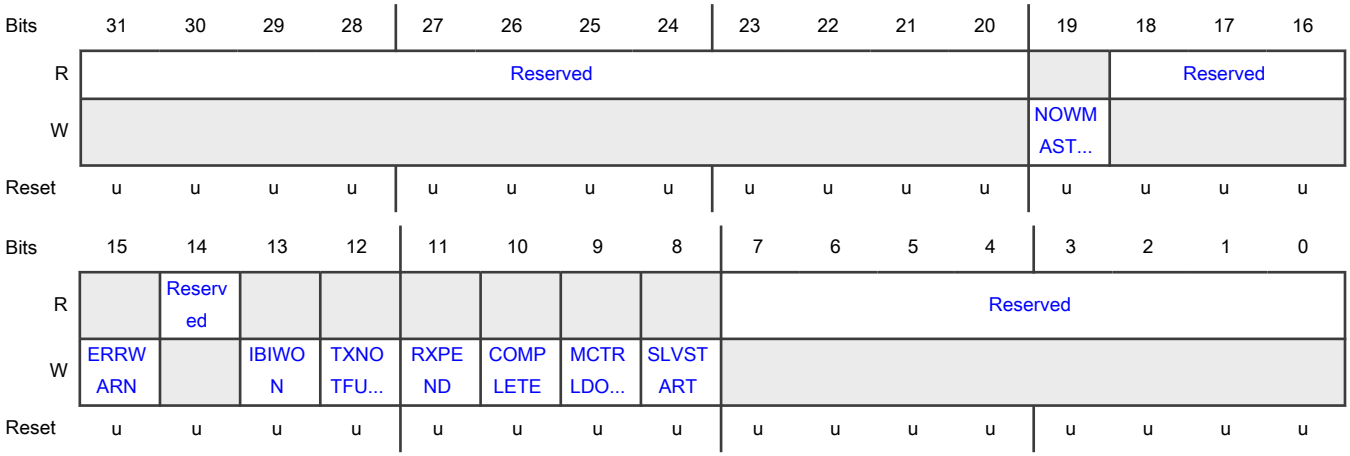
Offset

Register	Offset
MINTCLR	94h

Function

Clear interrupt enables for select MSTATUS bits. Writing a 1 clears the corresponding interrupt enable; writing a 0 has no effect.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	NOWMASTER interrupt enable clear
18-16 —	Reserved
15 ERRWARN	ERRWARN interrupt enable clear
14 —	Reserved
13 IBIWON	IBIWON interrupt enable clear
12 TXNOTFULL	TXNOTFULL interrupt enable clear
11 RXPEND	RXPEND interrupt enable clear
10	COMPLETE interrupt enable clear

Table continues on the next page...

Table continued from the previous page...

Field	Function
COMPLETE	
9 MCTRLDONE	MCTRLDONE interrupt enable clear
8 SLVSTART	SLVSTART interrupt enable clear
7-0 —	Reserved

49.7.31 Master Interrupt Mask Register (MINTMASKED)

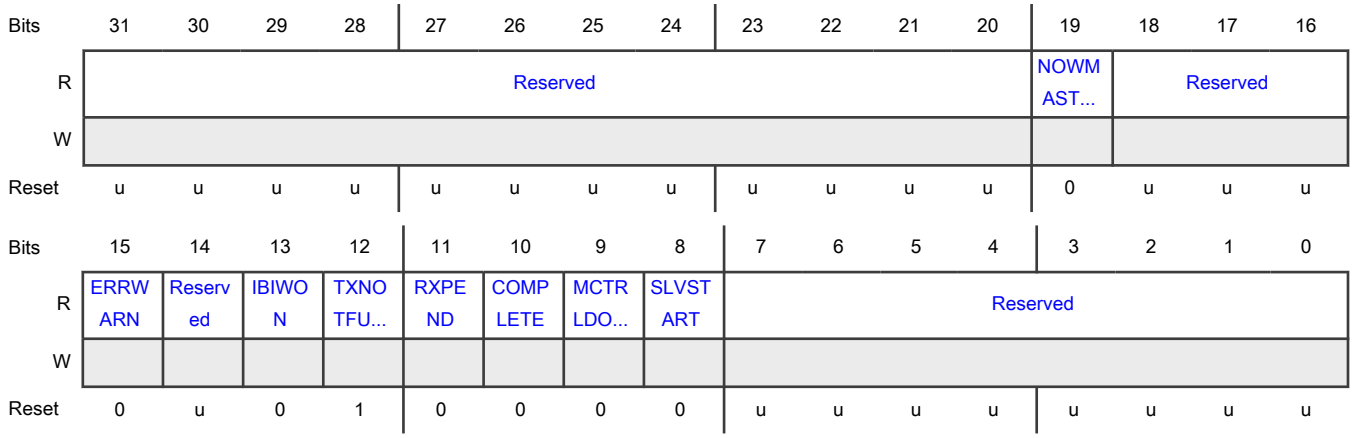
Offset

Register	Offset
MINTMASKED	98h

Function

Returns the status of enabled interrupts (the value of MSTATUS ANDed with MINTSET).

Diagram



Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 NOWMASTER	NOWMASTER interrupt mask
18-16 —	Reserved
15 ERRWARN	ERRWARN interrupt mask
14 —	Reserved
13 IBIWON	IBIWON interrupt mask
12 TXNOTFULL	TXNOTFULL interrupt mask
11 RXPEND	RXPEND interrupt mask
10 COMPLETE	COMPLETE interrupt mask
9 MCTRLDONE	MCTRLDONE interrupt mask
8 SLVSTART	SLVSTART interrupt mask
7-0 —	Reserved

49.7.32 Master Errors and Warnings Register (MERRWARN)

Offset

Register	Offset
MERRWARN	9Ch

Function

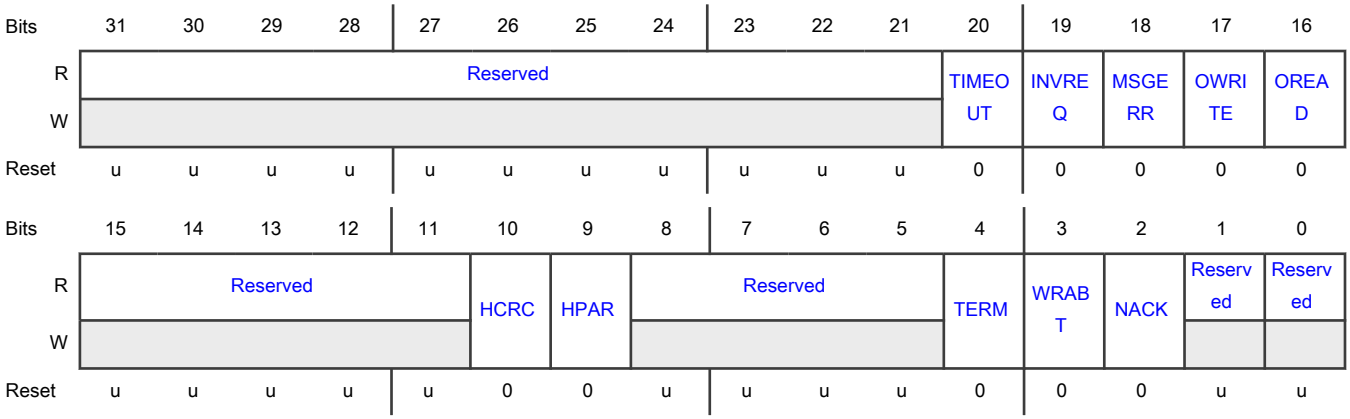
Contains errors and warnings from I3C/I2C protocol. When any errors or warnings are not 0, then the MSTATUS.ERRWARN bit will be set.

Parallel-to-slave ERRWARN:

- In Master mode, use this register (MERRWARN).
- In Slave mode, use the slave register SERRWARN.

The error bits in both registers (MERRWARN and SERRWARN) are similar in meaning.

Diagram



Fields

Field	Function
31-21 —	Reserved
20 TIMEOUT	TIMEOUT error The module has stalled too long in a frame. This happens: • When the TX FIFO or RX FIFO is not handled and the bus is stuck in the middle of a message, • When no STOP was issued and between messages, • When IBI manual is used and no decision was made. The maximum stall period is 10 KHz or 100 us.
19 INVREQ	Invalid request error Invalid use of a request: • Not using IBIAckNack when stopped in manual hold for IBI acknowledge. • Using other than ForceStop or ForceExit while in a message. Others are OK when the message is done. • Other mismatched uses (for example, IBIAckNack when in normal states).
18 MSGERR	Message error • Trying to write to or read from MWMSG_SDR register when in a DDR message (double data rate). • Trying to write to or read from MWMSG_DDR register when in a SDR message (single data rate). • Trying to read from HRMSG_SDR or HRMSG_DDR registers when no message has yet started.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 OWRITE	Over-write error Trying to write to the Master Write Data Byte Register (MWDATAB) when the FIFO is full.
16 OREAD	Over-read error Trying to read from the Master Read Data Byte Register (MRDATAB) when the FIFO is empty.
15-11 —	Reserved
10 HCRC	High data rate CRC error A Cyclic Redundancy Check (CRC) error occurred from a Double Data Rate (DDR) read.
9 HPAR	High data rate parity • Parity error from a Double Data Rate (DDR) read; this includes a bad preamble on a read. • Does not stop the read, because it is not safe to terminate (because the read data may get mis-framed). • Will end on a run of 1 second.
8-5 —	Reserved
4 TERM	Terminate error • This master terminated a slave read because athe read exceeded the count for the message; only valid when using the MWMSG_SDR or MWMSG_DDR register. • TERM self-clears if the MWMSG_SDR or MWMSG_DDR register is written.
3 WRABT	WRABT (Write abort) error • The I2C slave NACKed the write data, terminating the message. • WRABT self-clears if the MCTRL register is written. For example, the Master is writing in I2C and the Slave NACKed the write.
2 NACK	Not acknowledge (NACK) error • The slave or slaves NACKed (not acknowledged) the last address. • If 7E was the address, then all slaves NACKed the last address. • NACK self-clears if the MCTRL register is written.
1 —	Reserved
0 —	Reserved

49.7.33 Master DMA Control Register (MDMACTRL)

Offset

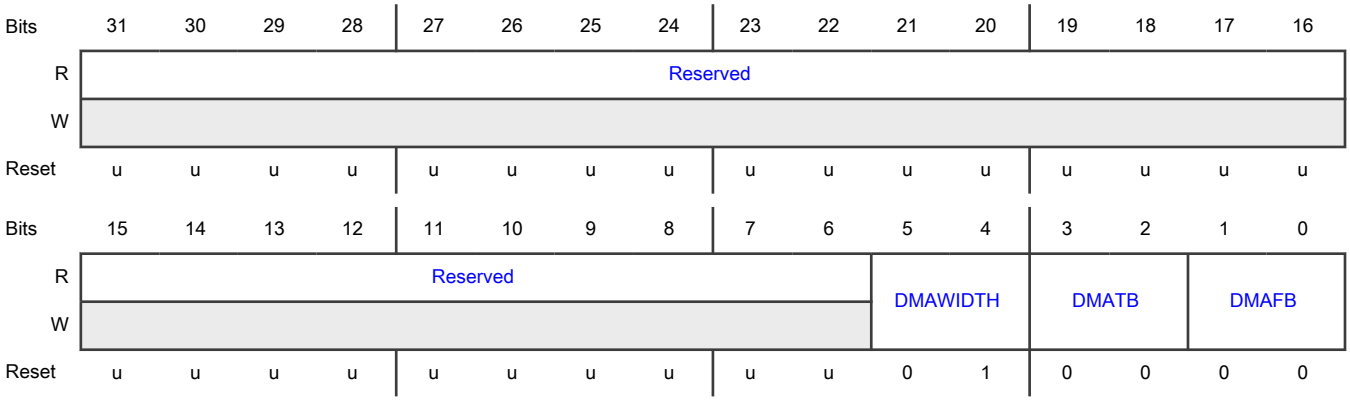
Register	Offset
MDMACTRL	A0h

Function

The DMA Control register allows for DMA to be used for inbound and outbound messages. DMA is much more useful for a Master than for a Slave, because the Master is directing the bus traffic and actions, so setting up DMA is more natural for a Master. DMA can be used with a Master in one of two ways:

- Push or Pull data to go with an MCTRL.REQUEST=EMITSTARTADDR request written by the processor.
- Implementing a message mode, to be completely DMA-controlled.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-4 DMAWIDTH	DMA width Specifies the data width of DMA operations. 00b - BYTE 01b - BYTE_AGAIN 10b - HALF_WORD. Half-word (16 bits). This will make sure that 2 bytes are free/available in FIFO. 11b - Reserved
3-2 DMATB	DMA to bus

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DMA Write (to-bus) trigger. If enabled with DMATB=1 or 2, then the I3C module will start request DMA on TX trigger. See the Master Data Control Register (MDATACTRL). The I3C module will request until full, unless DMA is set up as a trigger. DMAFB will cancel on MSTATUS.ERRWARN. 00b - NOT_USED. DMA is not used 01b - ENABLE_ONE_FRAME. DMA is enabled for 1 frame (ended by DMA or Terminated). DMATB auto-clears on STOP or START. See MCONFIG.MATCHSS. 10b - ENABLE. DMA is enabled until DMA is turned off. Normally DMA ENABLE should only be used in Master Message mode.
1-0 DMAFB	DMA from bus DMA Read (from-bus) trigger. If enabled with DMAFB=1 or 2, then the I3C module will request DMA on a RX trigger (see MDATACTL register). The I3C module will request until empty, unless the DMA is set up as a trigger. DMAFB will cancel on MSTATUS.ERRWARN. 00b - NOT_USED. DMA is not used 01b - ENABLE_ONE_FRAME. DMA is enabled for 1 frame. DMAFB auto-clears on STOP or repeated START. See MCONFIG.MATCHSS. 10b - ENABLE. DMA is enabled until the DMA is turned off.

49.7.34 Master Data Control Register (MDATACTRL)

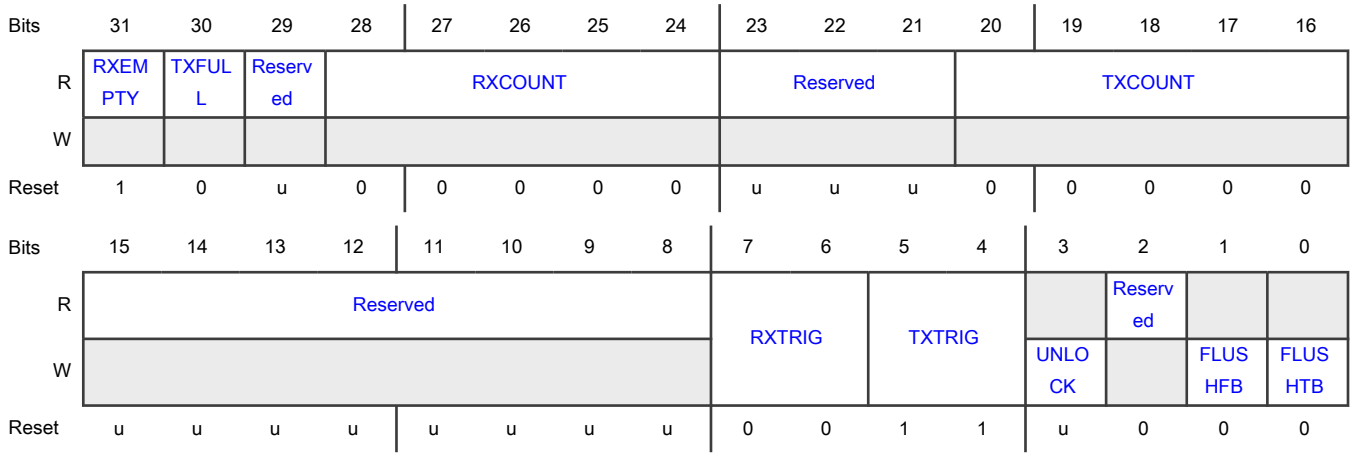
Offset

Register	Offset
MDATACTRL	ACh

Function

Controls data buffering and indicates the current buffer state. The MDATACTRL register is the alias of slave DATACTRL register (SDATACTRL). The MDATACTRL register is also the FIFO control.

Diagram



Fields

Field	Function
31 RXEMPTY	RX is empty • RXEMPTY=1 if RX is empty • RXEMPTY=0 if RX is not yet empty
30 TXFULL	TX is full • TXFULL=1 if TX is full • TXFULL=0 if TX is not yet full
29 —	Reserved
28-24 RXCOUNT	RX byte count The count of bytes in RX
23-21 —	Reserved
20-16 TXCOUNT	TX byte count The count of bytes waiting in the TXFIFO. This is how many bytes the application has written to the TXFIFO, that have not yet gone onto the I3C bus.
15-8 —	Reserved
7-6 RXTRIG	RX trigger level Trigger level for RX fullness when using a FIFO. Affects interrupts and DMA (if enabled). The default is off, and it triggers on any byte in the RXFIFO (meaning that the RXFIFO is not empty).

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 TXTRIG	TX trigger level Trigger level for TX emptiness when using a FIFO. Affects interrupts and DMA (if enabled). The default is 3.
3 UNLOCK	Unlock • If 0 is written to UNLOCK, then the RXTRIG and TXTRIG fields cannot be changed on a write. • If 1 is written to UNLOCK, then the RXTRIG and TXTRIG fields can be changed on a write.
2 —	Reserved
1 FLUSHFB	Flush from-bus buffer/FIFO Flushes the from-bus buffer/FIFO. FLUSHFB is not normally used.
0 FLUSHTB	Flush to-bus buffer/FIFO Flush the to-bus buffer/FIFO. Used when the master terminates a to-bus message (read) prematurely.

49.7.35 Master Write Data Byte Register (MWDATAB)

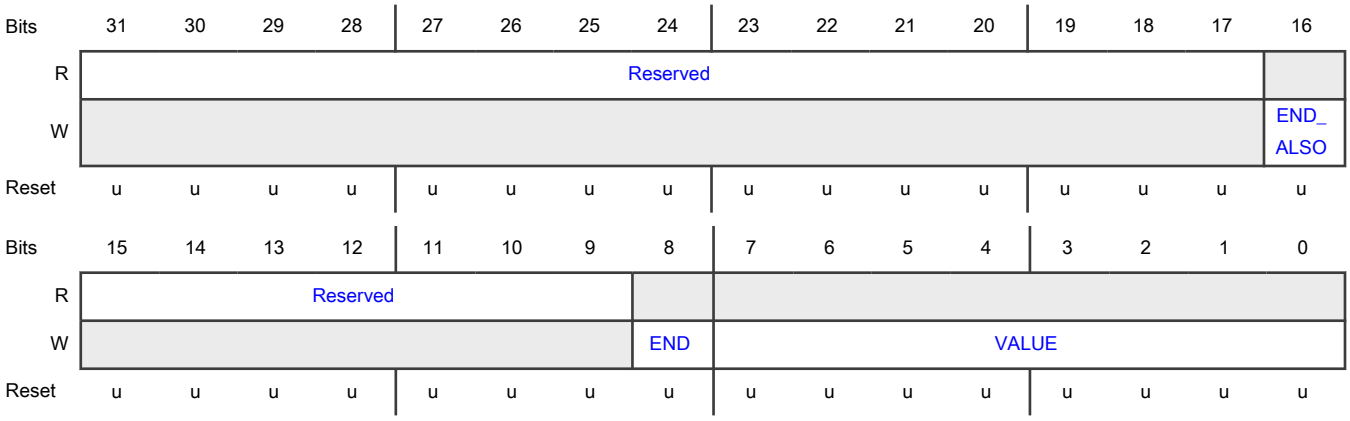
Offset

Register	Offset
MWDATAB	B0h

Function

Write a byte of data, possibly the last byte. The MWDATAB register is the alias of the WDATAB register. The MWDATAB register allows writing bytes to send onto the bus. This is only used when using MCTRL to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END_ALSO	End of message also End outbound message normally. Any message has to end, this just indicates that it is the last message to go out. This method can be used, and also the MDATABE register. • If END_ALSO=1, this marks the last byte of the message. • If END_ALSO=0, it is assumed there are more bytes. This is required to be used in I3C and is optional in I2C. For HDR-DDR (High Data Rate, Double Data Rate), the byte with the END_ALSO must be an even byte (2nd, 4th, 6th, etc) because DDR uses byte-pairs.
15-9 —	Reserved
8 END	End of message • If END=1, then the END bit marks the last byte of the message. • If END=0, then it is assumed there are more bytes. This is required to be used in I3C and is optional in I2C. For HDR-DDR (High Data Rate, Double Data Rate), the byte with the END must be an even byte (2nd, 4th, 6th, etc) because DDR uses byte-pairs.
7-0 VALUE	Data byte The byte to send to (write to) the master. • If I3C, the block will compute the parity. • If I2C, the block will handle the ACK/NACK.

49.7.36 Master Write Data Byte End Register (MWDATABE)

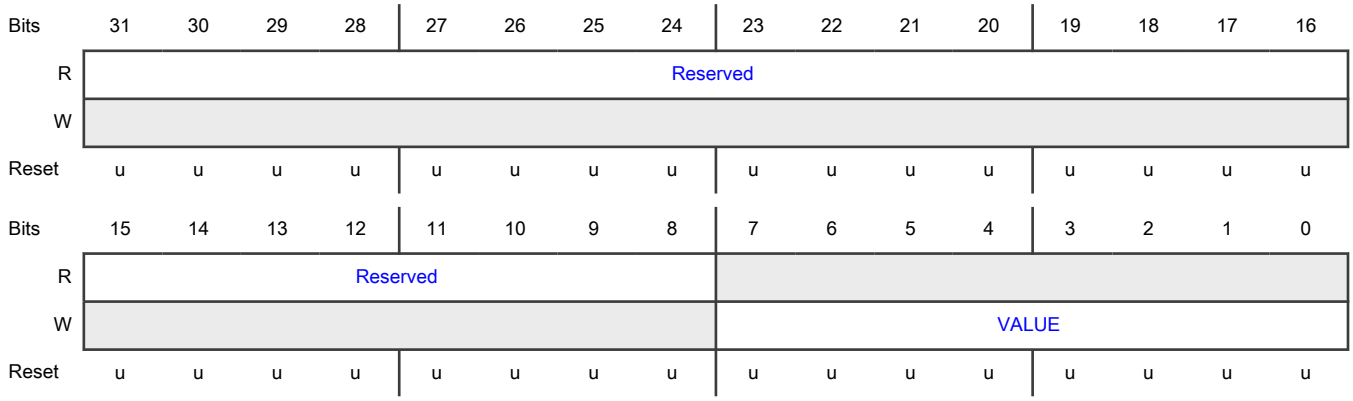
Offset

Register	Offset
MWDATABE	B4h

Function

The MWDATABE register is the alias of the Slave WDATABE register (SWDATABE). The MWDATABE register allows writing the last byte to send onto the bus. This is only used when using MCTRL to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively. Note that MWDATAB can also indicate END using bit 8.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	Data The last byte to send to (write to) the master. • If I3C, the block will compute the parity. • If I2C, the block will handle the ACK/NACK.

49.7.37 Master Write Data Half-word Register (MWDATAH)

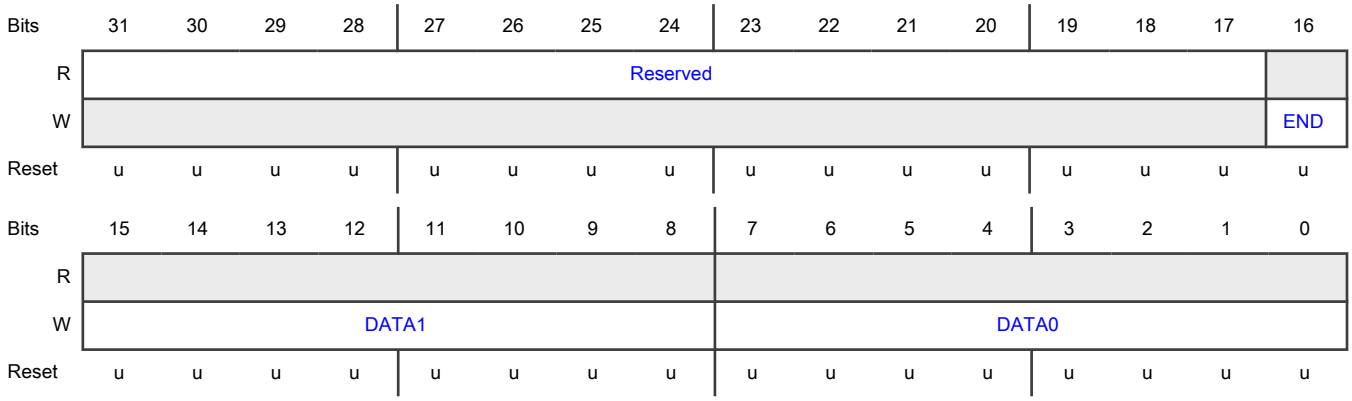
Offset

Register	Offset
MWDATAH	B8h

Function

Write a half-word of data, possibly the last half-word. The MWDATAH register is the alias of the WDATAH register. Thes Master Write Data Half-word register allows writing a half-word (pair of bytes) to the bus, unless an external FIFO is used. This takes a half-word, which will send out the Low byte and then the High byte. An end-of-data (last) marker bit is allowed (or must be 0). A half-word should not be written unless there is room for both, as indicated by use of TX FIFO level trigger or TXCOUNT available space in the MDATACTRL register.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of message • If END=1, then END bit marks the last byte of the message. • If END=0, then there are more bytes in the message. For this register, this always marks DATA1 as the end. END is required to be used in I3C, but is optional in I2C. For HDR-DDR, the byte with the END must be an even byte (2nd, 4th, 6th, etc.) because DDR uses byte-pairs.
15-8 DATA1	Data byte 1 The 2nd byte to send to the master.
7-0 DATA0	Data byte 0 The 1st byte to send to the master.

49.7.38 Master Write Data Byte End Register (MWDATAHE)

Offset

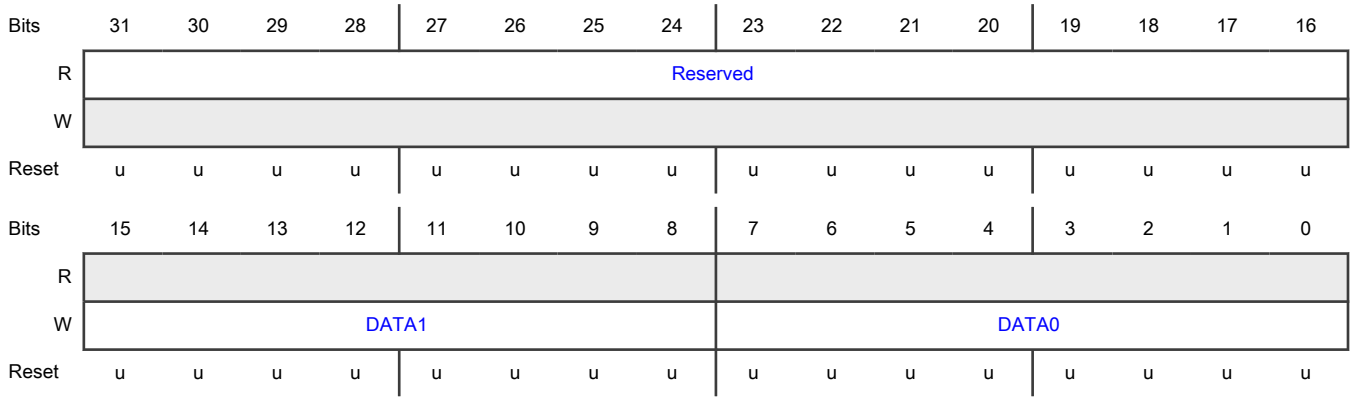
Register	Offset
MWDATAHE	BCh

Function

Write the last half-word of data. The MWDATAHE register is the alias of the Slave WDATAHE register (SWDATAHE). Write a half-word (byte pair) just like the MWDATAH register, but mark the 2nd byte as end-of-data (last byte). Note that for HDR-DDR, the byte with the END must be an even (2nd, 4th, 6th, etc) because DDR uses byte-pairs. A half-word should not

be written unless there is room for both half-words, as indicated by use of TX FIFO level trigger or TXCOUNT available space in the MDATACTRL register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DATA1	DATA 1 The 2nd byte to send to the master.
7-0 DATA0	DATA 0 The 1st byte to send to the master.

49.7.39 Master Read Data Byte Register (MRDATAB)

Offset

Register	Offset
MRDATAB	C0h

Function

Read a byte of data. The MRDATAB register is the alias of the Slave RDATAB register (SRDATAB). The MRDATAB register is used to read bytes written by the Slave on an SDR Read or DAA or DDR. This is only used when using MCTRL to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								VALUE							
W																
Reset	u	u	u	u	u	u	u	u	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	VALUE The byte read from the master (and written by the Slave).

49.7.40 Master Read Data Half-word Register (MRDATAH)

Offset

Register	Offset
MRDATAH	C8h

Function

Read a half-word of data. The MRDATAH register is the alias of the RDATAH register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSB								LSB							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 MSB	MSB The 2nd byte read from the master.
7-0 LSB	LSB The 1st byte read from the master.

49.7.41 Write Byte Data 1 (to bus) (MWDATAB1)

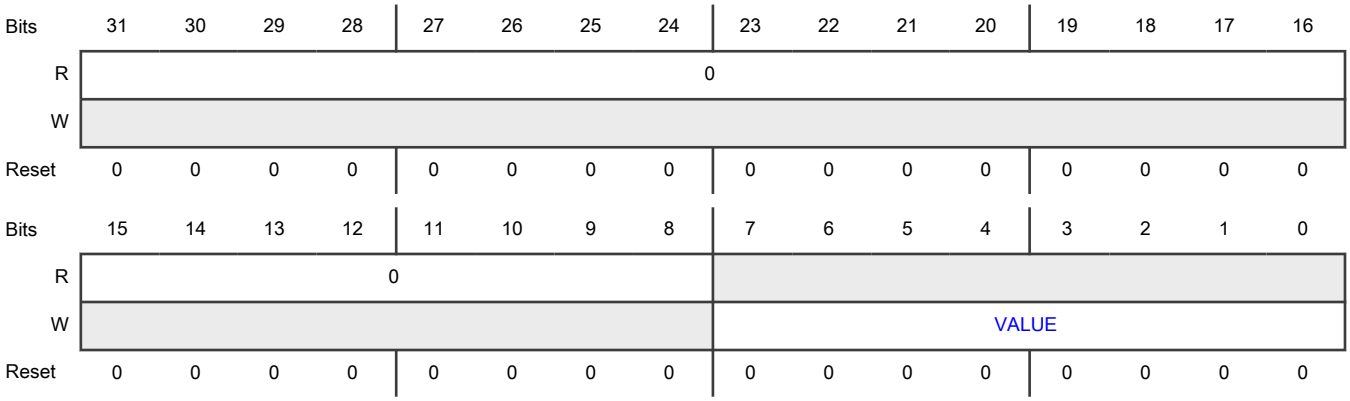
Offset

Register	Offset
MWDATAB1	CCh

Function

The MWDATAB1 register allows writing bytes to send onto the bus, and is intended for DMAs, which do not clear the upper bits of the word – MWDATAB1 does not have the END bits. MWDATAB1 is only used when using MCTRL to start the message. If MWMSG_SDR or MWMSG_DDR starts a message, that interface must be used exclusively.

Diagram



Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 VALUE	Value Byte to write out. The block computes the parity if I3C, or handles the ACK/NACK if I2C.

49.7.42 Master Write Message in SDR mode (MWMSG_SDR_CONTROL)

Offset

Register	Offset
MWMSG_SDR_CONTROL	D0h

Function

Set up and write 16-bit words in Single Data Rate (SDR) mode. The MWMSG_SDR_ register is modal, and has 2 modes: control and data.

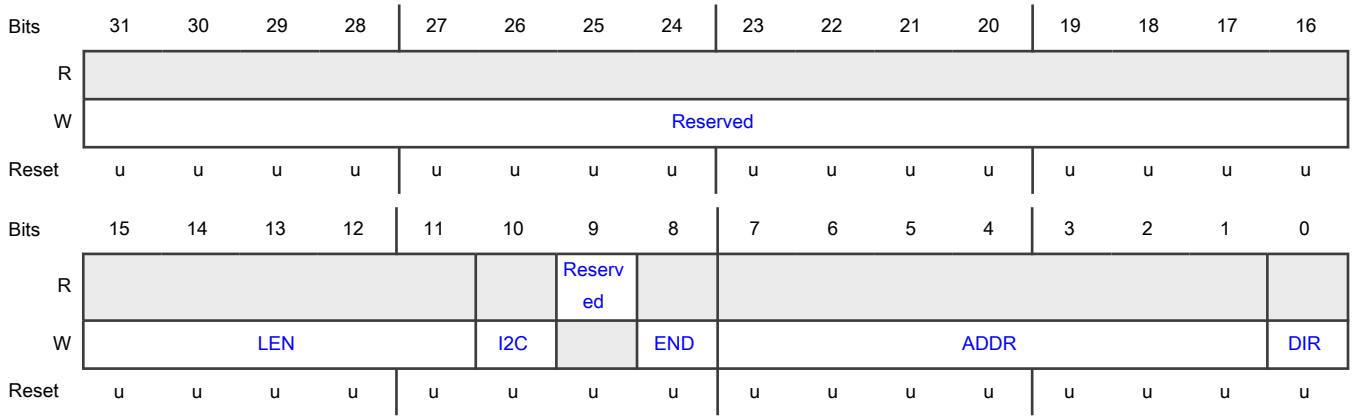
- On the first write to set up a new message, this register functions as the MWSMSG_SDR_CONTROL register.
- On subsequent writes, this register functions as the MWMSG_SDR_DATA register.

When not in the middle of a message, the MWMSG_SDR_CONTROL register is used to start a new message (or emit a STOP). After starting a message, the MWMSG_SDR_DATA register is used until the Length (see LEN field) counts down, or until data with END = 1 is used.

The MWMSG_SDR_CONTROL register is written with the 16-bit Control word if currently stopped or past the end of the previous message. If MSTATUS.STATE is MSGSDR and MSTAT.BETWEEN is 0, then the register (at this offset address) is functioning as the MWMSG_SDR_DATA register; otherwise, this register (at this offset address) is functioning as the MWMSG_SDR_CONTRL register, as long as MSTATUS.STATE is not in another mode. The control word contains the byte length (LEN), the address (ADDR), the direction (DIR), and how it ends (END). Note that if START and if an event (IBI, MR, HJ) occurs, the MCTRL register's IBIRESP field will be used to determine action, and the corresponding interrupt will occur. The message will be restarted in that case.

The MWMSG_SDR_CONTROL and MWMSG_SDR_DATA registers are oriented to DMA operations; but the MWMSG_SDR_CONTROL register can also be written by the processor if desired (instead of using MCTRL and MxDATAB).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-11 LEN	Length The byte length of the message. If LEN=0, then only END is used (and it will not use the address if STOPped). LEN=1 should not be used; the minimal LEN size is 2 bytes (LEN=2).
10 I2C	I2C Specifies if the message is I2C or I3C. 0b - I3C message 1b - I2C message
9 —	Reserved
8 END	End of SDR message • If END=1, then the single data rate message ends on STOP. • If END=0, then the single data rate message ends waiting for a new SDR message (will issue a repeated START for a new message). Sets MSTATUS.COMPLETE when done. This can be happen before LEN bytes are read in I3C if a slave ends sooner; or this can happen before LEN bytes are written in I2C if a slave NACKs.
7-1 ADDR	Address to be written to
0 DIR	Direction 0b - Write 1b - Read

49.7.43 Master Write Message Data in SDR mode (MWMSG_SDR_DATA)

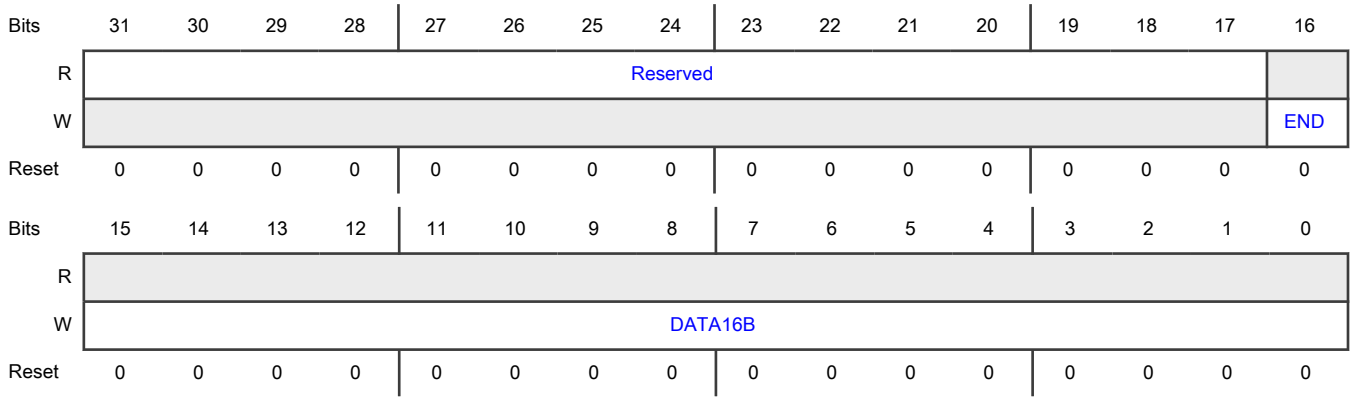
Offset

Register	Offset
MWMSG_SDR_DATA	D0h

Function

This is the 16-bit word to be written in Single Data Rate (SDR) mode. This register also functions as the MWMSG_SDR_CONTROL register.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of message • If END=1, then END bit marks the last byte of the message. • If END=0, then the normal message count is used. The default is to use END=0 and let the header count count down normally. END allows terminating a write earlier than MWMSG_SDR_CONTROL.LEN field does.
15-0 DATA16B	Data

49.7.44 Master Read Message in SDR mode (MRMSG_SDR)

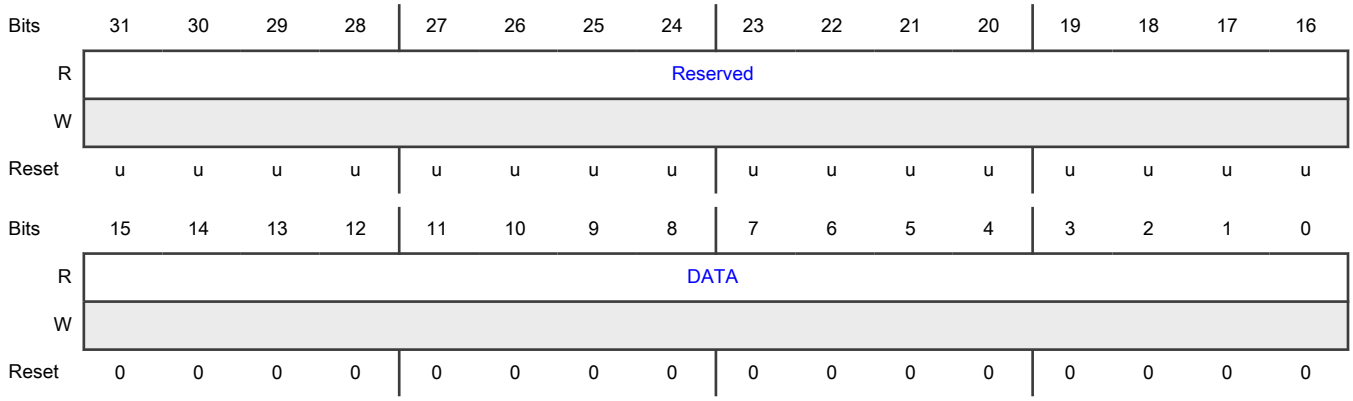
Offset

Register	Offset
MRMSG_SDR	D4h

Function

Read 16-bit words from a slave in Single Data Rate (SDR) message mode. The MRMSG_SDR register is used to read 16-bit words from an active message started with MWMSG_SDR. These words are intended to be read by DMA.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data DATA is the 16-bit word read from the slave. • If the length (LEN) was an uneven number, then the upper byte will be 0. • If the slave ends before LEN is finished, then it will treat the read as completed. • If the slave is not done before LEN is finished and END is not a continuation, then the read will be terminated.

49.7.45 Master Write Message in DDR mode (MWMSG_DDR_CONTROL)

Offset

Register	Offset
MWMSG_DDR_CONTROL	D8h

Function

Set up and write 16-bit words in Double Data Rate (DDR) mode. The MWMSG_DDR_ register is modal, and has 2 modes: control and data.

- On the first write to set up a new message, this register functions as the MWSMSG_DDR_CONTROL register.
- On subsequent writes, this register functions as the MWMSG_DDR_DATA register.

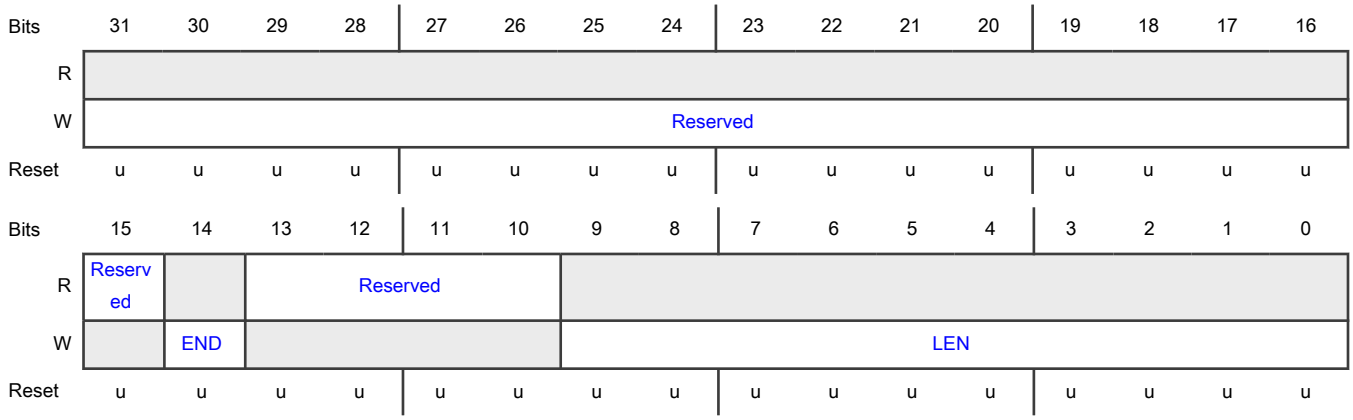
When not in the middle of a message, the MWMSG_DDR_CONTROL register is used to start a new message (or emit a STOP). After starting a message, the MWMSG_DDR_DATA register is used until the Length (see LEN field) counts down, or until data with END=1 is used.

The MWMSG_DDR_CONTROL register is written with the 16-bit Control word if currently stopped or past the end of the previous message. If MSTATUS.STATE is MSGDDR and MSTAT.BETWEEN is 0, then the register (at this offset address) is functioning as

the MWMSG_DDR_DATA register; otherwise, this register (at this offset address) is functioning as the MWMSG_DDR_CONTRL register, as long as MSTATUS.STATE is not in another mode. The main control word contains the 16-bit word length, and how it ends (stop, ready for next, continuation with more length). Then the 1st data word has the command and address for read or write. Note that if START and if an event (IBI, MR, HJ) occurs, then the MCTRL register's IBIRESP field will be used to determine action, and the corresponding interrupt will occur. The message will be restarted in that case. Note that the module handles preamble, parity, and CRC.

The MWMSG_DDR_CONTROL and MWMSG_DDR_DATA registers are oriented to DMA operations; but the MWMSG_DDR_CONTROL register can also be written by the processor if desired (instead of using MCTRL and MxDATAB).

Diagram



Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14 END	End of message • If END=1, then the double data rate message ends on HDR Exit. • If END=0, then the double data rate message ends waiting for a new DDR message (will issue a HDR Restart for the new message). Sets MSTATUS.COMPLETE when done. This can be happen before LEN bytes are read if the slave ends sooner.
13-10 —	Reserved
9-0 LEN	Length of message The length of the message (including the command) in half-words, so up to 2046 bytes. If LEN=0, then END is applied only. For Read, +1 for the CRC; for example, to read 4 bytes, use 1+2+1 for CMD + 2 halves + CRC.

49.7.46 Master Write Message Data in DDR mode (MWMSG_DDR_DATA)

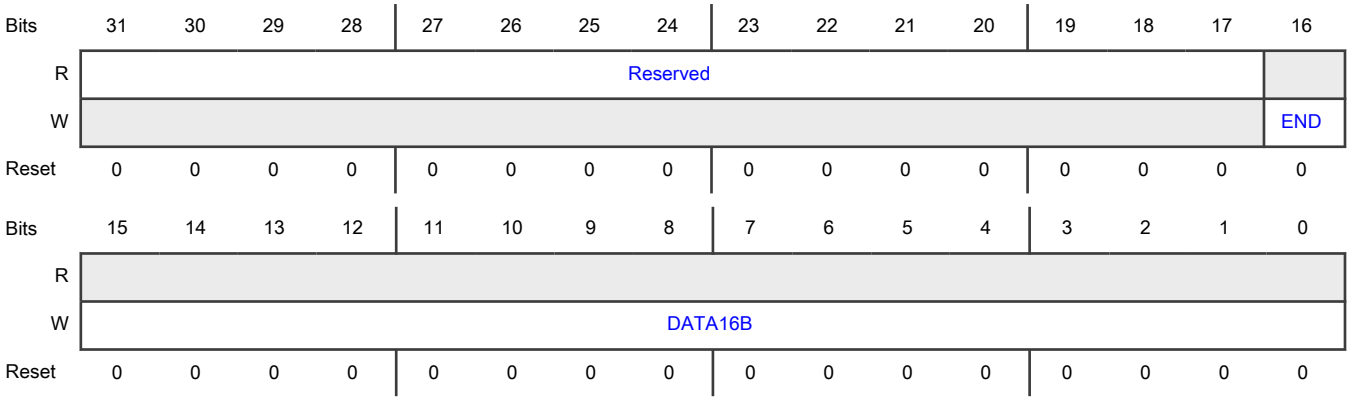
Offset

Register	Offset
MWMSG_DDR_DATA	D8h

Function

This is the 16-bit word to be written in Double Data Rate (DDR) mode. This register also functions as the MWMSG_DDR_CONTROL register.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of message • If END=1, then END bit marks the last byte of the message. • If END=0, then the normal message count is used. The default is to use END=0 and let the header count count down normally. END allows terminating a write earlier than MWMSG_DDR_CONTROL.LEN field does.
15-0 DATA16B	Data

49.7.47 Master Read Message in DDR mode (MRMSG_DDR)

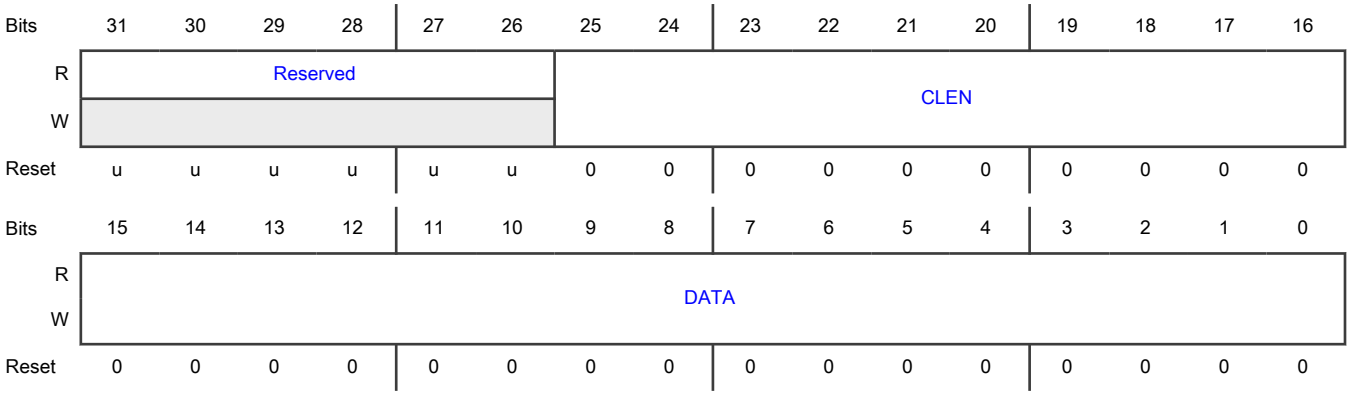
Offset

Register	Offset
MRMSG_DDR	DCh

Function

Read 16-bit words from a slave in Double Data Rate (DDR) message mode. The MRMSG_DDR register is used to read 16-bit words from an active message started with MWMSG_DDR. This is intended to be read by DMA.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 CLEN	Current length Current length. The DMA would normally only read 16 bits, and so not use the CLEN field. However, a processor can read CLEN to determine the current message length (for example, to see what the message ended on).
15-0 DATA	Data DATA is the 16-bit word read from a slave: the 1st byte is the LSB and the 2nd byte is the MSB. The 1st byte is in DATA[7:0] and the 2nd byte is in DATA[15:8]. • If the slave ends before the entire length of the message (MWMSG_DDR.LEN) is read, then the module will consider the DATA read as completed. In I3C, the slave can indicate the end of message (the last byte); otherwise, the master would terminate the message if the message is more than the master will accept. • If the slave has not yet finished sending DATA before the entire length of the message (MWMSG_DDR.LEN) is read and END is not a continuation, then the DATA read will be terminated.

49.7.48 Master Dynamic Address Register (MDYNADDR)

Offset

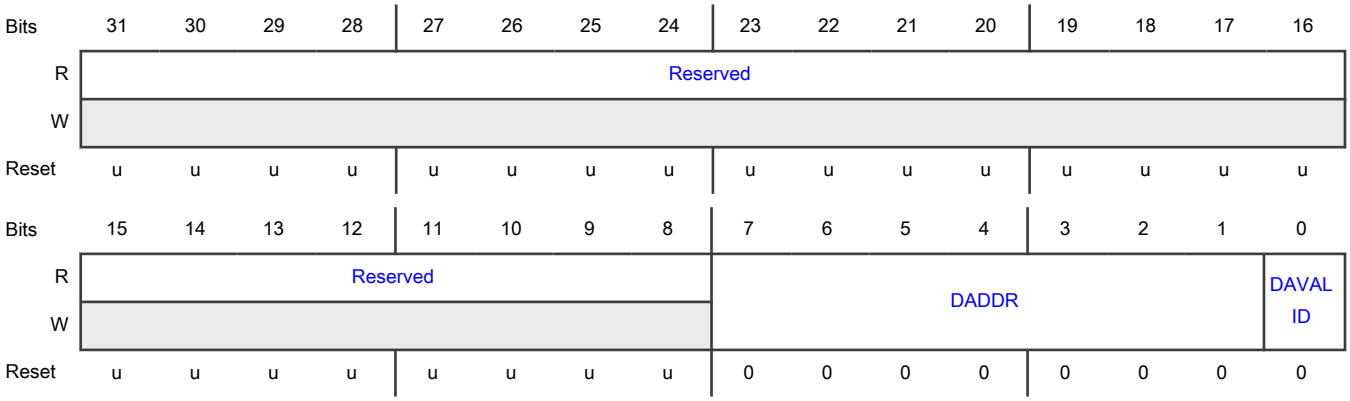
Register	Offset
MDYNADDR	E4h

Function

Allows an I3C module to write its own Dynamic Address when the I3C module changes from master mode to slave mode.

If this device is the Main Master (the Master at bus initialization), then this device may use this mechanism to assign itself its own Dynamic Address, for cases where the device hands off control to a secondary Master, and so becomes a Slave itself. This should be written before switching to Slave mode and should not be changed once in Slave mode (it is not clock safe for that). It should only be written with a valid address value in DADDR if DAVALID=1. Note that the Main master will also use DEFSLVS to define the slave addresses, including itself; this is how secondary Masters know this address. If the Master is not the Main Master, then this should not be used. If this peripheral needs to retain its DA for use during power cycle (when this peripheral is Slave), then the Slave DYNADDR register (SDYNADDR) should be used.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-1 DADDR	Dynamic address The assigned Dynamic Address when DAVALID is 1.
0 DAVALID	Dynamic address valid DAVALID=1 if a Dynamic Address is assigned.

49.7.49 Slave Module ID (SID)

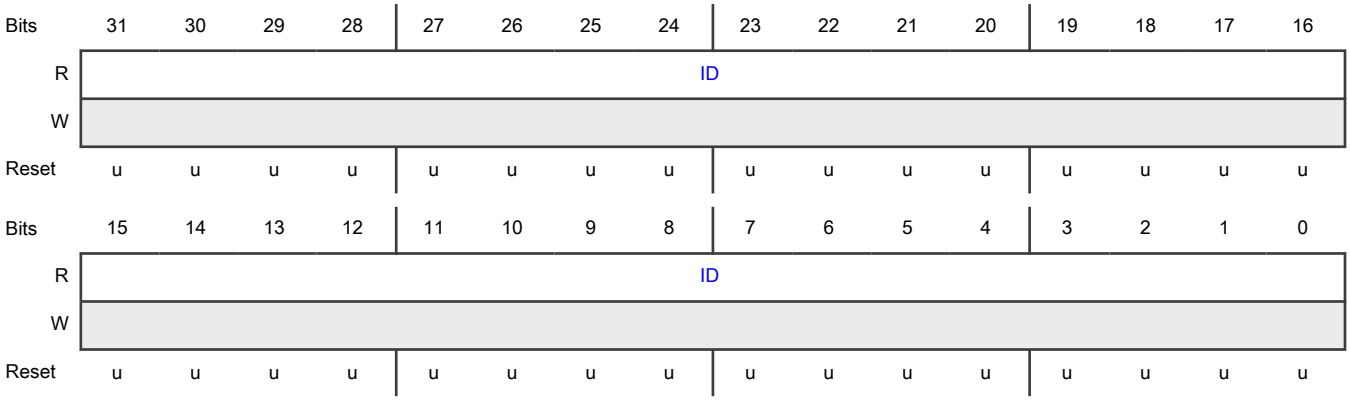
Offset

Register	Offset
SID	FFCh

Function

Allows software to detect the module and its version information.

Diagram



Fields

Field	Function
31-0 ID	ID The ID meaning is specific to each use of the I3C module.

Chapter 50
Low Power Serial Peripheral Interface (LPSPI)

50.1 Chip-specific LPSPI information

Table 300. Reference links to related information

Topic	Related module	Reference
Full description	LPSPI	LPSPI
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

50.1.1 Module instances

This device has two instances of the LPSPI module, LPSPI0, and LPSPI1.

50.1.2 Host request configuration

Both of the LPSPI0 and LPSPI1 support host request feature.

LPSPI0 does not have PCS1/HREQ pin, the related fields in CFGR0 are reserved which include LPSPI0_CFGR0[HRDIR] and 0b of the LPSPI0_CFGR0[HRSEL].

The input trigger that can be used as host request input is from TRGMUX, refer to Trigger multiplexer (TRGMUX) for more details.

50.2 Overview

LPSPI provides an efficient interface to a SPI bus, either as a master or slave. A SPI bus is a synchronous serial communication interface used in embedded systems. It is typically used to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing to Secure Digital cards and LCD displays.

50.2.1 Block diagram

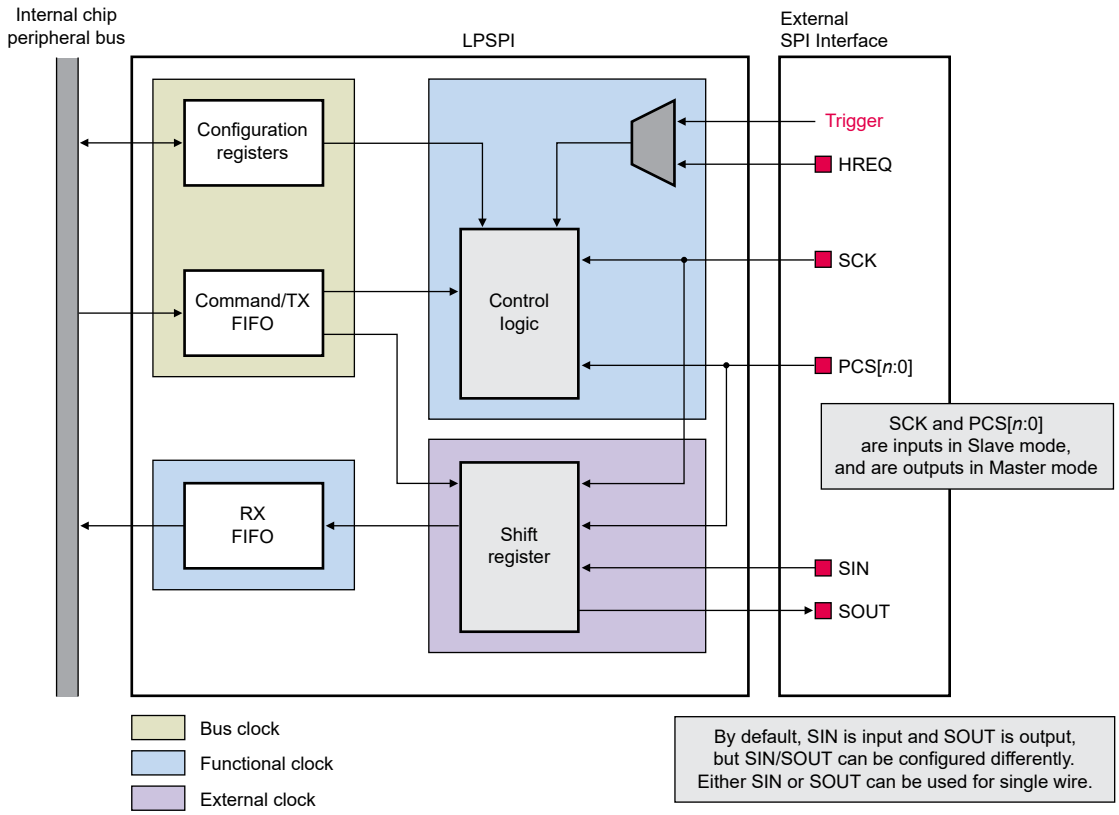


Figure 179. Block diagram

50.2.2 Features

- Requires minimal CPU overhead, with DMA offloading of FIFO register accesses
- Continues operating in Sleep mode, if configured to do so and an appropriate clock is available
- Supports DMA accesses and generates DMA requests
- 32-bit word size
- Configurable clock polarity and phase
- Master mode—supports 4 peripheral chip selects
- Slave mode
- 8-word transmit and command FIFO
- 8-word receive FIFO
- Flexible timing parameters in Master mode, including SCK frequency and duty cycle, and delays between PCS and SCK edges
- Continuous transfer option to keep PCS asserted across multiple frames
- Full-duplex transfers support 1-bit transmit and receive on each clock edge
- Half-duplex transfers support:
  - 1-bit transmit or receive on each clock edge
  - 2-bit transmit or receive on each clock edge

— 4-bit transmit or receive on each clock edge

- Can use host request to control the start of a SPI bus transfer
- Receive data match logic supports discard of non-matching data and interrupt on data match

50.3 Functional description

50.3.1 Master mode

50.3.1.1 Transmit and command FIFO commands

The transmit and command FIFO is a combined FIFO that includes both transmit data words and command words.

- You store transmit data words to the transmit and command FIFO, by writing to Transmit Data (TDR).
- You store command words to the transmit and command FIFO, by writing to Transmit Command (TCR).

When a command word is at the top of the transmit and command FIFO, the actions that can occur depend upon whether the LPSPI module is busy or between frames. See TCR[CONT] and TCR[CONTC].

Table 301. Possible actions when a command word is at the top of the transmit and command FIFO

Condition	Action
LPSPI is enabled and idle	The command word is pulled from the FIFO, and that command word controls all subsequent transfers.
LPSPI is busy and the Continuing Command field (TCR[CONTC]) is 0	The SPI frame completes at the end of the existing word, ignoring TCR[FRAMESZ]. The command word is then pulled from the FIFO and that command word controls all subsequent transfers (or until the next update to the command word). Note that a command word with TCR[CONTC] = 0 always terminates the existing transfer regardless of the previous TCR[CONT] value.
LPSPI is busy, the existing TCR[CONT] value is 1 and the new TCR[CONTC] value is 1	The command word must be updated at the frame boundary. The command word is pulled from the FIFO during the last SCK pulse of the existing frame (based on the FRAMESZ value), and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When TCR[CONTC] = 1, only the lower 24-bits of the command word are updated. If the command word is updated at a word boundary, then the transfer halts (stops) after that word. TCR[CONTC] is ignored when not at a frame boundary, so the frame ends prematurely.

About TCR[CONT] and TCR[CONTC]:

- TCR[CONT] = 1 keeps PCS asserted at end of frame, allowing the transfer to continue.
- TCR[CONTC] = 1 specifies that this command word should not terminate the existing frame, and the transfer can continue using the new command word.

TCR[CONTC] = 1 is restricted in the sense that the new command must load on a frame boundary, and the only way for a transfer to continue from a frame boundary is when the previous command has TCR[CONT] = 1.

You can read the current state of the existing command word from Transmit Command (TCR). It requires at least 3 LPSPI functional clock cycles for TCR to update after TCR is written (assuming an empty FIFO), and LPSPI must be enabled (CR[MEN] = 1).

Writing the TCR does not initiate a SPI bus transfer, unless the TCR[TXMSK] = 1. When the TCR[TXMSK] = 1, a new command word is not loaded until the end of the existing frame (based on the TCR[FRAMESZ] value); at the end of the transfer, TCR[TXMSK] transitions to 0.

In Master mode, the LPSPI command word in TCR controls SPI attributes based on selections in register fields.

Table 302. Command word in Master mode

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
CPOL	Clock polarity	Specifies the polarity of the SCK pin. Any change of CPOL value causes a transition on the SCK pin.	N
CPHA	Clock phase	Specifies the clock phase of the transfer.	N
PRESCALE	Prescaler value	Specifies a prescaler used to divide the LPSPI functional clock, to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS enables the LPSPI module to connect to different slave devices at different frequencies.	N
PCS	Peripheral chip select	Specifies which PCS pin asserts for the transfer; the polarity of PCS is static and specified by CFGR1[PCSPOL]. If CFGR1[PCSCFG] = 1, do not select PCS[3:2].	N
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.	Y
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.	Y
CONT	Continuous transfer	Configures LPSPI for a continuous transfer that keeps PCS asserted between frames (as specified by FRAMESZ). You must write a new command word to cause PCS to negate. Also supports changing the command word at the frame size boundaries.	Y
CONTC	Continuing command	Indicates that this is a new command word for the existing continuous transfer. When CONTC = 1, the command word must only be written to the transmit and command FIFO on a frame boundary.	Y
RXMSK	Receive data mask	Masks the receive data and does not store the masked receive data to the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.	Y
TXMSK	Transmit data mask	Masks the transmit data; masked transmit data is not pulled from the transmit FIFO, and the output data pin is 3-stated (unless otherwise configured by CFGR1[OUTCFG]). This option is useful for half-duplex transfers.	Y
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing to QuadSPI memory devices, and either TXMSK or RXMSK must also be 1.	Y
FRAMESZ	Frame size	Configures the frame size in number of bits equal to (FRAMESZ + 1).	Y

Table continues on the next page...

Table 302. Command word in Master mode (continued)

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
		• The minimum frame size is 8 bits. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remaining bits. For example, a 72-bit transfer consists of 3 words: the first and second words are 32 bits, and the third word is 8 bits.	

50.3.1.1.1 SPI bus transfers

LPSPI initiates a SPI bus transfer when all of the following conditions are true:

- Data is written to the transmit FIFO.
- The HREQ pin is asserted (or the HREQ function is disabled).
- LPSPI is enabled.

To perform the SPI bus transfer, LPSPI uses the attributes configured in Transmit Command (TCR) and uses the timing parameters in Clock Configuration (CCR).

The SPI bus transfer ends after the number of bits indicated by the FRAMESZ value have been transferred (provided CONT = 0), or at the end of a word when a new transmit command word is at the top of the transmit and command FIFO. When LPSPI is disabled, the SPI bus transfers end after the transmit FIFO is empty and LPSPI is idle.

The HREQ input is only checked when PCS is negated.

50.3.1.1.2 Circular FIFO

The transmit and command FIFO supports a circular FIFO feature. This feature enables the LPSPI master to (periodically) repeat a short data transfer that fits within the transmit and command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled (CFGR0[CIRFIFO] = 1), the current state of the FIFO read pointer is saved and the status flags do not update. After the FIFO is empty and LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit and command FIFO are not permanently pulled from the FIFO while Circular FIFO mode is enabled.

50.3.1.2 Receive FIFO and data match

The receive FIFO stores received data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of being stored in the receive FIFO.

- Received data is written to the receive FIFO when the last bit of the word is sampled.
- If the transmit FIFO is empty during a multiple-word or continuous transfer, then the receive data is written to the receive FIFO before the transfer stalls (assuming CFGR1[NOSTALL] = 0) while waiting for new transmit data or for a command word to be written.

LPSPI provides a receive data match function that can match received data against one of two words in DMR0 and DMR1, or against a masked data word. The received data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded due to the TCR[RXMSK] field cannot cause the data match flag to set, and delays the receive data match on the first received data word, until all discarded data is received.

- You can configure the receive data match function to discard all received data until a data match is detected, using CFGR0[RDMO].
- After a receive data match, to allow all subsequent data to be received, write 0 to the CFGR0[RDMO] field, then write 0 to SR[DMF].

50.3.1.3 Timing parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the TCR[PRESCALE] selection. Although you cannot change Clock Configuration (CCR) when the LPSPI module is busy, to support interfacing to different slave devices at different frequencies, you can change the TCR[PRESCALE] selection between SPI bus transfers using Transmit Command (TCR).

NOTE

The minimum value below is the minimum counter value, but the clock configuration register values must also satisfy the data sheet specs based on the LPSPI functional clock frequency and prescaler value.

Table 303. Timing parameters

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Min	Max
Field	Name			
SCKSET	SCK setup phase	Configures the SCK setup phase to (SCKSET + 1) cycles. The setup phase is the SCK high period when either CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. Otherwise, it is the SCK low period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
SCKHLD	SCK hold phase	Configures the SCK hold phase to (SCKHLD + 1) cycles. The hold phase is the SCK low period when either CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. Otherwise, it is the SCK high period. The SCK period is defined as (SCKSET + SCKHLD + 2) and the duty cycle is the difference between SCKSET and SCKHLD.	0 (1 cycle)	255 (256 cycles)
PCSPCS	PCS-to-PCS delay	Configures the minimum delay between PCS negation and the next PCS assertion to (PCSPCS + PCSPCS + 2) cycles. When the command word is updated between transfers, there is a minimum of (PCSPCS + 1) cycles between the command word update and any change on PCS pins.	0 (2 cycles)	255 (512 cycles)
SCKSCK	SCK-to-SCK delay	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (SCKSCK + 1) cycles. This is useful when the external slave requires a large delay between different words of an SPI bus transfer.	0 (1 cycle)	255 (256 cycles)

Table continues on the next page...

Table 303. Timing parameters (continued)

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Min	Max
Field	Name			
PCSSCK	PCS-to-SCK delay	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	SCK-to-PCS delay	Configures the minimum delay between the last SCK edge and the PCS negation to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

This figure shows the timing settings controlled by:

- TCR[CPHA]
- TCR[CPOL]
- CCR[SCKPCS]
- CCR[PCSSCK]
- CCR1[SCKSET]
- CCR1[SCKHLD]

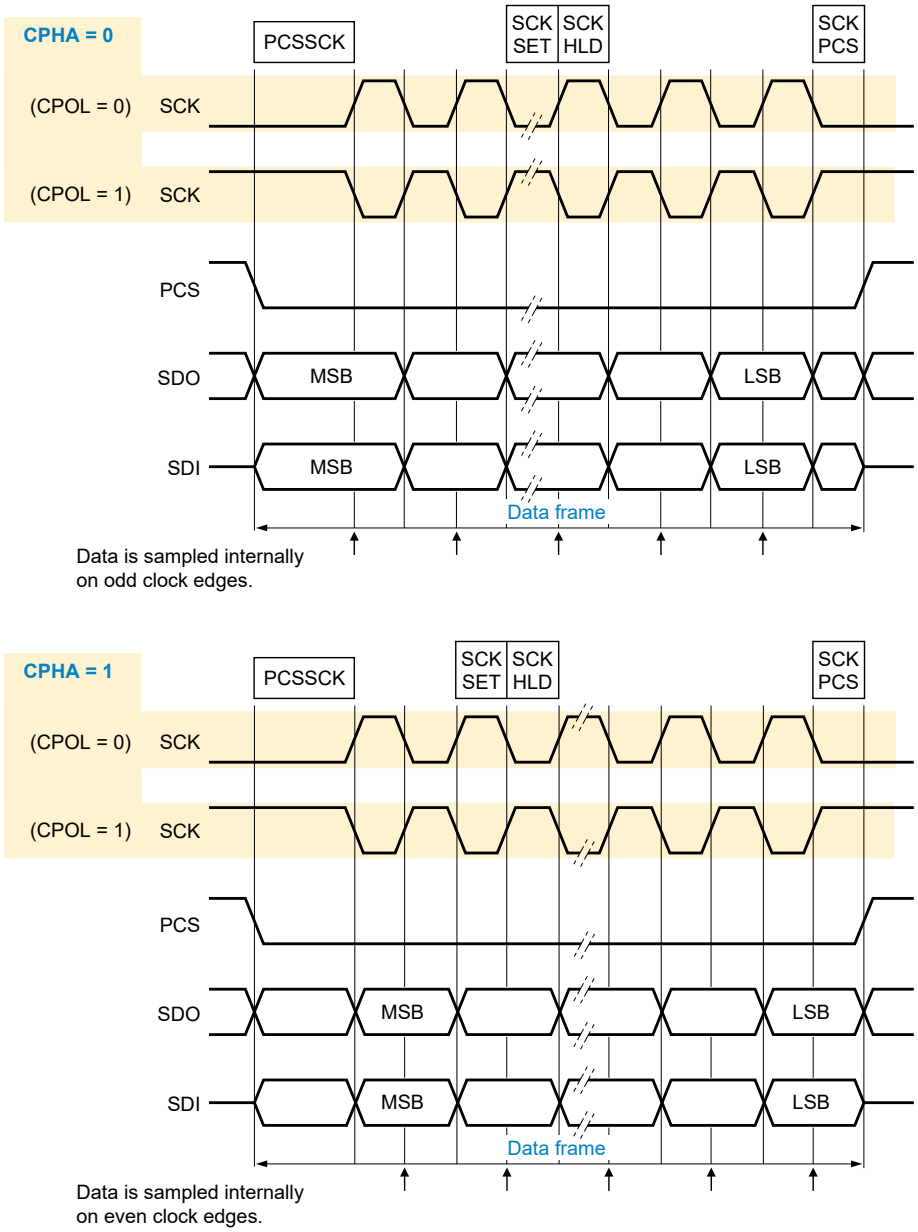


Figure 180. Clock phase (TCR[CPHA]) timing diagram example

50.3.1.4 Pin configuration

- To swap directions or support half-duplex transfers on the same pin, you can configure the SIN and SOUT pins using CFGR1[PINCFG].
- To specify whether an output data pin (SOUT, for example) 3-states when PCS is negated, or if the output data pin retains the last value, use CFGR1[OUTCFG].
- When configuring for half-duplex transfers, you must configure the output data pins to 3-state when PCS is negated (CFGR1[OUTCFG] = 1).
- When performing half-duplex 2-bit transfers, you can set CFGR1[PCSCFG] to any value.
- When performing half-duplex 4-bit transfers, you must set CFGR1[PCSCFG] to 1h.

50.3.1.5 Clock loopback

Configure the LPSPI master to use one of the following clocks to sample the input data:

- The SCK output clock
- A delayed version of the SCK output clock

The delayed version of the SCK is chosen by the SCK pin output delay, plus the SCK pin input delay, and is selected by writing 1 to CFGR1[SAMPLE]. Enabling the loopback version of the SCK can improve the setup time of the input data from the slave.

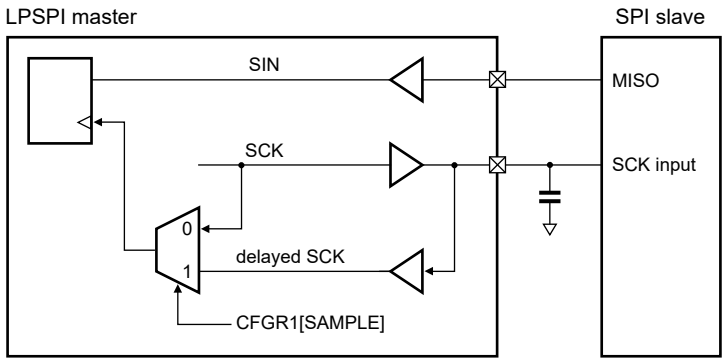


Figure 181. Clock loopback

See the chip data sheet for the specific input setup time in Master Loopback mode.

50.3.2 Slave mode

LPSPI slave mode:

- Uses the same shift register and logic that master mode uses.
- Does not use Clock Configuration Register (CCR).
- Requires that Transmit Command (TCR) remain static (unchanging) during SPI bus transfers.

50.3.2.1 Transmit and command FIFO commands

Before enabling LPSPI in Slave mode, initialize Transmit Command (TCR), although TCR does not update until after LPSPI is enabled. After LSPI is enabled, only change TCR when LPSPI is idle. In Slave mode, the LPSPI command word in TCR controls SPI attributes. Before the PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag sets.

Table 304. Command word in Slave mode

Transmit Command (TCR)		Description
Field	Name	
CPOL	Clock polarity	Specifies the polarity of the external SCK input.
CPHA	Clock phase	Specifies the clock phase of transfer.
PRESCALE	Prescaler value	Specifies the LPSPI functional clock prescaler.
PCS	Peripheral chip select	Specifies which PCS is used. The polarity of PCS is static and configured by CFGR1[PCSPOL]. If CFGR1[PCSCFG] is not equal to zero, then do not select the PCS[3:2] pins.

Table continues on the next page...

Table 304. Command word in Slave mode (continued)

Transmit Command (TCR)		Description
Field	Name	
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.
CONT	Continuous transfer	When continuous transfer is selected in Slave mode, after the number of bits indicated by FRAMESZ are transferred, the LPSPI will passthrough and transmit the received data until the next PCS negation. Whatever is shifted in on the receive data is shifted out as transmit data as if there were a 32-bit shift register.
CONTC	Continuing command	When continuing command is enabled in Slave mode, after the number of bits indicated by FRAMESZ are transferred, then RXMSK is considered equal to 1 and TXMSK is considered equal to 0 until the next PCS negation. CONTC can be used to change the direction of a transfer after the number of bits indicated by FRAMESZ.
RXMSK	Receive data mask	Masks the receive data; LPSPI does not store masked receive data to the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.
TXMSK	Transmit data mask	Masks the transmit data, so that the masked transmit data is not pulled from transmit FIFO, and the output data pin is 3-stated (unless otherwise specified by CFGR1[OUTCFG]). This option is useful for half-duplex transfers.
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing to QuadSPI memory devices, and at least one of TCR[TXMSK] or TCR[RXMSK] must be 1.
FRAMESZ	Frame size	Specifies the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contain the remainder bits. For example, a 72-bit transfer consists of 3 words: the first and second words are 32 bits, and the third word is 8 bits.

50.3.2.2 Receive FIFO and data match

The receive FIFO stores receive data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of storing the received data in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words in the DMR0 and DMR1 registers or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded because TCR[RXMSK] = 1 cannot cause the data match to set, and delays the match on the first received data word, until all discarded data is received.

- The receiver match function can also be configured to discard all received data until a data match is detected, using the CFGR0[RDMO] bit.
- After a receive data match, to allow all subsequent data to be received, first clear the CFGR0[RDMO] bit, then clear the SR[DMF] bit.

50.3.2.3 Partial received word

When the PCS pin deasserts and the receive shift register has shifted in a partial word, the receive shift register can be configured to either discard the partial word or to store it in the receive FIFO. You specify this using CFGR1[PARTIAL].

A partial word is defined as less than FRAMESZ bits (when FRAMESZ is equal or less than 32 bits, or it is the last word in a multi-word frame) or less than 32 bits (when FRAMESZ is greater than 32 bits and not the last word in a multi-word frame).

A single-bit frame is not supported. A partial received word of 1 bit is fine, but a partial received frame of 1 bit is not supported.

50.3.2.4 Clocked interface

LPSPI supports interfacing to external masters that provide only clock and data pins (PCS is not required). This interface requires:

- Writing 1 to TCR[CPHA] (data is changed on the leading edge of SCK and captured on the following edge).
- Configuring the PCS input to be always asserted (CFGR1[PCSPOLn] = 1). For example, to configure PCS[0] to be always asserted, write 1 to PCSPOL[0], and do not configure PCS[0] in the pin muxing. The chip-level drives PCS to a certain value (ideally 1), CFGR1[PCSPOL*n*] could be used to invert that value.
- Writing 1 to CFGR1[AUTOPCS] to enable automatic PCS generation. When CFGR1[AUTOPCS] = 1, a minimum of four LPSPI functional clock cycles (divided by the TCR[PRESCALE] selection) is required between the last SCK edge of one word and the first SCK edge of the next word.

50.3.3 Low-power modes

Table 305. Low-power modes

Chip mode	LPSPI operation
Run	Normal operation
Sleep/Wait	Can continue operating in Sleep mode if CR[DOZEN] = 0 and LPSPI is using an external or internal clock source that remains operating during Sleep/Wait modes
Low Power Stop (also called Deep Sleep)	Before entering Low Power Stop mode, LPSPI waits for the current transfer to finish any pending operation, temporarily ignoring CR[DOZEN].

50.3.4 Debug mode

Table 306. Debug mode

Chip mode	LPSPI operation
Debug (the core is in Debug/ Halted mode)	Can continue operating in Debug mode, if the CR[DBGEN] = 1.

50.3.5 Interrupts and DMA requests

The following table lists the Slave mode sources (status flags) that can generate LPSPI interrupts and LPSPI slave transmit and receive DMA requests.

Table 307. Interrupts and DMA requests

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit data flag	Data can be written to transmit FIFO, as configured by the transmit FIFO watermark FCR[TXWATER]	Y	TX	Y
RDF	Receive data flag	Data can be read from the receive FIFO, as configured by the receive FIFO watermark FCR[RXWATER]	Y	RX	Y
WCF	Word complete flag	Word is complete, the last bit of the word has been sampled	Y	N	Y
FCF	Frame complete flag	Frame is complete, and PCS has deasserted	Y	RX	Y
TCF	Transfer complete flag	Transfer is complete, PCS has deasserted, and the transmit and command FIFO is empty	Y	N	Y
TEF	Transmit error flag	Indicates a transmit and command FIFO underrun. In Master mode when CFGR1[NOSTALL] = 0 (transfers stall when transmit FIFO is empty), the Transmit Error Flag cannot set.	Y	N	Y
REF	Receive error flag	Indicates a receive FIFO overflow. In Master mode when CFGR1[NOSTALL] = 0 (transfers stall when receive FIFO is full), the Receive Error Flag cannot set.	Y	N	Y
DMF	Data match flag	Indicates that the received data has matched the configured data match value	Y	N	Y
MBF	Module busy flag	LPSPI is busy performing a SPI bus transfer	N	N	N

50.3.5.1 End of packet DMA transfer

The end of packet functionality is designed for serial interfaces where the size of the transfer may not be known by software in advance and the data is being pushed by an external device. Examples include UART receive, I2C Slave mode and SPI Slave mode. The end of packet processing is intended to ensure data does not become stranded in either the receive FIFO or the DMA receive buffer. Support for end of packet processing must be implemented in both the serial interfaces and the DMA controller.

The condition that signals the end of packet is different for each serial interface but is processed by the serial peripheral and DMA in the same way. For example, UART end of packet is signaled by idle line condition, I2C end of packet by STOP and/or Repeated START condition, and SPI end of packet by PCS negation.

When the serial peripheral is configured to signal end of packet condition to the DMA and an end of packet condition is detected by the serial interface, it asserts the DMA request for the receive FIFO irrespective of the watermark configuration. For larger watermark configurations, this ensures the last few words of the transfer are first flushed from the receive FIFO.

The DMA then reads the contents of the receive FIFO, depending on the first word in the FIFO:

- If the receive FIFO is empty, the serial interface signals an end of packet condition to the DMA controller.
- If the receive FIFO is not empty, but the first word in the FIFO is the start of a new packet, then data is not pulled from the receive FIFO and the serial interface signals an end of packet condition to the DMA controller.

- If the receive FIFO is not empty, and the first word in the FIFO is not the start of a new packet, the DMA will transfer the receive data as normal.

Since the DMA may be transferring multiple words on each request, the end of packet condition persists until the DMA minor loop has completed and no additional data is pulled from the receive FIFO. The status flag that triggered the end of packet condition is cleared when the minor loop completes following end of packet being signaled to the DMA controller.

When the DMA detects the end of packet condition, it writes all received words up to the end of packet into system memory and saves the destination address for the word after the last valid data. The DMA then terminates the channel as if the major loop completed, including final offsets and optional interrupts, channel linking and scatter/gather. The final destination address can optionally be saved to system memory or is available in the destination address register.

Since the DMA terminates the major loop, no servicing of the receive FIFO occurs until the DMA is reconfigured through either software or hardware (for example, channel linking or scatter-gather). This delay should be minimized to avoid receiver FIFO overrun. The automatic DMA end of packet processing is not recommended when there are only a few words transferred between end of packet conditions since the DMA will spend more time processing the end of packet than transferring the data. For example, the UART idle line length should be increased as needed to avoid an excessive number of idle conditions.

50.3.6 Clocks

Table 308. LPSPI clocks

LPSPI functional clock	• Asynchronous to the bus clock. • If the LPSPI functional clock remains enabled in low-power modes, then LPSPI can perform SPI bus transfers and low-power wakeups, in both Master and Slave modes. • LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least 2 times faster than the SPI external clock frequency (SCK).
External clock	• The LPSPI shift register is clocked directly by the SCK clock. • How the SCK clock is generated or supplied depends upon the mode (Master or Slave): — In Master mode: the SCK clock is generated internally. — In Slave mode: the SCK clock is supplied externally.
Bus clock	The bus clock is only used for bus accesses to the LPSPI control and configuration registers. The bus clock frequency must be high enough to support the data bandwidth requirements of the LPSPI registers, including the FIFOs.

See the chip-specific LPSPI information.

50.3.7 Resets

Table 309. Resets

Chip reset	Resets the LPSPI logic and registers to their default state.
Software reset	• Resets the LPSPI logic and registers to their default state, except for the Control Register. • The LPSPI software reset is controlled using CR[RST].
FIFO resets	• Resets the transmit and command FIFO and the receive FIFO. • The Reset Transmit FIFO field, CR[RTF], and the Reset Receive FIFO field, CR[RRF], are write only. • After being reset, a FIFO is empty.

50.3.8 Peripheral triggers

The connection of the LPSPI peripheral triggers to other peripherals depend upon the specific device being used.

Table 310. Peripheral triggers

Trigger	Description	Notes
Frame output trigger	The frame output trigger: • Asserts at the end of each frame (when PCS deasserts) • Remains asserted for one cycle of LPSPI functional clock divided by the PRESCALE configuration	LPSPI generates two output triggers that can be connected to other peripherals on the chip.
Word output trigger	The word output trigger: • Asserts at the end of each received word • Remains asserted for one cycle of LPSPI functional clock divided by the PRESCALE configuration	
Input trigger	To control the start of an LPSPI bus transfer, the LPSPI input trigger can be selected instead of the HREQ input. • The LPSPI input trigger is synchronized, and must assert for at least 2 cycles of the LPSPI functional clock divided by the PRESCALE configuration, so that the input trigger can be detected. • When the LPSPI module is busy, the HREQ input (and therefore the LPSPI input trigger) is ignored. • Both HREQ and the LPSPI input trigger are ignored when the module is busy. They are used to start a new transfer when the module is idle.	

50.4 Signals

Table 311. Signals

Signal	Name	Description	I/O
SCK	Serial clock	• Input in Slave mode • Output in Master mode	I/O
PCS[0]	Peripheral chip select	• Input in Slave mode • Output in Master mode	I/O
PCS[1] / HREQ	Peripheral chip select or host request	Host request pin is selected when CFGR0[HREN] = 1 and CFGR0[HRSEL] = 0 • Input in either Slave mode or when used as master host request • Output in either Master mode or when used as slave host request	I/O
PCS[2] / DATA[2]	Peripheral chip select or data pin 2 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Slave mode	I/O

Table continues on the next page...

Table 311. Signals (continued)

Signal	Name	Description	I/O
		• Output in Master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	
PCS[3] / DATA[3]	Peripheral chip select or data pin 3 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Slave mode • Output in Master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
SOUT / DATA[0]	Serial data output	Can be configured as serial data input signal • Used as data pin 0 in half-duplex parallel data transfers	I/O
SIN / DATA[1]	Serial data input	Can be configured as serial data output signal • Used as data pin 1 in half-duplex parallel data transfers	I/O

50.5 Memory map and registers

NOTE

- Writing to a read-only register or reading a write-only register can cause bus errors.
- LPSPI does not check values programmed in registers for validity, so you must take care to write valid values only.

50.5.1 LPSPI register descriptions

50.5.1.1 LPSPI memory map

LPSPI0 base address: 4003_6000h

LPSPI1 base address: 4003_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0004h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Parameter (PARAM)	32	R	0004_0303h
10h	Control (CR)	32	RW	0000_0000h
14h	Status (SR)	32	RW	0000_0001h
18h	Interrupt Enable (IER)	32	RW	0000_0000h
1Ch	DMA Enable (DER)	32	RW	0000_0000h
20h	Configuration 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match 0 (DMR0)	32	RW	0000_0000h
34h	Data Match 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration (CCR)	32	RW	0000_0000h
44h	Clock Configuration 1 (CCR1)	32	RW	0000_0000h
58h	FIFO Control (FCR)	32	RW	0000_0000h
5Ch	FIFO Status (FSR)	32	R	0000_0000h
60h	Transmit Command (TCR)	32	RW	0000_001Fh
64h	Transmit Data (TDR)	32	W	0000_0000h
70h	Receive Status (RSR)	32	R	0000_0002h
74h	Receive Data (RDR)	32	R	0000_0000h
78h	Receive Data Read Only (RDROR)	32	R	0000_0000h
3FCh	Transmit Command Burst (TCBR)	32	W	0000_0000h
400h - 5FCh	Transmit Data Burst (TDBR0 - TDBR127)	32	W	0000_0000h
600h - 7FCh	Receive Data Burst (RDBR0 - RDBR127)	32	R	0000_0000h

50.5.1.2 Version ID (VERID)

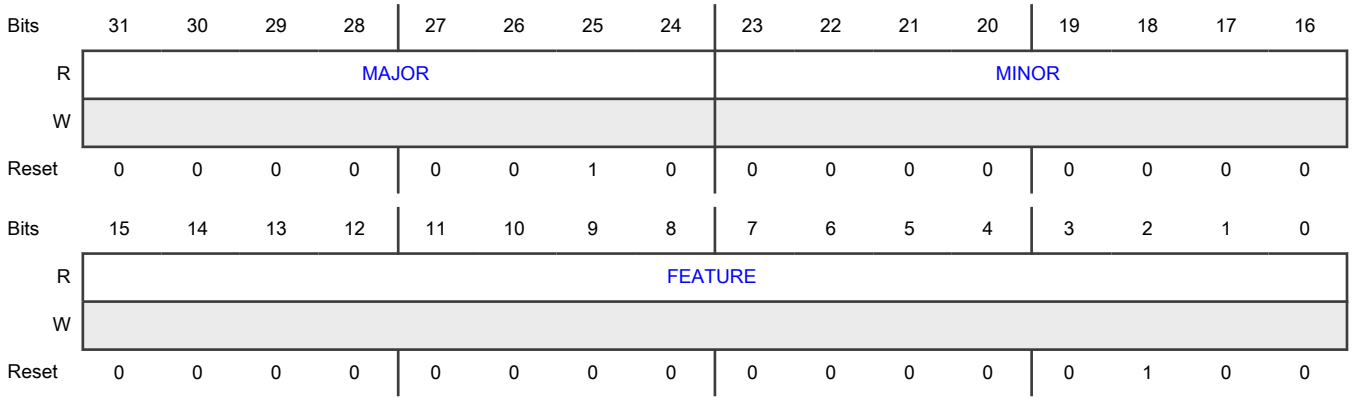
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for the module design and feature set.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.
15-0 FEATURE	Module Identification Number Indicates the feature set number. 0000_0000_0000_0100b - Standard feature set supporting a 32-bit shift register. All other values are reserved.

50.5.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains:

- Number of PCS pins
- Receive FIFO size
- Transmit FIFO size

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								PCSNUM							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Fields

Field	Function
31-24 —	Reserved
23-16 PCSNUM	PCS Number Indicates the number of PCS pins supported.
15-8 RXFIFO	Receive FIFO Size Indicates the maximum number of words in the receive FIFO. The maximum number of words is 2^{RXFIFO}.
7-0 TXFIFO	Transmit FIFO Size Indicates the maximum number of words in the transmit FIFO. The maximum number of words is 2^{TXFIFO}.

50.5.1.4 Control (CR)

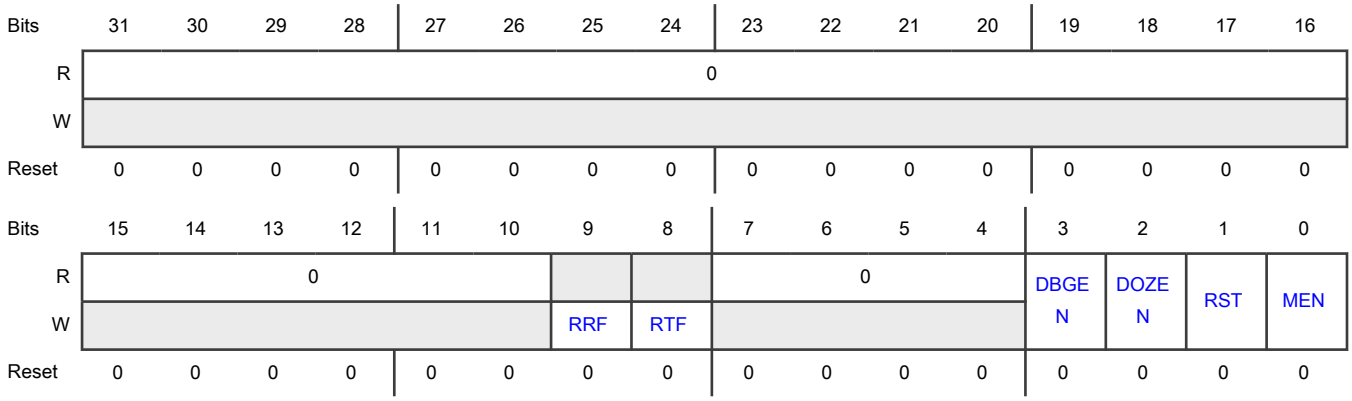
Offset

Register	Offset
CR	10h

Function

Contains fields that control module operation.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Deletes all entries in the receive FIFO. This field always reads zero. 0b - No effect 1b - Reset
8 RTF	Reset Transmit FIFO Deletes all entries in the transmit FIFO. This field always reads 0. 0b - No effect 1b - Reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables LPSPI in when the CPU is in Debug mode. When this field is 0, LPSPI is disabled when the CPU is halted; the PCS pin is deasserted after the transmit FIFO is empty regardless of the state of Transmit Command (TCR). Update this field only when the LPSPI module is disabled (MEN = 0). 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables LPSPI when the chip is in Doze mode. Update this field only when the LPSPI module is disabled (MEN = 0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
1 RST	Software Reset Resets all internal logic and registers, except Control (CR). The reset takes effect immediately and remains asserted until you write 0 to it. There is no minimum delay required before clearing the software reset by writing 0. 0b - Not reset 1b - Reset
0 MEN	Module Enable After writing 0, MEN remains set until the LPSPI has completed the current transfer and is idle. 0b - Disable 1b - Enable

50.5.1.5 Status (SR)

Offset

Register	Offset
SR	14h

Function
Contains data flow status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DMF	REF	TEF	TCF	FCF	WCF	0						RDF	TDF
W			W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-25 —	Reserved
24 MBF	Module Busy Flag In Master mode, indicates that there is data to transmit and LPSPI is able to transmit (for example, HREQ is asserted). It deasserts after the PCS pin deasserts and the LPSPI master has waited half the CCR[DBT] time with no new data to transmit. Slave mode asserts this flag when LPSPI is enabled and PCS is asserted. 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates that received data matches DMR0[MATCH0] and/or DMR1[MATCH1] (as configured by CFGR1[MATCFG]). 0b - No match 1b - Match
12 REF	Receive Error Flag Indicates a receive FIFO overflow error. When this flag is set: 1. End the transfer. 2. Empty the Receive FIFO. 3. Clear this flag. 4. Restart the transfer from the beginning. 0b - No overflow 1b - Overflow
11 TEF	Transmit Error Flag Indicates a Transmit FIFO underrun error. When this flag is set: 1. End the transfer. 2. Clear this flag. 3. Restart the transfer from the beginning. 0b - No underrun 1b - Underrun
10	Transfer Complete Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCF	In Master mode, indicates that all transfers have completed and that LPSPI has returned to Idle state and the transmit FIFO is empty. 0b - Not complete 1b - Complete
9 FCF	Frame Complete Flag Indicates that a frame transfer has completed, when the PCS deasserts. 0b - Not complete 1b - Complete
8 WCF	Word Complete Flag Indicates that the last bit of a received word is sampled. 0b - Not complete 1b - Complete
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates that the number of words in the receive FIFO is greater than the value in FCR[RXWATER]. 0b - Receive data not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag Indicates that the number of words in the transmit FIFO is equal to or less than the value in FCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data is requested

50.5.1.6 Interrupt Enable (IER)

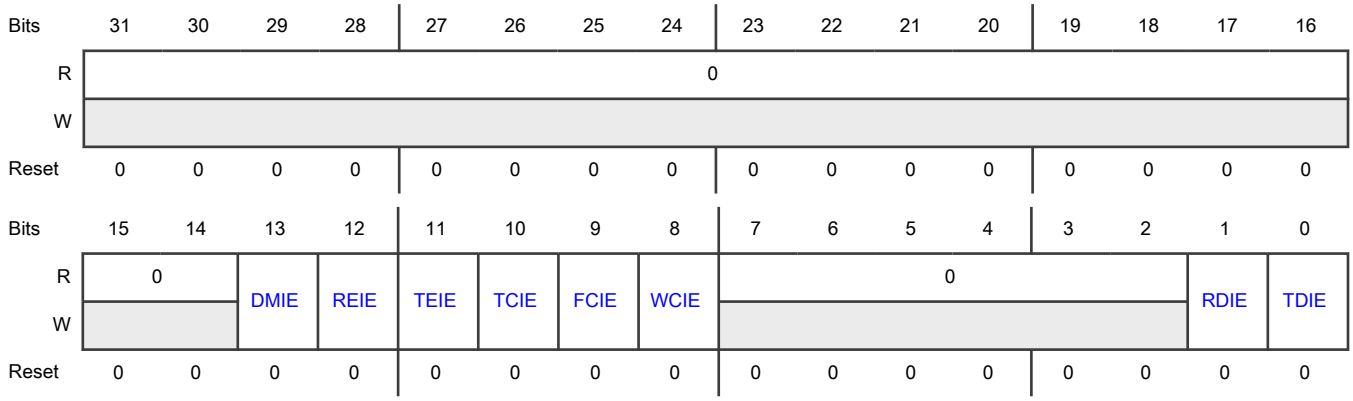
Offset

Register	Offset
IER	18h

Function

Enables interrupts based on data flow and errors.

Diagram



Fields

Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable 0b - Disable 1b - Enable
12 REIE	Receive Error Interrupt Enable 0b - Disable 1b - Enable
11 TEIE	Transmit Error Interrupt Enable 0b - Disable 1b - Enable
10 TCIE	Transfer Complete Interrupt Enable 0b - Disable 1b - Enable
9 FCIE	Frame Complete Interrupt Enable 0b - Disable 1b - Enable
8 WCIE	Word Complete Interrupt Enable 0b - Disable 1b - Enable
7-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 RDIE	Receive Data Interrupt Enable 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable 0b - Disable 1b - Enable

50.5.1.7 DMA Enable (DER)

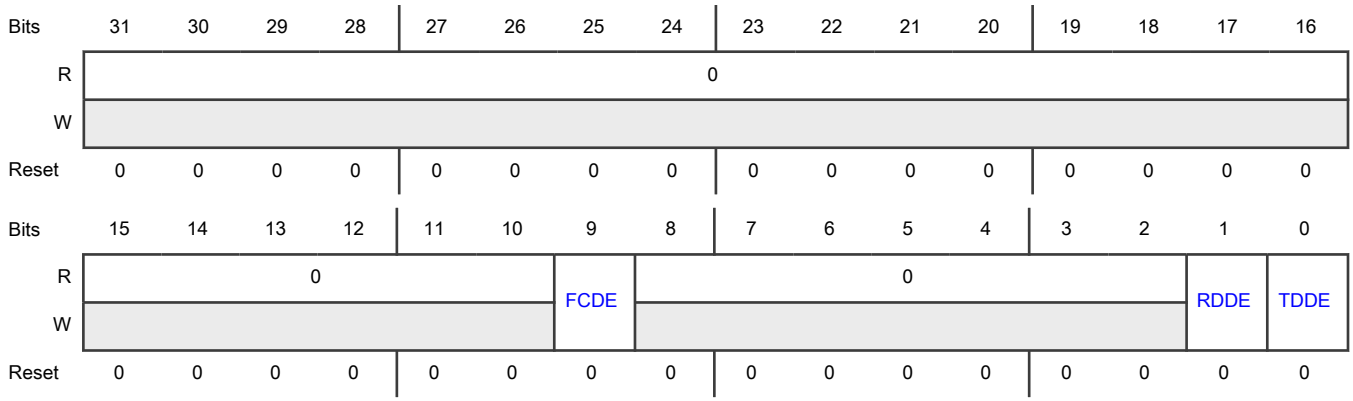
Offset

Register	Offset
DER	1Ch

Function

Enables DMA data flow.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 FCDE	Frame Complete DMA Enable Enables DMA end-of-packet processing. After the last word of a frame is read from the receive data FIFO, reading the receive data FIFO returns an end-of-packet signal with the receive data forced to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	FFFF_FFFFh. This continues until the DMA minor loop completes, and then SR[FCF] deasserts if the receive FIFO is empty or if LPSPI is busy (SR[MBF] = 1). 0b - Disable 1b - Enable
8-2 —	Reserved
1 RDDE	Receive Data DMA Enable 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable 0b - Disable 1b - Enable

50.5.1.8 Configuration 0 (CFGR0)

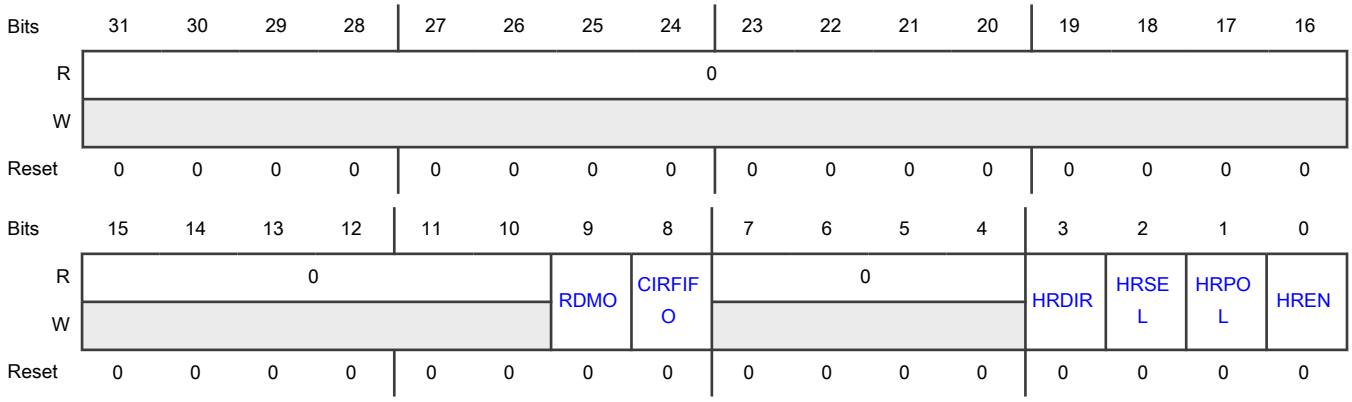
Offset

Register	Offset
CFGR0	20h

Function

Includes fields to configure LPSPI.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause the Data Match Flag (SR[DMF]) to assert is discarded. • Write 1 to this field when LPSPI is idle and SR[DMF] = 0. • After SR[DMF] = 1, this field is ignored. • To ensure that no receive data is lost when disabling RDMO, write 0 to this field before clearing SR[DMF]. See CFGR1[MATCFG] for the received data matching options. When disabled, all received data is stored in the receive FIFO. 0b - Disable 1b - Enable
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO is emptied as in normal operation, but when LPSPI is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This restoring of the read pointer causes the contents of the transmit FIFO to be cycled through repeatedly. **NOTE** The read pointer is restored for as long as this field is 1. Writing additional words to the FIFO when this field is 1 adds them to the end of the FIFO, up to the size of the transmit FIFO. 0b - Disable 1b - Enable
7-4 —	Reserved
3 HRDIR	Host Request Direction Specifies the direction of the HREQ pin. Only configure the HREQ pin as an output when LPSPI is in Slave mode. The HREQ pin direction must be an input for Master mode. 0b - Input 1b - Output
2 HRSEL	Host Request Select Specifies the source of the host request input. When the host request function is enabled with the HREQ pin, the PCS[1] function is disabled. 0b - HREQ pin 1b - Input trigger

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 HRPOL	Host Request Polarity Specifies the polarity of the HREQ pin or input trigger. 0b - Active high 1b - Active low
0 HREN	Host Request Enable In Master mode, allows LPSPI to start a new SPI bus transfer only if the host request input is asserted. When LPSPI is busy, the host request input is ignored. In Slave mode, causes the HREQ output pin to assert when data is available to be transmitted. 0b - Disable 1b - Enable

50.5.1.9 Configuration 1 (CFGR1)

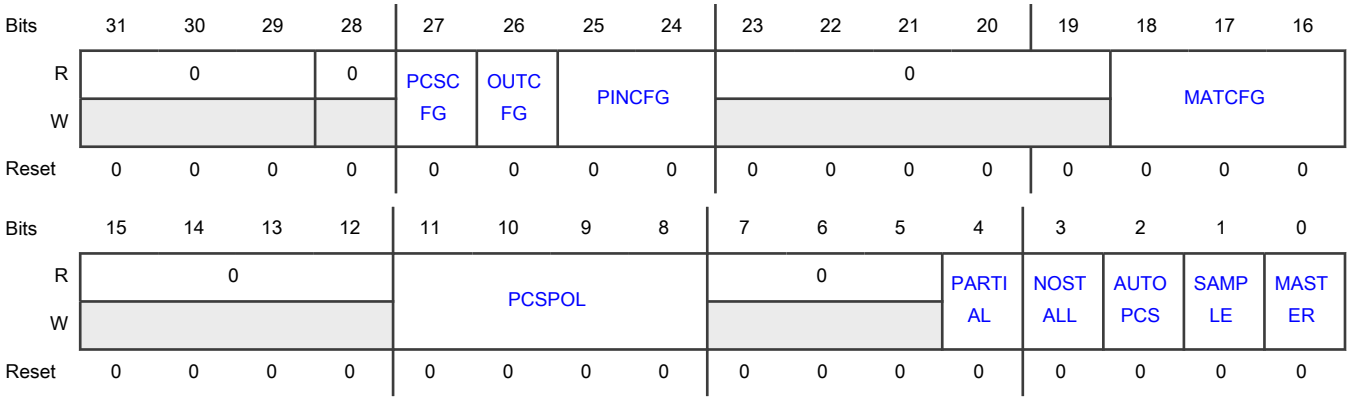
Offset

Register	Offset
CFGR1	24h

Function

Includes fields to configure LPSPI. Write to this register only when LPSPI is disabled.

Diagram



Fields

Field	Function
31-29	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28 —	Reserved
27 PCSCFG	Peripheral Chip Select Configuration Specifies the PCS pin configuration. When performing parallel transfers, this field must be configured to enable the desired transfer. 0b - PCS[3:2] are configured for chip select function 1b - PCS[3:2] are configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])
26 OUTCFG	Output Configuration Specifies whether the output data is 3-stated between accesses (when PCS is deasserted). When performing half-duplex transfers, this field must be 1. 0b - Output data retains last value. 1b - Output data is 3-stated.
25-24 PINCFG	Pin Configuration Specifies the pins used for input and output data during serial transfers. This field is ignored when performing parallel transfers. 00b - SIN is used for input data; SOUT is used for output data. 01b - SIN is used for both input and output data. Only half-duplex serial transfers are supported. 10b - SOUT is used for both input and output data. Only half-duplex serial transfers are supported. 11b - SOUT is used for input data; SIN is used for output data.
23-19 —	Reserved
18-16 MATCFG	Match Configuration Specifies the condition that causes SR[DMF] to assert.

Condition	Description
Match first data word with compare word	Match if first data word equals MATCH0 logically ORed with MATCH1 *first_data_word* == (MATCH0 || MATCH1)
Match any data word with compare word	Match if any data word equals MATCH0 logically ORed with MATCH1 *any_data_word* == (MATCH0 || MATCH1)

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Condition</th><th>Description</th></tr>
<tr><td>Sequential match, first data word</td><td>Match if first data word equals MATCH0, and second data word equals MATCH1
(first_data_word == MATCH0) && (second_data_word == MATCH1)</td></tr>
<tr><td>Sequential match, any data word</td><td>Match if any data word equals MATCH0, and the next data word equals MATCH1
(any_data_word == MATCH0) && (next_data_word == MATCH1)</td></tr>
<tr><td>Match first data word (masked) with compare word (masked)</td><td>Match if first data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1
(first_data_word && MATCH1) == (MATCH0 && MATCH1)</td></tr>
<tr><td>Match any data word (masked) with compare word (masked)</td><td>Match if any data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1
(any_data_word && MATCH1) == (MATCH0 && MATCH1)</td></tr>
</table>
000b - Match is disabled
001b - Reserved
010b - Match first data word with compare word
011b - Match any data word with compare word
100b - Sequential match, first data word
101b - Sequential match, any data word
110b - Match first data word (masked) with compare word (masked)
111b - Match any data word (masked) with compare word (masked)
</td></tr>
<tr><td>15-12
—</td><td>Reserved</td></tr>
<tr><td>11-8
PCSPOL</td><td>Peripheral Chip Select Polarity
Specifies the polarity of each PCS pin. Bit n in this field (the least-significant bit is bit 0) corresponds to PCS[n].
For each PCSPOL bit:
0b – PCS[n] pin is active low
1b – PCS[n] pin is active high
NOTE
The entire PCSPOL field is not fully supported in every LPSPI module instance. See the LSPI chip-specific information.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 PARTIAL	Partial Enable Specifies whether LSPI, when in Slave mode, stores a partial received word in the receive FIFO, or discards it, when PCS deasserts. 0b - Discard 1b - Store
3 NOSTALL	No Stall Disables a normal operating feature that causes LSPI, when in Master mode, to stall transfers when the transmit FIFO is empty or when the receive FIFO is full. This feature prevents transmit FIFO underruns and receive FIFO overruns. Writing 1 to this field disables this functionality. 0b - Disable 1b - Enable
2 AUTOPCS	Automatic PCS Enables automatic PCS generation. For correct operation in Slave mode, LPSPI requires the PCS signal to deassert between frames. Writing 1 to this field generates an internal PCS signal at the end of each transfer word when TCR[CPHA] = 1. When this field is 1, SCK must remain idle for at least 4 LPSPI functional clock cycles, divided by the prescaler selected (see TCR[PRESCALE]) between each word, to ensure correct operation. This field is ignored in Master mode. 0b - Disable 1b - Enable
1 SAMPLE	Sample Point Specifies the SCK clock edge on which LSPI, when in Master mode, samples input data. Writing 1 to this field causes LSPI to sample input data on a delayed loopback SCK clock edge, which improves the setup time when sampling data (see Clock loopback). In this configuration. The input data setup time in Master mode is equal to the input data setup time in Slave mode. In Slave mode, this field is ignored. 0b - SCK edge 1b - Delayed SCK edge
0 MASTER	Master Mode Specifies the LPSPI operating mode—either Master or Slave. This field directly controls the direction of the SCK and PCS pins. 0b - Slave mode 1b - Master mode

50.5.1.10 Data Match 0 (DMR0)

Offset

Register	Offset
DMR0	30h

Function

Specifies match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register while CFGR0[RDMO] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MATCH0	Match 0 Value MATCH0 value to be compared against received data.

50.5.1.11 Data Match 1 (DMR1)

Offset

Register	Offset
DMR1	34h

Function

Specifies match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register while CFGR0[RDMO] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MATCH1	Match 1 Value MATCH1 value to be compared against received data.

50.5.1.12 Clock Configuration (CCR)

Offset

Register	Offset
CCR	40h

Function

Contains clock configuration fields. These fields are only used in Master mode and you can only change them when LPSPI is disabled (CR[MEN] = 0).

Warning

Writing a 32-bit value to this register overwrites Clock Configuration 1 (CCR1); DBT and SCKDIV always read 0.

To avoid overwriting CCR1:

- Either write all 4 CCR register fields simultaneously and only once in a 32-bit data.
- Or to modify SCKPCS and/or PCSSCK values, write only these 2 upper bytes in a 16-bit data or one of them in an 8-bit data.
- Or to modify CCR1[PCSPCS] and CCR1[SCKSCK] fields only or CCR1[SCKSET] and CCR1[SCKHLD] fields only, write respectively CCR[DBT] or CCR[SCKDIV] in 8-bit data

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKPCS								PCSSCK							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0							
W	DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SCKPCS	SCK-to-PCS Delay In Master mode: configures the delay from the last SCK edge to the PCS negation. • The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. See Figure 180.
23-16 PCSSCK	PCS-to-SCK Delay In Master mode: configures the delay from the PCS assertion to the first SCK edge. • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. See Figure 180.
15-8 DBT	Delay Between Transfers Writing this field updates the contents of CCR1[PCSPCS] and CCR1[SCKSCK].
7-0 SCKDIV	SCK Divider Writing this field updates the contents of CCR1[SCKSET] and CCR1[SCKHLD].

50.5.1.13 Clock Configuration 1 (CCR1)

Offset

Register	Offset
CCR1	44h

Function

Contains clock configuration fields. These fields are only used in Master mode and you can only change them when LPSPI is disabled (CR[MEN] = 0).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKSCK								PCSPCS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SCKHLD								SCKSET							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SCKSCK	SCK Inter-Frame Delay In Master mode: • Configures the delay from the last SCK pulse of a frame and the first SCK pulse of the following frame, in a continuous transfer. • The delay is equal to (SCKSCK + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. **NOTE** For backward compatibility, writing CCR[DBT] updates CCR1[SCKSCK] with the value written.
23-16 PCSPCS	PCS to PCS delay In Master mode: • Configures the delay from the PCS negation to the next PCS assertion. • The delay is equal to (PCSPCS + PCSPCS + 2) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 2 cycles. • Half of the delay (PCSPCS + 1) occurs before PCS assertion and the other half of the delay (PCSPCS + 1) occurs after PCS negation. If the command word is updated between 2 transfers, then the command word is updated halfway between the PCS negation of the last transfer and PCS assertion of the next transfer. • The command word specifies which PCS signal is used, the polarity and phase of the SCK signal, and the prescaler selected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE For backward compatibility, writing CCR[DBT] updates this field with (DBT÷2) rounded up.
15-8 SCKHLD	SCK Hold In Master mode, this field configures the hold phase of the SCK pin. • The hold phase is the delay between the SCK edge that samples the receive data and the SCK edge that drives the transmit data. • This is the SCK low period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. This is the SCK high period when CPHA = 0, CPOL = 0 and CPHA = 1, CPOL = 1. • The SCK hold phase delay is equal to (SCKHLD + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD. Configure both fields to the same value for 50/50 duty cycle. See Figure 180. **NOTE** For backward compatibility, writing CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded down.
7-0 SCKSET	SCK Setup In Master mode, the SCK Setup configures the setup phase of the SCK pin. • The setup phase is the delay between the SCK edge that drives the transmit data and the SCK edge that samples the receive data. • This is the SCK high period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. This is the SCK low period when CPHA = 0 and CPOL = 0, or CPHA = 1 and CPOL = 1. • The SCK setup phase delay is equal to (SCKSET + 1) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The minimum delay is 1 cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the prescaler selected (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD, configure both fields to the same value for 50/50 duty cycle. See Figure 180. **NOTE** For backward compatibility, writing CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded up.

50.5.1.14 FIFO Control (FCR)

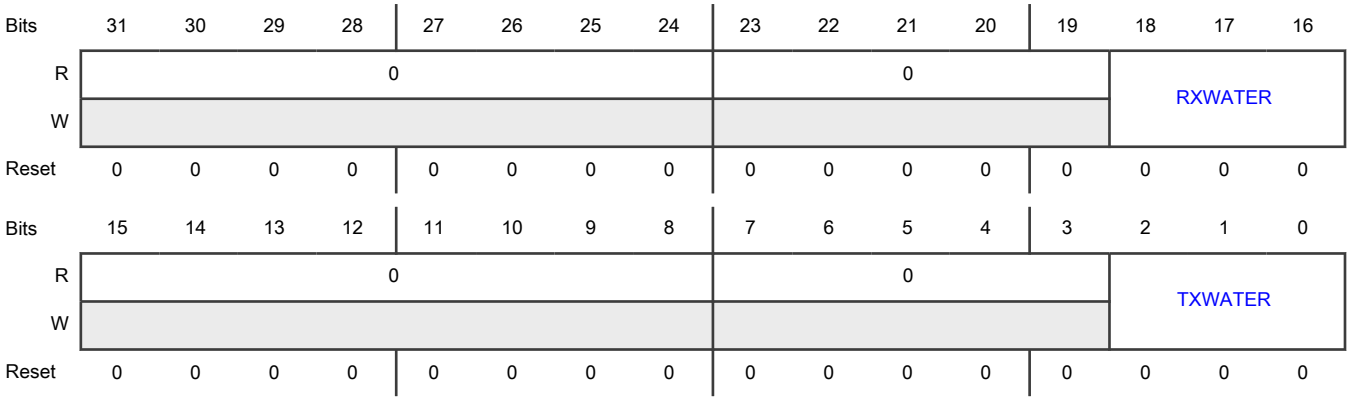
Offset

Register	Offset
FCR	58h

Function

Contains the receive FIFO and transmit FIFO watermark values.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-19 —	Reserved
18-16 RXWATER	Receive FIFO Watermark Causes LPSPI to set the Receive Data Flag (SR[RDF]) when the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size truncates the written value.
15-8 —	Reserved
7-3 —	Reserved
2-0 TXWATER	Transmit FIFO Watermark Causes LPSPI to set the Transmit Data Flag (SR[TDF]) when the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size truncates the written value.

50.5.1.15 FIFO Status (FSR)

Offset

Register	Offset
FSR	5Ch

Function

Contains fields that indicate the number of words currently stored in the receive FIFO and transmit FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				RXCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0				TXCOUNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-20 —	Reserved
19-16 RXCOUNT	Receive FIFO Count Indicates the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-4 —	Reserved
3-0 TXCOUNT	Transmit FIFO Count Indicates the number of words currently stored in the transmit FIFO.

50.5.1.16 Transmit Command (TCR)

Offset

Register	Offset
TCR	60h

Function

Writes to either this register or Transmit Data (TDR) push the data into the transmit FIFO, in the order written. Only write to this register using 32-bit writes. Writes are tagged and cause the command register to update, after that entry reaches the top of the FIFO and the LPSPI is enabled. This allows changes to the command word and the transmit data itself to be interleaved. That is, the writes to the two registers can be interleaved (write command word, then data word, then command word, and so on). Changing the command word causes all subsequent SPI bus transfers to be performed using the new command word.

- **In Master mode,** writing a new command word does not initiate a new transfer, unless TXMSK is 1. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK = 1). Hardware writes 0 to TXMSK when the PCS deasserts.
- **In Master mode,** if the command word is changed before an existing frame has completed, then the existing frame terminates and the command word then updates. The command word can be changed during a continuous transfer, if CONTC of the new command word is 1 and the command word is written on a frame size boundary.
- **In Slave mode,** the command word should be changed only when LPSPI is idle and there is no SPI bus transfer.

Avoid resetting the Transmit FIFO after writing to the Transmit Command Register, wait for the command register to update from the FIFO first.

Avoid register reading problems: Reading the Transmit Command Register returns the current state of the command register. Reading the Transmit Command Register at the same time that the Transmit Command Register is loaded from the transmit FIFO, can return an incorrect Transmit Command Register value. It is recommended to:

- Read the Transmit Command Register when the transmit FIFO is empty,
- Read the Transmit Command Register more than once and then compare the returned values.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CPOL	CPHA	PRESCALE			Reserved	PCS		LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				FRAMESZ											
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Fields

Field	Function
31 CPOL	Clock Polarity Specifies the value of SCK when it is idle. This field can only be updated when PCS deasserted. See Figure 180. 0b - Inactive low 1b - Inactive high
30 CPHA	Clock Phase This field can only be updated when PCS deasserted. See Figure 180. 0b - Captured. Data is captured on the leading edge of SCK and changed on the following edge of SCK 1b - Changed. Data is changed on the leading edge of SCK and captured on the following edge of SCK
29-27 PRESCALE	Prescaler Value For all SPI bus transfers, this value is applied to Clock Configuration (CCR) clock. This field can only be updated when PCS deasserted. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128
26 —	Reserved
25-24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. The Peripheral Chip Select field is only updated when PCS deasserted. **NOTE** The entire PCS field is not fully supported in every LPSPI module instance. See the chip-specific LPSPI information. 00b - Transfer using PCS[0] 01b - Transfer using PCS[1]

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Transfer using PCS[2] 11b - Transfer using PCS[3]
23 LSBF	LSB First 0b - Data is transferred MSB first 1b - Data is transferred LSB first
22 BYSW	Byte Swap Swaps the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers). 0b - Disabled 1b - Enabled
21 CONT	Continuous Transfer • In Master mode, CONT keeps the PCS asserted at the end of the frame size, until a command word is received that starts a new frame. • In Slave mode, when CONT is enabled, LPSPI only transmits the first FRAMESZ bits; after which LPSPI transmits received data (assuming a 32-bit shift register) until the next PCS negation. 0b - Continuous transfer is disabled 1b - Continuous transfer is enabled
20 CONTC	Continuing Command In Master mode, this field enables the command word to be changed within a continuous transfer. • The initial command word must enable continuous transfer (CONT = 1). • The continuing command must have CONTC = 1. • The continuing command word must be loaded on a frame size boundary. For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary. In Slave mode, the Continuing Command bit modifies the internal RXMSK/TXMSK configuration after the first FRAMESZ bits and until the PCS negation. • Receive data is discarded after the first FRAMESZ bits. If CONT is also 1 this does not block the transmission of received data. • Transmit data is not masked after the first FRAMESZ bits, this allows the first FRAMESZ bits to be received and a response transmitted. 0b - Command word for start of new transfer 1b - Command word for continuing transfer
19 RXMSK	Receive Data Mask Masks receive data (receive data is not stored in receive FIFO).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Normal transfer 1b - Receive data is masked
18 TXMSK	Transmit Data Mask Masks transmit data (no data is loaded from transmit FIFO and the output pin is 3-stated). In Master mode, TXMSK initiates a new transfer which cannot be aborted by another command word. TXMSK automatically transitions to 0 at the end of the transfer. 0b - Normal transfer 1b - Mask transmit data
17-16 WIDTH	Transfer Width Configures serial (1-bit) or parallel transfers. For half-duplex parallel transfers, either Receive Data Mask (RXMSK) or Transmit Data Mask (TXMSK) must be 1. 00b - 1-bit transfer 01b - 2-bit transfer 10b - 4-bit transfer 11b - Reserved
15-12 —	Reserved
11-0 FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remainder bits. For example, a 72-bit transfer consists of 3 words: the first and second words are 32 bits, and the third word is 8 bits.

50.5.1.17 Transmit Data (TDR)

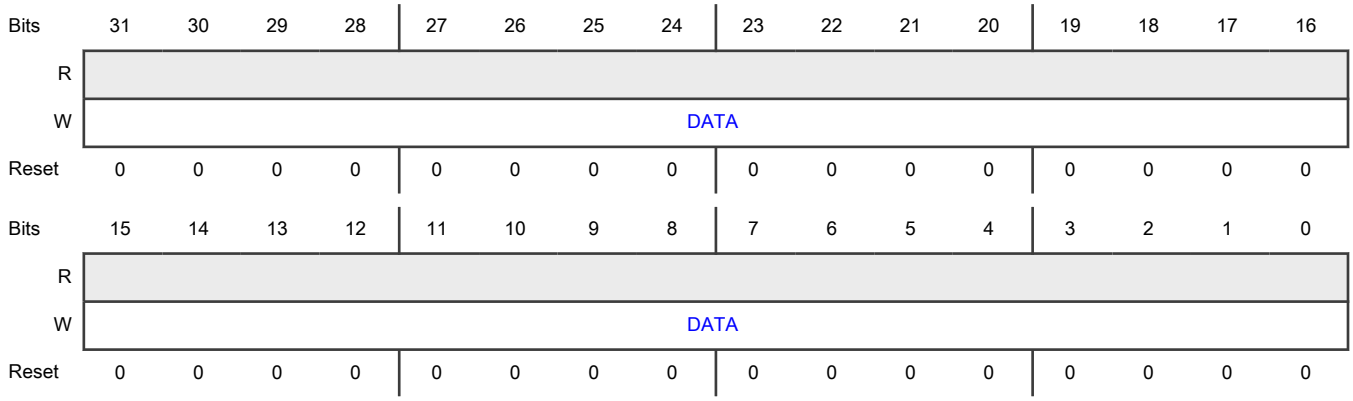
Offset

Register	Offset
TDR	64h

Function

Writes to either Transmit Command (TCR) or this register push the data into the transmit FIFO, in the order that the data is written. You can write to TDR using 32-, 16-, or 8-bit writes, but each write pushes data into the FIFO with zero pushed in unwritten bytes.

Diagram



Fields

Field	Function
31-0 DATA	Transmit Data Both 8- and 16-bit writes of transmit data zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8- and 16-bit writes (to 32 bits), means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

50.5.1.18 Receive Status (RSR)

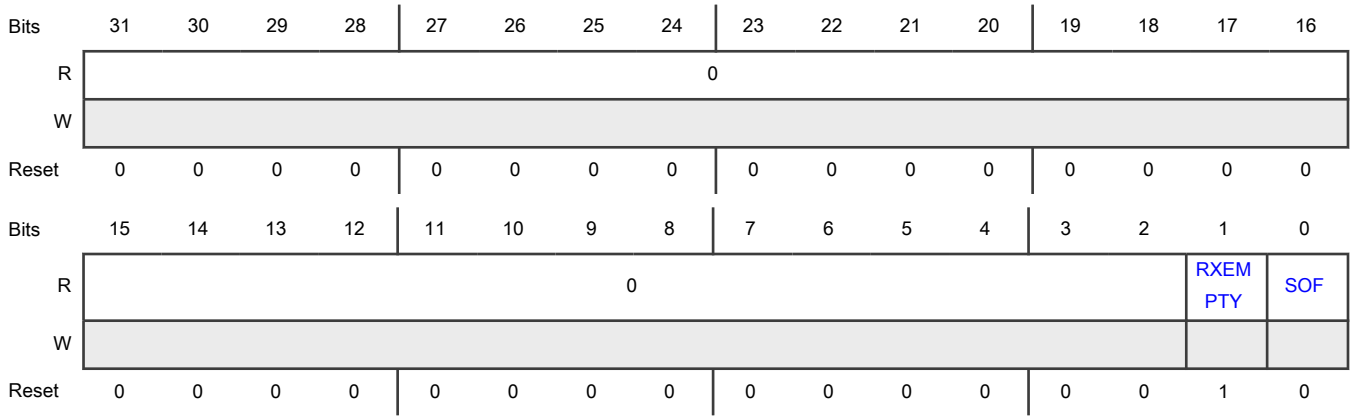
Offset

Register	Offset
RSR	70h

Function

Contains data flow status fields for receive FIFO.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty 0b - Not empty 1b - Empty
0 SOF	Start Of Frame Indicates that this is the first data word received after PCS assertion. 0b - Subsequent data word 1b - First data word

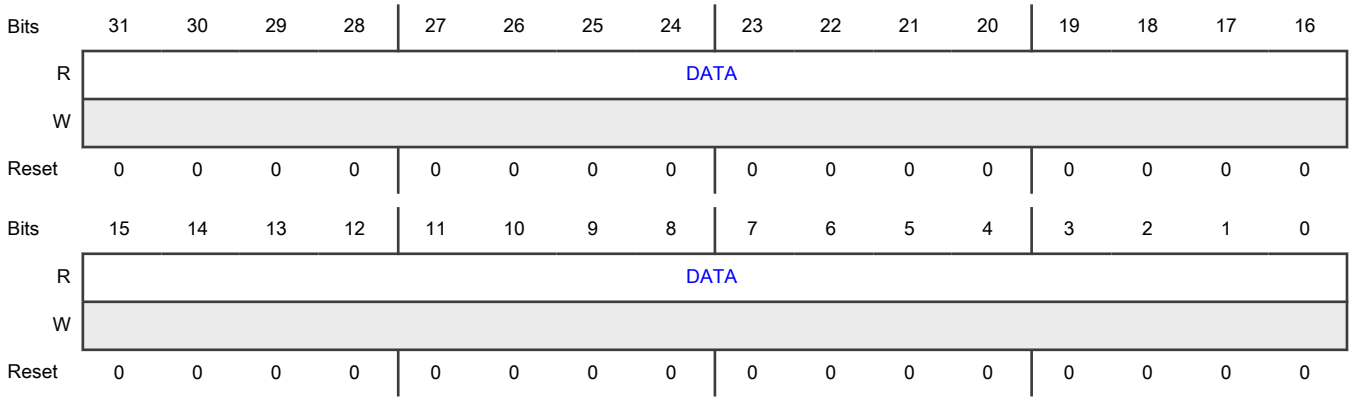
50.5.1.19 Receive Data (RDR)

Offset

Register	Offset
RDR	74h

Function
Reading this register pulls the first entry from the receive FIFO.

Diagram



Fields

Field	Function
31-0 DATA	Receive Data

50.5.1.20 Receive Data Read Only (RDROR)

Offset

Register	Offset
RDROR	78h

Function
Returns the first entry in the receive FIFO, but does not remove the data from the FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Receive Data

50.5.1.21 Transmit Command Burst (TCBR)

Offset

Register	Offset
TCBR	3FCh

Function
Supports burst transfers of command data to the transmit FIFO for use with the DMA controller.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Command Data Data is written to the Transmit Command Register.

50.5.1.22 Transmit Data Burst (TDBR0 - TDBR127)

Offset

For n = 0 to 127:

Register	Offset
TDBRn	400h + (n × 4h)

Function

Supports burst transfers of data to the transmit FIFO for use with the DMA controller. The size of this register is 512 bytes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Data Data is written to Transmit Data (TDR).

50.5.1.23 Receive Data Burst (RDBR0 - RDBR127)

Offset

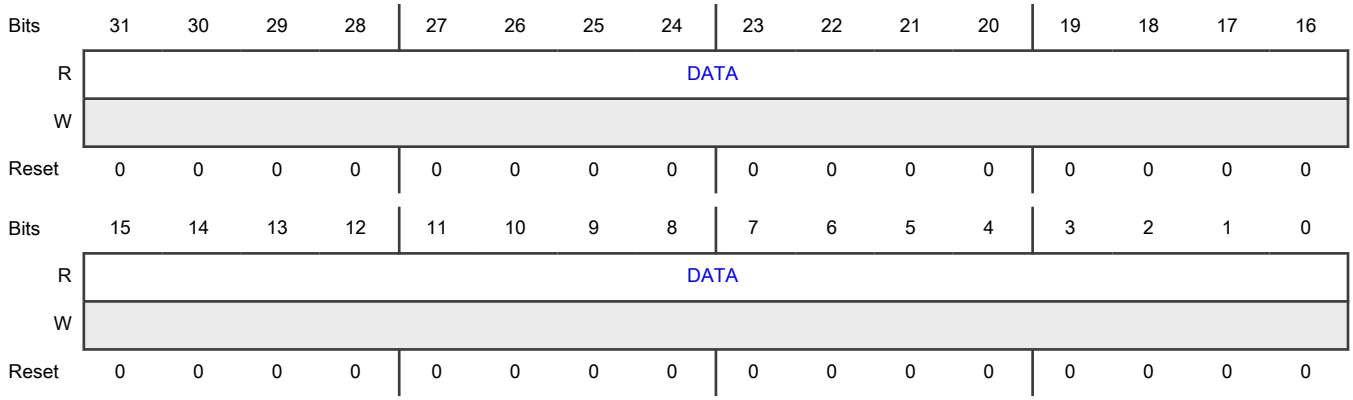
For n = 0 to 127:

Register	Offset
RDBRn	600h + (n × 4h)

Function

Supports burst transfers of data from the receive FIFO. The size of this register is 512 bytes.

Diagram



Fields

Field	Function
31-0 DATA	Data Data is read from the Receive Data Register.

Chapter 51
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

51.1 Chip-specific LPUART information

Table 312. Reference links to related information

Topic	Related module	Reference
Full description	LPUART	LPUART
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

51.1.1 Module instances

This device has two instances of the LPUART module, LPUART0, and LPUART1.

51.2 Overview

LPUART provides asynchronous, serial communication capabilities with external devices. It supports the non-return-to-zero (NRZ) encoding format and IrDA-compatible, low-speed serial infrared (SIR) protocol. LPUART can continue operating when the processor is in Low-Power mode, if an appropriate peripheral clock is available.

51.2.1 Block diagram

The following figure shows the transmitter portion of LPUART.

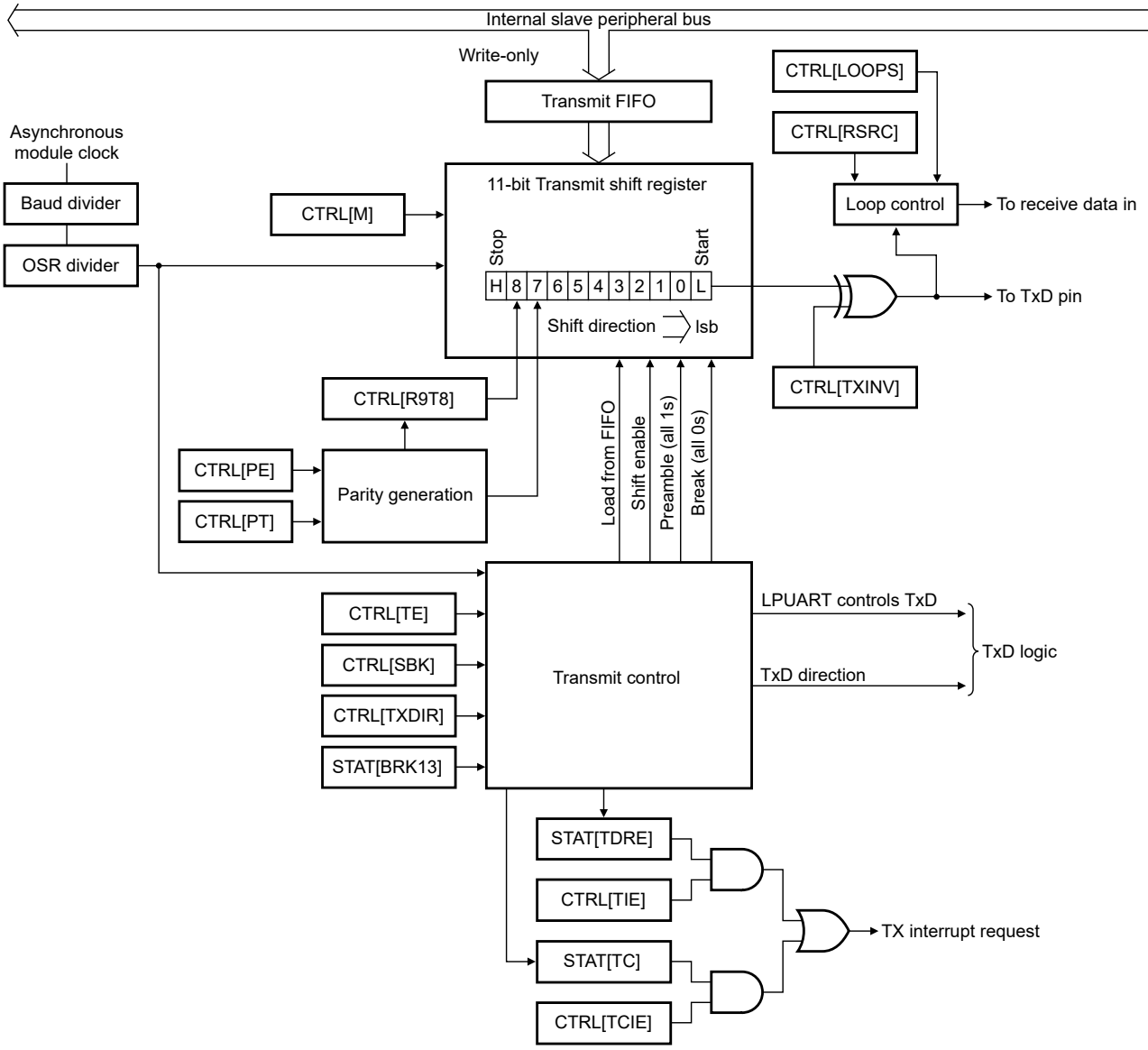


Figure 182. LPUART transmitter block diagram

The following figure shows the receiver portion of LPUART.

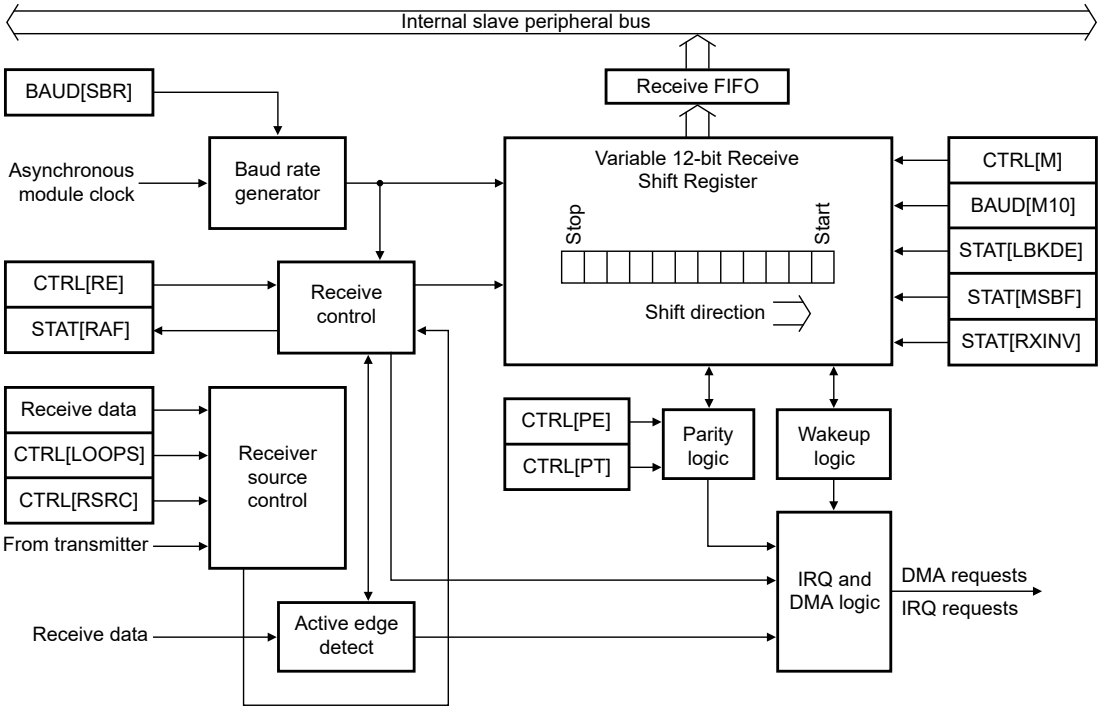


Figure 183. LPUART receiver block diagram

51.2.2 Features

- Full-duplex, standard NRZ format
- Programmable baud rates (13-bit modulo divider) with a configurable oversampling ratio (OSR) of 4× to 32×
- Asynchronous operations of transmit and receive baud rates with respect to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency.
  - Operation in Low-Power modes is supported.
- Interrupt, DMA, or polled operations:
  - Transmit data empty and transmission complete
  - Receive data full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 7-bit, 8-bit, 9-bit, or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Support for three receiver wakeup methods:
  - Idle line wakeup

  - Address mark wakeup
  - Receive data match
- Automatic address matching to reduce ISR overhead:
  - Address mark matching
  - Idle line address matching
  - Address match start, address match end
- Optional 13-bit and 11-bit break character generation
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64, or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with a programmable pulse width
- Independent FIFO structure for transmit and receive functions:
  - Separate configurable watermarks for receive and transmit requests
  - Option for receiver to assert request after a configurable number of idle characters, if receive FIFO is not empty

51.3 Register definition

LPUART includes registers to control baud rate, select options, report status, and store transmit and receive data. Access to an address outside the valid memory map generates a bus error.

NOTE

Writing to a read-only (RO) register or reading a write-only (WO) register can cause bus errors. This module does not check if programmed values in the registers are correct; you must write valid values to them.

51.3.1 LPUART register descriptions

51.3.1.1 LPUART memory map

LPUART0 base address: 4003_8000h

LPUART1 base address: 4003_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0403_0003h
4h	Parameter (PARAM)	32	R	0000_0303h
8h	Global (GLOBAL)	32	RW	0000_0000h
Ch	Pin Configuration (PINCFG)	32	RW	0000_0000h
10h	Baud Rate (BAUD)	32	RW	0F00_0004h
14h	Status (STAT)	32	RW	00C0_0000h
18h	Control (CTRL)	32	RW	0000_0000h
1Ch	Data (DATA)	32	RW	0000_1000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20h	Match Address (MATCH)	32	RW	0000_0000h
24h	MODEM IrDA (MODIR)	32	RW	0000_0000h
28h	FIFO (FIFO)	32	RW	00C0_0022h
2Ch	Watermark (WATER)	32	RW	0000_0000h
30h	Data Read-Only (DATARO)	32	R	0000_1000h

51.3.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Indicates the version integrated for this instance on the chip and also specifies the inclusion and exclusion of several optional features.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification.
23-16	Minor Version Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
MINOR	Returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number Returns the feature set number. 0000_0000_0000_0001b - Standard feature set 0000_0000_0000_0011b - Standard feature set with MODEM and IrDA support

51.3.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Indicates the parameter configuration for this instance on the chip.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Fields

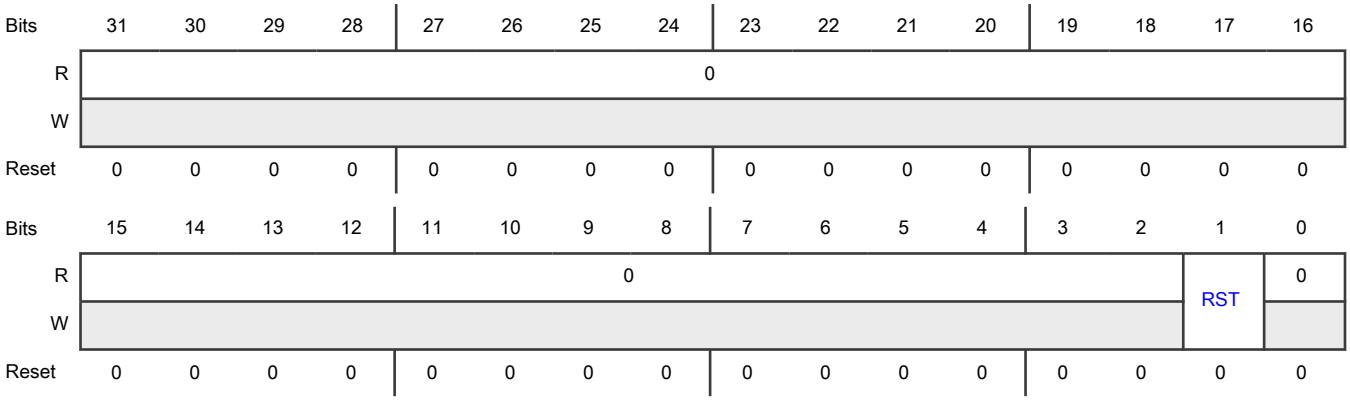
Field	Function
31-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size Indicates the number of characters in the receive FIFO, which is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size Indicates the number of characters in the transmit FIFO, which is 2^TXFIFO.

51.3.1.4 Global (GLOBAL)

Offset

Register	Offset
GLOBAL	8h

Diagram



Fields

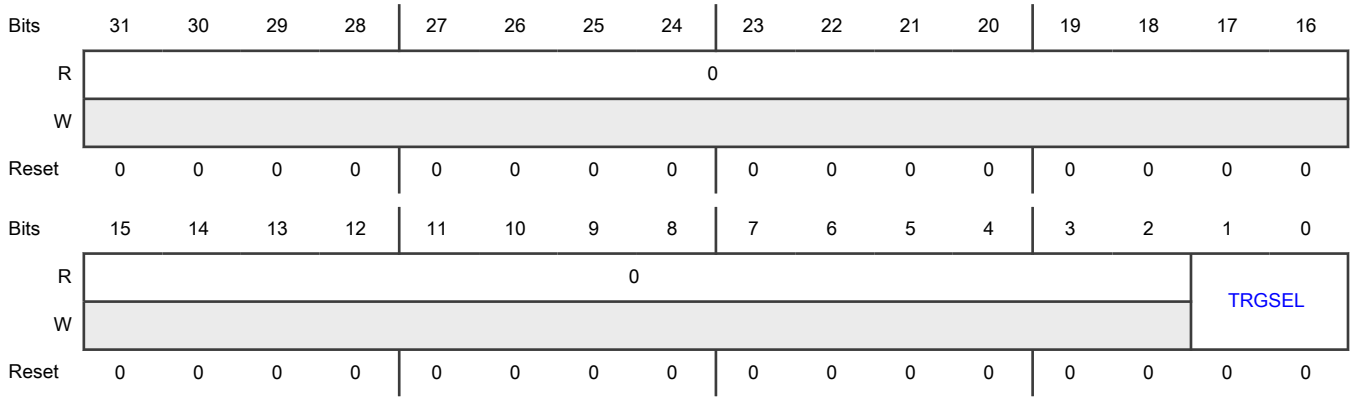
Field	Function
31-2 —	Reserved
1 RST	Software Reset Specifies whether the module is reset. This field resets all internal logic and registers, except Global (GLOBAL). The reset takes effect immediately and remains asserted until you negate it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 —	Reserved

51.3.1.5 Pin Configuration (PINCFG)

Offset

Register	Offset
PINCFG	Ch

Diagram



Fields

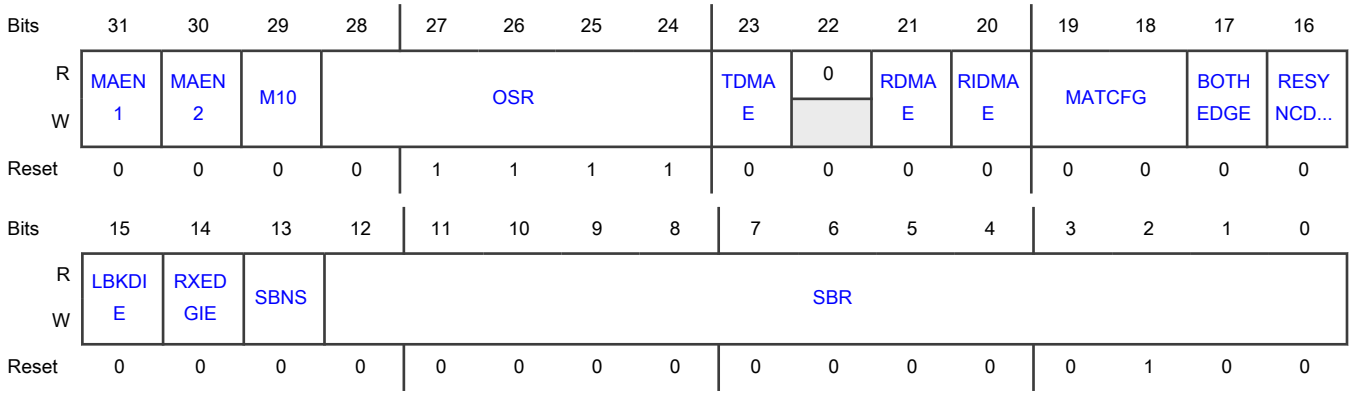
Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Input trigger disabled 01b - Input trigger used instead of the RXD pin input 10b - Input trigger used instead of the CTS_B pin input 11b - Input trigger used to modulate the TXD pin output, which (after TXINV configuration) is internally ANDed with the input trigger

51.3.1.6 Baud Rate (BAUD)

Offset

Register	Offset
BAUD	10h

Diagram



Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 Enables automatic address matching or data matching mode for MATCH[MA1]. If this field = 0, normal operation takes place. 0b - Disables 1b - Enables
30 MAEN2	Match Address Mode Enable 2 Enables automatic address matching or data matching mode for MATCH[MA2]. If this field = 0, normal operation takes place. 0b - Disables 1b - Enables
29 M10	10-Bit Mode Select Causes the tenth bit to be a part of the serial transmission. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters 1b - Receiver and transmitter use 10-bit data characters
28-24 OSR	Oversampling Ratio (OSR) Configures the OSR of the receiver. You must change the value of this field only when both the transmitter and receiver are disabled. **NOTE** BAUD[OSR] in this bit field results an OSR of BAUD[OSR]+1, for example, BAUD[OSR] = 0_0101b results in final divide by 6. 0_0000b - Results in an OSR of 16 0_0001b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0010b - Reserved 0_0011b - Results in an OSR of 4 (requires BAUD[BOTHEDGE] to be 1) 0_0100b - Results in an OSR of 5 (requires BAUD[BOTHEDGE] to be 1) 0_0101b - Results in an OSR of 6 (requires BAUD[BOTHEDGE] to be 1) 0_0110b - Results in an OSR of 7 (requires BAUD[BOTHEDGE] to be 1) 0_0111b - Results in an OSR of 8 0_1000b - Results in an OSR of 9 0_1001b - Results in an OSR of 10 0_1010b - Results in an OSR of 11 0_1011b - Results in an OSR of 12 0_1100b - Results in an OSR of 13 0_1101b - Results in an OSR of 14 0_1110b - Results in an OSR of 15 0_1111b - Results in an OSR of 16 1_0000b - Results in an OSR of 17 1_0001b - Results in an OSR of 18 1_0010b - Results in an OSR of 19 1_0011b - Results in an OSR of 20 1_0100b - Results in an OSR of 21 1_0101b - Results in an OSR of 22 1_0110b - Results in an OSR of 23 1_0111b - Results in an OSR of 24 1_1000b - Results in an OSR of 25 1_1001b - Results in an OSR of 26 1_1010b - Results in an OSR of 27 1_1011b - Results in an OSR of 28 1_1100b - Results in an OSR of 29 1_1101b - Results in an OSR of 30 1_1110b - Results in an OSR of 31 1_1111b - Results in an OSR of 32
23 TDMAE	Transmitter DMA Enable Configures STAT[TDRE] to generate a DMA request. 0b - Disables DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables DMA request
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable Configures STAT[RDRF] to generate a DMA request. 0b - Disables DMA request 1b - Enables DMA request
20 RIDMAE	Receiver Idle DMA Enable Configures STAT[IDLE] to generate a DMA request. If the value of this field = 1, reading Data (DATA) when either DATA[RXEMPT] or DATA[IDLINE] is 1 generates an end-of-packet response until the completion of the DMA minor loop. During an end-of-packet response, reading Data (DATA) returns 0000_33FFh and does not pull data from the receive FIFO. STAT[IDLE] becomes 0 on completion of the minor loop, provided an end-of-packet response is generated and either the receive FIFO is empty or the receiver is active. 0b - DMA request disabled 1b - DMA request enabled
19-18 MATCFG	Match Configuration Configures the match addressing mode used. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Address match wakeup 01b - Idle match wakeup 10b - Match on and match off 11b - Enables RWU on data match and match on/off for the transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given OSR. This field must be 1 for OSRs between x4 and x7 and is optional for higher OSRs. You must change the value of this field only when the receiver is disabled. 0b - Receiver samples input data using the rising edge of the baud rate clock 1b - Receiver samples input data using the rising and falling edges of the baud rate clock
16 RESYNCDIS	Resynchronization Disable Disables resynchronization of the received data word when a data one followed by data zero transition is detected. You must change the value of this field only when the receiver is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enables resynchronization 1b - Disables resynchronization
15 LBKDIE	LIN Break Detect Interrupt Enable Enables STAT[LBKDIF] to generate interrupt requests. 0b - Disables hardware interrupts from STAT[LBKDIF] (uses polling) 1b - Requests hardware interrupt when STAT[LBKDIF] is 1
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables STAT[RXEDGIF] to generate interrupt requests. Changing the value of CTRL[LOOPS] or CTRL[RSRC] when this field (RXEDGIE) is 1 can cause STAT[RXEDGIF] to become 1. 0b - Disables hardware interrupts from STAT[RXEDGIF] 1b - Requests hardware interrupts when STAT[RXEDGIF] is 1
13 SBNS	Stop Bit Number Select Determines whether data characters include one or two stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - One stop bit 1b - Two stop bits
12-0 SBR	Baud Rate Modulo Divisor Sets the modulo divide rate for the baud rate generator. If SBR is 1 - 8191, baud rate = baud clock ÷ ((OSR + 1) × SBR). You must update the 13-bit baud rate setting [SBR12:SBR0] only when both the transmitter and receiver are disabled (both CTRL[RE] and CTRL[TE] are 0).

51.3.1.7 Status (STAT)

Offset

Register	Offset
STAT	14h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	W1C	W1C										W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0				0		0						AME	LBKFE
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag Specifies whether a LIN break character is detected. This field becomes 1 when the LIN break detect circuitry is enabled and a LIN break character is detected. 0b - Not detected 1b - Detected
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag Specifies whether an active edge on the receive pin has occurred. This field becomes 1 whenever the receiver is enabled and an active edge (falling if STAT[RXINV] = 0, rising if STAT[RXINV] = 1) on the RXD pin occurs. 0b - Not occurred 1b - Occurred
29 MSBF	MSB First Specifies the first bit that is transmitted after the start bit. If the value of this field = 0, LSB (bit 0) is the first bit transmitted after the start bit (which means, the first bit received after the start bit is identified as bit 0). If the value of this field = 1, MSB (identified as bit 9, bit 8, bit 7, or bit 6) is the first bit that is transmitted, after the start bit, depending on the settings of CTRL[M], CTRL[PE], and BAUD[M10]. Writing 1 to this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - LSB 1b - MSB
28	Receive Data Inversion

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXINV	Specifies whether receive data is inverted. Writing 1 to this field reverses the polarity of the received data input. You must change the value of this field only when the receiver is disabled. **NOTE** Writing 1 to this field inverts the RXD input for all cases: data bits, start and stop bits, break, and idle. 0b - Inverted 1b - Not inverted
27 RWUID	Receive Wake Up Idle Detect Controls, for CTRL[RWU] on idle character detection, whether the idle character that wakes up the receiver writes 1 to STAT[IDLE]. For address match wakeup, this field controls whether STAT[IDLE] = 1 when the address does not match. You must change the value of this field only when the receiver is disabled. If the value of this field = 0, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] does not become 1 upon detection of an idle character. During address match wakeup, STAT[IDLE] does not become 1 when an address does not match. If the value of this field = 1, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] becomes 1 upon detection of an idle character. During address match wakeup, STAT[IDLE] becomes 1 when an address does not match. 0b - STAT[IDLE] does not become 1 1b - STAT[IDLE] becomes 1
26 BRK13	Break Character Generation Length Selects the longer transmitted break character length. The state of this field does not affect the detection of a framing error. You must change the value of this field only when the transmitter is disabled. You can send a break character by writing 1 to CTRL[SBK], or by writing the transmit FIFO when DATA[FRETSC] = 1 and DATA[R9T9] = 0. 0b - 9 to 13 bit times 1b - 12 to 15 bit times
25 LBKDE	LIN Break Detection Enable Enables LIN break detection. When the value of this field is 1, LIN break detect is enabled and the LIN break character is detected at a length of 11 bit times (if CTRL[M] = 0), 12 bit times (if CTRL[M] = 1), or 13 bit times (if BAUD[M10] = 1). When the value of this field is 0, LIN break detect is disabled, and only a normal break character can be detected. This field selects a longer break character detection length. While the field is 1, receive data is not stored in the receive FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field enables the LIN break detect circuit and disables writing receive data to FIFO. Therefore, it essentially ignores all characters except a LIN break. 0b - Disables 1b - Enables
24 RAF	Receiver Active Flag Specifies whether the LPUART receiver is idle or active. This field becomes 1 when the receiver detects the beginning of a valid start bit, and the field becomes 0 automatically when the receiver detects an idle line. 0b - Idle, waiting for a start bit 1b - Receiver active (RXD pin input not idle)
23 TDRE	Transmit Data Register Empty Flag Specifies whether the transmit FIFO level is greater than, equal to, or less than the watermark. After the transmit FIFO is enabled, this field becomes 1 when the number of datawords in the transmit FIFO is equal to, or less than the number that WATER[TXWATER] indicates. To make the value of this field 0, write to it until the number of words in the transmit FIFO is greater than the number that WATER[TXWATER] indicates. After the transmit FIFO is disabled, this field becomes 1 to indicate that the FIFO level is less than the watermark. To make the value of this field 0, write to Data (DATA). This register is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Greater than watermark 1b - Equal to or less than watermark
22 TC	Transmission Complete Flag Specifies whether the transmitter is active. This field becomes 0 when a transmission is in progress or a preamble or break character is loaded. The field becomes 1 when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When this happens, the transmit data output signal becomes idle (logic 1). This field becomes 0 after you write to Data (DATA) to transmit new data, queuing a preamble by first writing 0 and then writing 1 to CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active (sending data, a preamble, or a break) 1b - Transmitter idle (transmission activity complete)
21 RDRF	Receive Data Register Full Flag Specifies whether the receive FIFO level is less than, equal to, or greater than the watermark. This field becomes 1 when the number of datawords in the receive buffer is greater than the number that WATER[RXWATER] indicates and the receive FIFO is enabled. To write 0 to this field, read Data (DATA) until the number of datawords in the receive FIFO is equal to, or less than the number that

Table continues on the next page...

Table continued from the previous page...

Field	Function
	WATER[RXWATER] indicates. When the receive FIFO is disabled, this field (RDRF) becomes 1 if the receive buffer (Data (DATA)) is full. To make this field 0, read Data (DATA). A character that is in the process of being received does not cause a change in this field until the entire character is received. Even if this field is 1, the character continues to be received until an overrun condition occurs after the entire character is received. 0b - Less than watermark 1b - Equal to or greater than watermark
20 IDLE	Idle Line Flag Specifies whether an idle line is detected. This field becomes 1 when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] = 0, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bit time count towards the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. After CTRL[ILT] becomes 1, the receiver does not start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count towards the full character time of logic high needed for the receiver to detect an idle line. For this field to become 0, write 1 to it. After the field becomes 0, you cannot write 1 to it again until after a new character is stored in the receive buffer or a LIN break character writes 1 to STAT[LBKDIF]. This field becomes 1 only once, even if the receive line remains idle for an extended period. 0b - No idle line detected 1b - Idle line detected
19 OR	Receiver Overrun Flag Specifies whether there is receive overrun. This field becomes 1 when you cannot prevent STAT[RDRF] from overflowing with data. The field becomes 1 immediately after the stop bit is completely received for the dataword that overflows the buffer and all the other error fields (STAT[FE], STAT[NF], and STAT[PF]) are prevented from becoming 1. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If STAT[LBKDE] is enabled and a LIN break is detected, this field becomes 1 if STAT[LBKDIF] is not 0 before the next data character is received. When this field is 1, no additional data is stored in the receive FIFO even if sufficient room exists. 0b - No overrun 1b - Receive overrun (new LPUART data lost)
18 NF	Noise Flag (NF) Specifies whether noise is detected in the received character of Data (DATA). The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame, then noise is detected for that character. This field becomes 1 whenever the next character to be read from Data (DATA) is received with noise detected within the character.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No noise detected 1b - Noise detected
17 FE	Framing Error Flag (FE) Specifies whether a framing error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) is received with logic 0 detected where a stop bit was expected. 0b - No framing error detected (this does not guarantee that the framing is correct) 1b - Framing error detected
16 PF	Parity Error Flag (PF) Specifies whether a parity error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) received when parity is enabled (CTRL[PE] = 1) and the parity bit in the received character does not agree with the expected parity value. 0b - No parity error detected 1b - Parity error detected
15 MA1F	Match 1 Flag Specifies whether the received data is equal to MATCH[MA1]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA1]. 0b - Not equal to MA1 1b - Equal to MA1
14 MA2F	Match 2 Flag Specifies whether the received data is equal to MATCH[MA2]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA2]. 0b - Not equal to MA2 1b - Equal to MA2
13-10 —	Reserved
9-8 —	Reserved
7-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 AME	Address Mark Enable Configures the location of the address mark when configured for MSB first transfers. This field has no effect when configured for LSB first and you must change the value of this field only when both the transmitter and receiver are disabled. If the value of this field = 1, the address mark is stored in Data (DATA) at MSB (or MSB-1 when the parity bit is enabled). 0b - Address mark in character is MSB 1b - Address mark in character is the last bit before the stop bit (or parity bit when enabled)
0 LBKFE	LIN Break Flag Enable Enables the LIN break flag to assert whenever a LIN break character is detected. Unlike STAT[LBKDE], this does not impact data being stored in the receive data buffer, but does cause STAT[LBKDIF] to become 1 whenever a LIN break is detected. Because a LIN break is longer than a normal character, the LIN break triggers a write to STAT[RDRF] with the data fields as 0 and STAT[FE] = 1. The character following the LIN break has DATA[LINBRK] = 1 to indicate that the previous character was a LIN break. You must change the value of this field only when both the transmitter and receiver are disabled. If the value of this field = 1, the LIN break character is detected at a length of 11-bit times (if CTRL[M] = 0), 12 (if CTRL[M] = 1), or 13 (if BAUD[M10] = 1). 0b - Disables LIN break detect 1b - Enables LIN break detect

51.3.1.8 Control (CTRL)

Offset

Register	Offset
CTRL	18h

Function

Controls various optional features of the LPUART system.

You must write to the fields of this register only when both the transmitter and receiver are disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1IE	MA2IE	0		M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 R8T9	Receive Bit 8 Transmit Bit 9 Contains R8 and T9 that correspond to different functions. R8 is the ninth data bit received after you configure LPUART for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading Data (DATA). T9 is the tenth data bit transmitted after you configure LPUART for 10-bit data formats. When writing 10-bit data, write T9 before writing to Data (DATA). If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE** R8 is a read-only bit and T9 is a write-only bit; the value read is different from the value written.
30 R9T8	Receive Bit 9 Transmit Bit 8 Contains R9 and T8 that correspond to different functions. R9 is the tenth data bit received after you configure LPUART for 10-bit data formats. When reading 10-bit data, read R9 before reading Data (DATA). T8 is the ninth data bit transmitted after you configure LPUART for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing to Data (DATA). If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE** R9 is a read-only field and T8 is a write-only field; the value read is different from the value written.
29 TXDIR	TXD Pin Direction in Single-Wire Mode Determines the direction of data at the TXD pin when LPUART is configured for a single-wire half-duplex operation, (CTRL[LOOPS] = CTRL[RSRC] = 1). When writing 0 to this field, the transmitter finishes transmitting the current character (if any) before the receiver starts receiving data from the TXD pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - TXD pin is an input in Single-Wire mode 1b - TXD pin is an output in Single-Wire mode
28 TXINV	Transmit Data Inversion Specifies whether transmit data is inverted. Writing 1 to this field reverses the polarity of the transmitted data output. This action inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 0b - Not inverted 1b - Inverted
27 ORIE	Overrun Interrupt Enable Enables STAT[OR] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when OR interrupts are disabled. 0b - Disables 1b - Enables
26 NEIE	Noise Error Interrupt Enable Enables STAT[NF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when NF interrupts are disabled. 0b - Disables 1b - Enables
25 FEIE	Framing Error Interrupt Enable Enables STAT[FE] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when FE interrupts are disabled. 0b - Disables 1b - Enables
24 PEIE	Parity Error Interrupt Enable Enables STAT[PF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when PF interrupts are disabled. 0b - Disables 1b - Enables
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests if STAT[TDRE] = 1. 0b - Disables 1b - Enables
22	Transmission Complete Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCIE	Enables STAT[TC] to generate interrupt requests if STAT[TC] = 1. 0b - Disables 1b - Enables
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate hardware interrupt requests if STAT[RDRF] = 1. 0b - Disables 1b - Enables
20 ILIE	Idle Line Interrupt Enable Enables hardware interrupts. This field enables STAT[IDLE] to generate interrupt requests. 0b - Disables hardware interrupts from STAT[IDLE]; use polling 1b - Enables hardware interrupts when STAT[IDLE] = 1
19 TE	Transmitter Enable Enables the LPUART transmitter. Using this field, you can also queue an idle preamble by first writing 0 and then writing 1 to this field. After this field becomes 0, the field reads 1 until the transmitter has completed the current character and the TXD pin is tristated. You can also queue a single idle character by writing to the transmit FIFO with DATA[FRETSC] = 1 and DATA[R9T9] = 1. 0b - Disables 1b - Enables
18 RE	Receiver Enable Enables the LPUART receiver. After you write 0 to this field, this field becomes 1 until the receiver finishes receiving the current character (if any). 0b - Disables 1b - Enables
17 RWU	Receiver Wakeup Control Specifies whether the LPUART receiver in standby is waiting for a wakeup condition. You can write 1 to this field to place the LPUART receiver in a Standby state. The field becomes 0 automatically when an RWU event occurs, that is, in case of an idle event when CTRL[WAKE] = 0 or an address match when CTRL[WAKE] = 1 and STAT[RWUID] = 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You must write 1 to this field only when CTRL[WAKE] = 0 (wakeup on idle), if the channel is currently not idle. You can determine this by the value of STAT[RAF]. If the field is 1 to wake up an idle event and the channel is already idle, LPUART, possibly, discards the data. This is because the data must be received or a LIN break is detected after an IDLE is detected before IDLE is allowed to be reasserted. 0b - Normal receiver operation 1b - LPUART receiver in standby, waiting for a wakeup condition
16 SBK	Send Break Specifies whether queue break character(s) are to be sent. Writing a 1 and then a 0 to this field queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] = 1, and bit times of logic 0 are queued as long as this field is 1. Depending on the timing when this field is 1 and 0, relative to the character currently being transmitted, a second break character may be queued before you write 0 to this field. If the time taken to write 0 to this field is too long, for example, if the field does not become 0 by the end of the first break character, a second break character is sent. This is compared to queuing a break character through the transmit FIFO that guarantees only one break character is sent. You can also queue a single break character by writing to the transmit FIFO when DATA[FRETSC] = 1 and DATA[R9T9] = 0. 0b - Normal transmitter operation 1b - Queue break character(s) to be sent
15 MA1IE	Match 1 (MA1F) Interrupt Enable Enables the MA1F interrupt. 0b - Disables 1b - Enables
14 MA2IE	Match 2 (MA2F) Interrupt Enable Enables the MA2F interrupt. 0b - Disables 1b - Enables
13-12 —	Reserved
11 M7	7-Bit Mode Select Specifies the data characters that the receiver and transmitter use. You must change the value of this field only after both the transmitter and receiver are disabled. 0b - 8-bit to 10-bit data characters

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - 7-bit data characters
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before you write 1 to STAT[IDLE]. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 128
7 LOOPS	Loop Mode Select Selects Loop mode. After this field becomes 1, the RXD pin is disconnected from LPUART and the transmitter output is internally connected to the receiver input. The transmitter and receiver must be enabled to use the loop function. In Loop mode or Single-Wire mode, the transmitter outputs are internally connected to the receiver input (see CTRL[RSRC]). 0b - Normal operation: RXD and TXD use separate pins 1b - Loop mode or Single-Wire mode
6 DOZEEN	Enables LPUART in Doze mode. If this field is 1, LPUART remains active when not in Doze mode. 0b - Enables 1b - Disables
5 RSRC	Receiver Source Select Determines the source of the receiver shift register input if CTRL[LOOPS] = 1. This field has no effect unless CTRL[LOOPS] = 1. If the value of this field is 0, internal Loopback mode is selected. LPUART does not use the RXD pin. If the value of this field is 1, single-wire LPUART mode is selected where the TXD pin is connected to the transmitter output and receiver input. 0b - Internal Loopback mode 1b - Single-wire mode
4 M	9-Bit Or 8-Bit Mode Select Specifies the mode that the receiver and transmitter use. 0b - 8-bit data characters

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - 9-bit data characters
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes up LPUART when CTRL[RWU] = 1 and BAUD[MATCFG] = 0 (this field must be 1 when BAUD[MATCFG] = 11): • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character • An idle condition on the receive pin input signal 0b - Configures CTRL[RWU] for idle-line wakeup 1b - Configures CTRL[RWU] with address-mark wakeup
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or the stop bit. If the count begins after the start bit, a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE** In case you write 1 to this field, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - After the start bit 1b - After the stop bit
1 PE	Parity Enable Enables hardware parity generation and checking. If parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - Disables 1b - Enables
0 PT	Parity Type Selects the type of parity, even or odd, if parity is enabled (CTRL[PE] = 1): • Odd parity means that the total number of logic 1 bits in the data character, including the parity bit, is odd. • Even parity means that the total number of 1s in the data character, including the parity bit, is even. 0b - Even parity 1b - Odd parity

51.3.1.9 Data (DATA)

Offset

Register	Offset
DATA	1Ch

Function

Supports 8-bit, 16-bit, or 32-bit writes, each type of write performing a separate function. An 8-bit write to DATA[7:0] pushes {CTRL[R8T9], CTRL[R9T8], DATA[7:0]} the transmit FIFO with TSC clear. A 16-bit or 32-bit write pushes the data written into the FIFO and does not update the value of CTRL[R8T9] or CTRL[R9T8].

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status fields.

NOTE

Reads return the contents of the read-only receive FIFO and writes go to the write-only transmit FIFO, making this register work as a set of two separate registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NOISY	PARITYE	FRETSC	RXEMPT	IDLINE	LINBRK	R9T9	R8T8	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 NOISY	Noisy Data Received Indicates whether the current received dataword contained in DATA[R9:R0] is received with noise. 0b - Received without noise 1b - Received with noise
14 PARITYE	Parity Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether the current received dataword contained in DATA[R9:R0] is received with a parity error. 0b - Received without a parity error 1b - Received with a parity error
13 FRETSC	Frame Error Transmit Special Character Indicates the way the dataword is received. For reads, this field indicates that the current received dataword contained in DATA[R9:R0] is received with a frame error. For writes, the field indicates that a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 indicates a break character when it is 0 and indicates an idle character when it is 1. The contents of DATA[T8:T0] must be 0. 0b - Received without a frame error on reads or transmits a normal character on writes 1b - Received with a frame error on reads or transmits an idle or break character on writes
12 RXEMPT	Receive Buffer Empty Indicates whether the receive buffer contains valid data. This field becomes 1 when there is no data in the receive buffer. The field does not consider data in the receive shift register. 0b - Contains valid data 1b - Contains invalid data and is empty
11 IDLINE	Idle Line Indicates whether the receiver line was idle before receiving the character in DATA[9:0]. Unlike STAT[IDLE], you can write 1 to this field for the first character received when the receiver is first enabled. 0b - Received was not idle 1b - Receiver was idle
10 LINBRK	LIN Break Indicates whether the receiver line detected a LIN break before receiving the character in DATA[9:0]. This field requires the value of STAT[LBKDIF] to be 1. 0b - LIN break not detected or LIN break detect circuitry disabled 1b - LIN break detected
9 R9T9	Read Receive FIFO Bit 9 Or Write Transmit FIFO Bit 9
8 R8T8	Read Receive FIFO Bit 8 Or Write Transmit FIFO Bit 8
7	Read Receive FIFO Bit 7 Or Write Transmit FIFO Bit 7

Table continues on the next page...

Table continued from the previous page...

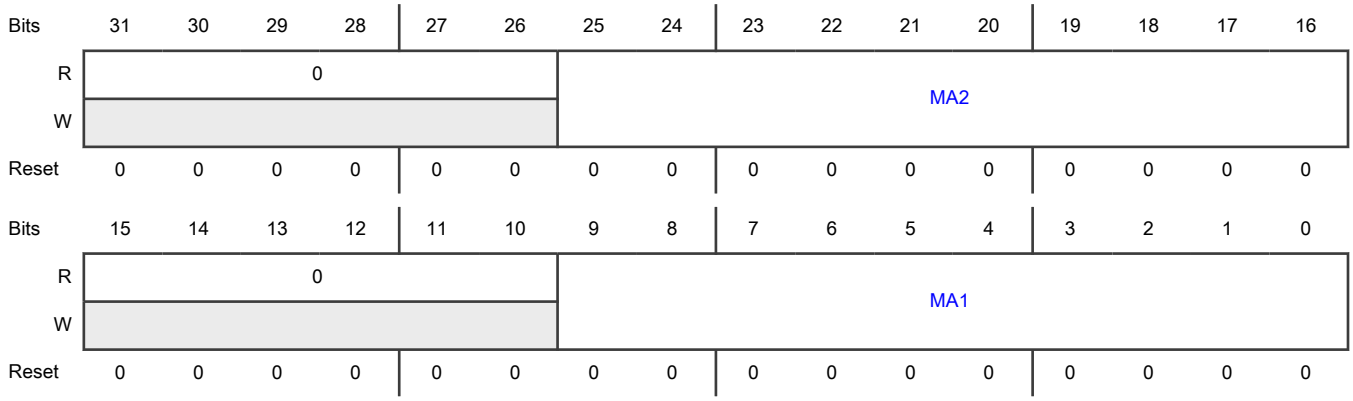
Field	Function
R7T7	
6 R6T6	Read Receive FIFO Bit 6 Or Write Transmit FIFO Bit 6
5 R5T5	Read Receive FIFO Bit 5 Or Write Transmit FIFO Bit 5
4 R4T4	Read Receive FIFO Bit 4 Or Write Transmit FIFO Bit 4
3 R3T3	Read Receive FIFO Bit 3 Or Write Transmit FIFO Bit 3
2 R2T2	Read Receive FIFO Bit 2 Or Write Transmit FIFO Bit 2
1 R1T1	Read Receive FIFO Bit 1 Or Write Transmit FIFO Bit 1
0 R0T0	Read Receive FIFO Bit 0 Or Write Transmit FIFO Bit 0

51.3.1.10 Match Address (MATCH)

Offset

Register	Offset
MATCH	20h

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] only when the associated Baud Rate (BAUD) field is 0.
15-10 —	Reserved
9-0 MA1	Match Address 1 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] fields only when the associated Baud Rate (BAUD) field is 0.

51.3.1.11 MODEM IrDA (MODIR)

Offset

Register	Offset
MODIR	24h

Function
Controls options for setting the MODEM configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													IREN	TNP	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				RTSWATER				0		TXCTSSRC	TXCTSC	RXRTSE	TXRTSPOL	TXRTSE	TXCTSE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18 IREN	IR Enable Enables IR modulation and demodulation. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Disables 1b - Enables
17-16 TNP	Transmitter Narrow Pulse Indicates whether LPUART transmits a 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulse when the IR pulse is enabled. You must change the value of this field only when both the transmitter and receiver are disabled. The IR pulse width must be configured to less than half of the OSR. Common pulse widths are 3 ÷ 16, 1 ÷ 16, 1 ÷ 32, or 1 ÷ 4 of the bit length. You can configure these by selecting the appropriate OSR and pulse width. 00b - 1 ÷ OSR 01b - 2 ÷ OSR 10b - 3 ÷ OSR 11b - 4 ÷ OSR
15-11 —	Reserved
10-8 RTSWATER	Receive RTS Configuration Configures the assertion and negation of the receiver's RTS_B output. The receiver's RTS_B output negates when the number of empty words in the receive FIFO is greater or equal to the value of this field. If this field is 0, the RTS_B pin negates when the receive FIFO is full. For the purpose of receive RTS_B generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between RTS_B negation and the external transmitter ceasing transmission. If both receive RTS_B and address or data matching is enabled, RTS_B could assert at the end of a character if there exists no match. You must change the value of this field only when the receiver is disabled.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - The CTS_B pin 1b - An internal connection to the receiver address match result

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 TXCTSC	Transmit CTS Configuration Configures whether the CTS state or input is checked or sampled at the start of each character or only when the transmitter is idle. 0b - Sampled at the start of each character 1b - Sampled when the transmitter is idle
3 RXRTSE	Receiver RTS Enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. You must change the value of this field only when the receiver is disabled. If the value of this field = 0, the receiver has no effect on RTS. If the value of this field = 1, RTS is deasserted if STAT[RDRF] = 1 or a start bit is detected that causes STAT[RDRF] to become 1. RTS is asserted if STAT[RDRF] = 0 and has not detected a start bit that causes STAT[RDRF] to become 1. **NOTE** Do not write 1 to both MODIR[RXRTSE] and MODIR[TXRTSE]. 0b - Disables 1b - Enables
2 TXRTSPOL	Transmitter RTS Polarity Controls the polarity of the transmitter RTS. This field does not affect the polarity of the receiver RTS that remains negated in the active-low state unless MODIR[TXRTSE] = 1. You must change the value of this field only when the transmitter is disabled. 0b - Transmitter RTS is active low 1b - Transmitter RTS is active high
1 TXRTSE	Transmitter RTS Enable Controls the operation of RTS before and after a transmission. You must change the value of this field only when the transmitter is disabled. When the value of this field = 1, a character is placed into an empty transmit shift register. RTS asserts 1-bit time before the start bit is transmitted and deasserts 1-bit time after all characters in the transmitter FIFO and shift register are completely sent, including the last stop bit. If the value of this field = 0, the transmitter has no effect on RTS. 0b - Disables 1b - Enables
0 TXCTSE	Transmitter CTS Enable Enables the operation of the transmitter. You can write 1 to this field irrespective of the states of MODIR[TXRTSE] and MODIR[RXRTSE]. If the value of this field = 1, the transmitter checks the state of the CTS signal each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and

Table continues on the next page...

Table continued from the previous page...

Field	Function
	transmission is delayed until CTS is asserted. Changes in CTS, when a character is being sent, do not affect its transmission. 0b - Disables 1b - Enables

51.3.1.12 FIFO (FIFO)

Offset

Register	Offset
FIFO	28h

Function

Provides you the ability to turn on and turn off the FIFO functionality.

This register also provides you the size of the FIFO that has been implemented. You can read this register at any time and must write to it only when CTRL[RE] and CTRL[TE] are 0 and the FIFO is empty.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXEMPT	RXEMPT	0				TXOF	RXUF
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	RXIDEN			TXOFE	RXUFE	TXFE	TXFIFOSIZE			RXFE	RXFIFOSIZE		
W	TXFLUSH	RXFLUSH														
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0

Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit FIFO Or Buffer Empty Specifies whether the transmit buffer is empty.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 when there is no data in the transmit FIFO or buffer. The field does not consider data in the transmit shift register. 0b - Not empty 1b - Empty
22 RXEMPT	Receive FIFO Or Buffer Empty Specifies whether the receive buffer is empty. This field becomes 1 when there is no data in the receive FIFO or buffer. The field does not consider data in the receive shift register. 0b - Not empty 1b - Empty
21-18 —	Reserved
17 TXOF	Transmitter FIFO Overflow Flag Specifies whether more data has been written to the transmit FIFO than it can hold. If the value of this field = 1, at least one transmit FIFO overflow has occurred since the last time the field was cleared. If the value of this field = 0, no transmit FIFO overflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[TXOFE]. However, an interrupt is issued to the host only if FIFO[TXOFE] = 1. 0b - No overflow 1b - Overflow
16 RXUF	Receiver FIFO Underflow Flag Specifies whether more data has been read from the receive FIFO than was present. If the value of this field = 1, at least one receive FIFO underflow has occurred since the last time the field was cleared. If the value of this field = 0, no receive FIFO underflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[RXUFE]. However, an interrupt is issued to the host only if FIFO[RXUFE] = 1. 0b - No underflow 1b - Underflow
15 TXFLUSH	Transmit FIFO Flush Causes all data that is stored in the transmit FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the transmit FIFO or buffer clears out. This does not affect data in the transmit shift register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - All data is flushed out
14 RXFLUSH	Receive FIFO Flush Causes all data that is stored in the receive FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the receive FIFO or buffer clears out. This does not affect data in the receive shift register. 0b - No effect 1b - All data is flushed out
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable Enables STAT[RDRF] to become 1 when the receiver is idle for a number of idle characters and the FIFO is not empty. 000b - Disables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle 001b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for one character 010b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for two characters 011b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for four characters 100b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for eight characters 101b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 16 characters 110b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 32 characters 111b - Enables STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 64 characters
9 TXOFE	Transmit FIFO Overflow Interrupt Enable Enables FIFO[TXOF] to generate an interrupt to the host. 0b - Disables 1b - Enables
8	Receive FIFO Underflow Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
RXUFE	Enables FIFO[RXUF] to generate an interrupt to the host. 0b - Disables 1b - Enables
7 TXFE	Transmit FIFO Enable Enables the transmit FIFO. When the value of this field = 1, the built-in FIFO structure for the transmit buffer is enabled. FIFO[TXFIFOSIZE] indicates the size of the FIFO structure. If the value of this field = 0, the transmit buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[TXFIFOSIZE]. Both CTRL[TE] and CTRL[RE] must be 0 before you change the value of this field. 0b - Disables; buffer depth is 1 1b - Enables; FIFO[TXFIFOSIZE] indicates the buffer depth
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth Indicates the maximum number of transmit datawords that can be stored in the transmit buffer. 000b - Transmit FIFO buffer depth = 1 dataword 001b - Transmit FIFO buffer depth = 4 datawords 010b - Transmit FIFO buffer depth = 8 datawords 011b - Transmit FIFO buffer depth = 16 datawords 100b - Transmit FIFO buffer depth = 32 datawords 101b - Transmit FIFO buffer depth = 64 datawords 110b - Transmit FIFO buffer depth = 128 datawords 111b - Transmit FIFO buffer depth = 256 datawords
3 RXFE	Receive FIFO Enable Enables the receive FIFO. When the value of this field = 1, the built-in FIFO structure for the receive buffer is enabled. FIFO[RXFIFOSIZE] indicates the size of the FIFO structure. If RXFE is 0, the receive buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[RXFIFOSIZE]. Both CTRL[RE] and CTRL[TE] must be 0 before you change the value of this field. 0b - Disables; buffer depth is 1 1b - Enables; FIFO[RXFIFOSIZE] indicates the buffer depth
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth Contains the maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. 000b - Receive FIFO buffer depth = 1 dataword

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Receive FIFO buffer depth = 4 datawords 010b - Receive FIFO buffer depth = 8 datawords 011b - Receive FIFO buffer depth = 16 datawords 100b - Receive FIFO buffer depth = 32 datawords 101b - Receive FIFO buffer depth = 64 datawords 110b - Receive FIFO buffer depth = 128 datawords 111b - Receive FIFO buffer depth = 256 datawords

51.3.1.13 Watermark (WATER)

Offset

Register	Offset
WATER	2Ch

Function

Provides the ability to set a programmable threshold for notification, or sets the programmable thresholds to indicate that transmit data can be written or receive data can be read.

You may read this register at any time but must write to it only when CTRL[TE] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				RXCOUNT				0				RXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TXCOUNT				0				TXWATER			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-24 RXCOUNT	Receive Counter Indicates the number of datawords in the receive FIFO or buffer. If a dataword is being received in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate the room left in the receive FIFO or buffer.
23-19 —	Reserved
18-16 RXWATER	Receive Watermark Generates a DMA request if the number of datawords in the receive FIFO or buffer is greater than the value of this field. For proper operation, the value of this field must be less than the size of the receive FIFO or buffer, as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE].
15-12 —	Reserved
11-8 TXCOUNT	Transmit Counter Indicates the number of datawords in the transmit FIFO or buffer. If a dataword is being transmitted to the transmit shift register, it is not included in the count. This value may be used in conjunction with the value of FIFO[TXFIFOSIZE] to calculate the room left in the transmit FIFO or buffer.
7-3 —	Reserved
2-0 TXWATER	Transmit Watermark Generates a DMA request when the number of datawords in the transmit FIFO or buffer is equal to or less than the value of this field. For proper operation, the value of this field must be less than the size of the transmit buffer or FIFO, as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

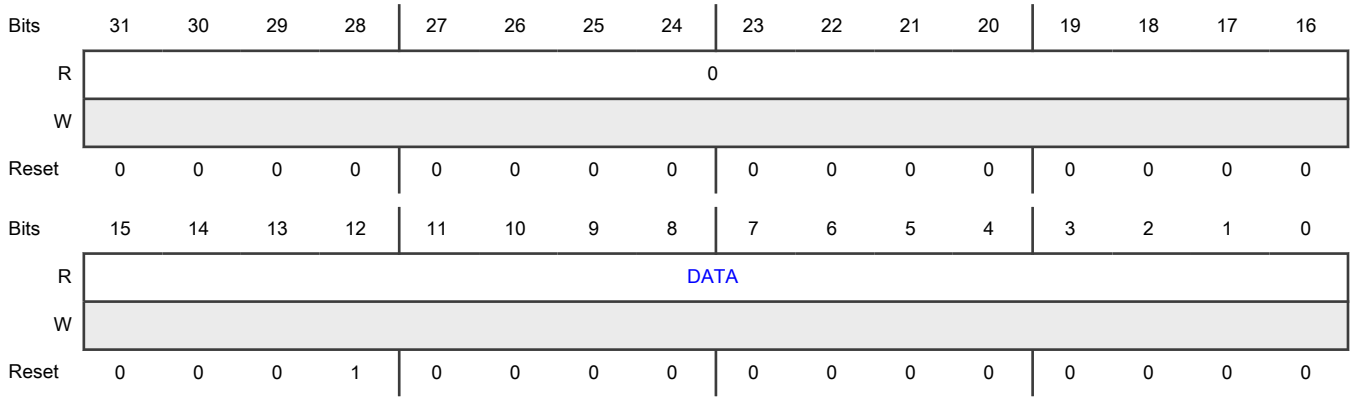
51.3.1.14 Data Read-Only (DATARO)

Offset

Register	Offset
DATARO	30h

Function
Returns the first entry in the receive FIFO, but does not pull data from the FIFO.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Receive Data Returns the first entry from the receive FIFO. This register has the same functionality as that of Data (DATA).

51.4 Functional description

LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following sections describe all LPUART blocks.

51.4.1 Low-Power modes

LPUART remains functional during Low-Power modes, provided CTRL[DOZEEN] = 0 and the LPUART functional clock is enabled. LPUART can generate an interrupt or DMA request to cause a wakeup from Low-Power modes.

You can configure LPUART to be disabled in Low-Power modes, when CTRL[DOZEEN] = 1. In this case, the transmitter and receiver finish transmitting and receiving the current word.

If LPUART is disabled in Low-Power modes, it can generate a wakeup via STAT[RXEDGIF] if the receiver detects an active edge.

NOTE

See the chip-specific information for specific Low-Power modes available on your chip.

51.4.2 Debug mode

LPUART remains functional in Debug mode.

51.4.3 Baud rate generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and transmitter. The value, ranging from 1 to 8191, written to BAUD[SBR] determines the baud clock divisor for the asynchronous LPUART baud clock. The baud rate clock drives the receiver, while a bit clock, generated from the baud rate clock divided by the OSR, drives the transmitter. Depending on the OSR, the receiver has an acquisition rate of 4 to 32 samples per bit time.

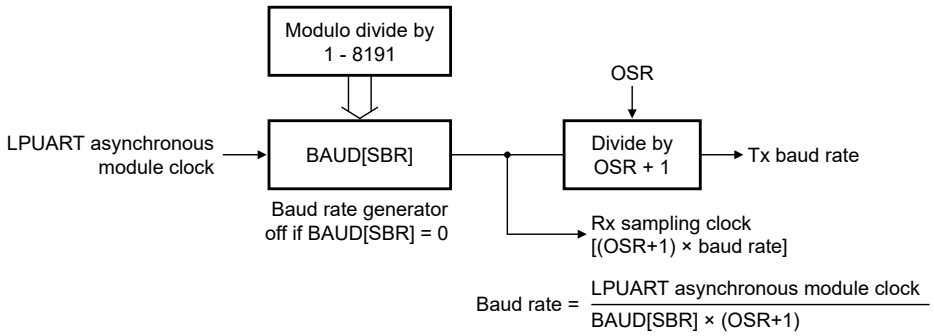


Figure 184. Baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can lead to a phase shift.

Baud rate generation is a free-running counter that continues whenever the transmitter or receiver is enabled. The transmitter bit clock continues whenever the transmitter is enabled; each transmitted character aligns to the next edge of the transmit bit clock.

In general, configuring OSR for a higher ratio and/or sampling on both edges of the clock slightly improves LPUART's tolerance to baud rate mismatch between the received data and LPUART configured baud rate. However, the three data samples in each bit are also closer together, which may impact noise sensitivity.

51.4.4 Transmitter functional description

This section discusses the functioning of the LPUART transmitter, as shown in the transmitter portion of Block diagram, as well as specialized functions for sending break and idle characters.

The transmitter output (TXD) idle state defaults to logic high; the transmitter output is inverted when you write 1 to CTRL[TXINV], which becomes 0 following reset. You can enable the transmitter by writing 1 to CTRL[TE]. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit FIFO. Programs store data in the transmit FIFO by writing to Data (DATA).

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13-bit long depending on the settings of CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS]. Going forward in this discussion, assume that CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] are 0, selecting the normal 8-bit Data mode, in which the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in STAT[TDRE] is transferred to the transmit shift register, synchronized with the baud rate clock, and STAT[TDRE] becomes 1 to indicate that another character may be written to the transmit FIFO at Data (DATA).

If no new character is waiting in the transmit FIFO after a stop bit is shifted out of the TXD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TXD high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character, or break character), although the transmitter does not start transmitting another character.

51.4.4.1 Send break and queued idle

CTRL[SBK] sends break characters, originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 9-bit to 12-bit times, including the start and stop bits. You can enable a longer break of 13-bit times by writing 1 to STAT[BRK13]. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, the program writes 1 and then writes 0 to CTRL[SBK]. This action queues a break character to be sent as soon as the shifter is available. If CTRL[SBK] remains 1 when the queued break moves into the shifter,

synchronized with the baud rate clock, an additional break character is queued. When LPUART is the receiving module, it receives a break character as 0s in all data bits and a framing error (STAT[FE] = 1) is detected.

You can also transmit a break character by writing to Data (DATA) with DATA[FRETSC] = 1 and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows DMA to transmit a break character.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, the program writes 0 and then writes 1 to CTRL[TE]. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while CTRL[TE] becomes 0, the LPUART transmitter does not release control of the TXD pin.

You can also write to Data (DATA) to transmit an idle character, with DATA[FRETSC] and DATA[R9T9] = 1 and the values of all the other fields = 0. This supports transmitting the idle character as part of the normal data stream and also allows DMA to transmit an idle character.

As shown in the following table, STAT[BRK13], CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] affect the length of the break character.

Table 313. Break character length

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
0	0	0	0	0	10
0	0	0	0	1	11
0	0	0	1	0	9
0	0	0	1	1	10
0	1	0	—	0	11
0	1	0	—	1	12
0	—	1	—	0	12
0	—	1	—	1	13
1	0	0	0	0	13
1	0	0	0	1	13
1	0	0	1	0	12
1	0	0	1	1	12
1	1	0	—	0	14
1	1	0	—	1	14
1	—	1	—	0	15
1	—	1	—	1	15

51.4.4.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS_B. If the CTS operation is enabled, the character is transmitted when CTS_B is asserted. If CTS_B is deasserted in the middle of a transmission with characters remaining in the transmitter FIFO, the character in the transmit shift register is complete. Any characters in the FIFO wait for CTS_B to assert again, and TXD remains in the mark state (idle state) until CTS_B is reasserted. The CTS_B pin must assert for longer than one bit period to guarantee that a new transmission is started when the transmitter is idle with DTS.

If the CTS operation is disabled, the transmitter ignores the state of CTS_B.

The transmitter's CTS_B signal can be enabled even if the same LPUART receiver's RTS_B signal is disabled.

51.4.4.3 Transceiver driver enable

The transmitter can use RTS_B as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS_B for details. If the RTS operation is enabled, when a character is placed into an empty transmit shift register, RTS_B asserts 1-bit time before the start bit is transmitted. RTS_B remains asserted for the whole time that the transmit shift register has any characters. RTS_B deasserts 1-bit time after all characters in the transmit FIFO and shift register are completely sent, including the last stop bit. In other words, when RTS_B is used as a transceiver enable, RTS_B asserts 1-bit time before the transmitter starts transmitting and negates 1-bit time after the transmitter goes idle.

Transmitting a break character also asserts RTS_B, with the same assertion and deassertion timing as having a character in the transmit shift register.

The transmitter's RTS_B signal asserts only when the transmitter is enabled. However, the transmitter's RTS_B signal is unaffected by its CTS_B signal. RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled mid-way through a data transfer.

51.4.4.4 Transceiver driver enable using RTS_B

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is three-stated unless the LPUART is driving. The transmitter can use the RTS_B signal to enable the driver of a transceiver. The polarity of RTS_B can be matched to the polarity of the transceiver's driver enable signal.

In the following figure, the receiver enable signal is asserted. Another option for this connection is to connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled while driving. A pullup can pull RXD to a non-floating value during this time. You can refine this option further by operating LPUART in Single-Wire mode, freeing the RXD pin for other uses.

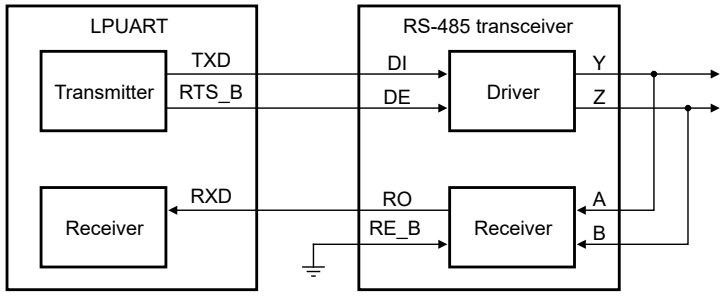


Figure 185. Transceiver driver enable using RTS_B

51.4.5 Receiver functional description

This section discusses the functioning of the LPUART receiver, as shown in the receiver portion of Block diagram. The section also discusses:

- The data sampling technique used to reconstruct receiver data
- Different variations of the receiver wakeup function

You can invert the receiver input by writing 1 to STAT[RXINV] and enable the receiver by writing 1 to CTRL[RE]. Character frames consist of a start bit of logic 0, sever to 10 data bits (MSB or LSB first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit, or 10-bit Data mode, see Data modes. Going forward in this discussion, assume that LPUART is configured for a normal 8-bit Data mode.

After receiving the stop bit into the receive shifter, and provided STAT[RDRF] = 0, the data character is transferred to STAT[RDRF] resulting into its value becoming 1. However, if STAT[RDRF] is already 1, indicating that the receive data buffer is already full, STAT[OR] becomes 1 and the new data is lost.

Because the LPUART receiver is separate from the receive FIFO, the receive shift register can receive the next word when the receive FIFO is full, and it is only at the end of the character that the next data is written into the receive FIFO, potentially triggering the overrun flag if the FIFO is full.

When a program detects that STAT[RDRF] = 1, it gets the data from STAT[RDRF] by reading Data (DATA). See Interrupts and status fields for details about flag clearing.

51.4.5.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by considering logic level samples at the oversampling rate times the baud rate to search for a falling edge on the RXD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR ÷ 2), (OSR ÷ 2) + 1, and (OSR ÷ 2) + 2 to ensure that this is a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR ÷ 2), (OSR ÷ 2) + 1, and (OSR ÷ 2) + 2, to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, STAT[NF] becomes 1 when the received character is transferred to the receive FIFO.

When the LPUART receiver is configured to sample on both edges of the baud rate clock (that is, when BAUD[BOTHEDGE] = 1), the number of segments in each received bit is effectively doubled (from 1 to OSR× 2). The start and data bits are then sampled at OSR, OSR + 1, and OSR + 2. You must enable sampling on both edges of the clock for oversampling rates of 4× to 7×. This sampling is optional for higher oversampling rates.

The synchronization feature of LPUART synchronizes the internal oversampling counter with a detected falling edge on the receive signal, and to adjust the data sampling window. The falling edge detection needs three consecutive 1s prior to the 1->0 transition. After the initial falling edge detection for the start bit, the circuit continuously monitors the next falling edge, and resets the counter after another falling edge is detected. This is called resynchronization.

When BAUD[RESYNCDIS] = 0, you perform this falling edge detection and resynchronization not only for the start bit but also for the rest of the character reception after the start bit.

When BAUD[RESYNCDIS] = 1, you perform the falling edge detection and resynchronization only for the start bit. The use case for disabling the resynchronization is protocols that require this (for example, LIN 2.1 prohibits resynchronization within a byte).

The following table and figure explain LPUART resynchronization.

Table 314. LPUART resynchronization settings

Resynchronization	BAUD[RESYNCDIS] = 0	BAUD[RESYNCDIS] = 1
For the starting bit falling edge	Yes	Yes
For all falling edges after the start bit	Yes	No

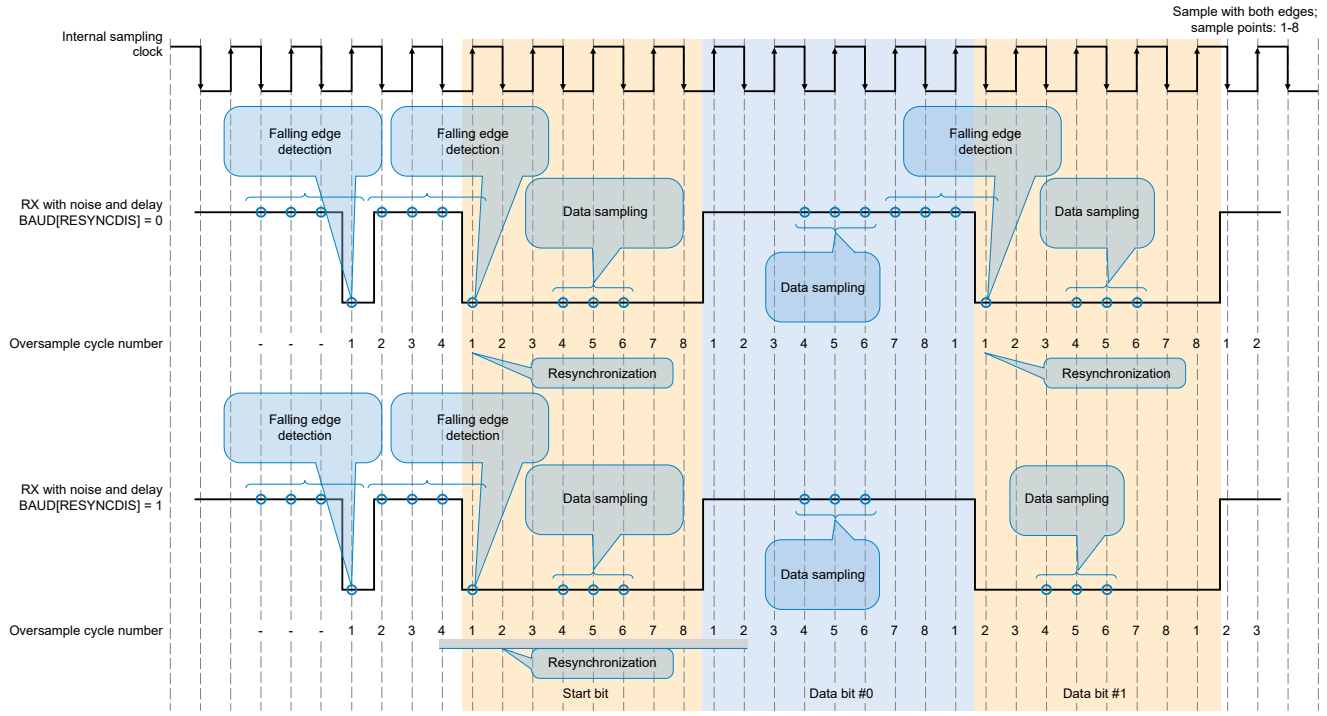


Figure 186. LPUART resynchronization diagram

51.4.5.2 Receiver wakeup operation

Receiver wakeup and receiver address matching are hardware mechanisms that allow an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write 1 to CTRL[RWU].

When CTRL[RWU] and STAT[RWUID] are 1, the status fields associated with the receiver, with the exception of STAT[IDLE], are inhibited from becoming 1, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, all receivers automatically force CTRL[RWU] to become 0. This results in all receivers waking up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver ignores all characters that do not meet the address match requirements.

Table 315. Receiver wakeup options

CTRL[RWU]	BAUD[MAEN1] | BAUD[MAEN2]	BAUD[MATCFG]	CTRL[WAKE]: STAT[RWUID]	Receiver wakeup
0	0	X	X	Normal operation
1	0	00	00	Receiver wakeup on idle line, STAT[IDLE] = 0
1	0	00	01	Receiver wakeup on idle line, STAT[IDLE] = 1

Table continues on the next page...

Table 315. Receiver wakeup options (continued)

CTRL[RWU]	BAUD[MAEN1] | BAUD[MAEN2]	BAUD[MATCFG]	CTRL[WAKE]: STAT[RWUID]	Receiver wakeup
1	0	00	10	Receiver wakeup on address mark
1	1	11	10	Receiver wakeup on data match
0	1	00	X0	Address mark address match, STAT[IDLE] = 0 for discarded characters
0	1	00	X1	Address mark address match, STAT[IDLE] = 1 for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Match on and match off, STAT[IDLE] = 0 for discarded characters
0	1	10	X1	Match on and match off, STAT[IDLE] = 1 for discarded characters

51.4.5.2.1 Idle-line wakeup

When CTRL[WAKE] = 0, you can configure the receiver for an idle-line wakeup. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a full character time of the idle-line level.

CTRL[M], CTRL[M7], and BAUD[M10] select 7-bit to 10-bit Data mode and BAUD[SBNS] selects a 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 9 to 13 bit times because of the start and stop bits.

When CTRL[RWU] = 1 and STAT[RWUID] = 0, the idle condition that wakes up the receiver does not lead to STAT[IDLE] becoming 1. The receiver wakes up and waits for the first data character of the next message that leads to STAT[RDRF] becoming 1 and generates an interrupt if enabled. When STAT[RWUID] = 1, any idle condition leads to STAT[IDLE] becoming 1 and generates an interrupt if enabled, regardless of whether CTRL[RWU] is 0 or 1.

The following are ways to detect an idle line:

- When CTRL[ILT] = 0, the idle bit counter starts after the start bit so that the stop bit and any logic 1s at the end of a character count to calculate the full character time of idle.
- When CTRL[ILT] = 1, the idle bit counter does not start until after the stop bit time so that the data in the last character of the previous message does not impact the idle detection.

51.4.5.2.2 Address-mark wakeup

When CTRL[WAKE] = 1, you can configure the receiver for an address-mark wakeup. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a logic 1 in the most significant bit of the received character. When parity is enabled, the second most significant bit is used for address-mark wakeup.

Address-mark wakeup allows messages to contain idle characters, but requires one bit to be reserved for use in address frames. The logic 1 in the most significant bit (or second most significant bit when parity is enabled) of an address frame writes 0 to

CTRL[RWU] and writes 1 to STAT[RDRF]. In this case, the character with the address-mark bit is received even if the receiver is sleeping during most of this character time.

51.4.5.2.3 Data-match wakeup

When CTRL[RWU] and CTRL[WAKE] are 1, and BAUD[MATCFG] equals 11, the receiver is configured for a data-match wakeup. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a character that matches MATCH[MA1] when BAUD[MAEN1] = 1, or that matches MATCH[MA2] when BAUD[MAEN2] = 1.

51.4.5.2.4 Address match operation

You can enable the address match operation when either BAUD[MAEN1] or BAUD[MAEN2] = 1 and BAUD[MATCFG] = 0. In this function, a character that the RXD pin receives with a logic 1 in the most significant bit (or second most significant bit when parity is enabled) is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and STAT[RDRF] becomes 1, if the comparison matches. All subsequent characters received with a logic 0 in the most significant bit (or second most significant bit when parity is enabled) are considered to be data associated with the address and are transferred to the receive FIFO. If no marked address match occurs, no transfer is made to the receive FIFO, and all the characters that follow, with logic 0 in the most significant bit (or second most significant bit when parity is enabled), are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

The address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2]:

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, a marked address is compared only to the associated Match Address (MATCH) field and data is transferred to the receive FIFO only on a match.
- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, a marked address is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

51.4.5.2.5 Idle match operation

You can enable the idle match operation when either BAUD[MAEN1] or BAUD[MAEN2] = 1 and BAUD[MATCFG] = 1. In this function, the first character that the RXD pin receives after an idle line condition is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2] field. The character is transferred only to the receive buffer, and STAT[RDRF] becomes 1, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive FIFO until the next idle line condition is detected. If no address match occurs, no transfer is made to the receive FIFO, and all the frames that follow, until the next idle condition, are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

An idle match operation functions in the same way for both MATCH[MA1] and MATCH[MA2]:

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, the first character after an idle line is compared only to the associated Data (DATA) field and data is transferred to the receive FIFO only on a match.
- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, the first character after an idle line is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

51.4.5.2.6 Match on, match off operation

The match on, match off operation is enabled when both BAUD[MAEN1] and BAUD[MAEN2] are 1 and BAUD[MATCFG] = 10. In this function, a character that the RXD pin receives matches MATCH[MA1] and is transferred to the receive buffer, and STAT[RDRF] becomes 1. All subsequent characters are considered to be data and are also transferred to the receive FIFO, until a character that matches MATCH[MA2] is received. The character that matches MATCH[MA2], along with all subsequent characters, is discarded; and this continues until another character that matches MATCH[MA1] is received. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

NOTE

The match on, match off operation requires both BAUD[MAEN1] and BAUD[MAEN2] to be 1.

51.4.5.3 Hardware flow control

To support hardware flow control, you can program the receiver to automatically assert and deassert RTS_B:

- RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS_B for more information.
- If the receiver RTS functionality is enabled, the receiver automatically deasserts RTS_B if STAT[RDRF] = 1 or a start bit is detected that causes STAT[RDRF] to become 1.
- The receiver asserts RTS_B when STAT[RDRF] = 0 and has not detected a start bit that causes STAT[RDRF] to become 1. There is no impact if STAT[RDRF] = 1 already.
- Even if RTS_B is deasserted, the receiver continues to receive characters until the receive FIFO is overrun.
- If the receiver RTS functionality is disabled, the receiver's RTS_B remains deasserted.

51.4.6 Additional LPUART functions

51.4.6.1 Data modes

The LPUART transmitter and receiver can be configured to operate in 7-bit Data mode by writing 1 to CTRL[M7], 9-bit Data mode by writing 1 to CTRL[M], or 10-bit Data mode by writing 1 to BAUD[M10]. In 9-bit Data mode, there exists a ninth data bit and in 10-bit mode, there exists a tenth data bit.

When performing 8-bit writes to the transmit FIFO, the ninth and tenth bits are pushed into the FIFO from CTRL[T8] and CTRL[T9]. For coherent 8-bit writes, you must write to CTRL[T8] and CTRL[T9] before writing to Data (DATA)[7:0]. However, if the values in CTRL[T8] or CTRL[T9] do not need to change, it is not necessary to update CTRL[T8] and CTRL[T9] before every 8-bit write to Data (DATA).

When performing 16-bit or 32-bit writes to the transmit FIFO, all 10 bits are pushed into the transmit FIFO from the write data.

When performing 8-bit reads of the receive FIFO, the ninth and tenth bits are held in CTRL[R8] and CTRL[R9] but should be read before reading Data (DATA). A 16-bit or 32-bit read of the receive FIFO returns all 10 bits in Data (DATA).

The 9-bit Data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with the address-mark wakeup so that the ninth data bit can serve as the wakeup bit. The 10-bit Data mode is typically used with parity and address-mark wakeup so that the ninth data bit can serve as the wakeup bit and the tenth bit can serve as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

51.4.6.2 Idle length

An idle character is one where the start bit, all data bits, and stop bits are in the mark position (idle state, generally logic 1). You can configure CTRL[ILT] to start detecting an idle character from the previous start bit (any data bits and stop bits count for idle character detection) or from the previous stop bit.

You can also use CTRL[IDLECFG] to configure the number of idle characters that must be received before an idle line condition is detected. This field configures the number of idle characters that must be received before STAT[IDLE] becomes 1, STAT[RAF] becomes 0, and DATA[IDLINE] becomes 1 with the next received character.

CTRL[IDLECFG] also affects the idle-line wakeup and idle match operations. When either the address match or match on/off operation is enabled, writing 1 to STAT[RWUID] causes any discarded characters to be treated as idle characters.

51.4.6.3 Loop mode

When CTRL[LOOPS] = 1, CTRL[RSRC] selects between Loop mode (CTRL[RSRC] = 0) or Single-Wire mode (CTRL[RSRC] = 1). You, sometimes, use Loop mode to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and LPUART does not use the RXD pin.

51.4.6.4 Single-Wire mode

When CTRL[LOOPS] = 1, CTRL[RSRC] selects either Loop mode (CTRL[RSRC] = 0) or Single-Wire mode (CTRL[RSRC] = 1). Single-Wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TXD pin (the RXD pin is not used).

In Single-Wire mode, CTRL[TXDIR] controls the direction of serial data on the TXD pin. When CTRL[TXDIR] becomes 0, the TXD pin is an input to the receiver and the transmitter is temporarily disconnected from the TXD pin so that an external device can send serial data to the receiver. When CTRL[TXDIR] = 1, the TXD pin is an output that the transmitter drives. The internal loop back connection is disabled, and as a result, the receiver is unable to receive characters that the transmitter sends out.

51.4.7 Infrared (IR) interface

LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses, transforming them to serial bits, which are then sent to LPUART. The IrDA physical layer specification defines a half-duplex IR communication link for exchanging data. The full standard includes data rates up to 16 Mbit/s. The LPUART IrDA support is limited to SIR mode that supports data rates only between 2.4 kbit/s and 115.2 kbit/s.

LPUART has an infrared transmit encoder and a receive decoder. The infrared decoder converts the received character from the IrDA format to the NRZ format, which the receiver uses. It also has an OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received. LPUART transmits serial bits of data, which the infrared submodule encodes, to transmit a narrow pulse for every zero bit. No pulse is transmitted for every single bit. When receiving data, an IR photo diode (external to LPUART) detects the IR pulses. The IR receive decoder transforms them to CMOS levels. The infrared receive decoder then stretches the narrow pulses to get back to a serial bit stream that LPUART receives. You can invert the polarity of transmitted pulses and expected receive pulses so that a direct connection can be made to external IrDA transceiver modules that use active-high pulses.

The IR submodule receives its clock sources from LPUART. The submodule selects one of these clocks to generate either 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulses during transmission.

51.4.7.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from the transmit shift register to the TXD signal. A narrow pulse is transmitted for a 0 bit and no pulse is transmitted for a 1 bit. The narrow pulse is sent at the start of the bit with a duration of 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR of a bit time. A narrow low pulse is transmitted for a 0 bit when CTRL[TXINV] = 0, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] = 1.

51.4.7.2 Infrared receive decoder

The infrared receive block converts data from the RXD signal to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. A narrow low pulse is expected for a 0 bit when STAT[RXINV] = 0, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] = 1. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

51.4.7.3 Start bit detection

When STAT[RXINV] = 0, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently of each other.

51.4.7.4 Noise filtering

The decoder ignores any rising edges detected during the first half of the infrared decoder counter, and can leave any pulses less than one oversampling baud clock as undetected. This is regardless of whether the pulse is seen in the first or second half of the count.

51.4.7.5 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as 0, which is sent to the receiver. The decoder counter is also reset.

51.4.7.6 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, the decoder sends a 1 to the receiver.

If the next bit is 0, which arrives late, a low bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, the delay of a 0 is not recorded as noise.

51.4.8 Interrupts and status fields

The LPUART transmitter has two status fields that can optionally generate hardware interrupt requests. STAT[TDRE] = 1 indicates that there is room in the transmit FIFO to write another transmit character to Data (DATA). If CTRL[TIE] = 1, a hardware interrupt is requested when STAT[TDRE] = 1.

STAT[TC] indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TXD at the inactive level. This field is often used in systems with modems to determine when it is safe to turn off the modem. If CTRL[TCIE] = 1, a hardware interrupt is requested when STAT[TC] = 1. Instead of hardware interrupts, software polling may be used to monitor STAT[TDRE] and STAT[TC] if the corresponding CTRL[TIE] or CTRL[TCIE] are 0.

When a program detects that STAT[RDRF] = 1, it gets the data from this field by reading Data (DATA). The field becomes 0 by reading Data (DATA).

STAT[IDLE] includes logic that prevents it from becoming 1 repeatedly when the RXD line remains idle for an extended period of time. STAT[IDLE] becomes 0 when you write 1 to it, and cannot become 1 again until the receiver has received at least one new character and has STAT[RDRF] = 1.

If the associated error is detected in the received character that caused STAT[RDRF] to become 1, STAT[NF], STAT[FE], and STAT[PF] become 1 at the same time STAT[RDRF] becomes 1. These flags do not become 1 in overrun cases.

If STAT[RDRF] is already 1 when a new character is ready to be transferred from the receive shifter to the receive FIFO, STAT[OR] becomes 1, instead of the data along with any associated STAT[NF], STAT[FE], or STAT[PF] condition is lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2], then STAT[MA1F] and/or STAT[MA2F] become 1 at the same time that STAT[RDRF] becomes 1.

At any time, an active edge on the RXD serial data input pin causes STAT[RXEDGIF] to become 1. STAT[RXEDGIF] becomes 0 when you write 1 to it. This function depends on the receiver being enabled (CTRL[RE] = 1).

51.4.8.1 End-of-packet DMA transfers

The end-of-packet functionality is designed for serial interfaces where you may not know the size of the transfer in advance and the data is being pushed by an external device. Examples include UART receive, I2C Slave mode, and SPI Slave mode. The end of packet processing ensures that data does not become stranded in either the receive FIFO or the DMA receive buffer. Support for end-of-packet processing must be implemented in both the serial interfaces and DMA controller.

The condition that signals the end of packet is different for each serial interface, but the serial peripheral and DMA process it in the same way. For example, an idle line condition signals the UART end of packet, the Stop and/or Repeated Start condition signals the I2C end of packet, and PCS negation signals the SPI end of packet.

When the serial peripheral is configured to signal the end-of-packet condition to the DMA and the serial interface detects an end-of-packet condition, it asserts the DMA request for the receive FIFO irrespective of the watermark configuration. For larger watermark configurations, this ensures that the last few words of the transfer are first flushed from the receive FIFO.

The DMA then reads the contents of the receive FIFO, depending on the first word in the FIFO:

- If the receive FIFO is empty, the serial interface signals an end of packet condition to the DMA controller.

- If the receive FIFO is not empty, but the first word in the FIFO is the start of a new packet, data is not pulled from the receive FIFO and the serial interface signals an end-of-packet condition to the DMA controller.
- If the receive FIFO is not empty, and the first word in the FIFO is not the start of a new packet, the DMA transfers the receive data as normal.

Because the DMA may be transferring multiple words on each request, the end-of-packet condition persists until the DMA minor loop has completed and no additional data is pulled from the receive FIFO. The field that triggered the end-of-packet condition becomes 0 when the minor loop completes following the end of packet being signaled to the DMA controller.

When the DMA detects the end-of-packet condition, it writes all received words up to the end of packet into the system memory and saves the destination address for the word after the last valid data. The DMA then terminates the channel as if the major loop completed, including final offsets and optional interrupts, channel linking, and scatter-gather. You can, optionally, save the final destination address to system memory. The final destination address is also available in the destination address register.

Because the DMA terminates the major loop, no servicing of the receive FIFO occurs until you or the hardware reconfigures the DMA (for example, channel linking or scatter-gather). You must minimize this delay to avoid receiver FIFO overrun. The automatic DMA end-of-packet processing is not recommended when there are only a few words transferred between end-of-packet conditions. This is because the DMA spends more time processing the end of packet than transferring the data. For example, the UART idle line length must be increased as needed to avoid an excessive number of idle conditions.

51.4.9 Peripheral triggers

The connection of the LPUART peripheral triggers with other peripherals is chip-specific.

Output triggers

LPUART generates the following output triggers that can be connected to other peripherals on the chip:

- The transmit word trigger asserts at the end of each transmitted word and negates after 1-bit period.
- The transmit data trigger is identical to the TXD pin output, but without support for input trigger modulation.
- The receive word trigger asserts at the end of each received word that is written to the receive FIFO, for one oversampling clock period.
- The receive idle trigger asserts when STAT[IDLE] becomes 1, for one oversampling clock period.

Input trigger

LPUART supports one peripheral input trigger that you can configure in one of the following ways:

- By enabling the CTS function: You can connect the input trigger instead of the CTS_B pin input. The input trigger must assert for longer than 1-bit clock period when the transmitter is idle, with data to send, to guarantee a new transmission.
- By making the input trigger modulate the transmit data output (trigger is logically ANDed with the TXD output): The input trigger is expected to be a free running clock (carrier signal) that generates from a timer or PWM source with a frequency that is greater than the bit-clock frequency. The carrier signal must not toggle faster than the maximum supported bit time.
- By connecting the input trigger instead of the RXD pin input: The input trigger is expected to be generated from a receive data source, such as an analog comparator or external pin.

51.5 Clocking and resets

Table 316. Clocks

LPUART functional clock	Is asynchronous to the bus clock and can remain enabled in Low-Power modes to support transmit and/or receive functions, including low-power wakeups.
Bus clock	Is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPUART transmit and receive registers, including the FIFOs.

Table 317. Resets

Chip reset	Enables the logic and registers for the LPUART transmitter and receiver to reset to their default states.
Software reset	Resets the LPUART logic and registers to their default states, except for Global (GLOBAL). GLOBAL[RST] controls the LPUART software reset.
FIFO reset	Implements write-only control fields that reset the transmit FIFO (FIFO[TXFLUSH]) and receive FIFO (FIFO[RXFLUSH]). After a FIFO is reset, that FIFO becomes empty.

51.6 External signals

Table 318. External signals

Signal	Description	I/O
TXD	Transmit data: This pin is normally an output, but is an input (tristated) in Single-Wire mode whenever the transmitter is disabled or transmit direction is configured for receive data.	I/O
RXD	Receive data	I
CTS_B	Clear-to-send	I
RTS_B	Request-to-send	O

Chapter 52
CAN (FlexCAN)

52.1 Chip-specific FlexCAN information

Table 319. Reference links to related information

Topic	Related module	Reference
Full description	CAN	CAN
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

52.1.1 Module instances

The parts with CAN module have one instance: CAN0. Refer to Ordering information for parts have CAN module.

52.1.2 Enhanced Rx FIFO under DMA operation

In order to use the DMA feature, the ERFCR[ERFWM] must be configured as 0b.

ERFSR[ERFDA] is set by the hardware when there is at least one frame available to be read from the Enhanced Rx FIFO, and ERFSR[ERFWMI] is set by the hardware when the number of frames stored in the FIFO is greater than 0 (the watermark defined by ERFCR[ERFWM]). A DMA request is generated when ERFSR[ERFWMI] is set. Upon receiving the request, the DMA controller can read the message in the Enhanced Rx FIFO output. The DMA controller performs one read loops in the FIFO output defined by ERFCR[ERFWM]. Also, each message reading process must end by the address defined in ERFCR[DMALW].

52.1.3 Maximum data rate

The maximum data rate for CAN in this device is 8 Mbps. Refer to device's Data sheet for more information.

52.1.4 Wakeup source from low-power mode on 40-pin package

In the 40-pin HVQFN package that supports FlexCAN interface, the pins PTC5 and PTC6 are tied together inside the device. The PTC5 works as a CAN interface while the PTC6 pin works as a wake-up source only. In order to wake up from low-power mode, the user uses PTC5 as CAN0_RX interface and must configure the PTC6 signal as input and wake up source in the target wake-up domain. When a CAN frame is received in pin PTC5, a wake-up interrupt will be triggered asserting PTC6.

NOTE

Do not configure PTC5 and PTC6 pins as output at the same time in 40-pin HVQFN packages. When not in use, they must be configure as disabled (ALT0).

NOTE

In 40-pin HVQFN package, PTC6 shall only be configured as an input wake up source if in use.

52.1.5 FlexCAN reset

The chip level hard reset input of FlexCAN is controlled by MRCC_CANn[RSTB], therefore a FlexCAN hard reset leads to reset all the module which results in all registers being affected by this reset. The chip level soft reset input is not supported on this device, use MCR[SOFTRST] to initiate a soft reset of the FlexCAN.

52.2 Overview

FlexCAN is a communication controller implementing the CAN protocol according to the ISO 11898-1:2015 standard and CAN 2.0 Part B protocol specifications.

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific real-time processing and reliable operation requirements in the electromagnetic interference (EMI) environment of a vehicle. FlexCAN is a full implementation of the CAN protocol specification, the CAN with flexible data rate (CAN FD) protocol, and the CAN version 2.0 Part B protocol. It supports both standard and extended message frames and long payloads.

NOTE

Legacy Receive (RX) FIFO cannot be used in Flexible Data (FD) mode.

NOTE

In CAN FD mode, use the Enhanced Receive FIFO feature instead of the Legacy Receive FIFO.

52.2.1 Block diagram

Figure 187 shows the main submodules implemented in FlexCAN.

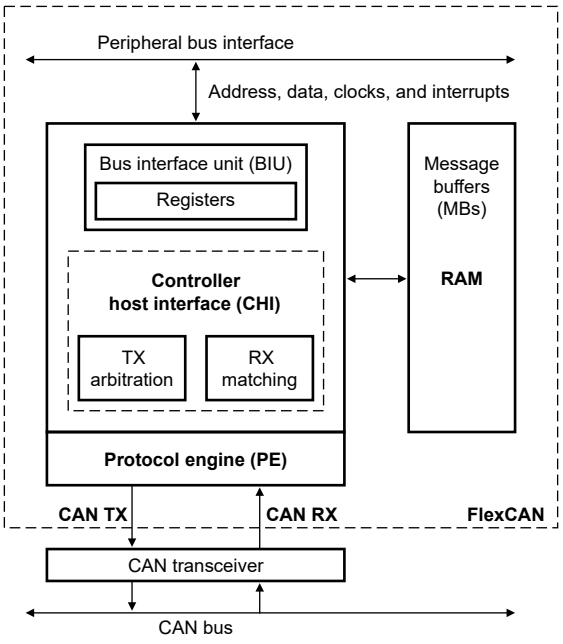


Figure 187. Block diagram

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- RAM access for receiving and transmitting message frames
- Receive message validation
- Error handling.
- CAN FD message detection

The Controller Host Interface (CHI) submodule manages:

- Message buffer (MB) selection for reception and transmission
- Arbitration and ID matching algorithms for both CAN FD and non-CAN FD message formats

The Bus Interface Unit (BIU) submodule controls access to and from the internal interface bus to establish connection to the CPU and to other blocks. The BIU manages access to:

- Clocks
- Address and data buses
- Interrupt outputs
- DMA
- Test signals

52.2.2 Features

- Full implementation of *CAN with Flexible data rate (CAN FD) protocol specification* and *CAN Specification Version 2.0, Part B*
  - Standard data frames
  - Extended data frames
  - Data length of 0–64 bytes
  - Programmable bit rate (see chip-specific FlexCAN information for specific maximum rate configuration)
  - Content-related addressing
- Compliance with ISO 11898-1:2015 standard
- Flexible message buffers that can be configured to store a payload of 0, 8, 16, 32, or 64 bytes
  - Increasing the payload size decreases the number of supported message buffers (see FlexCAN memory partition for CAN FD).
  - Message buffers are configurable as receive or transmit, supporting standard and extended messages.
- Individual Receive Mask registers for each message buffer
- Full-featured Legacy RX FIFO with storage capacity for six frames and automatic internal pointer handling with DMA support
- Full-featured Enhanced RX FIFO with storage capacity of 12 CAN FD frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Optional general purpose RAM space, using RAM not used by reception or transmission structures
- Listen-Only mode
- Programmable loopback mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Timestamp based on 16-bit free-running timer
- Global network time, synchronized by specific message
- Maskable interrupts
- Independence from transmission medium (external transceiver is assumed)
- Short latency time due to arbitration scheme for high-priority messages
- Low-power modes, with programmable wake-up on bus activity or matching with received frames (Pretended Networking)
- Transceiver delay compensation when transmitting CAN FD messages at faster data rates
- Management of remote request frames, automatically or by software
- Restriction only to write CAN bit time settings and configuration bits in Freeze mode
- Transmit message buffer status (lowest-priority buffer or empty buffer)

- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- ESR1[SYNCH] to indicate module is synchronous with CAN bus
- CRC status for transmitted message
- Legacy RX FIFO Global Mask register
- Selectable priority between message buffers and receive FIFO during matching process
- Powerful Legacy RX FIFO ID filtering, capable of matching incoming IDs against either 128 extended IDs, 256 standard IDs, or 512 partial (8-bit) IDs, with 32 individual masking capability
- Powerful Enhanced RX FIFO ID filtering, capable of matching incoming IDs against either 16 extended or 32 standard ID filter elements with three filtering schemes: mask plus filter, range, and two filters without mask.
- Complete backward compatibility with previous FlexCAN version
- Pretended Networking functionality in low power: Doze mode, Stop mode

52.3 Functional description

FlexCAN is a CAN protocol engine with a flexible message buffer system for transmitting and receiving CAN frames. The system is a set of message buffers (MBs) that stores configuration and control data, timestamp, message ID, and data (see Message buffer structure).

For classical CAN frames, FlexCAN supports simultaneous reception through Legacy FIFO and message buffer. For CAN FD frames, FlexCAN supports reception through message buffer and enhanced receive FIFO.

For message buffer reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed in the ID field. A masking scheme makes it possible to match the ID programmed on the message buffer with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of message buffers to be transmitted based on the message ID (optionally augmented by three local priority bits) or the message buffer ordering.

A message buffer is active at a given time if it can participate in both the matching and arbitration processes. A receive message buffer with a 0b code is inactive (see Table 354). A transmit message buffer with a 1000b or 1001b code is also inactive (see Table 355).

FlexCAN can receive and transmit messages in CAN FD format. The message buffers are sized to store the quantity of data bytes selected by FDCTRL[MBDSRn]. The quantity of FD message buffers available for a given quantity of data bytes is described in the CAN FD Control (FDCTRL) register. See also FlexCAN memory partition for CAN FD.

52.3.1 Modes of operation

FlexCAN has these functional modes:

Table 320. Functional modes

Mode	Description
Normal	In Normal mode, FlexCAN receives and transmits message frames, manages errors normally, and enables all CAN Protocol functions.
Freeze	Freeze mode is enabled when MCR[FRZ] = 1. If enabled, FlexCAN enters Freeze mode when MCR[HALT] is 1 or when is requested at chip level and FlexCAN writes 1 to MCR[FRZACK]. In this mode, no transmission or reception of frames is done, and synchronicity to the CAN bus is lost. See Freeze mode.
Loopback	FlexCAN enters this mode when CTRL1[LPB] becomes 1. In this mode, FlexCAN performs an internal loopback that can be used for self-test. The bit stream output of the transmitter is internally fed back to the receiver input. The receiving CAN input pin is ignored and the transmitting CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting and treats its own

Table continues on the next page...

Table 320. Functional modes (continued)

Mode	Description
	transmitted message as a message received from a remote node. To ensure proper reception of its own message, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field. Both transmit and receive interrupts are generated.
Listen-Only	FlexCAN enters this mode when CTRL1[LOM] becomes 1. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station are received. If FlexCAN detects an unacknowledged message, it flags a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.
CAN FD Active	In this mode, FlexCAN can transmit and receive all messages formatted according to the CAN FD Standard (2.0) and 2.0B Protocol in an interleaved fashion. The CPU can put FlexCAN into CAN FD Active mode by configuring MCR[FDEN] in Freeze mode.

Some features available in classical CAN are unavailable in CAN FD mode.

Table 321. Differences between classical CAN and CAN FD

Feature	Classical CAN	CAN FD
Legacy RX FIFO DMA	Yes	No
Legacy RX FIFO	Yes	No
Enhanced RX FIFO DMA	Yes	Yes
Enhanced RX FIFO	Yes	Yes
Pretended network support	Yes	No

FlexCAN can operate in these low-power modes:

Table 322. Low-power modes

Mode	Description
Module Disable	FlexCAN enters this mode when the CPU writes 1 to MCR[MDIS] and FlexCAN writes 1 to MCR[LPMACK]. After FlexCAN is disabled, it issues a request to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Writing 0 to MCR[MDIS] exits this mode. See Module Disable mode.
Doze	FlexCAN enters this mode when MCR[DOZE] is 1, Doze mode is requested at chip level, and FlexCAN writes 1 to MCR[LPMACK]. When in Doze mode, FlexCAN issues a request to disable the clocks to the CAN Protocol Engine and the CAN Controller-Host Interface submodules. This mode is exited when: • MCR[DOZE] becomes 0 • The chip is removed from Doze mode • Or activity is detected on the CAN bus and the Self-Wake-up mechanism is enabled. See Doze mode.
Stop	FlexCAN enters this mode when Stop mode is requested at chip level and FlexCAN writes 1 to MCR[LPMACK]. When in Stop mode, FlexCAN puts itself in an inactive state and then informs the CPU that the clocks can be shut down globally. Exit from this mode occurs when the Stop mode request is

Table continues on the next page...

Table 322. Low-power modes (continued)

Mode	Description
	removed, or when activity is detected on the CAN bus and the Self-Wake-up mechanism is enabled. See Stop mode.
Pretended Network (PN)	FlexCAN can operate in this mode with Doze mode or Stop mode. Before entering one of these low-power modes, you must write 1 to MCR[PNET_EN]. When in a low-power mode, CHI subblock clocks are shut down and CAN_PE subblock remains clocked. The receive process remains active to filter incoming messages (see Receive process in Pretended Networking mode) as defined by the configuration registers (see Pretended Networking Control 1 (CTRL1_PN)). Upon detecting a wake-up event, a wake-up interrupt is issued to the system. When MCR[PNET_EN] becomes 1, the CPU must disable Self-Wake-up by writing 0 to MCR[SLFWAK] (see Module Configuration (MCR).

52.3.1.1 Modes of operation details

FlexCAN has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following subsections contain functional details about Freeze mode and low-power modes.

CAUTION

FlexCAN does not support "Permanent Dominant" failure on the CAN bus line. If a Low-Power request or Freeze mode request occurs during a "Permanent Dominant" condition, the corresponding acknowledgment field can never be 1.

52.3.1.1.1 Freeze mode

This mode is requested either by the CPU writing 1 to MCR[HALT] or when the chip is put into Debug mode. MCR[FRZ] must be 1 and the module must not be in a low-power mode.

To obtain acknowledgment, FlexCAN writes 1 to MCR[FRZACK]. The CPU must only consider FlexCAN to be in Freeze mode when both request and acknowledgment conditions are satisfied.

When Freeze mode is requested, FlexCAN:

1. Waits to be in either Intermission, Error Passive, Bus Off, or Idle state.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in does not prevent entering Freeze mode.
3. Ignores the receive input pin and drives the transmit pin as recessive.
4. Stops the prescaler, halting all CAN protocol activities.
5. Grants write access to the Error Counters register, which is read-only in other modes.
6. Writes 1 to MCR[NOTRDY] and MCR[FRZACK].

After requesting Freeze mode, you must wait for MCR[FRZACK] to become 1 before executing any other action, otherwise FlexCAN may operate unpredictably. In Freeze mode, all memory-mapped registers are accessible.

Freeze mode is exited in one of these conditions:

- CPU writes 0 to MCR[FRZ].
- The chip is removed from Debug mode or the MCR[HALT] becomes 0.

MCR[FRZACK] becomes 0 after the protocol engine recognizes the negation of the freeze request. After leaving Freeze mode, FlexCAN tries to resynchronize to the CAN bus by waiting for 11 consecutive recessive bits.

52.3.1.1.2 Module Disable mode

This low-power mode is normally used to disable a complete FlexCAN block temporarily, with no power consumption. The CPU requests this mode by writing 1 to MCR[MDIS], and FlexCAN acknowledges the request by writing 1 to MCR[LPMACK]. The CPU must only consider FlexCAN to be in Disable mode when both the request and acknowledgment conditions are satisfied.

If FlexCAN is disabled during Freeze mode, the module requests to disable the clocks to the PE and CHI submodules, writes 1 to MCR[LPMACK] and writes 0 to MCR[FRZACK].

It is not recommended to use Module Disable mode under Pretended Networking mode. Write 0 to MCR[MDIS] and wait for MCR[LPMACK] to become 0 before writing 1 to MCR[PNET_EN].

If the module is disabled during transmission or reception, FlexCAN:

1. Waits to be in either Idle or Bus Off state, or waits for the third bit of Intermission and then checks that it is recessive.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. FlexCAN does not take a pending move-in into account.
3. Ignores its receive input pin and drives its transmit pin as recessive.
4. Shuts down the clocks to the PE and CHI submodules.
5. Writes 1 to MCR[NOTRDY] and MCR[LPMACK].

In this mode, the Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except for:

- The Receive Message Buffers Global Mask registers
- The Receive Buffer 14 Mask register
- The Receive Buffer 15 Mask register

When in Disable mode, these items may not be accessed:

- The message buffers
- The Receive Individual Mask registers
- The reserved words within RAM

To exit this mode, the CPU writes 0 to MCR[MDIS], causing FlexCAN to request to resume the clocks and write 0 to MCR[LPMACK]. This write occurs after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

52.3.1.1.3 Doze mode

This is a system low power mode in which the CPU bus is kept alive and a global Doze mode request is sent to all peripherals asking them to enter low-power mode. When Doze mode is globally requested, MCR[DOZE] needs to have been asserted previously for Doze mode to be triggered. The acknowledgement is obtained through the assertion by the FlexCAN of MCR[LPMACK]. The CPU must only consider the FlexCAN to be in Doze mode when both request and acknowledgement conditions are satisfied.

If Doze mode is triggered during Freeze mode, FlexCAN requests to shut down the clocks to the PE and CHI submodules, sets MCR[LPMACK] and negates MCR[FRZACK]. If Doze mode is triggered during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive.
- Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive.
- Shuts down the clocks to the PE and CHI submodules.
- Sets MCR[NOTRDY] and MCR[LPMACK].

The Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except the Rx Mailboxes Global Mask registers, the Rx Buffer 14 Mask register, the Rx Buffer 15 Mask register. The message buffers, the Rx Individual Mask registers, and the reserved words within RAM may not be accessed when the module is in Doze mode.

Exiting Doze mode is done in one of the following ways:

- CPU removing the Doze mode request
- CPU negating MCR [DOZE]
- Self Wake mechanism

In the Self Wake mechanism, if MCR[SLFWAK] was set at the time FlexCAN entered Doze mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN negates the DOZE bit, requests to resume its clocks and negates the LPMACK after the CAN protocol engine recognizes the negation of the Doze mode request. It also sets ESR [WAKINT] and, if enabled by MCR[WAKMSK], generates a Wake-Up interrupt to the CPU. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLFWAK and WAKMSK upon wakeup from Doze mode.

Table 323. Wakeup from Doze mode

SLFWAK	WAKINT	WAKMSK	FlexCAN clocks enabled	Wakeup interrupt generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	Yes	No
1	1	1	Yes	Yes

52.3.1.1.4 Stop mode

In this low-power mode for the system, all chip clocks can be stopped for maximum power savings. Stop mode is globally requested by the CPU and FlexCAN writes 1 to a Stop Acknowledgment signal to indicate acknowledgment. The CPU must only consider FlexCAN to be in Stop mode when both request and acknowledgment conditions are satisfied.

If FlexCAN receives the global Stop mode request during Freeze mode, it shuts down the clocks globally by:

1. Writing 1 to MCR[LPMACK].
2. Writing 0 to MCR[FRZACK].
3. Sending the Stop Acknowledge signal to the CPU.

If Stop mode is requested during transmission or reception, FlexCAN:

1. Waits to be in either Idle or Bus Off state, or waits for the third bit of Intermission and verifies that it is recessive.
2. Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. FlexCAN does not take a pending move-in into account.
3. Ignores its receive input pin and drives its transmit pin as recessive.
4. Writes 1 to MCR[NOTRDY] and MCR[LPMACK].
5. Sends a Stop Acknowledge signal to the CPU, so that the CPU can shut down the clocks globally.

FlexCAN exits Stop mode when the CPU resumes the clocks and removes the Stop mode request. This exit can occur as a result of the Self-Wake mechanism.

In Self-Wake, if MCR[SLFWAK] = 1 when FlexCAN enters Stop mode, then upon detecting a recessive-to-dominant transition on the CAN bus, FlexCAN writes 1 to ESR1[WAKINT]. If enabled by MCR[WAKMSK], FlexCAN generates a wake-up interrupt to the CPU. Upon receiving the interrupt, the CPU should resume the clocks and remove the Stop mode request. FlexCAN waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, FlexCAN does not receive the frame that woke it up. Table 324 details the effect of MCR[SLFWAK] and MCR[WAKMSK] upon wakeup from Stop mode. Waking from Stop mode only works when both fields are 1.

After the CAN protocol engine recognizes the negation of the Stop mode request, FlexCAN writes 0 to MCR[LPMACK]. FlexCAN then waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, FlexCAN does not receive the frame that woke it up.

Table 324. Waking from Stop mode

SLFWAK	WAKINT	WAKMSK	Chip clocks enabled	Wake-up interrupt generated
0	—	—	No	No
0	—	—	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	No	No
1	1	1	Yes	Yes

NOTE

When FlexCAN is in the Bus Off state and Low-Power mode, FlexCAN may take an extra 128 IDLE cycles to leave the Bus Off state after exiting Low-Power mode.

52.3.1.1.5 Pretended Networking (PN) mode

This special low-power mode receives wake-up messages with low power consumption. This mode can be selected to operate together with Doze mode or Stop mode. Before entering a low-power mode, MCR[PNET_EN] must be 1. After entering a low-power mode, the CHI subblock is shut down and the CAN_PE subblock is kept active. The receive process is still active to filter incoming messages (see Receive process in Pretended Networking mode) as defined by the configuration registers (see Pretended Networking Control 1 (CTRL1_PN)). Upon detecting a wake-up event, FlexCAN issues a wake-up interrupt to the system.

To enter Pretended Networking mode, FlexCAN must be in Normal mode, not in Freeze or Module Disable mode. In Pretended Networking mode, FlexCAN keeps itself synchronized with the CAN bus in Doze or Stop mode. Then, when either Doze or Stop mode is requested, FlexCAN:

1. Waits to be in the Idle state, or waits for the third bit of Intermission, then verifies that it is recessive.
2. Writes 1 to MCR[LPMACK].
3. Requests the shutdown of the CHI submodule clock, while keeping the PE submodule clock active.

FlexCAN can exit Pretended Networking mode in one of these ways:

- The CPU removes the Doze mode request.
- The CPU writes 0 to MCR[DOZE].
- The CPU removes the Stop mode request.
- FlexCAN waits until Bus Idle, or until the third bit of Intermission state, to write 0 to MCR[LPMACK].

The above exit events can be triggered:

- By FlexCAN, after detecting a wake-up event and issuing the respective interrupt.
- By the CPU itself, when another method wakes it up.

Consequently, FlexCAN waits until the Bus Idle state, or until the third bit of the Intermission state, to write 0 to MCR[LPMACK] and resume Normal mode. This procedure ensures that FlexCAN is synchronized to the CAN bus after exiting Pretended Networking mode. The CPU must wait for MCR[LPM_ACK] to become 0 before performing any access to FlexCAN.

When MCR[PNET_EN] becomes 1, the CPU must disable the self-wake-up by writing 0 to MCR[SLFWAK] (see Module Configuration (MCR)).

NOTE

For more information about the difference between FD and non-FD regarding this feature, see Table 321.

52.3.2 Transmission process

NOTE

Instances of MB_CS in this topic refer to items in message buffers. See Message buffer structure for details.

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt flag is set, and clear it if necessary.
2. If the message buffer is active (transmission pending), request an abort of the transmission. Write the ABORT code (1001b) to the CODE field of the Control and Status word.
3. Poll the IFLAG register until the corresponding IFLAG flag is set, or wait for the interrupt request (if enabled by the respective IMASK field).
4. Read the CODE field to identify whether the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIOEN]).
7. Write payload data bytes.
8. Configure the Control and Status word as needed.
    a. Set ID type via MB_CS[IDE].
    b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
    c. If MCR[FDEN] = 1, configure MB_CS[EDL] and MB_CS[BRS]. For details about the relationship between the written value and transmitted value of MB_CS[ESI], see Table 335.[2]
    d. Configure Data Length Code in bytes via MB_CS[DLC]. See Table 357 for detailed information.
    e. To transmit the CAN frame, activate the message buffer by writing Ch to MB_CS[CODE].

NOTE

To maximize software performance, configure all the fields in the MB_CS word in a single 32-bit write operation. If the fields are configured in separate writes, MB_CS[CODE] must be the last write in the Control and Status word.

When the MB is activated, it participates in the arbitration process and its contents are eventually transmitted according to its priority. When the DLC value stored in the MB selected for transmission exceeds the MB payload size, FlexCAN completes the expected DLC by adding a constant CCh pattern.

[2] There is no need to write the ESI field; it is automatically transmitted as dominant by error active nodes and as recessive by error passive nodes. If FlexCAN is operating as a network gateway, however, the CPU writes MB_CS[ESI] according to the error status of the node that sent the message.

After a successful transmission:

1. The value of the free-running timer is written into the timestamp field.
2. The CODE field in the Control and Status word is updated.
3. Both Cyclic Redundancy Check (CRCR) and CAN FD CRC (FDCRC) are updated.
4. A status flag is set in the Interrupt Flag register.
5. If allowed by the corresponding Interrupt Mask Register field, an interrupt is generated.

After transmission, the new CODE field depends on the code used to activate the message buffer (see Table 354 and Table 355 in Message buffer structure).

When the Abort feature is enabled (MCR[AEN] is 1), after the Interrupt flag is set for an MB configured as transmit buffer, the message buffer is blocked. The CPU cannot update the message buffer until the Interrupt Flag is cleared by the CPU. The CPU must clear the corresponding IFLAG flag before preparing this MB for a new transmission or reception.

NOTE

For backward compatibility (MCR[AEN] is 0), write the INACTIVE code (1000b) to the CODE field to deactivate the MB. In this case, the pending frame may be transmitted without notification (see Message buffer inactivation).

52.3.3 Arbitration process

The arbitration process scans the message buffers, searching for the transmission MB that holds the message to be sent at the next opportunity. This MB is called the arbitration winner. The scan starts from the lowest number MB and continues to the higher ones.

The arbitration process is triggered when at least one of the following occur:

- CRC field of the CAN frame arrives. The starting point depends on the value of CTRL2[TASD].
- Error Delimiter field of a CAN frame is in progress.
- Overload Delimiter field of a CAN frame is in progress.
- The winner of the arbitration is deactivated, and the CAN bus has not reached the first bit of the Intermission field.
- CPU writes to the Control and Status word of a winner MB, and the CAN bus has not reached the first bit of the Intermission field.
- CHI is in the Idle state and the CPU writes to the Control and Status word of any message buffer.
- FlexCAN exits Bus Off state.
- FlexCAN leaves Freeze mode or a low-power mode.

If the arbitration process does not evaluate all message buffers before the CAN bus reaches the first bit of the Intermission field, the temporary arbitration winner is invalidated. FlexCAN will not compete for the CAN bus at the next opportunity.

The arbitration process selects the winner among the active transmission message buffers at the end of the scan according to the values of CTRL1[LBUF] and MCR[LPRIOEN].

See Arbitration process (continued) for more information about this process.

52.3.3.1 Lowest-number message buffer first

If CTRL1[LBUF] is 1, the first (lowest number) active transmission message buffer found is the arbitration winner. MCR[LPRIOEN] has no effect when CTRL1[LBUF] is 1.

52.3.3.2 Highest-priority message buffer first

If CTRL1[LBUF] is 0, the arbitration process searches for the active transmission message buffer with the highest priority. The frame of this message buffer has a higher probability of winning the arbitration on the CAN bus when multiple external nodes compete for the bus simultaneously.

The sequence of bits considered for this arbitration is called the arbitration value of the message buffer. The transmission message buffer with the lowest arbitration value among all transmission message buffers has the highest priority.

If two or more message buffers have equivalent arbitration values, the message buffer with the lowest number is the arbitration winner.

The composition of the arbitration value depends on MCR[LPRIOEN].

52.3.3.2.1 Local priority disabled

If MCR[LPRIOEN] = 0, the arbitration value is built using the exact sequence of bits that would be transmitted in a CAN frame where local priority is disabled.

Table 325. Composition of the arbitration value when local priority is disabled

Format	Message buffer arbitration value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	— (18 bits)	— (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

52.3.3.2.2 Local priority enabled

To enable local priority, MCR[LPRIOEN] must be 1. In this case, the message buffer PRIO field (see Message buffer structure) is included at the very left of the arbitration value.

Table 326. Composition of the arbitration value when local priority is enabled

Format	Message buffer arbitration value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	— (18 bits)	— (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

Because the PRIO field is the most significant part of the arbitration value, message buffers with low PRIO values have higher priority than message buffers with high PRIO values. This priority is maintained regardless of the rest of their arbitration values.

The PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

52.3.3.3 Arbitration process (continued)

After the arbitration winner is found (see Arbitration process), its content is copied to a hidden auxiliary message buffer called a transmit serial message buffer (TX SMB). The TX SMB has the same structure as a normal message buffer, but is not user-accessible. This copy operation is called move-out. After it is done, write access to the Control and Status word of the corresponding MB is blocked (if MCR[AEN] = 1). Write access is restored in one of the following events:

- The CPU clears the corresponding IFLAG flag after the message buffer is transmitted.
- FlexCAN enters Freeze mode or Bus Off state.
- FlexCAN loses the bus arbitration, or there is an error during the transmission.

At the first opportunity window on the CAN bus, the message on the TX SMB is transmitted according to the CAN protocol rules.

The arbitration process can be triggered under the following conditions:

- During RX and TX frames from CAN CRC field to end of frame. The value of CTRL2[TASD] may be changed to optimize the arbitration start point.

- During CAN Bus Off state from TX_ERR_CNT = 124 to 128. The value of CTRL2[TASD] may be changed to optimize the arbitration start point.
- During Control and Status write by CPU in Bus Idle. The first Control and Status write starts the arbitration process, and a second Control and Status write during this same arbitration restarts the process. If other Control and Status writes are performed, the transmission arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and Bus Idle state starts, then an arbitration process is triggered. In this case, the first and second Control and Status writes in Bus Idle do not restart the arbitration process. If there is not enough time to finish arbitration in the Wait For Bus Idle state and the next state is Idle, the scan is not interrupted. That scan is completed during Bus Idle state. During this arbitration, a Control and Status write does not cause an arbitration restart.
- Deactivation of an arbitration winner during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the Wait For Bus Idle state). If a resynchronization occurs during Wait For Bus Idle, the arbitration process is restarted.

The arbitration process stops when:

- All message buffers are scanned.
- A transmission message buffer is found (if lowest buffer feature is enabled).
- An arbitration winner deactivates or aborts during any arbitration process.
- There is not enough time to finish the transmission arbitration process (for instance, when a deactivation is performed near the end of frame). In this case, the arbitration process is pending.
- An error or overload flag occurs in the bus.
- A low-power or Freeze mode request occurs in Idle state.

Arbitration is considered pending when:

- It is not possible to finish arbitration process in time.
- A Control and Status write occurs during arbitration, when that write is performed in an MB whose number is lower than the transmission arbitration pointer.
- Any Control and Status write occurs when there is no transmission arbitration process in progress.
- RX Match has updated an RX code to TX code.
- FlexCAN enters the Bus Off state.

If a Control and Status write is performed in the arbitration winner, a new process is restarted immediately.

If a Control and Status write is performed in an MB whose number is higher than the transmission arbitration pointer, the ongoing arbitration process scans this MB as normal.

52.3.4 Receive process

To be able to receive CAN frames into a message buffer, the CPU must prepare it for reception by executing the following steps:

1. If the message buffer is active (either TX or RX), deactivate it (see Message buffer inactivation), preferably with a safe deactivation (see Transmission abort mechanism).
2. Write the ID word into the message buffer.
3. To activate the message buffer, write the EMPTY code (0100b) to the CODE field of the Control and Status word. No setup is required for EDL, BRS, and ESI fields. The respective fields in the received message overwrite these fields.

After the MB is activated, it can receive frames that match the programmed filter. At the end of a successful reception, the move-in process updates the message buffer (see Move-in) as follows:

1. The received data field (up to eight bytes for Classical CAN message format and up to 64 bytes for CAN FD message format) is stored.

2. The received Identifier field is stored.
3. The value of the free-running timer when the second bit of the Identifier field of the frame is written into the Timestamp field of the MB.
4. The received SRR, IDE, RTR, EDL, BRS, ESI, and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 354 and Table 355 in Section Message buffer structure).
6. If allowed by the corresponding Interrupt Mask field, a status flag is set in the Interrupt Flag register and an interrupt is generated.

The recommended way for the CPU to service (read) the frame received in a message buffer is:

1. Read the Control and Status word of that message buffer.
2. Verify that the BUSY bit is 0, indicating that the message buffer is locked. If it is not 0, repeat step 1 until it becomes 0. See Message buffer lock mechanism.
3. Read the contents of the message buffer. After the message buffer is locked, FlexCAN move-in processes do not modify its contents. See Move-in.
4. Acknowledge the proper flag in the IFLAG registers.
5. Unlock the message buffer by reading the free-running timer.

To verify frame reception, the CPU should poll the status flag bit for that specific message buffer in one of the IFLAG registers, not the CODE field of that message buffer. Polling the CODE field does not work in this case. After a frame is received and the CPU services the message buffer (by reading the Control and Status word and unlocking the message buffer), the CODE field does not return to EMPTY. It remains FULL, as explained in Table 354. If the CPU tries to work around this behavior by writing to the Control and Status word to force an EMPTY code after reading the message buffer without a prior safe deactivation, a newly received frame matching the filter of that message buffer may be lost.

CAUTION

In summary: never poll by reading the Control and Status word of the message buffers directly. Instead, read the IFLAG registers.

The Identifier field of the received frame is always stored in the matching message buffer. If the match was due to masking, the contents of the ID field in a message buffer may change. When MCR[SRXDIS] becomes 1, FlexCAN does not store frames transmitted by itself in any MB, even if it contains a matching receive MB. Also, no interrupt flag or interrupt signal is generated. Otherwise, when MCR[SRXDIS] becomes 0, if a matching receive MB exists, FlexCAN can receive frames transmitted by itself.

When MCR[DMA]) becomes 1, upon receiving a frame in the Legacy FIFO, IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Legacy RX FIFO in DMA Operation). The IMASK1 fields in the Legacy RX FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 80h address, optional).
2. Read the ID field (read 84h address, optional).
3. Read all data bytes (start read at 88h address, optional).

52.3.5 Matching process

The matching process scans the message buffer memory for RX MBs programmed with the same ID as the one received from the CAN bus. The matching starts from the lowest number message buffer and continues toward the higher-numbered ones.

For enhanced RX FIFO, see Enhanced RX FIFO.

For legacy RX FIFO, see Legacy RX FIFO.

As the frame is received, it is stored in a hidden auxiliary MB called Receive Serial Message Buffer (RX SMB).

The starting point of the matching process depends on the following conditions:

- If the received frame is a remote frame, the starting point is the CRC field of the frame.
- If the received frame is a data frame with DLC field equal to zero, the starting point is the CRC field of the frame.
- If the received frame is a data frame and the DLC field has a nonzero value, the starting point is the DATA field of the frame.

If a matching ID is found in one of the message buffers, the move-in process transfers the contents of the RX SMB to the matched MB. If any CAN protocol error is detected, no match results are transferred to the matched MB at the end of reception.

The matching process scans all matching elements of the active receive MBs (CODE is EMPTY, FULL, OVERRUN, or RANSWER). The process searches for a successful comparison to the matching elements of the RX SMB that is receiving the frame on the CAN bus. The RX SMB has the same structure as a message buffer. The reception structures (message buffers) associated with the matching elements that had a successful comparison are the matched structures. The matching winner is selected at the end of the scan among those matched structures. The matching winner depends on conditions described in Table 327.

Table 327. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Message buffer	0	—	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY, FULL, or OVERRUN
Message buffer	0	—	1	cmp_msk	cmp_msk	cmp_msk	EMPTY, FULL, or OVERRUN
Message buffer	1	0	—	cmp	no_cmp	cmp	RANSWER
Message buffer	1	1	0	cmp	no_cmp	cmp_msk	EMPTY, FULL, or OVERRUN
Message buffer	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY, FULL, or OVERRUN

1. For message buffer structure, if SMB[IDE] is 1, the ID is 29 bits (ID Standard plus ID Extended). If SMB[IDE] is 0, the ID is 11 bits (ID Standard). For Legacy RX FIFO structure, the ID depends on IDAM.
2. cmp: Compares the RX SMB contents to the MB contents regardless of masks.
3. no_cmp: The RX SMB contents are not compared to the MB contents.
4. cmp_msk: Compares the RX SMB contents to MB contents, accounting for masks.

NOTE

For Enhanced RX FIFO, see Enhanced RX FIFO matching process.

A reception structure is free-to-receive when any of the following conditions is satisfied:

- The CODE field of the message buffer is EMPTY.
- The CODE field of the message buffer is either FULL or OVERRUN, and it has already been serviced (the CPU has read the Control and Status word and unlocked it as described in Message buffer lock mechanism).
- The CODE field of the message buffer is either FULL or OVERRUN and an inactivation (see Message buffer inactivation) is performed.

The scan order for message buffers is from the matching element with the lowest number to the higher ones.

52.3.5.1 Matching priority

MCR[IRMQ] affects the matching winner search for MBs. If the field is 0, the matching winner is the first matched MB whether it is free-to-receive or not. If it is 1, the matching winner is selected according to this priority:

1. The first free-to-receive matched message buffer
2. The last non-free-to-receive matched message buffer

52.3.5.2 Special cases

If a non-safe MB inactivation (see Message buffer inactivation) occurs during the matching process and the inactivated MB is the temporary matching winner, the temporary matching winner is invalidated. The matching elements scan is not stopped and not restarted; it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist. In this case, a message may be lost.

Consider an example where:

- MCR[IRMQ] is 1.
- There are two message buffers with the same ID: the second and fifth MBs in the array.
- FlexCAN starts receiving messages with that ID.

When the first message arrives, the matching algorithm finds the first match in message buffer number 2. The code of this message buffer is EMPTY, so the message is stored in that MB. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive." It continues looking, finds MB number 5, and stores the message in that MB. If yet another message with the same ID arrives, the matching algorithm finds no matching free-to-receive MBs, so it overwrites the last matched message buffer (MB number 5). In doing so, it updates the CODE field of the message buffer to OVERRUN.

The ability to match the same ID in more than one MB can be used to implement a reception queue to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the message buffers. The CPU can examine the Timestamp field of the message buffers to determine the order in which the messages arrived.

Matching a range of IDs is possible via ID acceptance masks. FlexCAN supports individual masking per message buffer (see Receive Individual Mask (RXIMR0 - RXIMR31)). During the matching algorithm, if a mask field is 1, the corresponding ID bit is compared. If the mask field is 0, the corresponding ID bit is a "don't care". Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed when the module is in Freeze mode; otherwise, the module blocks them.

FlexCAN also supports an alternate masking scheme with only RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), and Receive 15 Mask (RX15MASK) for backward compatibility with legacy applications. This alternate masking scheme is enabled when the MCR[IRMQ] = 0.

52.3.6 Receive process in Pretended Networking mode

Pretended Networking mode adds specific wake-up functionality in low-power modes (Doze and Stop mode). When Pretended Networking (PN) mode is enabled by writing 1 to MCR[PNET_EN], FlexCAN continues processing received CAN messages in low-power mode. FlexCAN is able to detect specific wake-up messages by filtering them against identifier (ID) and payload target values using preselected matching criteria.

NOTE

Wake-up functionality is not available for messages in CAN FD format. When in PN mode, CAN FD format messages are ignored.

PN registers are located in the 0B00h–0B7Ch address range and can be written only in Freeze mode. These registers are used for writing PN configuration (both control and target values) prior to entering PN mode. They are also used for reading wake-up flags and the received message ID and data when returning to Normal mode after wake-up. The CPU must wait for MCR[LPMACK] to become 0 before performing any access to FlexCAN PN registers.

PN control registers are described in Pretended Networking Control 1 (CTRL1_PN) and Pretended Networking Control 2 (CTRL2_PN). The control fields that configure the filtering criteria are:

- Payload filtering selection: CTRL*n*_PN[PLFS]
- ID filtering selection: CTRL*n*_PN[IDFS]
- Filtering combination selection: CTRL*n*_PN[FCS]

Table 328. PN target values

Value	Description
FLT_IDE	IDE target value used to filter the incoming message by its format (standard or extended)
FLT_RTR	RTR target value used to filter the incoming message by its type (data or remote frame)
FLT_DLC_HI and FLT_DLC_LO	Target DLC range used to filter the size of the payload of an incoming message
FLT_ID1	ID target value used to filter the incoming message ID (equal to, smaller than or equal to, greater than or equal to, or the lower limit value in an ID range)
FLT_ID2	ID target value used as the upper limit in an ID range
PL1	Payload target value used to filter the incoming message payload (equal to, smaller than or equal to, greater than or equal to, or the lower limit value in a payload range)
PL2	Payload target value used as the upper limit in a payload range

IDE, RTR, ID, and payload filters have their respective masks. The ones in these masks determine which bits are considered in equality comparisons. The zeroes in these masks determine which bits are don't care. ID and payload masks are used only for exact ID and exact payload comparisons.

52.3.6.1 ID filtering

The IDs of incoming messages can be filtered based on the following criteria:

- Match the exact ID value, found by comparing the ID field of the incoming message to the content of target Pretended Networking ID Filter 1 (FLT_ID1). The ID mask is used.
- Less than or equal to the maximum range of ID. That is, any message with ID value smaller than or equal to the content of target FLT_ID1 is accepted. The ID mask is not used.
- Greater than or equal to the minimum range of ID. That is, any message with ID value greater than or equal to the content of target FLT_ID1 is accepted. The ID mask is not used.
- Inside a range of IDs. Any message with an ID value greater than or equal to the content of target FLT_ID1 and smaller than or equal to the content of target Pretended Networking ID Filter 2 or ID Mask (FLT_ID2_IDMASK) is accepted. The ID mask is not used.

See CTRL1_PN[IDFS].

The above criteria for ID filtering must be coherent with FLT_IDE and FLT_RTR target values in Pretended Networking ID Filter 1 (FLT_ID1). Only RX frames that match the respective IDE and RTR bits to the contents of FLT_ID1[FLT_IDE] and FLT_ID1[FLT_RTR] are compared. When a range of IDs is selected (CTRL1_PN[IDFS] = 11), both FLT_ID1 and FLT_ID2 are referred to the same FLT_IDE and FLT_RTR fields in FLT_ID1.

The ID mask is applied only to the exact ID comparison filtering option (CTRL1_PN[IDFS] = 00) to determine which bits are considered in the comparison. For the exact match option, the mask can select any bit within the ID field. For maximum range, minimum range, and inside range comparisons, the ID mask is not considered.

The IDE and RTR masks are applied in both exact and range ID comparison filtering options to determine which bits are considered in comparison.

52.3.6.2 Payload filtering

Similar to the ID criteria, 64-bit data or payloads (PL) of incoming messages can be filtered based on the following criteria:

- A match with the exact payload value, found by comparing the payload field of the incoming message to the content of PL1. The payload mask is used.
- Less than or equal to the maximum range of payload. That is, any message with payload value smaller than or equal to the content of PL1 is accepted. The payload mask is not used.
- Greater than or equal to the minimum range of payload. That is, any message with payload value greater than or equal to the content of PL1 is accepted. The payload mask is not used.
- Inside a range of payloads. Any message with a payload value greater than or equal to the content of PL1 and smaller than or equal to the content of PL2 is accepted. The payload mask is not used.

See CTRL1_PN[PLFS].

The above criteria for payload filtering must be coherent with upper and lower limit values in Pretended Networking Data Length Code (DLC) Filter (FLT_DLC). The payload of an incoming message is filtered using the selected criteria only if the DLC value of the incoming message is inside a DLC range:

- Greater than or equal to FLT_DLC[FLT_DLC_LO] (lower limit)
- Lower than or equal to FLT_DLC[FLT_DLC_HI] (upper limit)

A DLC value outside the specified range results in a mismatch. If you configure FLT_DLC_LO = FLT_DLC_HI, only payloads of the specified quantity of bytes are filtered. DLC is not maskable.

When the inside range of payloads option is selected (CTRL1_PN[PLFS] = 11), both PL1 and PL2 are considered with the 8-byte data length. All data bytes excluded by the DLC of the received message are considered to have value zero.

The payload mask is only used in the exact match option (CTRL1_PN[PLFS] = 00). This mask determines which bits or bytes in the 8-byte data field of both incoming message and the contents of PL1 register are used for matching. Mask length must meet the expected range of DLC values. For maximum range, minimum range, and inside range comparisons, the payload mask is not considered.

When FlexCAN receives a remote frame and CTRL1_PN[FCS] is configured to select payload comparison, payload filtering is not considered and the comparison results in a mismatch.

52.3.6.3 Other filtering

Incoming messages can also be filtered based on the quantity and rate of message reception, specifically:

- Several messages that match the filtering criteria for ID or payload for a predefined number of times. This number can be from 1 to 255. See Pretended Networking Control 1 (CTRL1_PN).
- No message matching the filtering criteria for ID or payload up to a timeout trigger. That is, non-reception of a matching message for a defined quantity of time. See Pretended Networking Control 2 (CTRL2_PN).

When the counter reaches the predefined timeout value, FlexCAN can generate a wake-up timeout event from an internal timer with associated comparator circuitry capable of generating a timeout flag. This behavior is specified in CTRL2_PN[MATCHTO].

The above filtering criteria can be used together as follows:

- Message ID filtering only
- Message ID filtering and Payload filtering
- Message ID filtering only occurring *n* times
- Message ID filtering and Payload filtering occurring *n* times

The timeout counter runs concurrently with the reception filtering process. Both engines (timeout counter and message filtering) are independent. If an incoming message matches the selected filter criteria, the timeout counter continues counting until the CPU wakes up. If the timeout counter reaches the target value, the message filtering process continues to filter incoming messages until the CPU wakes up. WU_MTC[MCOUNTER] reports the number of matched messages that occurred in Pretended Networking mode up to the moment the CPU wakes up.

In Pretended Networking mode, the wake-up event that may occur sets the respective wake-up flag (see Pretended Networking Wake-Up Match (WU_MTC)):

- WU_MTC[WUMF] indicates a successful match meeting the selected filtering criteria.
- WU_MTC[WTOF] indicates a timeout trigger.

Any of these flags generates interrupts to the CPU, provided the respective mask bits are enabled (CTRL1_PN[WUMF_MSK] = 1 or CTRL1_PN[WTOF_MSK] = 1).

There are four Wake-up Message Buffers (WMBs) used to store incoming messages in Pretended Networking mode. Up to four messages can be stored (see Wake-Up Message Buffer (WMB0_CS - WMB3_CS)).

When CTRL1_PN[NMATCH] = 1, only one message is received if the filtering criteria are matched. This message is stored in WMB0.

If CTRL1_PN[NMATCH] is between two and four, WMB1, WMB2, and WMB3 store the second, third, and fourth matching messages, respectively.

If CTRL1_PN[NMATCH] is greater than four, the last four matching messages are stored in the WMBs. The WMB index indicates the arrival order. The last message is stored in WMB3.

Only the valid data bytes of the incoming match message are stored in the data field of WMBs. The non-valid data bytes are read as zero. If DLC = 0 and RTR = 1, the data field is filled with zeroes. In any of the above cases, the wake-up interrupt is generated only when the filtering criteria are completed and CTRL1_PN[WUMF_MSK] = 1.

When a non-match wake-up event occurs (timeout or external) and WU_MTC[MCOUNTER] ≥ 4, the message stored in WMB0 does not have valid content. WMB0 is used as a buffer for the current message in the CAN bus. Messages received during Pretended Networking mode do not have time stamps, and the respective field in the WMB structure must be ignored.

In low-power mode (Doze or Stop), all processes are shut down except for the PN functionality inside the CAN_PE subblock, which remains clocked by the oscillator clock (see Clock domains and restrictions). FlexCAN continues to receive incoming messages, but only compares them to the predefined target values according to the selected filtering criteria. The matching, arbitration, move-in, and move-out processes, normally available in Normal mode, are not performed in Pretended Networking mode.

FlexCAN in Pretended Networking reacts to messages on the CAN bus in the same manner as in Normal mode. It generates acknowledge bits, detect and count errors, and so on.

52.3.7 Move process

There are two types of move process: move-in and move-out.

52.3.7.1 Move-in

The move-in process is the copying of a message received by an RX SMB to an RX message buffer that has matched it. Each RX SMB has its own move-in process, but only one is performed at a given time. The move-in starts only when the message held by the RX SMB has a corresponding match (see Matching process) and all of the following conditions are true:

- The CAN bus has reached or already gone past:
  - The second bit of Intermission field next to the frame that carried the message that is in the RX SMB.
  - The first bit of an overload frame next to the frame that carried the message in the RX SMB.
- There is no ongoing matching process.
- The CPU is not locking the destination message buffer.
- No move-in process from another RX SMB is ongoing. If more than one move-in process is to be started at the same time, both are performed and the newest process substitutes for the oldest.

The term "pending move-in" is used throughout the documentation and stands for a move-to-be that does not satisfy all of the above conditions.

If any of the following conditions is satisfied, the move-in is canceled and the RX SMB is able to receive another message:

- The destination message buffer is inactivated after the CAN bus has reached the first bit of the Intermission field next to the frame that carried the message. Also, its matching process has finished.
- There is a previous pending move-in to the same destination message buffer.
- The RX SMB is receiving a frame transmitted by FlexCAN itself and self-reception is disabled (MCR[SRXDIS] = 1).
- Any CAN protocol error is detected.

If the module enters Freeze or Low-Power mode, the pending move-in is not canceled. It remains on hold, waiting for Freeze and Low-Power mode to be exited and for the module to be unlocked. If a message buffer is unlocked during Freeze mode, the move-in occurs immediately.

The move-in process is FlexCAN executing the following steps:

1. Read all data words from the RX SMB in accordance with the selected payload size for the RX storage element.
2. Write all data words to the RX message buffer according to the selected payload size for the RX storage element. If the data size of the storage element is smaller than the original payload size described in the DLC field of the message, the payload is truncated. The high-order bytes that do not fit the destination size are lost.
3. Read the Control and Status and ID words from the RX SMB.
4. Write the Control and Status and ID words to the RX message buffer, and update the CODE field.

The move-in process is not atomic; the inactivation of the destination message buffer immediately cancels it (see Message buffer inactivation). In this case, the message buffer may remain partially updated, and therefore incoherent.

To alert the CPU that the message buffer content is temporarily incoherent, the BUSY Bit (least significant bit of the CODE field) of the destination message buffer becomes 1 during move-in.

52.3.7.2 Move-out

The move-out process is the copying of content from a TX message buffer to the TX SMB when a message for transmission is available (see Arbitration process). The move-out occurs in the following conditions:

- In the first bit of Intermission field
- During the Bus Off state, when TX Error Counter is in the 124 to 128 range
- During the Bus Idle state
- During the Wait For Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently outside the Bus Idle state. In Bus Idle, the move-out has the lowest priority of the concurrent memory accesses.

52.3.8 Data coherence

In order to maintain data coherency and proper FlexCAN operation, the CPU must obey the rules described in Transmission process and Receive process.

52.3.8.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abortion of a pending transmission. A feedback mechanism is provided to inform the CPU whether the transmission was aborted or the frame could not be aborted and was transmitted instead.

These primary conditions must be fulfilled in order to abort a transmission:

- MCR[AEN] must be 1.
- The first CPU action must be the writing of abort code (1001b) into the CODE field of the Control and Status word.

Active message buffers configured for transmission must be aborted before they can be updated. The write operation is blocked and the transmission is not disturbed when the abort code is written to:

- A message buffer currently being transmitted.

- A message buffer that was already loaded into the TX SMB for transmission.

In this case, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration.
- There is an error during the transmission.
- The module is put into Freeze mode.
- The module enters the Bus Off state.
- There is an overload frame.

If none of the conditions above are reached:

1. The message buffer is transmitted correctly.
2. The interrupt flag is set in the proper IFLAG register.
3. If enabled, an interrupt to the CPU is generated.

The abort request is automatically cleared when the interrupt flag is set. If only one of the above conditions is reached, the frame is not transmitted. In this case:

1. The abort code is written into the CODE field.
2. The interrupt flag is set in the proper IFLAG register.
3. Optionally, an interrupt is generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, the write operation is not blocked. The MB is updated and the interrupt flag is set. In this way, the CPU only needs to read the abort code to verify that the active MB was safely inactivated. In this case, although MCR[AEN] = 1 and the CPU wrote the abort code, the MB is inactivated and not aborted, because the transmission did not start yet. A message buffer is aborted only when the abort request is captured and kept pending until one of the previous conditions is satisfied.

52.3.8.2 Message buffer inactivation

Inactivation protects the message buffer against updates by FlexCAN internal processes. It allows the CPU to rely on message buffer data coherence after having updated it, even in Normal mode.

Inactivation of transmission message buffers must be performed only when MCR[AEN] = 0.

If a message buffer is inactivated, it does not participate in the arbitration process or the matching process until it is reactivated. See Transmission process and Receive process for detailed instructions on how to inactivate and reactivate a message buffer.

To inactivate a message buffer, the CPU must update its CODE field to INACTIVE (either 0b or 1000b).

Because you cannot synchronize the CODE field update with FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of an inactivated RX message buffer may be lost without notice. This loss can occur even if there are other message buffers with the same filter.
- A frame containing the message within an inactivated TX message buffer may be transmitted without setting the respective IFLAG flag.

To perform a safe inactivation and avoid the above consequences for TX message buffers, the CPU must use the transmission abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the message buffer (see Message buffer lock mechanism).

52.3.8.3 Message buffer lock mechanism

In addition to message buffer inactivation, FlexCAN uses a message buffer lock mechanism to maintain data coherence for the receive process. When the CPU reads the Control and Status word of an RX message buffer with codes FULL or OVERRUN,

FlexCAN is configured to allow the CPU to read the whole message buffer in an atomic operation. FlexCAN sets an internal lock flag for that message buffer.

The lock is released in any of the following events:

- The CPU reads the free-running timer (global unlock operation).
- The CPU reads the Control and Status word of another message buffer, regardless of its code.
- The CPU writes into the Control and Status word. This procedure is not recommended for normal unlocking, because it cancels a pending move and may lose a received message.

The message buffer lock prevents a new frame from being written into the message buffer when the CPU is reading it.

NOTE

The locking mechanism applies only to RX message buffers that have a code other than INACTIVE (0b) or EMPTY[1] (0100b). TX message buffers cannot be locked.

Consider an example where:

- The second and the fifth message buffers of the array are programmed with the same ID.
- FlexCAN has already received and stored messages into these two message buffers.
- The CPU reads message buffer number 5 while another message with the same ID is arriving.

When the CPU reads the Control and Status word of message buffer number 5, this message buffer is locked. The new message arrives and the matching algorithm finds no free-to-receive message buffers, so it overrides message buffer number 5. This message buffer is locked, so the new message cannot be written to it. The message remains in the RX SMB until the message buffer is unlocked, and only then is it written to the message buffer.

If the message buffer remains locked and another new message with the same ID arrives, the new message overwrites the one in the RX SMB. There is no indication of lost messages in the CODE field of the message buffer or in Error and Status 1 (ESR1).

When the message is moved from the RX SMB to the message buffer, the BUSY bit on the CODE field becomes 1. If the CPU reads the Control and Status word and identifies that the BUSY bit is 1, it must wait until the BUSY bit becomes 0 to access the MB.

NOTE

If the BUSY bit is 1 or the message buffer is empty, reading the Control and Status word does not lock the message buffer.

Inactivation takes precedence over locking. If the CPU inactivates a locked receive message buffer, then its lock status is negated and the message buffer is marked as invalid for the current matching round. Any pending message on the RX SMB is not transferred to the message buffer. A message buffer is unlocked when the CPU reads Free-Running Timer (TIMER) or the Control and Status word of another message buffer.

The lock and unlock mechanisms have the same functionality in Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. If unlocking occurs during a low-power mode, however, the move-in is postponed (see Modes of operation). Move-in takes place only when the module returns to Normal or Freeze mode.

52.3.9 Enhanced RX FIFO

FlexCAN supports an enhanced RX FIFO engine which can store up to 12 CAN FD messages. The region 2000h– 204Bh contains the output of the FIFO, which the CPU should read. To enable the enhanced RX FIFO, write 1 to ERFCR[ERFEN]. FlexCAN has two FIFO options, Legacy RX FIFO and Enhanced RX FIFO, but both options cannot be enabled at the same time. See Legacy RX FIFO for additional information.

To configure the enhanced RX FIFO watermark, write a value to ERFCR[ERFWM]. If ERFCR[ERFWM] is configured, the CPU is notified only if a minimum number of messages is stored in the FIFO. When the number of stored messages is greater than the

[1] In previous FlexCAN versions, reading the Control and Status word locked the message buffer even when CODE indicates it is EMPTY. This behavior is maintained when the IRMQ bit is 0.

value in ERFCR[ERFWM], the module sets ERFSR[ERFWMI]. Optionally, if MCR[DMA] or ERFIER[ERFWMIIE] are enabled, a DMA transfer or an interrupt can be triggered, respectively.

For the enhanced RX FIFO to receive, the CPU must execute the configuration procedure below. If the CPU must change any configurations of the Enhanced RX FIFO, the same procedure must be followed.

Prerequisites

MCR[RFEN] must be 0.

Procedure

Step	Purpose	Programming	Notes
1	Enter Freeze mode.	See Freeze mode.	—
2	If enhanced RX FIFO is not already enabled, enable it.	Write 1 to ERFCR[ERFEN].	—
3	Reset enhanced RX FIFO engine.	Write 1 to ERFSR[ERFCLR].	—
4	If the enhanced RX FIFO error flags are set, clear them.	Write 1 to these flags: • ERFSR[ERFUFW] • ERFSR[ERFOVF] • ERFSR[ERFWMI] • ERFSR[ERFDA]	—
5	Specify the total number of enhanced RX FIFO filter elements to be used in Enhanced RX FIFO reception.	Write the number to ERFCR[NFE].	—
6	Specify the number of extended ID and standard ID filter elements to be used in Enhanced RX FIFO reception.	Write the number to ERFCR[NEXIF].	ERFCR[NEXIF] ≤ ERFCR[NFE] + 1.
7	If you are using DMA, enable DMA.	Write 1 to MCR[DMA].	—
8	If you are using DMA, specify the number of words to transfer for each Enhanced RX FIFO data element.	Write the number to ERFCR[DMALW].	—
9	Specify the Enhanced RX FIFO watermark.	Write the number to ERFCR[ERFWM].	If MCR[DMA] = 1, ERFCR[ERFWM] should be 0h.
10	If you are using interrupts, enable the interrupts.	Write 1 to the interrupt enables in Enhanced RX FIFO Interrupt Enable (ERFIER).	—
11	Configure the filter elements.	Write to the ERFFEL*n* registers.	ERFFEL*n* registers are implemented in RAM; you must explicitly initialize them before prior any reception.
12	Exit Freeze mode.	See Freeze mode.	—

There are two types of enhanced RX FIFO filter elements that can be stored in ERFFEL*n* registers: extended-ID filter elements and standard-ID filter elements. Each extended-ID filter element is stored in two ERFFEL*n* registers, and each standard-ID filter element is stored in one ERFFEL*n* register. ERFCR[NFE] defines the total number of Enhanced RX FIFO filter elements.

In addition, the filter memory space can be split into two regions: one for extended-ID filter elements and another for standard-ID filter elements, according to ERFCR[NEXIF]. Figure 188 shows how the enhanced RX filter elements are defined. See Enhanced RX FIFO matching process for information about the Enhanced RX FIFO matching process and filter element formats.

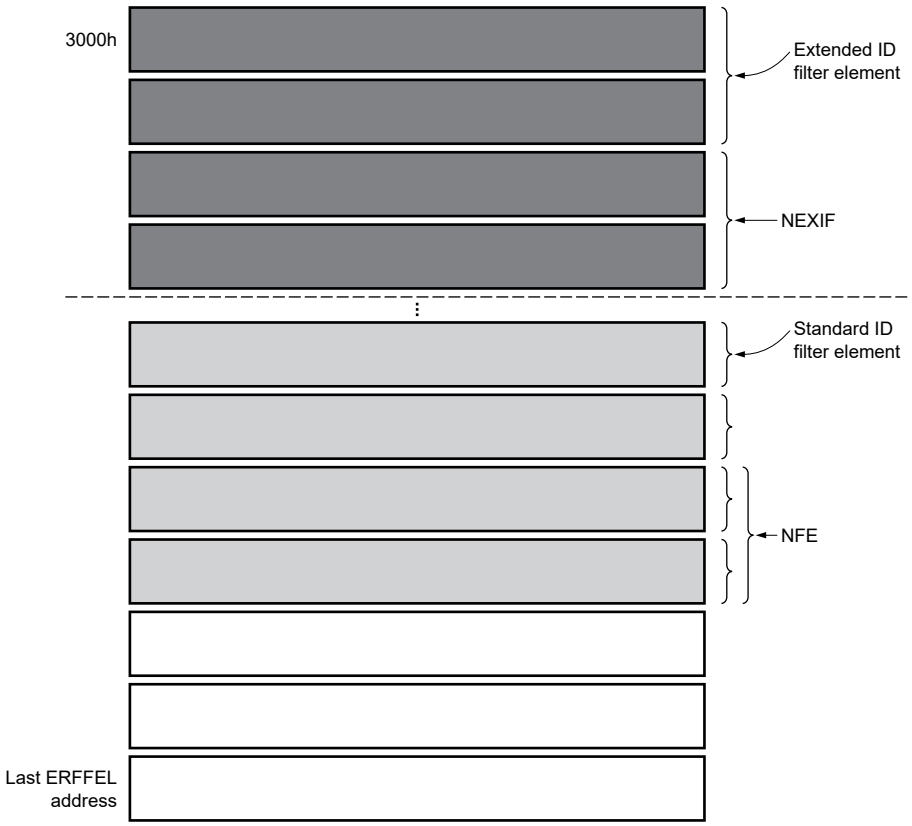


Figure 188. Number of enhanced RX FIFO filter elements

52.3.9.1 Enhanced RX FIFO matching process

When ERFCR[ERFEN] = 1, FlexCAN scans the ERFFEL*n* memory region. If at least one filter element satisfies the matching criteria, the CAN message content is transferred to the enhanced RX FIFO memory. If multiple filters match the incoming message ID, the first matching filter found by the matching process must be indicated in IDHIT.

Each ERFFEL*n* register can store one standard filter element. ERFFEL*n*[FEL][31:30], also called FSCH, determines the matching criteria in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base-frame format (IDE = 0).
  2. (ID[*n*] = STD ID filter [*n*]) or (STD ID Mask[*n*] = 0) for each bit *n* from 0 to 10.
  3. (RTR = RTR Filter) or (RTR MASK = 0).

In this explanation, RTR and ID are the Remote Transmit Request field and the ID from a CAN message, respectively.

If FSCH = b00, the filters and masks are defined as shown in Table 329.

Table 329. Standard ID filter element with filter and mask scheme (FSCH = b00)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b00		Reserved		RTR Filter	STD ID Filter		Reserved		RTR MASK	STD ID MASK	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0).
  2. ID ≥ STD ID Filter1.
  3. ID ≤ STD ID Filter2.
  4. (RTR = RTR filter) or (RTR MASK = 0).

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b01, the filters and mask are defined as shown in Table 330.

Table 330. Standard ID filter element with range scheme (FSCH = b01)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b01		Reserved		RTR Filter	STD ID Filter2		Reserved		RTR MASK	STD ID Filter1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0).
  2. (ID[*n*] = STD ID Filter1[*n*]) or (ID[*n*] = STD ID Filter2[*n*]) for each bit *n* from 0 to 10.
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2).

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b10, the filters are defined as shown in Table 331.

Table 331. Standard ID filter element with two-filter scheme (FSCH = b10)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH = b10		Reserved		RTR Filter 2	STD ID Filter2		Reserved		RTR Filter 1	STD ID Filter1	

Each pair of ERFFEL*n* registers can store one extended filter element. ERFFEL*n*[FSCH] determines the matching criteria in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. (ID[*n*] = EXT ID filter [*n*]) or (EXT ID Mask[*n*] = 0) for each bit *n* from 0 to 28.
  3. (RTR = RTR Filter) or (RTR MASK = 0).

If FSCH = b00, the filters and masks are defined as shown in Table 332.

Table 332. Extended ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	0
FSCH		RTR Filter	EXT ID Filter	
Reserved		RTR MASK	EXT ID MASK	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches an extended ID filter element only if the following criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. ID ≥ EXT ID Filter1.
  3. ID ≤ EXT ID Filter2.
  4. (RTR = RTR Filter) or (RTR MASK = 0).

If FSCH = b01, the filters and masks are defined as shown in Table 333.

Table 333. Extended ID filter element with range scheme (FSCH = b01)

31	30	29	28	0
FSCH		RTR Filter	EXT ID Filter2	
Reserved		RTR MASK	EXT ID Filter 1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1).
  2. (ID[*n*] = EXT ID Filter1[*n*]) or (ID[*n*] = EXT ID Filter2[*n*]) for each bit *n* from 0 to 28.
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2).

If FSCH = b10, the filters are defined as shown in Table 334.

Table 334. Extended ID filter element with two-filter scheme (FSCH = b10)

31	30	29	28	0
FSCH		RTR Filter2	EXT ID Filter2	
Reserved		RTR Filter1	EXT ID Filter 1	

52.3.9.2 Enhanced RX FIFO under DMA operation

You can enable the DMA feature by writing 1 to both ERFCR[ERFEN] and MCR[DMA]. The DMA controller can read the received message by reading a message buffer structure at the enhanced FIFO output port at the address range defined in Enhanced RX FIFO structure.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

For proper FIFO engine operation, the CPU should not access the Enhanced FIFO output port address range during DMA operation. Before writing 1 to MCR[DMA], the CPU must service Enhanced RX FIFO status bits. Otherwise, these bits may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before writing 0 to MCR[DMA], the CPU must first clear the ERFSR[ERFUFW], ERFSR[ERFOVF], ERFSR[ERFWMI], and ERFSR[ERFDA] flags. It must then clear the enhanced RX FIFO engine by writing one to ERFSR[ERFCLR].

When there is one frame available to be read from the Enhanced RX FIFO, FlexCAN sets ERFSR[ERFDA]. Upon receiving the request, the DMA controller can read the message in the Enhanced RX FIFO output. Each message reading process must end by the address defined in ERFCR[DMALW].

Follow these rules for Enhanced RX FIFO DMA operation:

- Because a DMA transfer cannot be changed dynamically, program ERFCR[DMALW] so the enhanced RX FIFO element can store the largest CAN message present on the CAN bus.
- Data bytes are valid according to the DLC field. See Table 357.

Each time the DMA controller reads one message from the FIFO, FlexCAN clears ERFSR[ERFDA]. If there is at least one message stored in the FIFO, FlexCAN sets it again.

Consider an example where the maximum number of bytes in the data field of a CAN frame for a certain application is eight. In that case, the last enhanced RX FIFO address offset can be found in Table 367 and Table 368. Using this address offset, ERFCR[DMALW] can be determined in this way:

- Maximum number of data bytes = 8
- Last address offset = TS_OFF = 2010h
- DMALW = 4

52.3.9.3 Enhanced RX FIFO clear operation

When ERFCR[ERFEN] is 1, the CPU can clear the Enhanced RX FIFO by writing 1 to ERFSR[ERFCLR]. The clear operation resets the internal FIFO pointers, but the FIFO content stored in RAM is not changed. This operation can only be performed in Freeze mode; the module blocks the operation in other modes. This operation does not clear ERFSR[ERFUFW], ERFSR[ERFOVF], ERFSR[ERFDA], or ERFSR[ERFWMI]. The CPU must service all these fields before executing the clear FIFO operation.

52.3.10 Legacy RX FIFO

The Legacy RX FIFO is receive-only. To enable it, write 1 to MCR[RFEN]. To maintain software backward compatibility with previous versions of FlexCAN that did not have the Legacy FIFO feature, the reset value of this field is zero.

CAUTION

Do not enable Legacy RX FIFO when the CAN FD feature is enabled.

The Legacy FIFO is six messages deep. The memory region the Legacy FIFO structure occupies (both message buffers and Legacy FIFO engine) is described in Legacy RX FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a message buffer structure at the output of the Legacy FIFO.

IFLAG1[BUF5I] (Frames Available in Legacy RX FIFO) is set when at least one frame is available to be read from the Legacy FIFO. If the corresponding mask bit enables it, an interrupt is generated. Upon receiving the interrupt, the CPU can read the message (accessing the output of the Legacy FIFO as a message buffer) and Legacy RX FIFO Information (RXFIR), then clear the interrupt. If there are more messages in the Legacy FIFO, clearing the interrupt:

1. Updates the output of the Legacy FIFO with the next message.
2. Updates RXFIR with the attributes of that message.
3. Reissues the interrupt to the CPU.

Otherwise, the flag remains cleared. The output of the Legacy FIFO is valid only when IFLAG1[BUF5I] is set.

IFLAG1[BUF6I] (Legacy RX FIFO Warning) is set when the Legacy RX FIFO receives a new message that increases the number of unread messages from four to five. This change means that the Legacy RX FIFO is almost full. The flag remains set until the CPU clears it.

IFLAG1[BUF7I] (Legacy RX FIFO Overflow) is set when an incoming message is lost because the Legacy RX FIFO is full. The flag is not set when the Legacy RX FIFO is full and a message buffer captures the message. The flag remains set until the CPU clears it.

Clearing one of the three flags above does not affect the state of the other two.

If an IFLAG flag is set and the corresponding mask bit is 1, an interrupt is generated.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing workload. The filtering criteria are specified by programming a table of up to 128 32-bit registers, according to CTRL2[RFFN]. This table can be configured to one of the following formats (see also Legacy RX FIFO structure):

- Format A: 128 Identifier Acceptance Filters (IDAFs) — extended or standard IDs including IDE and RTR
- Format B: 256 IDAFs — standard IDs or extended 14-bit ID slices including IDE and RTR
- Format C: 512 IDAFs — standard or extended 8-bit ID slices

NOTE

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the Legacy RX FIFO has a corresponding Identifier Acceptance Filter Hit Indicator (IDHIT). The IDHIT can be read in the IDHIT field in the Control and Status word, as shown in the Legacy RX FIFO Structure description. The CPU can also obtain this information by accessing Legacy RX FIFO Information (RXFIR). RXFIR[IDHIT] refers to the message at the output of the Legacy FIFO, and is valid when IFLAG1[BUF5I] is set. RXFIR must be read only before clearing the flag, guaranteeing that the information refers to the correct frame within the Legacy FIFO.

The Individual Mask Registers (RXIMR*n*) individually affect up to 32 elements of the filter table, according to the value of CTRL2[RFFN]. This configuration allows very powerful filtering criteria to be defined. If MCR[IRMQ] is 0, the Legacy RX FIFO filter table is affected by Legacy RX FIFO Global Mask (RXFGMASK).

NOTE

See Table 321 for information about the difference between FD and non-FD regarding this feature.

52.3.10.1 Legacy RX FIFO in DMA Operation

The receive-only Legacy FIFO can support DMA. To enable this feature, write 1 to both MCR[RFEN] and MCR[DMA]. To maintain backward compatibility with previous versions of the module that did not have the DMA feature, the reset value of MCR[DMA] is zero.

The DMA controller can read the received message by reading a message buffer structure at the Legacy FIFO output port in the 80h–8Ch address range.

NOTE

FlexCAN supports 32-bit access only for DMA transfers.

When MCR[DMA] = 1, the CPU must not access the Legacy FIFO output port address range. Before writing 1 to MCR[DMA], the CPU must service the IFLAG flags set in the Legacy RX FIFO region. Otherwise, these flags may indicate that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before writing 0 to MCR[DMA], the CPU must perform a clear Legacy FIFO operation.

When at least one frame available to be read from the FIFO, IFLAG1[BUF5I] (Frames available in Legacy RX FIFO) is set. A DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the Legacy FIFO as a message buffer). The DMA reading process must end by reading address 8Ch. This read operation:

- Clears IFLAG1[BUF5I].
- Updates the FIFO output with the next message (if the FIFO is not empty).
- Updates Legacy RX FIFO Information (RXFIR) with the attributes of the new message.

If there are more messages stored in the FIFO, IFLAG1[BUF5I] is reasserted and another DMA request is issued. Otherwise, the flag remains cleared.

IFLAG1[BUF6I] and IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Legacy RX FIFO interruption and must not clear the related IFLAG flags. The related IMASK bits are not used to mask the generation of DMA requests.

NOTE

See Table 321 for information about the difference between FD and non-FD regarding this feature.

52.3.10.2 Clear Legacy FIFO

When MCR[RFEN] = 1, you can use the clear Legacy FIFO operation to empty Legacy FIFO contents. When the CPU writes 1 to IFLAG1[BUF0I], the clear FIFO operation occurs. This operation can only be performed in Freeze mode; FlexCAN blocks it in other modes. This operation does not clear the FIFO IFLAG flags; the CPU must service all FIFO IFLAG flags before executing the clear FIFO operation.

When Legacy RX FIFO is working with DMA, the clear FIFO operation clears IFLAG1[BUF5I], and the DMA request is canceled.

CAUTION

The clear Legacy FIFO operation does not clear IFLAG flags, except when MCR[DMA] = 1; in this case, only IFLAG1[BUF5I] is cleared.

52.3.11 Fault reaction

This situation is catastrophic, and in response to the fault reaction, FlexCAN goes into a safe state (Freeze) and drives a safe value (recessive bit) on the bus.Either CTRL2[FLT_RXN] or an input "fault reaction" signal from another module can trigger a fault reaction. FlexCAN remains in a safe state as long as CTRL2[FLT_RXN] = 1 or the "fault reaction" signal is asserted and MCR[FRZ] = 1.

A fault can occur at any time, including during frame transmission, reception, and BUS IDLE. If a fault occurs during frame transmission, FlexCAN stops the frame transmission and goes into a safe state.

If a fault occurs during frame reception, FlexCAN stops receiving the frame and goes into a safe state. If FlexCAN is in loopback or self-reception mode, the effect of the fault reaction is the same as described above.

When the fault condition is removed, FlexCAN remains in Freeze mode until the CPU writes 0 to MCR[FRZ].

52.3.12 Retransmission

If a message transmission is unsuccessful due to a reason other than a loss of arbitration or being aborted by the host, FlexCAN attempts to retransmit the message a number of times determined by the value of CTRL2[RETRY].

When a message transmission is unsuccessful, a counter is decremented. When that counter reaches zero, FlexCAN aborts the message. FlexCAN sets the IFLAG for the aborted message and an interrupt is generated (if enabled). FlexCAN sets the IFLAG and updates the CODE field of the control and status word by abort code. The host can read the IFLAG flag and CODE field to identify which message FlexCAN aborted due to retransmission attempts greater than the value of CTRL2[RETRY].

52.3.13 CAN protocol-related features

52.3.13.1 CAN FD ISO compliance

The CAN FD protocol has been improved to increase the failure-detection capability that was in the original CAN FD protocol. This original protocol is also called non-ISO CAN FD, by CAN in Automation (CiA). A three-bit stuff counter and a parity bit have been introduced in the improved CAN FD protocol, now called ISO CAN FD. The CRC calculation has also been modified. All these improvements make the ISO CAN FD protocol incompatible with the non-ISO CAN FD protocol. FlexCAN still supports non-ISO CAN FD, so it can be used during an intermediate phase, for evaluation and development purposes.

It is recommended that you configure FlexCAN with the ISO CAN FD protocol by writing 1 to CTRL2[ISOCANFDEN].

52.3.13.2 CAN FD frames

ISO 11898-1:2015 specifies the Classical Frame format compliant to ISO 11898-1:2003 (2003) and introduces the CAN Flexible Data Rate Frame format (CAN FD). The Classical Frame format allows bit rates up to one Mbit/s and payloads up to eight bytes per frame. The Flexible Data Rate Frame format allows bit rates faster than one Mbit/s and payloads longer than eight bytes per frame. FlexCAN can receive and transmit CAN FD messages interleaved with Classical CAN messages.

There are additional control bits in the CAN FD frame:

- The Extended Data Length (EDL) bit enables a longer data payload with different data length coding.
- The Bit Rate Switch (BRS) bit decides whether the bit rate is switched inside a CAN FD format frame.
- The Error State Indicator (ESI) flag is transmitted dominant by error active nodes, and recessive by error passive nodes.

There are no Remote Frames (see Remote frames) in the CAN FD format. A message configured to transmit a Remote Frame is always sent out in Classical CAN format. When an FD frame is received and matches a message buffer, the RTR bit in the receiving message buffer becomes 0. The RTR bit must be considered in classical frames only.

52.3.13.2.1 CAN FD messages

CAN FD messages may be formatted as long frames where the data field exceeds eight bytes, and may range from 12 up to 64 bytes. They can also be configured to support bit rate switching. In this case, the control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and end of the frame. Messages in Classical CAN format are limited to transport a maximum payload of eight bytes at nominal rate. Figure 189 illustrates the message formats for Classical and FD frames with either standard or extended ID.

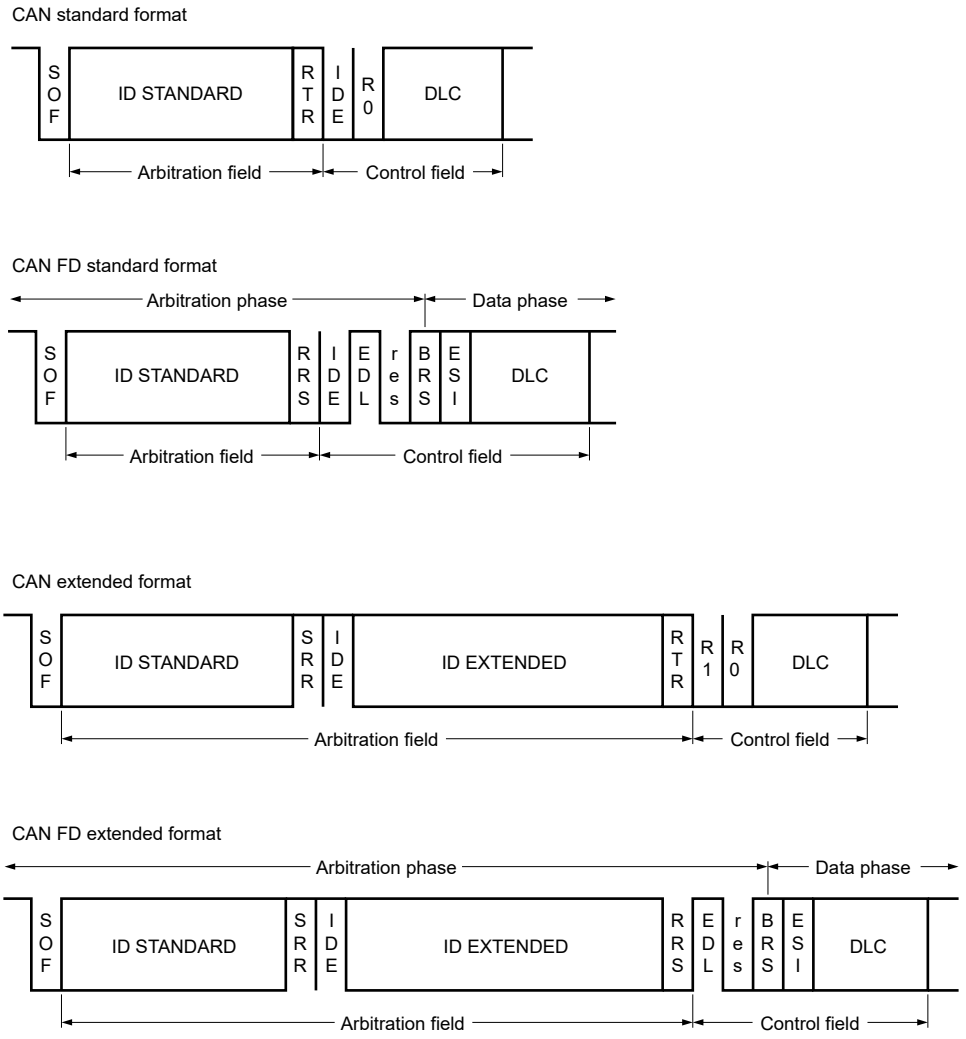


Figure 189. CAN message formats

MCR[FDEN] enables the ability to receive and transmit CAN FD messages. A recessive R0 bit in CAN frames with 11-bit identifiers, or a recessive R1 bit in CAN frames with 29-bit identifiers, is decoded as an EDL bit (not a reserved one). A recessive EDL bit identifies a CAN FD frame, and a dominant EDL bit identifies a Classical CAN frame. The BRS bit specifies whether this frame switches the bit rate in its data phase. A long frame is decoded according to the DLC field value (see DLC definition in Message buffer structure).

CAN FD messages can be transmitted with two different bit rates. The first part of a CAN FD frame, from the Start of Frame (SOF) bit until the Bit Rate Switch (BRS) bit is called the arbitration phase. This part is transmitted with the nominal bit rate based on a set of nominal CAN bit timing configuration values. The second part, from the BRS bit until the CRC Delimiter bit, is called the data phase. When this second part is transmitted, a second set of CAN data bit timing configuration values determines the data bit rate. Finally, from the CRC delimiter until the Intermission bits, the transmission returns to nominal bit rate.

52.3.13.2.2 BRS in CAN FD

In CAN FD frames with bit rate switching, the bit timing changes inside the frame at the sample point of the BRS bit if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by CAN Bit Timing (CBT). (Control 1 (CTRL1) also defines this timing for backward compatibility.) Upon detecting a recessive BRS bit, the CAN data bit timing is used as defined by CAN FD Bit Timing (FDCBT).

NOTE

If the time quantum length in nominal bit timing and in the data bit timing are not identical, a quantization error of up to one time quantum of the arbitration phase may be present as a phase error. This situation can occur after the switch from arbitration to data phase, and it lasts until the next synchronization event. The length of the time quantum should be the same in nominal and data bit timing. This configuration minimizes the chance of error frames on the CAN bus, and optimizes the clock tolerance in networks that use FD frames.

If BRS = 1 in the selected TX MB, FDCTRL[FDRATE] enables the transmission of all frames with bit rate switching. If FDCTRL[FDRATE] = 0, the transmission is performed at nominal rate regardless of the BRS bit value. FDCTRL[FDRATE] can be written at any time but takes effect only for the next message transmitted or received.

Nominal bit timing is resumed at the sample point of the CRC Delimiter bit or when an error is detected, whichever occurs first. Figure 190 describes the mechanism for entering and leaving the data phase when the BRS bit is recessive.

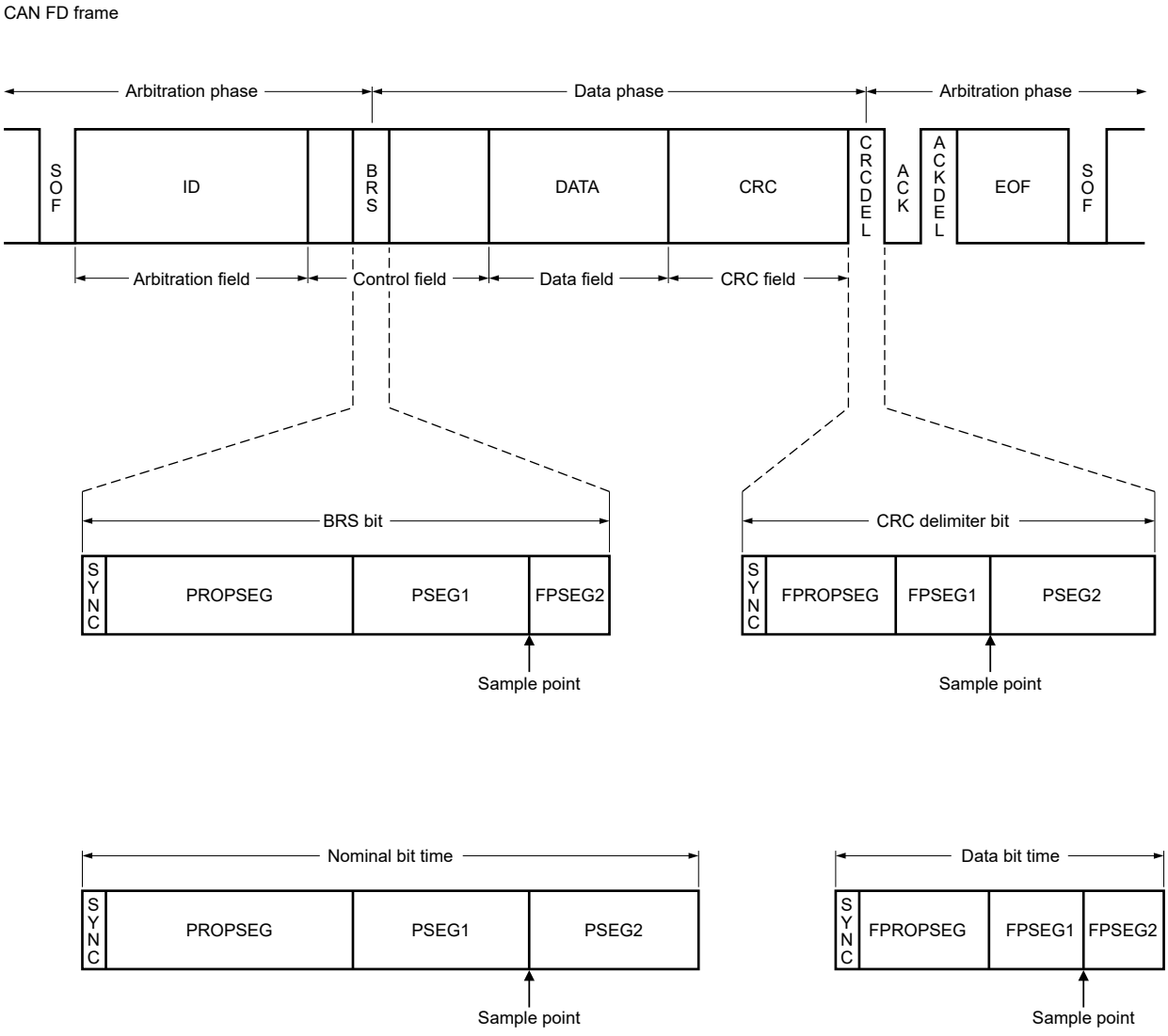


Figure 190. Bit rate switching mechanism for CAN FD messages

NOTE

In Classical CAN frames, the CRC delimiter is one recessive bit. In CAN FD frames, the CRC delimiter may consist of one or two recessive bits. FlexCAN sends only one recessive bit as the CRC delimiter. It accepts two recessive bits before the recessive-to-dominant edge that starts the acknowledge slot. As a receiver, FlexCAN sends its acknowledge bit after the first CRC delimiter bit. In CAN FD frames, FlexCAN accepts a two-bit dominant ACK slot as a valid ACK to compensate for phase shifts between the receivers.

The maximum configurable bit rate in the CAN FD data phase depends on the clock frequency of the CAN_PE subblock. For example, for a CAN_PE clock frequency of 40 MHz and the shortest configurable bit time of 5 time quanta, the bit rate in the data phase is 8 Mbit/s.

NOTE

The frequency used in this example may not be supported on this chip. It is shown only to demonstrate how the maximum configurable bit rate is calculated.

52.3.13.2.3 ESI in CAN FD

The value of the ESI bit is determined:

- By the error state of the transmitter at the start of the transmission, if the frame is originated in the FlexCAN node,
- Or by the original transmitting node when FlexCAN is acting as a gateway for the message.

If the transmitter is error-passive, ESI is transmitted recessive; otherwise, it is transmitted dominant. The permutations of the relationship between the written value and the transmitted value of the ESI are shown in Table 335.

Table 335. Written versus transmitted values of ESI field

FlexCAN fault confinement status at start of frame	ESI bit of TX MB	Transmitted ESI
Error active	0	0 (Error Active)
Error passive	0	1 (Error Passive)
Error active	1	1 (Error Passive)
Error passive	1	1 (Error Passive)

52.3.13.2.4 CRC calculations in CAN FD

Different CAN frame formats have different CRC polynomials. The first polynomial, CRC_15, is used for all frames in Classical CAN format. The second, CRC_17, is used for frames in CAN FD format with a data field up to 16 bytes long. The third, CRC_21, is used for frames in CAN FD format with a data field longer than 16 bytes. Each polynomial results in a Hamming distance of 6. At the start of the frame, all three CRC polynomials are calculated concurrently. The values of the EDL bit and the DLC field select the CRC sequence to be transmitted. When receiving a message, FlexCAN decodes EDL and DLC to select the adequate CRC polynomial to check for a CRC error.

In CAN FD format frames, stuff bits are included in the bit stream for CRC calculation. In Classical CAN format frames, stuff bits are not included. After the transmission of the last bit relevant to the CRC calculation, CAN FD CRC (FDCRC) stores the calculated CRC for the transmitted message. This storage is performed with adequate length for the type of message, for CAN FD and non-FD messages. Cyclic Redundancy Check (CRCR) reports a valid CRC for Classical CAN messages only.

In CAN FD format frames, the CAN bit stuffing method changes for the CRC sequence, so the stuff bits are inserted at fixed positions. When FlexCAN is transmitting a CAN FD frame, a fixed stuff bit is inserted just before the first bit of the CRC sequence. This insertion occurs even if the last bits of the preceding field do not fulfill the CAN stuff condition. Additional stuff bits are inserted after each fourth bit of the CRC sequence. The value of any fixed stuff bit is the inverse value of its preceding bit. When FlexCAN receives a CAN FD frame, it discards the fixed stuff bits from the bit stream for the CRC check. A stuff error is detected if the fixed stuff bit has the same value as its preceding bit.

52.3.13.2.5 CAN FD errors

FlexCAN detects errors in CAN FD frames the same way as in Classical CAN frames. The error counters ECR[RXERRCNT] and ECR[TXERRCNT] accumulate the counts of RX and TX errors, respectively, for both FD and non-FD frames indiscriminately. Two extra error counters, ECR[RXERRCNT_FAST] and ECR[TXERRCNT_FAST], accumulate RX and TX errors occurring in the data phase of CAN FD frames with BRS = 1 only. The rules for updating the error counters are the same for both CAN FD and non-FD frames (see Error Counter (ECR)).

These error flags report errors in both CAN FD and non-FD frames:

- ESR1[BIT1ERR]
- ESR1[BIT0ERR]
- ESR1[ACKERR]
- ESR1[CRCERR]
- ESR1[FRMERR]
- ESR1[STFERR]

If CTRL1[ERRMSK] = 1, they also generate the ERRINT interrupt.

These additional error flags indicate the occurrence of errors in the data phase of CAN FD frames with BRS = 1:

- ESR1[BIT1ERR_FAST]
- ESR1[BIT0ERR_FAST]
- ESR1[CRCERR_FAST]
- ESR1[FRMERR_FAST]
- ESR1[STFERR_FAST]

No ACKERR is detected in the data phase of a CAN FD frame. Fault confinement status reported in ESR1[FLTCONF] is the same for both CAN FD and Classical CAN frames, and is based on ECR[RXERRCNT] and ECR[TXERRCNT] only. Information in ECR[RXERRCNT_FAST] and ECR[TXERRCNT_FAST] may be considered as status to help detect the error nature related to the bit rate value.

When FlexCAN detects an error while transmitting or receiving a CAN FD message in the data phase, it immediately switches:

- Back to the arbitration phase, and
- Back to the nominal rate to start an error flag.

52.3.13.2.6 CAN FD synchronization

Resynchronization and hard synchronization occur in CAN FD frames in the same way as in Classical CAN ones. A hard synchronization is also performed at the recessive-to-dominant edge from EDL to R0 in CAN FD format frames. FlexCAN does not resynchronize when transmitting in the CAN FD data phase.

52.3.13.3 Transceiver delay compensation

The CAN FD protocol allows the transmission and reception of data at a higher bit rate than the nominal rate used in the arbitration phase, when BRS = 1 in the message. This feature enables the use of rates up to 8 Mbit/s.

During the data phase of a CAN FD frame, if the transmitter cannot receive its own latest transmitted bit at the sample point of that bit, it detects a bit error. When bit rate switching is enabled (BRS = 1), the CAN bit time in the data phase can become shorter than the loop delay of the transceiver. This condition impedes the correct comparison between the transmitted bit and the received bit within the current CAN bit time interval.

The transceiver delay compensation (TDC) process defines a secondary sample point where the transmitted bit is correctly compared to the received bit to check for bit errors.

You can enable the TDC mechanism via FDCTRL[TDCEN] or ETDC[ETDCEN]. The TDC mechanism is effective only during the data phase of FD frames with BRS = 1. It has no effect on either non-FD frames or FD frames transmitted at the normal bit rate. When the transmitted message has BRS = 1, TDC is active from the sample point of the BRS bit until the sample point of the CRC Delimiter bit. When TDC is active, the real received bit is compared to the delayed transmitted bit, where the delay is calculated based on the measured transceiver loop delay.

NOTE

The transmitters using TDC disregard the value of the CRC delimiter bit. A global error at the end of the CRC field causes the receivers to send error frames that the transmitter detects during Acknowledge or End of Frame.

For every transmitted FD frame with BRS = 1, the transition from the recessive EDL bit to the dominant R0 bit triggers the delay measurement (as shown in Figure 191). The loop delay is measured in Protocol Engine (PE) clock periods (CANCLK, see Protocol timing), from the transmitted EDL-R0 edge to the received EDL-R0 edge. The measured loop delay time added to an offset value specified in FDCTRL[TDCOFF] or ETDC[ETDCOFF] determines the position of the secondary sample point. FDCTRL[TDCVAL] or ETDC[ETDCVAL] stores the result of this calculation. The TDCVAL and ETDCVAL value saturates at its maximum value of 63 CANCLK and 255 CANCLK when the delay measurement is too long.

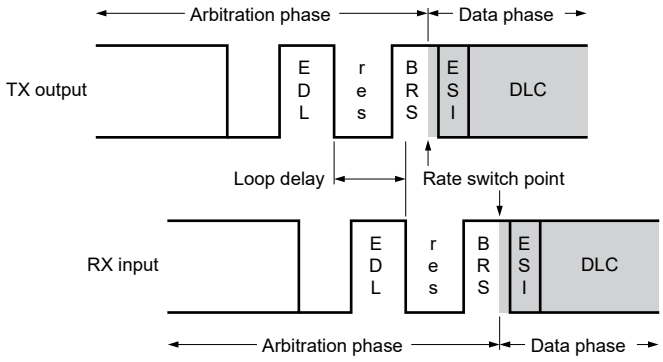


Figure 191. Transceiver loop delay measurement

The measured loop delay is not enough to define the secondary sample point, because it relates to the CAN bit edges. The transceiver delay compensation offset FDCTRL[TDCOFF] or ETDC[ETDCOFF] is used to shift the secondary sample point to an intermediate point inside the bit time, far away from its edges. The value of FDCTRL[TDCOFF] or ETDC[ETDCOFF] cannot be larger than the CAN bit duration in the data phase.

If the secondary sample point is set very near the CAN bit edge (SYNC field), problems may occur during the bit sampling in the data phase. For the TDC to work reliably, the offset must use optimal settings. To ensure that bit sampling is performed in the best region, configure the TDC offset as shown in this equation:

$$Offset = (FPSEG1 + FPROPSEG + 2) \times (FPRESDIV + 1)$$
or
$$Offset = (DTSEG1 + 2) \times (EDPRESDIV + 1), \textit{ if ETDCEN}$$

Equation 1. TDC offset calculation

Figure 192 shows the SSP position when these settings are used.

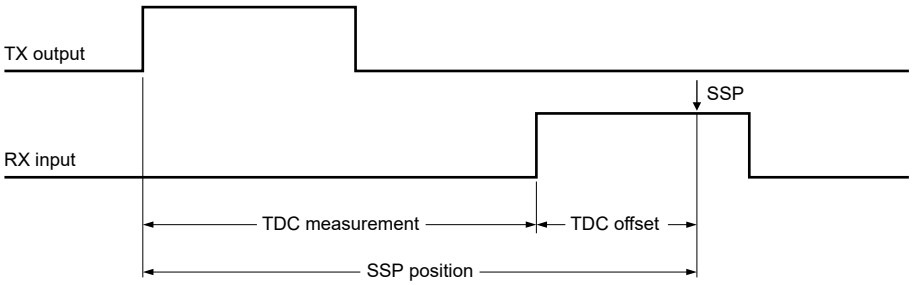


Figure 192. SSP position with optimal values

Alternatively, if CTRL2[BTE] and ETDC[ETDCEN] are 1, you can write 1 to ETDC[TDMDIS] to disable the transceiver delay measurement. In this case, only ETDC[ETDCOFF] defines the SSP position. Figure 193 shows the secondary sample point position when the transceiver delay measurement is disabled.

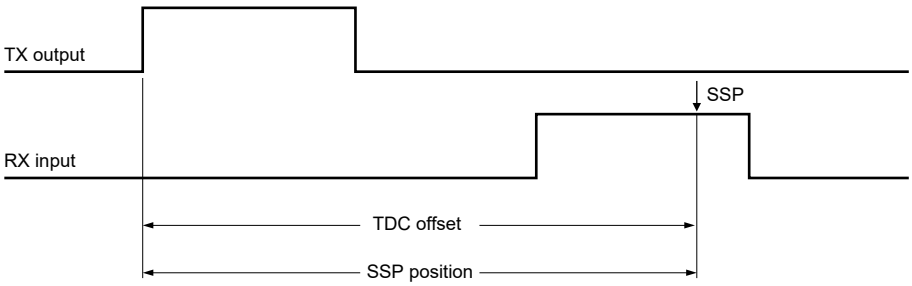


Figure 193. SSP position with TDC measurement disabled by ETDCEN and TDMDIS

During the data phase of CAN FD frames with bit rate switching enabled, at the onset of every TX CAN bit:

- The transmitted TX bit value is temporarily stored in a buffer.
- A time countdown based on FDCTRL[TDCVAL] or ETDC[ETDCVAL] is started. This countdown ends with the comparison of the received RX bit (delayed by the external loop delay plus the specified offset) to the stored TX bit.

If a bit error is detected at the secondary sample point, FlexCAN issues an error flag to the CAN bus at the next sample point.

During the arbitration phase, delay compensation is always disabled. During the data phase, the TDC mechanism of FlexCAN can compensate a maximum delay of 3 CAN bit times − 2 Tq. Beyond this limit, the FDCTRL[TDCFAIL] or ETDC[ETDCFAIL] flag is set. The flag indicates when the TDC mechanism is out of range and is unable to compensate the transceiver loop delay.

52.3.13.4 Remote frames

A remote frame is a special type of frame. You can program a message buffer to be a remote request frame by configuring the message buffer as Transmit with the RTR = 1. After the remote request frame is transmitted successfully, the message buffer becomes a receive message buffer, with the same ID as before.

When FlexCAN receives a remote request frame, the frame can be treated in different ways, depending on remote request storing (CTRL2[RRS]) and RX FIFO Enable (MCR[RFEN]):

- If RRS = 0, the ID of the frame is compared to the IDs of the transmit message buffers with the CODE field 1010b. If a matching ID exists, this message buffer frame is transmitted. If the matching message buffer has the RTR = 1, FlexCAN transmits a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response.

  The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. If a remote request frame is received and matches a message buffer, this message buffer immediately enters the internal arbitration process. However, it is considered a normal TX message buffer, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.

- If CTRL2[RRS] = 1, the ID of the frame is compared to the IDs of the receive message buffers with the CODE field 0100b, 0010b, or 0110b. If a matching ID exists, this message buffer stores the remote frame in the same fashion as a data frame. No automatic remote response frame is generated. The mask registers are used in the matching process.
- If MCR[RFEN] = 1, FlexCAN does not generate an automatic response for remote request frames that match the Legacy FIFO filtering criteria. If the remote frame matches one of the target IDs, it is stored in the Legacy FIFO and presented to the CPU.

  For filtering formats A and B (see Legacy RX FIFO structure), it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted if they match the ID. Remote request frames are considered as normal frames. They generate a Legacy FIFO overflow when a successful reception occurs and the Legacy FIFO is already full.
- If ERFCR[ERFEN] = 1, FlexCAN does not generate an automatic response for remote request frames that match the Enhanced RX FIFO filtering criteria. Remote Request Frames are considered normal frames. They generate an Enhanced RX FIFO overflow when a successful reception occurs and the enhanced RX FIFO is already full.

NOTE

There is no remote frame in the CAN FD format. A fixed dominant RRS bit replaces the RTR bit. FlexCAN receives and transmits remote frames in the Classical CAN format.

52.3.13.5 Overload frames

When a dominant bit is detected on the CAN bus in these locations, FlexCAN transmits overload frames:

- The first or second bit of Intermission.
- The seventh bit (last) of End of Frame field (RX frames).
- The eighth bit (last) of Error Frame Delimiter or Overload Frame Delimiter.

52.3.13.6 Timestamp

The value of the free-running timer is sampled at the beginning of the Identifier field on the CAN bus. This value is stored at the end of move-in in the TIME_STAMP field, providing network behavior regarding time.

The FlexCAN bit clock clocks the free-running timer, which defines the baud rate on the CAN bus. During a message transmission or reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The free-running timer is not incremented during Disable, Doze, Stop, and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See CTRL1[TSYN].

52.3.13.7 Protocol timing

Figure 194 shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule.

NOTE

To identify the proper clock source, see the clock distribution chapter (module clocks table).

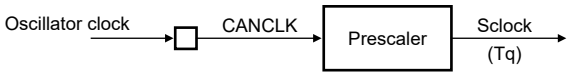


Figure 194. CAN engine clocking scheme

52.3.13.7.1 Bit timing configuration

FlexCAN supports various means to configure bit timing parameters required by the CAN protocol. Control 1 (CTRL1) has various fields to control bit timing parameters:

- CTRL1[PRESDIV]

- CTRL1[PROPSEG]
- CTRL1[PSEG1]
- CTRL1[PSEG2]
- CTRL1[RJW]

CAN Bit Timing (CBT) extends the range of the CAN bit timing variables in CTRL1. Enhanced Data Phase CAN Bit Timing (EDCBT) provides a second set of CAN bit timing variables to be applied at the data phase of CAN FD frames with the Bit Rate Switch (BRS) = 1.

Enhanced Nominal CAN Bit Timing (ENCBT) extends the range of CAN bit timing variables in CBT. Enhanced Nominal CAN Bit Timing (ENCBT) extends the range of CAN bit timing variables in FDCBT. When using ENCBT and EDCBT, you must program the nominal bit timing and data phase serial clock (Sclock) dividers in Enhanced CAN Bit Timing Prescalers (EPRS).

NOTE

When the CAN FD feature is enabled, always write 1 to CBT[BTF] or CTRL2[BTE] and specify the CAN bit timing variables in CBT or ENCBT. See CAN Bit Timing (CBT) or Enhanced Nominal CAN Bit Timing (ENCBT).

CTRL1[PRESDIV], and its extended range CBT[EPRESDIV] (or EPRS[ENPRESDIV]) and FDCBT[FPRESDIV] (or EPRS[EDPRESDIV]) for the data phase bits of CAN FD messages, defines the prescaler value that generates the serial clock (Sclock). (See Equation 2.) The period of Sclock defines the time quantum used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time managed by the CAN engine. It is the smallest time unit for all configuration values.

$$Tq = \frac{(PRESDIV + 1)}{f_{CANCLK}}$$

Equation 2. Time quantum

The bit rate, which defines the rate the CAN message is received or transmitted, is calculated with the formula:

$$CAN\ bit\ time = (Number\ of\ time\ quanta\ in\ 1\ bit\ time) \times Tq$$

$$Bit\ rate = \frac{1}{CAN\ bit\ time}$$

Equation 3. CAN bit time and baud rate

52.3.13.7.2 Bit time segments

A bit time is subdivided into three segments as shown in Figure 195. See also Figure 196 , Figure 197, and Table 336.

NOTE

For further explanation of the underlying concepts, see ISO 11898-1:2015. See also *CAN Specification Version 2.0, Part A and Part B* for bit timing.

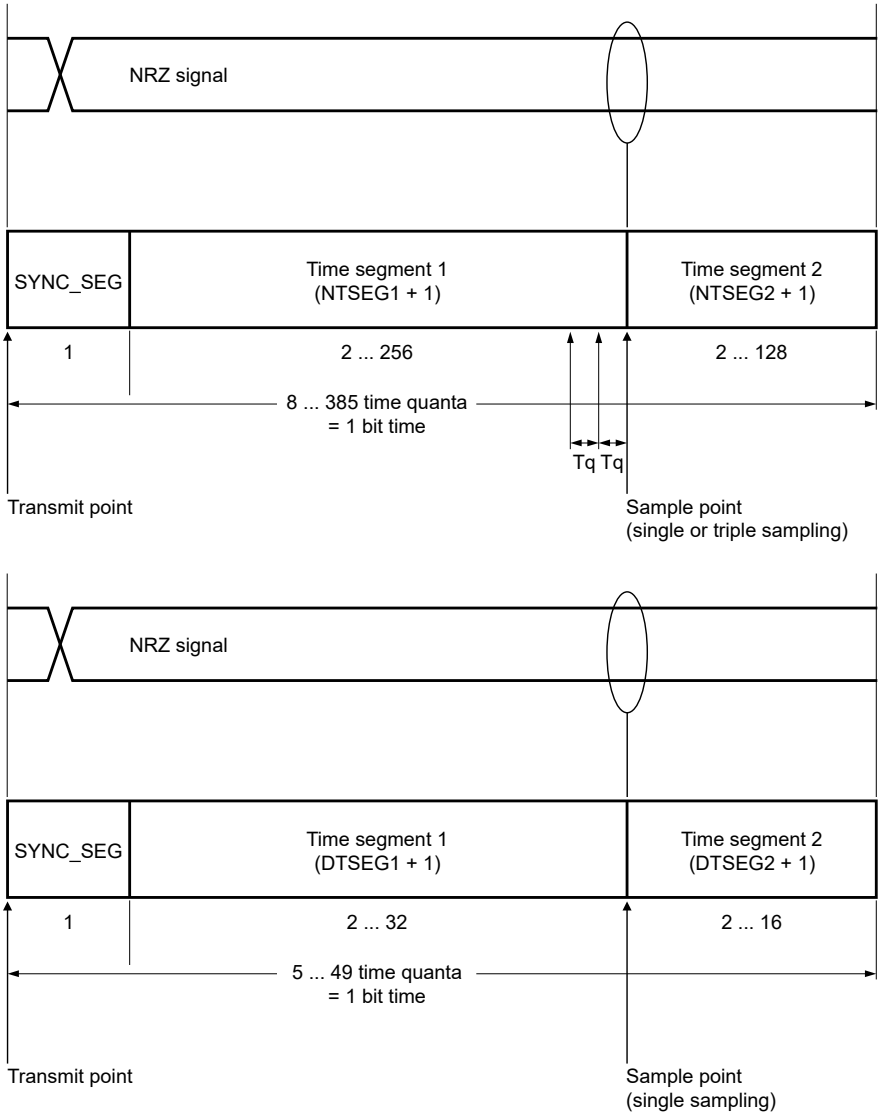


Figure 195. Segments within the bit timing (example using ENCBT and EDCBT bit timing variables)

The three bit time segments are:

- SYNC_SEG—this segment has a fixed length of one time quantum. Signal edges are expected to occur within this section.
- Time Segment 1—this segment includes the propagation segment and the phase segment 1 of the CAN standard.

  It can be programmed by configuring CTRL1[PROPSEG] and CTRL1[PSEG1] so that the sum (plus 2) is 2–16 time quanta. When CBT[BTF] = 1, FlexCAN uses CBT[EPROPSEG] and CBT[EPSEG1] so that the sum (plus 2) is 2–96 time quanta. For messages in CAN FD format with the BRS = 1, FlexCAN uses FDCBT[FPROPSEG] and FDCBT[FPSEG1] so that the sum (plus 1) is 2–39 time quanta.

  If CTRL2[BTE] = 1, FlexCAN uses ENCBT[NTSEG1] to configure time segment 1 to 2–256 time quanta. For the data phase in CAN FD messages with BRS = 1, EDCBT[DTSEG1] must be used for configuring time segment 1 to 2–32 time quanta.
- Time Segment 2—this segment represents the phase segment 2 of the CAN standard.

  It can be programmed by configuring CTRL1[PSEG2] (plus 1) to be 2–8 time quanta. When CBT[BTF] = 1, FlexCAN uses CBT[EPSEG2] so that its value (plus 1) is 2–32 time quanta. For messages in CAN FD format with the BRS = 1, FlexCAN uses

FDCBT[FPSEG2] instead, so that its value (plus 1) is 2–8 time quanta. Time segment 2 cannot be smaller than the Information Processing Time (IPT), which is 2 time quanta in FlexCAN.

If CTRL2[BTE] = 1, FlexCAN uses ENCBT[NTSEG2] to configure time segment 2 to 2–128 time quanta. For the data phase in CAN FD messages with BRS = 1, EDCBT[DTSEG2] must configure time segment 2 to 2–16 time quanta.

NOTE

The bit time defined by the above time segments must not be smaller than five time quanta. For bit time calculations, use an Information Processing Time (IPT) of two, which is the value implemented in the FlexCAN module.

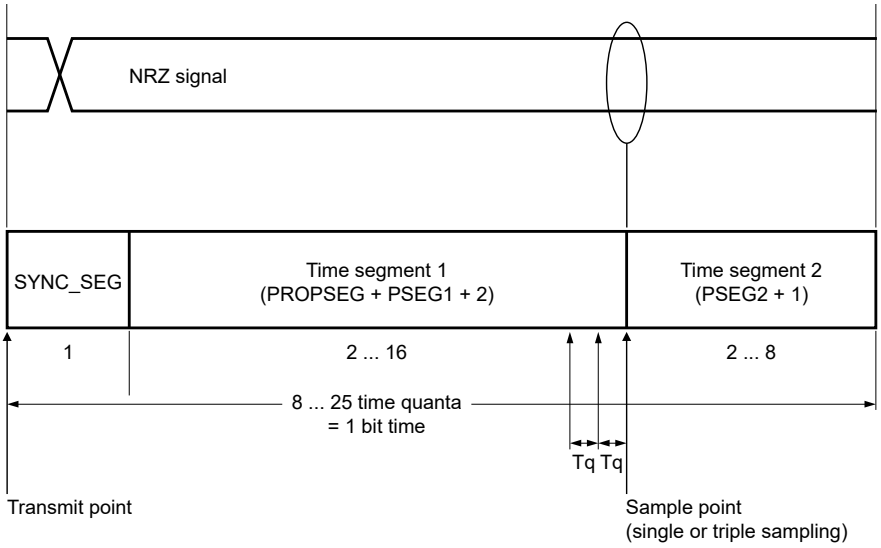


Figure 196. Segments within the bit time (example using CTRL1 bit timing variables for Classical CAN format)

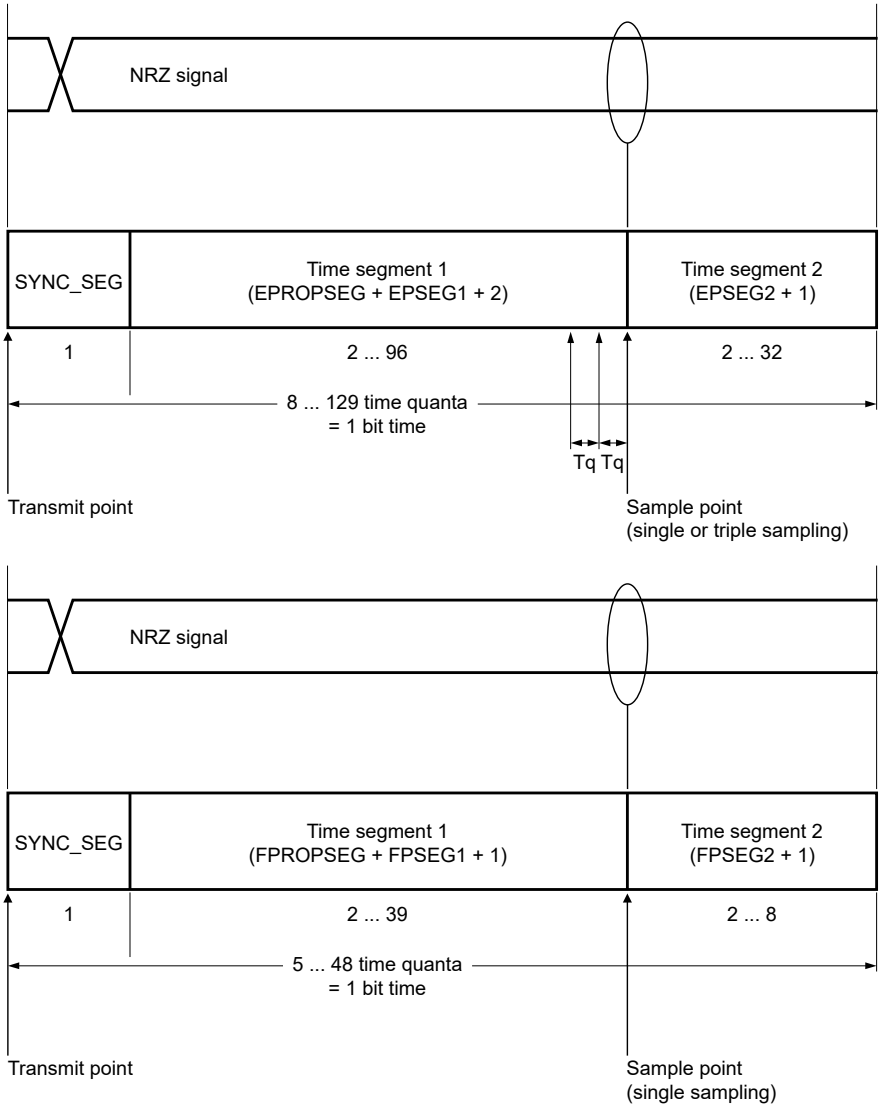


Figure 197. Segments within the bit time (example using CBT and FDCBT bit timing variables for CAN FD format)

Table 336. Time segment syntax

Syntax	Description
SYNC_SEG	Period during which the system expects transitions to occur on the bus
TSEG1	Period corresponding to the sum of PROPSEG and PSEG1
TSEG2	Period corresponding to the PSEG2 value
Transmit point	Point at which a node in Transmit mode transfers a new value to the CAN bus
Sample point	Point at which a node samples the bus. If the option of three samples per bit is selected, this point marks the position of the third sample.

Table 337 gives some examples of the CAN-compliant segment settings for Classical CAN format (Bosch CAN 2.0B) (non-FD) messages.

Table 337. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Resynchronization jump width
5 to 10	2	1 to 2
4 to 11	3	1 to 3
5 to 12	4	1 to 4
6 to 13	5	1 to 4
7 to 14	6	1 to 4
8 to 15	7	1 to 4
9 to 16	8	1 to 4

NOTE

You must ensure the bit time settings comply with the CAN Protocol standard (ISO 11898-1:2015).

52.3.13.7.3 Calculating peripheral clocks

A CAN bit can be used as a measure of duration (for example, estimating the occurrence of a CAN bit event in a message). When a CAN bit is used in this way, the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

Equation 4. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 0

Or, if CTRL2[BTE] = 1:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (ENPRESDIV + 1) \times (NTSEG1 + NTSEG2 + 3)$$

Equation 5. Number of peripheral clocks per CAN bit when CTRL2[BTE] = 1

Where:

- NumClkBit is the number of peripheral clocks in one CAN bit.
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure 194), in Hz.
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz.
- PSEG1 is the value of CTRL1[PSEG1].
- PSEG2 is the value of CTRL1[PSEG2].
- PROPSEG is the value of CTRL1[PROPSEG].
- PRESDIV is the value in CTRL1[PRESDIV].
- ENPRESDIV is the value of EPRS[ENPRESDIV].
- NTSEG1 is the value of ENCBT[NTSEG1].
- NTSEG2 is the value of ENCBT[NTSEG2].

The formula above is also applicable to the alternative CAN bit timing variables described in:

- CAN Bit Timing (CBT)

- Enhanced Nominal CAN Bit Timing (ENCBT)
- CAN FD Bit Timing (FDCBT)
- Enhanced Nominal CAN Bit Timing (ENCBT)

For example, 180 CAN bits = (180 × NumClkBit) peripheral clock periods.

52.3.13.8 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in, and move-out processes are executed during certain time windows inside the CAN frame. These windows are shown in the following figures.

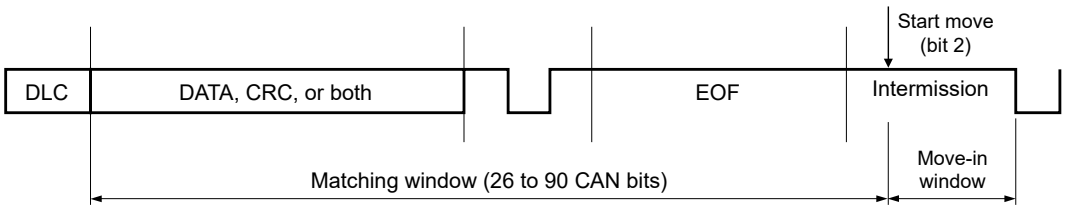


Figure 198. Matching and move-in time windows

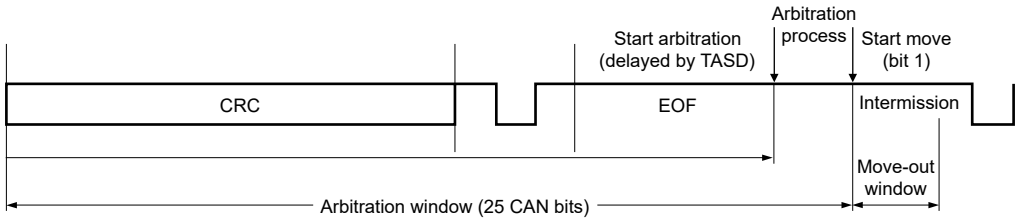


Figure 199. Arbitration and move-out time windows

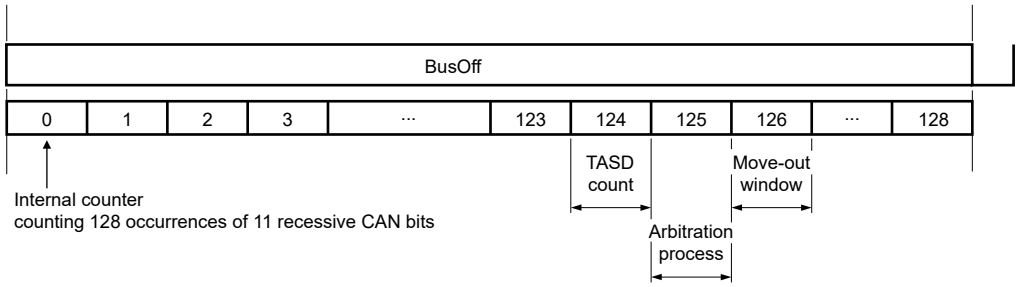


Figure 200. Arbitration at the end of bus off and move-out time windows

NOTE

In these figures, the matching and arbitration timing do not consider delays caused by concurrent memory access due to the CPU or other internal FlexCAN subblocks.

52.3.13.9 TX arbitration start delay

TX Arbitration Start Delay (CTRL2[TASD]) indicates the number of CAN bits that FlexCAN uses to delay the TX arbitration process starting point from the first bit of the CRC field of the current frame. This variable can be written only in Freeze mode; FlexCAN blocks it in other modes.

The ability of the CPU to reconfigure message buffers for transmission after the end of the internal arbitration process impacts transmission performance. In the arbitration process, FlexCAN finds the winner MB for transmission (see Arbitration process). If

the arbitration ends too early (before the first bit of the Intermission field) the CPU may reconfigure some TX message buffers. It is possible that the winning message buffer is no longer the best candidate to be transmitted.

TASD can optimize the transmission performance by defining the arbitration start point, as shown in Figure 201, based on factors such as:

- Peripheral-to-oscillator clock ratio
- CAN bit timing variables that determine the CAN bit rate
- Number of message buffers in use by the matching and arbitration processes

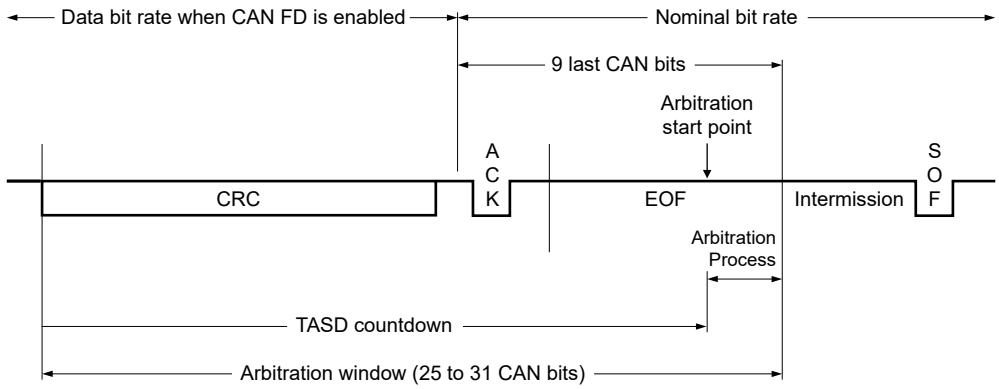


Figure 201. Optimal TX arbitration start point

The duration of an arbitration process, in terms of CAN bits, is:

- Directly proportional to the number of available message buffers
- Directly proportional to the CAN bit rate
- Inversely proportional to the peripheral clock frequency

The optimal arbitration timing occurs when the last message buffer is scanned immediately before the first bit of the Intermission field of a CAN frame. For instance, if the following are true:

- There are few message buffers.
- The peripheral-to-oscillator clock ratio is high.
- The CAN baud rate is low.

Then the arbitration can be placed closer to the end of the frame, adding more delay to its starting point, and vice versa.

If CTRL2[TASD] = 0, the arbitration start is not delayed, and more time is reserved for arbitration. Alternatively, if CTRL2[TASD] is close to 24, the CPU can configure a TX message buffer later, and less time is reserved for arbitration. If too little time is reserved for arbitration, FlexCAN may not be able to find a winner MB in time. The transmitted arbitration winner may not have the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

For CAN FD frames and (MAXMB + 1) ≤ NMB_{END}

$$TASD = 31 - \frac{2 * (MAXMB + 1) + 4}{CPCB_N}$$

For CAN FD frames and (MAXMB + 1) > NMB_{END}

$$TASD = 22 - \frac{2 * (MAXMB + 1) - NMB_{END}}{CPCB_F}$$

For non-FD frames

$$TASD = 25 - \frac{2 * (MAXMB + 1) + 4}{CPCB}$$

Equation 6. Optimal value for TASD

Where:

$$NMB_{END} = \frac{(9 \times CPCB_N) - 4}{2}$$

$$BITRATE_N = \left(\frac{f_{CANCLK}}{[1 + (EPSEG1 + 1) + (EPSEG2 + 1) + (EPROPSEG + 1)] \times (EPRESDIV + 1)} \right)$$

$$BITRATE_F = \left(\frac{f_{CANCLK}}{[1 + (FPSEG1 + 1) + (FPSEG2 + 1) + FPROPSEG] \times (FPRESDIV + 1)} \right)$$

$$CPCB_N = \frac{f_{SYS}}{BITRATE_N}$$

$$CPCB_F = \frac{f_{SYS}}{BITRATE_F}$$

$$CPCB = CPCB_N$$

Equation 7. Variables used in TASD calculation

- MAXMB is the value in MCR[MAXMB].
- NMB_{END} is the number of message buffers that the arbitration process can scan during the last nine CAN bits at the end of a frame. (See Equation 7.)
- $BITRATE_N$ is the CAN bit rate in bits per second calculated by the nominal CAN bit time variables.
- $BITRATE_F$ is the CAN bit rate in bits per second calculated by the data CAN bit time variables.
- $CPCB_N$ is the number of peripheral clocks per CAN bit in nominal bit rate for CAN FD frames.
- $CPCB_F$ is the number of peripheral clocks per CAN bit in data bit rate for CAN FD frames.
- CPCB is the number of peripheral clocks per CAN bit for non-FD frames.
- f_{CANCLK} is the oscillator clock, in Hz.
- f_{SYS} is the peripheral clock, in Hz.
- EPSEG1 is the value in CBT[EPSEG1] (CTRL1[PSEG1] can also be used).
- EPSEG2 is the value in CBT[EPSEG2] (CTRL1[PSEG2] can also be used).
- EPROPSEG is the value in CBT[EPROPSEG] (CTRL1[PROPSEG] can also be used).
- EPRESDIV is the value in CBT[EPRESDIV] (CTRL1[PRESDIV] can also be used).
- FPSEG1 is the value in FDCBT[FPSEG1].

- FPSEG2 is the value in FDCBT[FPSEG2].
- FPROPSEG is the value in FDCBT[FPROPSEG].
- FPRESDIV is the value in FDCBT[FPRESDIV].
- NTSEG1 is the value in ENCBT[NTSEG1].
- NTSEG2 is the value in ENCBT[NTSEG2].
- ENPRESDIV is the value in EPRS[ENPRESDIV].
- DTSEG1 is the value in EDCBT[DTSEG1].
- DTSEG2 is the value in EDCBT[DTSEG2].
- EDPRESDIV is the value in EPRS[EDPRESDIV].

If CTRL2[BTE] = 1, then:

$$BITRATE_N = \frac{f_{CANCLK}}{[1 + (NTSEG1 + 1) + (NTSEG2 + 1)] \times (ENPRESDIV + 1)}$$

Equation 8. Nominal baud rate when CTRL2[BTE] = 1

$$BITRATE_F = \frac{f_{CANCLK}}{[1 + (DTSEG1 + 1) + (DTSEG2 + 1)] \times (EDPRESDIV + 1)}$$

Equation 9. Fast baud rate when CTRL2[BTE] = 1

See also Protocol timing for more details.

52.3.13.9.1 TASD configuration examples

The following tables show the TASD value calculated for some configuration cases.

Case 1:

- Clock ratio = 2:1 (for example, peripheral clock 80 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 338. TASD values in Case 1

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	24	8.0

Case 2:

- Clock ratio = 1:1 (for example, peripheral clock 40 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 339. TASD values in Case 2

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	6.67

Case 3:

- Clock ratio = 2:1 (for example, peripheral clock 40 MHz and oscillator clock 20 MHz)

- Bit rate in arbitration phase = 1 Mbaud

Table 340. TASD values in Case 3

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	4.0

52.3.14 Clocks

The following table describes the clock sources for FlexCAN. See the chip clocking chapter for clock setting, configuration, and gating information.

Table 341. FlexCAN clocks

Clock name	Description
MODULE_CLK (system_clk)	Peripheral clock
MODULE_CLK_CHI (host_clock)	Control Host Interface (CHI) clock
MODULE_CLK_PE (protocol_engine_clock)	Protocol Engine (PE) clock
MODULE_CLK_PE_NOGATE (protocol_engine_clock_nogate)	Protocol Engine clock (no gating)
MODULE_CLK_S (system_clock_nogate)	Peripheral access clock

52.3.14.1 Clock domains and restrictions

FlexCAN has two clock domains asynchronous to each other:

- The bus domain feeds the Control Host Interface (CHI) submodule.
- The oscillator domain feeds the CAN Protocol Engine (PE) submodule.

When the two domains are connected to clocks with different frequencies or phases, the frequency relationship between the two clock domains is restricted. In asynchronous operation, the bus domain clock frequency must always be greater than the oscillator domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When performing matching and arbitration, FlexCAN must scan the whole message buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. To provide sufficient time for the scan, observe the following requirements:

- The peripheral clock frequency cannot be less than the oscillator clock frequency.
- There must be a minimum number of peripheral clocks per CAN bit, as specified in Table 342.

Table 342. Minimum number of peripheral clocks per CAN bit for Classical CAN format

Number of message buffers	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
16	0	0	16
32	0	0	16

Table continues on the next page...

Table 342. Minimum number of peripheral clocks per CAN bit for Classical CAN format (continued)

Number of message buffers	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
16	1	0	16
32	1	0	17
16	0	1	16
32	0	1	19

For classical frame format, the minimum number of peripheral clocks per CAN bit specified in Table 342 determines the minimum peripheral clock frequency for a given number of message buffers and for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit. This number can be defined by adjusting one or more of the bit timing values contained in:

- Control 1 (CTRL1)
- CAN Bit Timing (CBT)
- Enhanced Nominal CAN Bit Timing (ENCBT)

The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be eight, so the oscillator clock frequency should be at least eight times the CAN bit rate.

52.3.14.1.1 Clock restrictions for CAN FD

For CAN FD frame format, some constraints must be satisfied. The equation below calculates the number of peripheral clocks per CAN bit in nominal bit rate (NumClkNomBit).

$$NumClkNomBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

$$= \frac{f_{SYS}}{NomBitRate}$$

Equation 10. Number of peripheral clocks per nominal CAN bit

Where PRESDIV, PSEG1, and PSEG2 are CAN bit time values in Control 1 (CTRL1). Alternatively, EPRESDIV, EPSEG1, and EPSEG2 values in CAN Bit Timing (CBT) or the values of EPRS[ENPRESDIV], ENCBT[NTSEG1], and ENCBT[NTSEG2] can be used instead. NumClkNomBit can also be calculated as a function of the expected nominal bit rate used in the arbitration phase (NomBitRate), as shown in the equation above.

The number of CAN bits in the data phase of an FD frame with BRS = 1 (fast CAN bits) depends on the number of data bytes in the payload. The number of fast CAN bits (NumOfFastBits) can be determined in Table 343. Having fewer data bytes means having fewer fast CAN bits. It also means that less time is available for FlexCAN to scan the whole message buffer memory during the internal matching and arbitration processes.

Table 343. Number of fast CAN bits in a CAN FD frame

Minimum number of data bytes	DLC field	NumOfFastBits
0	0h	21
1	1h	29
2	2h	37
3	3h	45
4	4h	53

Table continues on the next page...

Table 343. Number of fast CAN bits in a CAN FD frame (continued)

Minimum number of data bytes	DLC field	NumOfFastBits
5	5h	61
6	6h	69
7	7h	77
8	8h	85
12	9h	117
16	Ah	149
20	Bh	186
24	Ch	218
32	Dh	282
48	Eh	410
64	Fh	538

The critical part of a CAN FD frame is during the data phase, where the CAN bit rate is faster than in the arbitration phase. The minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated to guarantee that enough time is available for FlexCAN to scan the message buffer memory during reception and transmission. The equation below calculates this constraint.

$$MinNumClkFastBit_A = \frac{(8.5 \times MaxNumOfMb) + [ERFEN \times (2 \times NFE + 4)] + 64 - (9 \times NumClkNomBit)}{NumOfFastBits}$$

Equation 11. Minimum number of peripheral clocks per fast CAN bit for FlexCAN scan process

Where MaxNumOfMb is the maximum number of available message buffers defined in MCR[MAXMB]. NFE and ERFEN are the fields defined in Enhanced RX FIFO Control (ERFCR).

The clock-domain-crossing circuit between the CHI and PE subblocks also imposes a minimum number of peripheral clocks per fast CAN bit. This minimum is required for the handshake mechanism to work properly without losing status information through the interface, as shown in the equation below.

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{f_{SYS}}{f_{CANCLK}}\right)$$

Equation 12. Minimum number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

Therefore, the larger of the two values calculated above determines the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit).

$$MinNumClkFastBit = Maximum\ (MinNumClkFastBitA, MinNumClkFastBitB)$$

Equation 13. Minimum number of peripheral clocks per fast CAN bit

Then, the maximum CAN bit rate in the data phase of CAN FD frames (DataBitRateMAX) can be calculated as below.

$$DataBitrate_{MAX} = \frac{f_{CANCLK}}{ROUNDUP\left(\frac{MinNumClkFastBit \times f_{CANCLK}}{f_{SYS}}\right)}$$

Equation 14. Maximum achievable baud rate for data phase

These factors affect the maximum data bit rate attainable by FlexCAN in CAN FD mode:

- The peripheral and oscillator clock frequencies
- The maximum number of message buffers
- The expected nominal bit rate

Also, the data bit rate depends on the minimum payload size of FD frames used in a given application.

To illustrate how the configuration of FlexCAN variables affects the CAN FD bit rate, consider this application example:

- The peripheral clock frequency is set to 50 MHz
- The oscillator clock frequency is set to 40 MHz

1. Considering the nominal bit rate as 1 Mbit/s, the number of peripheral clocks per CAN bit in nominal bit rate is calculated as below.

$$NumClkNomBit = \frac{50 \times 10^6}{1 \times 10^6} = 50$$

Equation 15. Calculation example for number of peripheral clocks per nominal CAN bit

2. The number of fast CAN bits (NumOfFastBits) is determined in Table 343. For example, if the minimum payload in FD frames is 8 bytes, there are 85 CAN bits in the data phase.
3. Assuming the maximum number of message buffers is 96, and Enhanced RX FIFO is disabled, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated.

$$MinNumClkFastBit_A = \frac{(8.5 \times 96) + 64 - (9 \times 50)}{85} = 5.06$$

Equation 16. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN scan process

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{50}{40}\right) = 6.75$$

Equation 17. Calculation example for number of peripheral clocks per fast CAN bit for FlexCAN clock domain interface

$$MinNumClkFastBit = Maximum\ (\ 5.06, 6.75\ ) = 6.75$$

Equation 18. Calculation example for number of peripheral clocks per fast CAN bit

4. The maximum CAN bit rate in the data phase can finally be found.

$$DataBitRate_{MAX} = \frac{40 \times 10^6}{ROUNDUP\left(\frac{6.75 \times 40 \times 10^6}{50 \times 10^6}\right)} = 6.667\ Mbps$$

Equation 19. Calculation example for maximum achievable baud rate

Even though the oscillator clock frequency (40 MHz) is adequate to generate a data rate of 8 Mbit/s in CAN FD mode, the specific FlexCAN configuration limits this rate to 6.667 Mbit/s. This limitation is mainly due to the low peripheral clock frequency that imposes the MinNumClkFastBitB bound.

Table 344 shows the maximum data rate for CAN FD with Enhanced RX FIFO disabled according to clock frequencies, payload size, and number of available message buffers. For some cases, if the number of available message buffers is reduced, FlexCAN can then achieve a data rate up to 8 Mbit/s.

Table 344. Maximum CAN bit rate in data phase on CAN FD frames with Enhanced RX FIFO disabled

Peripheral clock frequency (MHz)	Payload size	Number of available message buffers	Maximum data rate (Mbit/s)
40	8	94	6.667
40	8	114	>5.0
40	12	>117	6.667
40	12	128	5.714
50	12–64	128	6.667
60	8	126	8.0
60	12	128	8.0
67	6	128	8.0
80	3	128	8.0
100	0	128	8.0

52.3.15 Reset

You can reset FlexCAN in the following ways:

- Soft reset, in one of two ways:
  - MCR[SOFTRST], which resets some of the memory-mapped registers synchronously. To see which registers soft reset affects, see Table 347.
  - Chip-level soft reset, which has the same effect as MCR[SOFTRST].

Soft reset is synchronous and must follow an internal request-and-acknowledge procedure across clock domains. Therefore, it may take some time to propagate its effects fully. MCR[SOFTRST] remains 1 when soft reset is pending, so software can poll this field to identify when the reset has completed. Soft reset cannot be applied when clocks are shut down in a low-power mode. The low-power mode should be exited and the clocks resumed before applying soft reset.

When the module is enabled (MCR[MDIS] becomes 0), FlexCAN automatically enters Freeze mode. In Freeze mode:

1. FlexCAN is unsynchronized to the CAN bus.
2. MCR[HALT] and MCR[FRZ] become 1.
3. The internal state machines are disabled.
4. MCR[FRZACK] and MCR[NOTRDY] become 1.

The TX pin is in the recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Reset does not affect the message buffers and the RX Individual Mask registers, so they are not automatically initialized.

52.3.16 Interrupts

FlexCAN has many interrupt sources:

- Interrupts due to message buffers

- Interrupts due to interrupts combined via an OR operator from:
  - Message buffers
  - Bus Off
  - Bus Off Done
  - Error
  - Error Fast (errors detected in the data phase of CAN FD format messages with BRS = 1)
  - Wake Up
  - Wake Up Match
  - Wake Up Timeout
  - TX Warning
  - RX Warning

If its corresponding IMASK bit is 1, each message buffer can be an interrupt source. There is no distinction between TX and RX interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each buffer has an assigned flag bit in the IFLAG registers. When the corresponding buffer completes a successful transfer, the flag is set. When the CPU writes 1 to it, the flag is cleared (unless another interrupt is generated at the same time).

NOTE

The CPU must clear only the bit causing the current interrupt. For this reason, do not use bit manipulation instructions (BSET) to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt handler.

If the Legacy RX FIFO is enabled (MCR[RFEN] = 1) and DMA is disabled (MCR[DMA] = 0), the interrupts corresponding to message buffers 0–7 have different meanings.

- Bit of Interrupt Flags 1 (IFLAG1) becomes the Legacy FIFO Overflow flag
- Bit becomes the Legacy FIFO Warning flag
- Bit becomes the Frames Available in Legacy FIFO flag
- Bits are unused.

See Interrupt Flags 1 (IFLAG1) for more information.

If both Legacy RX FIFO and DMA are enabled (MCR[RFEN] = 1 and MCR[DMA] = 1), FlexCAN does not generate any Legacy FIFO interrupt. Bit of IFLAG1 still indicates Frames Available in Legacy FIFO and generates a DMA request. Bits are unused.

CAUTION

Legacy FIFO cannot be enabled when CAN FD is enabled.

When multiple message buffer interrupt sources are combined via an OR operator into a single interrupt, the interrupt is generated when any associated message buffer (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the IFLAG registers to determine which message buffer or FIFO source caused the interrupt.

These interrupt sources generate interrupts like the message buffer interrupt sources, and can be read from Error and Status 1 (ESR1):

- Bus Off
- Bus Off Done
- Error
- Error Fast
- Wake Up
- TX Warning

- RX Warning

The Bus Off, Error, TX Warning, and RX Warning interrupt masks are located in Control 1 (CTRL1); the wake-up interrupt mask is located in Module Configuration (MCR).

The interrupt sources for Pretended Networking (Wake-up by Match Flag and Wake-up by Timeout Flag) can be read in Pretended Networking Wake-Up Match (WU_MTC). The respective interrupt mask bits are located in Pretended Networking Control 1 (CTRL1_PN).

52.3.17 Bus interface

CPU access to FlexCAN registers is subject to the following rules:

- Read and write access to implemented reserved address space results in an access error.
- Write access to positions whose bits are all currently read-only results in an access error. If at least one of the bits is not read-only, no access error is issued. Write permission to specific positions or some of their bits can change depending on the mode of operation or transitory state. See register and field descriptions for details.
- Read and write access to unimplemented address space results in an access error.
- Read and write access to RAM-located positions during Low-Power mode results in an access error.
- The RXIMR memory region can be considered as general-purpose memory and available for access via these methods:
  - If you write 0 to MCR[IRMQ], the individual masks (RXIMR) are disabled. In this case, the RXIMR memory region is considered general-purpose memory.
  - If MCR[MAXMB] is programmed with a value smaller than the available number of message buffers, the unused memory space can be used as general-purpose RAM space. Reserved words within RAM cannot be used.
    For example, suppose the RAM in FlexCAN can support up to 16 message buffers, CTRL2[RFFN] = 0h, and MCR[MAXMB] = 0.
    - In this case, the maximum number of message buffers becomes one.
    - The RAM starts at 0080h, and the space 0080h–008Fh is used by the one message buffer.
    - The memory space 0090h–017Fh is available.
    - The space 0180h–087Fh is reserved.
    - The space 0880h–0883h is used by the one individual mask and the available memory in the mask register space is 0884h–08BFh.
    - In the space from 08C0h–09DFh, there are reserved words for internal use which cannot be used as general-purpose RAM.

    As a rule, free memory space for general purpose depends only on MCR[MAXMB].
  - If MCR[FDEN] = 1, general-purpose memory can be used only outside Freeze mode.

Table 345. Access permissions

Modes of operation	Normal	Freeze	Low-power
MCR	Bus error	Bus error	Bus error
CTRL1	Read and write	Read and write	Read and write
TIMER	Read and write	Read and write	Read and write
TCR	Bus error	Bus error	Bus error
RXMGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation

Table continues on the next page...

Table 345. Access permissions (continued)

Modes of operation	Normal	Freeze	Low-power
RX14MASK[1]	Bus error for write operation	Read and write	Bus error for write operation
RX15MASK[1]	Bus error for write operation	Read and write	Bus error for write operation
ECR	Bus error for write operation	Read and write	Bus error for write operation
ESR1	Read and write	Read and write	Read and write
IMASK1	Read and write	Read and write	Read and write
IFLAG1	Read and write	Read and write	Read and write
CTRL2	Read and write	Read and write	Read and write
ESR2	Read and write	Read and write	Read and write
CRCR	Bus error for write operation	Bus error for write operation	Bus error for write operation
RXFGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation
RXFIR[1]	Bus error for write operation	Bus error for write operation	Bus error for write operation
CBT	Bus error for write operation	Read and write	Bus error for write operation
DBG1	Bus error for write operation	Bus error for write operation	Bus error for write operation
DBG2	Bus error for write operation	Bus error for write operation	Bus error for write operation
MB[1] [2]	Read and write	Read and write	Read and write
Legacy FIFO header[1]	Read and write	Read and write	Read and write
Legacy FIFO reserved space[1]	Bus error	Bus error	Bus error
Legacy FIFO filters[1]	Read and write	Read and write	Read and write
RXIMR[1]	Bus error	Read and write	Bus error
CTRL1_PN	Read and write [3]	Read and write	Read and write [3]
CTRL2_PN	Read and write [3]	Read and write	Read and write [3]
WU_MTC	Read and write	Read and write	Read and write
FLT_ID1	Read and write [3]	Read and write	Read and write [3]
FLT_DLC	Read and write [3]	Read and write	Read and write [3]
PL1_LO	Read and write [3]	Read and write	Read and write [3]
PL1_HI	Read and write [3]	Read and write	Read and write [3]

Table continues on the next page...

Table 345. Access permissions (continued)

Modes of operation	Normal	Freeze	Low-power
FLT_ID2_IDMASK	Read and write [3]	Read and write	Read and write [3]
PL2_PLMASK_LO	Read and write [3]	Read and write	Read and write [3]
PL2_PLMASK_HI	Read and write [3]	Read and write	Read and write [3]
WMB0	Read and write [3]	Read and write [3]	Read and write [3]
WMB1	Read and write [3]	Read and write [3]	Read and write [3]
WMB2	Read and write [3]	Read and write [3]	Read and write [3]
WMB3	Read and write [3]	Read and write [3]	Read and write [3]
EPRS	Bus error for write operation	Read and write	Bus error for write operation
ENCBT	Bus error for write operation	Read and write	Bus error for write operation
EDCBT	Bus error for write operation	Read and write	Bus error for write operation
ETDC	Bus error for write operation	Read and write	Bus error for write operation
FDCTRL	Read and write	Read and write	Read and write
FDCBT	Centralize [3]	Read and write	Read and write [3]
FDCRC	Bus error for write operation	Bus error for write operation	Bus error for write operation
ERFCR	Centralize [3]	Read and write	Read and write [3]
ERFIER	Read and write	Read and write	Read and write
ERFSR	Read and write	Read and write	Read and write
Enhanced Rx FIFO header[4]	Bus error for write operation	Bus error for write operation	Bus error for write operation
Enhanced Rx FIFO reserved space[4]	Bus error	Bus error	Bus error
ERFFEL	Bus error for write operation	Read and write	Bus error for write operation
General purpose RAM[1]	Read and write	Read and write	Read and write
Reserved space (used)[1]	Bus error	Bus error	Bus error
Reserved space (empty)[1]	Bus error	Bus error	Bus error

1. Access in low power is only possible if RAM_clk and MODULE_CLK are enabled.
2. If MCR[RFEN] = 1, see Legacy FIFO access rules below.
3. Write operation has no effect.
4. If MCR[RFEN] = 1, do not access Enhanced RX FIFO.

52.4 External signal descriptions

FlexCAN has two I/O signals connected to the external chip pins. These signals are summarized in Table 346 and described in the following subsections.

Table 346. FlexCAN signal descriptions

Signal	Description	I/O
CAN RX	CAN receive pin	Input
CAN TX	CAN transmit pin	Output

52.4.1 CAN RX

This pin is the receive pin from the CAN bus transceiver. Logic level 0 represents its dominant state. Logic level 1 represents its recessive state.

52.4.2 CAN TX

This pin is the transmit pin to the CAN bus transceiver. Logic level 0 represents its dominant state. Logic level 1 represents its recessive state.

52.5 Memory map and register definition

This section describes the registers and data structures in FlexCAN. The base address of the module depends on the particular memory map of the chip.

52.5.1 FlexCAN memory mapping

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0080h.

Each individual register is identified by its complete name and corresponding mnemonic.

NOTE

An invalid register access results in a bus error. Invalid accesses include reading a write-only register, writing a read-only register, and accessing an invalid address.

NOTE

To update the parity bits in memory properly, all FlexCAN memory must be initialized before reading registers which are implemented in memory. You must also initialize RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), Receive 15 Mask (RX15MASK) and Legacy RX FIFO Global Mask (RXFGMASK). MCR[RFEN] and ERFCR[ERFEN] must not be 1 during memory initialization.

Table 347. Register reset information

Register	Affected by soft reset
Module Configuration (MCR)	Yes
Control 1 (CTRL1)	No
Free-Running Timer (TIMER)	Yes
RX Message buffers Global Mask (RXMGMASK)	No

Table continues on the next page...

Table 347. Register reset information (continued)

Register	Affected by soft reset
RX Buffer 14 Mask (RX14MASK)	No
RX Buffer 15 Mask (RX15MASK)	No
Error Counter (ECR)	Yes
Error and Status 1 (ESR1)	Yes
Interrupt Masks 1 (IMASK1)	Yes
Interrupt Flags 1 (IFLAG1)	Yes
Control 2 (CTRL2)	No
Error and Status 2 (ESR2)	Yes
Cyclic Redundancy Check (CRCR)	Yes
RX FIFO Global Mask (RXFGMASK)	No
RX FIFO Information (RXFIR)	No
CAN Bit Timing (CBT)	No
Message buffers	No
RX Individual Masks	No
Pretended Networking Control 1 (CTRL1_PN)	Yes
Pretended Networking Control 2 (CTRL2_PN)	Yes
Pretended Networking Wake-Up Match (WU_MTC)	Yes
Pretended Networking ID Filter 1 (FLT_ID1)	Yes
Pretended Networking DLC Filter (FLT_DLC)	Yes
Pretended Networking Payload Low Filter 1 (PL1_LO)	Yes
Pretended Networking Payload High Filter 1 (PL1_HI)	Yes
Pretended Networking ID Filter 2 or ID Mask (FLT_ID2_IDMASK)	Yes
Pretended Networking Payload Low Filter 2 or Payload Low Mask (PL2_PLMASK_LO)	Yes
Pretended Networking Payload High Filter 2 or Payload High Mask (PL2_PLMASK_HI)	Yes
Pretended Networking Wake Up Message Buffer 0 (WMB0)	No

Table continues on the next page...

Table 347. Register reset information (continued)

Register	Affected by soft reset
Pretended Networking Wake Up Message Buffer 1 (WMB1)	No
Pretended Networking Wake-Up Message Buffer 2 (WMB2)	No
Pretended Networking Wake-Up Message Buffer 3 (WMB3)	No
Enhanced CAN Bit Timing Prescalers (EPRS)	No
Enhanced Nominal CAN Bit Timing (ENCBT)	No
Enhanced Data Phase CAN bit Timing (EDCBT)	No
Enhanced Transceiver Delay Compensation (ETDC)	No
CAN FD Control (FDCTRL)	No
CAN FD Bit Timing (FDCBT)	No
CAN FD CRC (FDCRC)	Yes
Enhanced RX FIFO Control (ERFCR)	Yes
Enhanced RX FIFO Interrupt Enable (ERFIER)	Yes
Enhanced RX FIFO Status (ERFSR)	Yes
Enhanced RX FIFO	No
Enhanced RX FIFO Filter Element (ERFFEL)	No

FlexCAN can store CAN messages for transmission and reception using message buffers and RX FIFO structures.

52.5.2 CAN register descriptions

The table below shows the FlexCAN memory map.

The address range from offset 80h–27Fh allocates the thirty-two 128-bit message buffers. The memory maps for the message buffers are in FlexCAN message buffer memory map.

The address range from offset 2000h–2048h allocates the Enhanced RX FIFO output, and the address range from offset 204Ch–238Ch allocates the rest of Enhanced RX FIFO 11 elements. The memory map for the Enhanced RX FIFO is in Enhanced RX FIFO structure.

52.5.2.1 CAN memory map

CAN0 base address: 4003_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	D890_040Fh
4h	Control 1 (CTRL1)	32	RW	0000_0000h
8h	Free-Running Timer (TIMER)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	RX Message Buffers Global Mask (RXMGMASK)	32	RW	See section
14h	Receive 14 Mask (RX14MASK)	32	RW	See section
18h	Receive 15 Mask (RX15MASK)	32	RW	See section
1Ch	Error Counter (ECR)	32	RW	0000_0000h
20h	Error and Status 1 (ESR1)	32	RW	0000_0000h
28h	Interrupt Masks 1 (IMASK1)	32	RW	0000_0000h
30h	Interrupt Flags 1 (IFLAG1)	32	RW	0000_0000h
34h	Control 2 (CTRL2)	32	RW	00A0_001Ch
38h	Error and Status 2 (ESR2)	32	R	0000_1000h
44h	Cyclic Redundancy Check (CRCR)	32	R	0000_0000h
48h	Legacy RX FIFO Global Mask (RXFGMASK)	32	RW	See section
4Ch	Legacy RX FIFO Information (RXFIR)	32	R	See section
50h	CAN Bit Timing (CBT)	32	RW	0000_0000h
78h	External Timer (ET)	32	R	0000_0000h
7Ch	Fault Confinement Interrupt Enable (FLTCONF_IE)	32	RW	0000_0000h
880h - 8FCh	Receive Individual Mask (RXIMR0 - RXIMR31)	32	RW	See section
B00h	Pretended Networking Control 1 (CTRL1_PN)	32	RW	0000_0100h
B04h	Pretended Networking Control 2 (CTRL2_PN)	32	RW	0000_0000h
B08h	Pretended Networking Wake-Up Match (WU_MTC)	32	RW	0000_0000h
B0Ch	Pretended Networking ID Filter 1 (FLT_ID1)	32	RW	0000_0000h
B10h	Pretended Networking Data Length Code (DLC) Filter (FLT_DLC)	32	RW	0000_0008h
B14h	Pretended Networking Payload Low Filter 1 (PL1_LO)	32	RW	0000_0000h
B18h	Pretended Networking Payload High Filter 1 (PL1_HI)	32	RW	0000_0000h
B1Ch	Pretended Networking ID Filter 2 or ID Mask (FLT_ID2_IDMASK)	32	RW	0000_0000h
B20h	Pretended Networking Payload Low Filter 2 and Payload Low Mask (PL2_PLMASK_LO)	32	RW	0000_0000h
B24h	Pretended Networking Payload High Filter 2 and Payload High Mask (PL2_PLMASK_HI)	32	RW	0000_0000h
B40h	Wake-Up Message Buffer (WMB0_CS)	32	R	0000_0000h
B44h	Wake-Up Message Buffer for ID (WMB0_ID)	32	R	0000_0000h
B48h	Wake-Up Message Buffer for Data 0–3 (WMB0_D03)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B4Ch	Wake-Up Message Buffer Register Data 4–7 (WMB0_D47)	32	R	0000_0000h
B50h	Wake-Up Message Buffer (WMB1_CS)	32	R	0000_0000h
B54h	Wake-Up Message Buffer for ID (WMB1_ID)	32	R	0000_0000h
B58h	Wake-Up Message Buffer for Data 0–3 (WMB1_D03)	32	R	0000_0000h
B5Ch	Wake-Up Message Buffer Register Data 4–7 (WMB1_D47)	32	R	0000_0000h
B60h	Wake-Up Message Buffer (WMB2_CS)	32	R	0000_0000h
B64h	Wake-Up Message Buffer for ID (WMB2_ID)	32	R	0000_0000h
B68h	Wake-Up Message Buffer for Data 0–3 (WMB2_D03)	32	R	0000_0000h
B6Ch	Wake-Up Message Buffer Register Data 4–7 (WMB2_D47)	32	R	0000_0000h
B70h	Wake-Up Message Buffer (WMB3_CS)	32	R	0000_0000h
B74h	Wake-Up Message Buffer for ID (WMB3_ID)	32	R	0000_0000h
B78h	Wake-Up Message Buffer for Data 0–3 (WMB3_D03)	32	R	0000_0000h
B7Ch	Wake-Up Message Buffer Register Data 4–7 (WMB3_D47)	32	R	0000_0000h
BF0h	Enhanced CAN Bit Timing Prescalers (EPRS)	32	RW	0000_0000h
BF4h	Enhanced Nominal CAN Bit Timing (ENCBT)	32	RW	0000_0000h
BF8h	Enhanced Data Phase CAN Bit Timing (EDCBT)	32	RW	0000_0000h
BFCh	Enhanced Transceiver Delay Compensation (ETDC)	32	RW	0000_0000h
C00h	CAN FD Control (FDCTRL)	32	RW	8000_0100h
C04h	CAN FD Bit Timing (FDCBT)	32	RW	0000_0000h
C08h	CAN FD CRC (FDCRC)	32	R	0000_0000h
C0Ch	Enhanced RX FIFO Control (ERFCR)	32	RW	0000_0000h
C10h	Enhanced RX FIFO Interrupt Enable (ERFIER)	32	RW	0000_0000h
C14h	Enhanced RX FIFO Status (ERFSR)	32	RW	0000_0000h
3000h - 307Ch	Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL31)	32	RW	See section

52.5.2.2 Module Configuration (MCR)

Offset

Register	Offset
MCR	0h

Function

Defines global system configurations, including the module operation modes and the maximum message buffer configuration.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MDIS	FRZ	RFEN	HALT	NOTRDY	WAKMSK	SOFTRST	FRZACK	Reserved	SLFWAK	WRNEN	LPMACK	WAKSRC	DOZE	SRXDIS	IRMQ
W																
Reset	1	1	0	1	1	0	0	0	1	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA	PNET_EN	LPRIOEN	AEN	FDEN	TPOV	IDAM		TPOE	MAXMB						
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31 MDIS	Module Disable Disables FlexCAN. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Soft reset does not affect this field. See the chip-specific FlexCAN information for reset value of MCR[MDIS] and state of FlexCAN (enabled or disabled). 0b - Enable 1b - Disable
30 FRZ	Freeze Enable Specifies FlexCAN behavior when MCR[HALT] = 1 or when Debug mode is requested at chip level. When this field becomes 1, FlexCAN can enter Freeze mode. Writing 0 to this field causes FlexCAN to exit from Freeze mode. 0b - Disable 1b - Enable
29 RFEN	Legacy RX FIFO Enable Enables the Legacy RX FIFO feature. When this field is 1, message buffers 0–5 cannot be used for normal reception and transmission. The corresponding memory region (80h–DCh) is used by the FIFO engine and additional message buffers (up to 32, depending on CTRL2[RFFN]). These message buffers are used as Legacy RX FIFO ID filter table elements. This field also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 342. This field can be written in Freeze mode only; the module blocks it in other modes. **NOTE** When CAN FD operation is enabled (see MCR[FDEN]), you cannot write 1 to this field. **NOTE** This field must not be 1 if ERFCR[ERFEN] = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
28 HALT	Halt FlexCAN Puts FlexCAN into Freeze mode. The CPU should write 0 to this field after initializing the message buffers and Control 1 (CTRL1) and Control 2 (CTRL2). FlexCAN performs no reception or transmission before this field becomes 0. Freeze mode cannot be entered when FlexCAN is in a low-power mode. 0b - No request 1b - Enter Freeze mode, if MCR[FRZ] = 1.
27 NOTRDY	FlexCAN Not Ready Indicates whether FlexCAN is in Disable mode, Doze mode, Stop mode or Freeze mode. When FlexCAN has exited these modes, this field becomes 0. Soft reset does not affect this field. 0b - FlexCAN is in Normal mode, Listen-Only mode, or Loopback mode. 1b - FlexCAN is in Disable mode, Doze mode, Stop mode, or Freeze mode.
26 WAKMSK	Wake-up Interrupt Mask Enables the wake-up interrupt generation under the Self Wake-Up mechanism. 0b - Disabled 1b - Enabled
25 SOFTRST	Soft Reset Resets internal state machines of FlexCAN and some memory-mapped registers. The CPU can write 1 to this field directly. This field also becomes 1 when global soft reset is requested at the chip level. Because soft reset is synchronous and must follow a request-and-acknowledge procedure across clock domains, it may take some time to propagate its effect fully. When reset is pending, this field remains 1; it automatically becomes 0 when reset completes. You can poll this field to know when the soft reset has completed. Soft reset cannot be applied when clocks are shut down in a low-power mode. Transfer the module out of the low-power mode before applying soft reset. Soft reset does not affect this field. **NOTE** This field becomes 0 within 2 CAN bits after assertion of this bit. 0b - No reset 1b - Soft reset affects reset registers
24 FRZACK	Freeze Mode Acknowledge Indicates whether FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore you can poll this field to know when FlexCAN has entered Freeze mode. If the Freeze mode request is negated, this field becomes 0 after the FlexCAN prescaler is running again. If Freeze mode is requested when FlexCAN is in

Table continues on the next page...

Table continued from the previous page...

Field	Function
	a low-power mode, this field becomes 1 only when the low-power mode is exited. See Freeze mode. Soft reset does not affect this field. **NOTE** This field becomes 1 within 180 CAN bits if current transmission or reception is ongoing for classical frames and 730 CAN bits if current transmission or reception is ongoing for FD frames after the low-power mode requested by CPU. This field becomes 0 within 2 CAN bits after the Freeze mode request removal (see Protocol timing). 0b - Not in Freeze mode, prescaler running. 1b - In Freeze mode, prescaler stopped.
23 —	Reserved
22 SLFWAK	Self Wake-up Enables the Self Wake-up feature when FlexCAN is in a low-power mode other than Disable mode. When this feature is enabled, the FlexCAN module monitors the bus for a wake-up event (that is, a recessive-to-dominant transition). If a wake-up event is detected during Doze mode, FlexCAN requests to resume its clocks. If enabled to do so, it also generates a wake-up interrupt to the CPU. If a wake-up event is detected during Stop mode, FlexCAN generates, if enabled to do so, a wake-up interrupt to the CPU. The CPU can exit Stop mode globally and FlexCAN can request to resume the clocks. When FlexCAN is in a low-power mode other than Disable mode, this field cannot be written, as the module blocks it. When MCR[PNET_EN] = 1, this feature must be disabled. 0b - Disable 1b - Enable
21 WRNEN	Warning Interrupt Enable Enables the generation of the flags ESR1[TWRNINT] and ESR1[RWRNINT]. When this field is 1, TWRNINT and RWRNINT flags are set when the respective error counter transitions from less than 96 to greater than or equal to 96. When this field is 0, the TWRNINT and RWRNINT flags are always zero, independent of the values of the error counters. No warning interrupt is generated. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
20 LPMACK	Low-Power Mode Acknowledge Indicates whether FlexCAN is in a low-power mode (Disable mode, Doze mode, Stop mode). A low-power mode cannot be entered until all current transmission and reception processes have finished. The CPU can poll this field to know when FlexCAN has entered low-power mode. Soft reset does not affect this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field becomes 1 within 180 CAN bits if current transmission or reception is ongoing for classical frames and 730 CAN bits if current transmission or reception is ongoing for FD frames after the low-power mode requested by CPU. This field becomes 0 within 2 CAN bits after the low-power mode request removal (see Protocol timing). When FlexCAN is in Pretended Networking mode, this field becomes 0 within 180 CAN bits after the low-power mode request removal. 0b - Not in a low-power mode 1b - In a low-power mode
19 WAKSRC	Wake-Up Source Determines whether the integrated low-pass filter is applied to the RX input that FlexCAN uses to detect recessive-to-dominant edges on the CAN bus. This filter can protect the RX CAN input from spurious wake-up. This field can be written only in Freeze mode. The module blocks it in other modes. 0b - No filter applied 1b - Filter applied
18 DOZE	Doze Mode Enable Determines whether FlexCAN can enter low-power mode when Doze mode is requested at chip level. This field is automatically reset when FlexCAN wakes from Doze mode upon detecting activity on the CAN bus (self wake-up enabled). 0b - Disable 1b - Enable
17 SRXDIS	Self-Reception Disable Determines whether FlexCAN can receive frames transmitted by itself. If 1, frames transmitted by the module are not stored in any MB, regardless of whether the MB is programmed with an ID that matches the transmitted frame. No interrupt flag or interrupt signal is generated due to the frame reception. This field can be written only in Freeze mode; the module blocks it in other modes. 0b - Enable 1b - Disable
16 IRMQ	Individual RX Masking and Queue Enable Indicates whether RX matching process is based on individual masking and queue, or based on a masking scheme with RX Message Buffers Global Mask (RXMGMASK), Receive 14 Mask (RX14MASK), Receive 15 Mask (RX15MASK), and Legacy RX FIFO Global Mask (RXFGMASK). When this field is disabled, for backward compatibility with legacy applications, reading the Control and Status word locks the MB even if it is empty. This field can be written in Freeze mode only. The module blocks it in other modes. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 DMA	DMA Enable Enables DMA. The DMA feature can only be used in Legacy RX FIFO or Enhanced RX FIFO, so MCR[RFEN] or ERFCR[ERFEN] must be 1. When DMA and RFEN are 1, IFLAG1[BUF5I] generates the DMA request, and no RX FIFO interrupt is generated. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
14 PNET_EN	Pretended Networking Enable Enables Pretended Networking functionality. When in Doze mode or Stop mode, the PE subblock is kept operational. It can process RX message filtering as defined by the Pretended Networking configuration registers. See Receive process in Pretended Networking mode. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable
13 LPRIOEN	Local Priority Enable Enables the local priority feature. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word. However, the actual transmitted ID is 11 bits for standard frames and 29 bits for extended frames. This field can be written only in Freeze mode; the module blocks it in other modes. This bit is provided for backward compatibility with legacy applications. 0b - Disable 1b - Enable
12 AEN	Abort Enable Enables the TX abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** When this field is 1, only use the abort mechanism (see Transmission abort mechanism) to update message buffers configured for transmission. **CAUTION** Writing the abort code into RX message buffers can cause unpredictable results when this field is 1. 0b - Disabled 1b - Enabled
11	CAN FD Operation Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FDEN	Enables the CAN with flexible data rate (CAN FD) operation. This field can be written in Freeze mode only. FlexCAN can receive and transmit messages in CAN 2.0 format. If this field is enabled, FlexCAN can also receive and transmit messages in CAN FD format. FlexCAN can transmit FD frame format according to ISO 11898-1:2015. **NOTE** If the value of this field is 1, the Legacy RX FIFO Enable (MCR[RFEN]) field cannot be 1. 0b - Disable 1b - Enable
10 TPOV	TX Pin Override Value Indicates forced value on the TX pin. This bit is affected by soft reset. 0b - TX pin is forced to be dominant 1b - TX pin is forced to be recessive
9-8 IDAM	ID Acceptance Mode Identifies the format of the Legacy RX FIFO ID filter table elements. This field configures all elements of the table at the same time; they are all the same format. See Legacy RX FIFO structure. This field can be written only in Freeze mode; the module blocks it in other modes. 00b - Format A: One full ID (standard and extended) per ID filter table element. 01b - Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID filter table element. 10b - Format C: Four partial 8-bit standard IDs per ID filter table element. 11b - Format D: All frames rejected.
7 TPOE	TX Pin Override Enable Enables forcing of TX pin to recessive or dominant. This bit is affected by soft reset. **NOTE** This bit must not be enabled during the functional mode. 0b - TX pin forcing is disabled 1b - TX pin forcing is enabled
6-0 MAXMB	Number of the Last Message Buffer Defines the number of the last message buffer that takes part in the matching and arbitration processes. The reset value (0Fh) is equivalent to a 16-MB configuration. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** You must write a value smaller than or equal to the number of available message buffers to this field, as described in FlexCAN memory partition for CAN FD.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Additionally, the MAXMB value must consider the region of message buffers occupied by Legacy RX FIFO and its ID filters table space defined by CTRL2[RFFN]. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit, as described in Table 342.

52.5.2.3 Control 1 (CTRL1)

Offset

Register	Offset
CTRL1	4h

Function

Contains specific FlexCAN control features related to the CAN bus. These features include bit rate, programmable sampling point within an RX bit, Loopback mode, Listen-Only mode, Bus Off recovery behavior, and interrupt enabling (Bus-Off, Error, Warning). It also determines the division factor for the clock prescaler.

The CAN bit timing variables (CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW]) can also be configured in CAN Bit Timing (CBT), which extends the range of all these variables. If CBT[BTF] = 1, CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] become read-only.

If CTRL2[BTE] = 1, CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] are not used by the module. Instead, these fields are read as zero, and a write operation to them has no effect.

NOTE

When the CAN FD feature is enabled, do not use CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] for CAN bit timing. Instead use CBT[EPRESDIV], CBT[ERJW], CBT[EPSEG1], CBT[EPSEG2], and CBT[EPROPSEG].

The CAN bit variables in CTRL1 and in CBT are stored in the same internal register.

Soft reset does not affect the contents of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRESDIV								RJW		PSEG1			PSEG2		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BOFF MSK	ERRM SK	Reserv ed	LPB	TWRN MSK	RWRN MSK	0	ROM	SMP	BOFF REC	TSYN	LBUF	LOM	PROPSEG		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 PRESDIV	Prescaler Division Factor Determines the ratio between the PE clock frequency and the serial clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The maximum value of this field is FFh, which gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (PRESDIV + 1).
23-22 RJW	Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time. One time quantum is equal to one Sclock period. The valid programmable values are 0–3. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = RJW + 1.
21-19 PSEG1	Phase Segment 1 Defines the length of phase segment 1 in the bit time. The valid programmable values are 0–7. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) × Time Quanta.
18-16 PSEG2	Phase Segment 2 Defines the length of phase segment 2 in the bit time. The valid programmable values are 1–7. See Protocol timing for more information. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time Quanta.
15 BOFFMSK	Bus Off Interrupt Mask Provides a mask for the Bus Off interrupt ESR1[BOFFINT]. 0b - Interrupt disabled 1b - Interrupt enabled
14 ERRMSK	Error Interrupt Mask Provides a mask for the Error interrupt ESR1[ERRINT]. 0b - Interrupt disabled 1b - Interrupt enabled
13 —	Reserved **NOTE** Writeable only if the module is disabled. Otherwise the access type is read-only.
12	Loopback Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPB	Configures FlexCAN to operate in Loopback mode. In this mode, FlexCAN performs an internal loopback that can be used for self-test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The RX CAN input pin is ignored and the TX CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field. It generates an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This field can be written only in Freeze mode; the module blocks it in other modes. **NOTE** In this mode, MCR[SRXDIS] cannot become 1, because it would impede the self-reception of a transmitted message. **NOTE** FDCTRL[TDCEN] and MCR[ETDCEN] must be 0 when this field is 1. 0b - Disabled 1b - Enabled
11 TWRNMSK	TX Warning Interrupt Mask Provides a mask for the TX Warning interrupt associated with the ESR1[TWRNINT] flag. When MCR[WRNEN] = 0, this field is read as 0. This field can be written only if MCR[WRNEN] = 1. 0b - Disabled 1b - Enabled
10 RWRNMSK	RX Warning Interrupt Mask Provides a mask for the RX Warning interrupt associated with the ESR1[RWRNINT] flag. When MCR[WRNEN] = 0, this field is read as 0. This field can be written only if MCR[WRNEN] = 1. 0b - Disabled 1b - Enabled
9 —	Reserved
8 ROM	Restricted Operation Mode Enables restricted operation mode. In restricted operation mode, FlexCAN is able to receive and acknowledge CAN messages, but it cannot send error frames and overload frames. If an error or overload condition is detected, FlexCAN treats it as a protocol exception and enters an integrating state. In restricted operation mode, FlexCAN does not increment or decrement the error counters. Besides, FlexCAN is able to transmit time reference messages of potential time masters in a network, but other types of frames must not be configured to be transmitted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field can be written in freeze mode only. 0b - Restricted operation is disabled 1b - Restricted operation is enabled
7 SMP	CAN Bit Sampling Determines the sampling mode of CAN bits at the RX input. This field can be written in Freeze mode only; the module blocks it in other modes. **NOTE** For proper operation, to write 1 to this field, you must guarantee a minimum value of two time quanta in CTRL1[PSEG1] (or CBT[EPSEG1]).This bit cannot become 1 when CAN FD is enabled (MCR[FDEN] = 1). 0b - One sample is used to determine the bit value. 1b - Three samples are used to determine the value of the received bit: the regular one (sample point) and two preceding samples. A majority rule is used.
6 BOFFREC	Bus Off Recovery Determines how FlexCAN recovers from Bus Off state. If 0, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If 1, automatic recovering from Bus Off is disabled. The module remains in Bus Off state until you write 1 to this field. If this field becomes 0 before 128 sequences of 11 recessive bits are detected on the CAN bus, Bus Off recovery happens as if this field had never become 1. If this field becomes 0 after 128 sequences of 11 recessive bits occurred, FlexCAN resynchronizes to the bus. It waits for 11 recessive bits before joining the bus. After this field becomes 0, it can become 1 again during Bus Off, but it will only be effective the next time the module enters Bus Off. If this field becomes 0 when the module is in Bus Off, writing 1 to this field is not effective for the current Bus Off recovery. **NOTE** See Bus Off in the CAN Protocol standard (ISO 11898-1:2015) for details. 0b - Enabled 1b - Disabled
5 TSYN	Timer Sync Enables a mechanism that resets the free-running timer each time a message is received in message buffer 0. This feature provides the means to synchronize multiple FlexCAN stations with a special "SYNC" message (that is, global network time). If MCR[RFEN] = 1 (Legacy RX FIFO enabled), the first available message buffer, according to CTRL2[RFFN], is used for timer synchronization instead of MB0. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 LBUF	Lowest Buffer Transmitted First Determines the ordering mechanism for message buffer transmission. When 1, MCR[LPRIOEN] does not affect the priority arbitration. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Buffer with highest priority is transmitted first. 1b - Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode Configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in Error Counter (ECR) are frozen, and the module operates in CAN Error Passive mode. Only messages acknowledged by another CAN station are received. If FlexCAN detects an unacknowledged message, it flags a BIT0 error without changing the receive error counter (ECR[RXERRCNT]), as if it is trying to acknowledge the message. FlexCAN acknowledges Listen-Only mode when ESR1[FLTCONF] = 1, indicating the Error Passive state. There can be some delay between the Listen-Only mode request and its acknowledgment. This field can be written in Freeze mode only; the module blocks it in other modes. 0b - Listen-Only mode is deactivated. 1b - FlexCAN module operates in Listen-Only mode.
2-0 PROPSEG	Propagation Segment Defines the length of the propagation segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation segment time = (PROPSEG + 1) × Time Quanta. One Time Quantum = one Sclock period.

52.5.2.4 Free-Running Timer (TIMER)

Offset

Register	Offset
TIMER	8h

Function

Represents a 16-bit free-running counter that the CPU can read and write. The timer starts from 0h after reset, counts linearly to FFFFh, and wraps around.

The CAN bit clock increments the timer, which defines the baud rate on the CAN bus. During a message transmission or reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable, Doze, Stop, Pretended Networking, and Freeze modes.

The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the timestamp entry in a message buffer after a successful reception or transmission of a message.

If CTRL1[TSYN] = 1, the timer is reset whenever a message is received in the first available message buffer, according to CTRL2[RFFN].

The CPU can write to this register anytime. However, if the write occurs simultaneously with the timer being reset by a reception in the first message buffer, the write value is discarded.

Reading this register affects the message buffer unlocking procedure (see Message buffer lock mechanism).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 TIMER	Timer Value Contains the free-running counter value.

52.5.2.5 RX Message Buffers Global Mask (RXMGMASK)

Offset

Register	Offset
RXMGMASK	10h

Function

Masks the filter bits of all RX message buffers, excluding MB14 and MB15, which have individual mask registers.

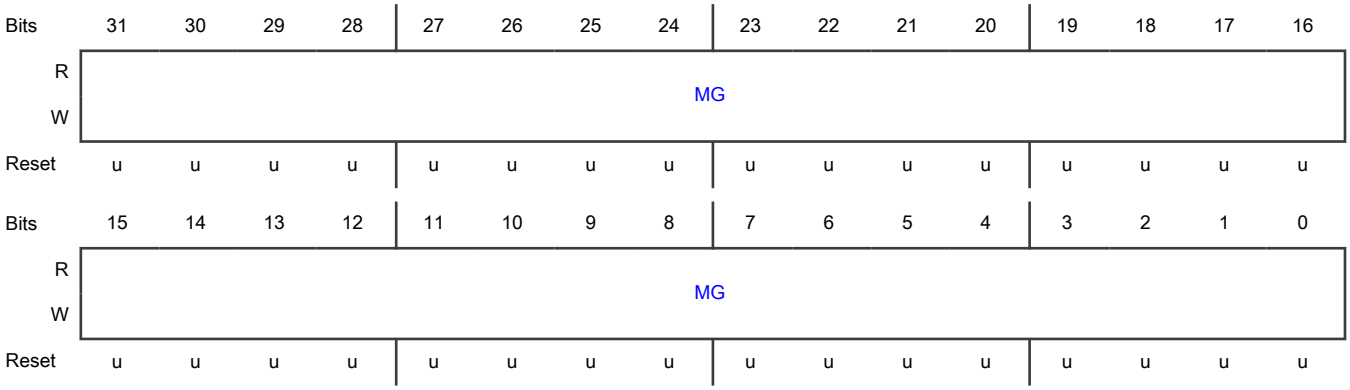
This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When MCR[IRMQ] is 0, RXMGMASK is always in effect. The bits in RXMGMASK[MG] mask the MB filter bits.
- When MCR[IRMQ] is 1, RXMGMASK has no effect. The bits in RXMGMASK[MG] do not mask the MB filter bits.

This register can only be written in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 MG	Global Mask for RX Message Buffers Masks the message buffer filter bits. The alignment with the ID word of the message buffer is imperfect. The two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the MB. The following table shows which MG bits mask each MB filter field.

SMB[RTR] [1]	CTRL2[RRS]	CTRL2[EACEN]	Message buffer filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	-	0	Note [2]	Note [3]	MG[28:0]	MG[31:29]
0	-	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	-	-	-	-	MG[31:0]
1	1	0	-	-	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

1. RTR bit of the incoming frame. It is saved into an auxiliary MB called RX serial message buffer (RX SMB).
2. If CTRL2[EACEN] is 0, the RTR bit of MB is never compared with the RTR bit of the incoming frame.
3. If CTRL2[EACEN] is 0, the IDE bit of MB is always compared with the IDE bit of the incoming frame.

0b - The corresponding bit in the filter is "don't care."

1b - The corresponding bit in the filter is checked.

52.5.2.6 Receive 14 Mask (RX14MASK)

Offset

Register	Offset
RX14MASK	14h

Function

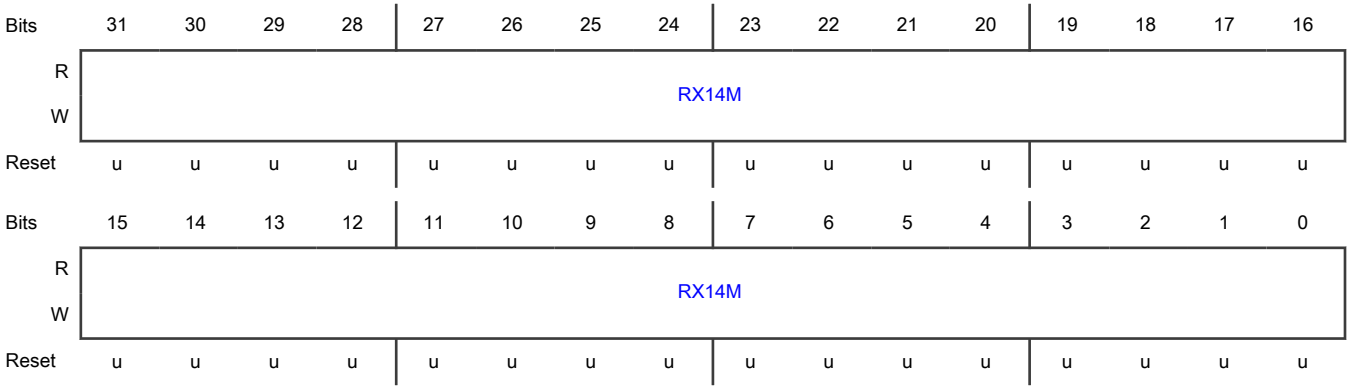
Masks the filter fields of MB14.

This register is located in RAM.

RX14MASK is provided for legacy application support. When MCR[IRMQ] = 1, RX14MASK has no effect.

This register can only be programmed when the module is in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 RX14M	RX Buffer 14 Mask Bits Masks the corresponding MB14 filter field in the same way that RX Message Buffers Global Mask (RXMGMASK) masks the filters of the other message buffers. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

52.5.2.7 Receive 15 Mask (RX15MASK)

Offset

Register	Offset
RX15MASK	18h

Function

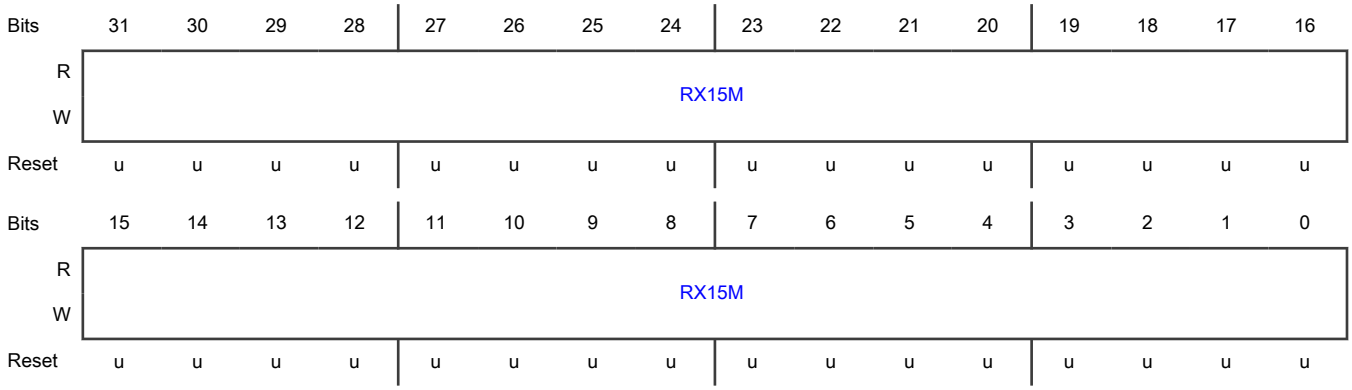
Masks the filter fields of MB15.

This register is located in RAM.

RX15MASK is provided for legacy application support. When MCR[IRMQ] = 1, RX15MASK has no effect.

This register can be programmed only when the module is in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-0 RX15M	RX Buffer 15 Mask Bits Masks the corresponding MB15 filter field in the same way that RX Message Buffers Global Mask (RXMGMASK) masks the filters of other message buffers. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

52.5.2.8 Error Counter (ECR)

Offset

Register	Offset
ECR	1Ch

Function

Contains error counters for received and transmitted messages.

TXERRCNT and RXERRCNT consider all errors in both CAN FD and non-FD message formats. TXERRCNT_FAST and RXERRCNT_FAST count only the errors that occur in the data phase of CAN FD frames that have BRS = 1.

The Fault Confinement state (ESR1[FLTCONF]) is updated based on TXERRCNT and RXERRCNT counters only. The rules for increasing and decreasing these counters are described in the CAN protocol and are entirely implemented in FlexCAN.

The basic rules for FlexCAN bus state transitions are:

- If the value of TXERRCNT or RXERRCNT becomes greater than or equal to 128, ESR1[FLTCONF] is updated to reflect Error Passive state.
- If the state of FlexCAN is Error Passive, and TXERRCNT or RXERRCNT decrements to a value less than 128 when the other already satisfies this condition, ESR1[FLTCONF] is updated to reflect Error Active state.
- If the value of TXERRCNT becomes greater than 255, ESR1[FLTCONF] is updated to reflect Bus Off state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in Bus Off, TXERRCNT is cascaded with another internal counter to count the occurrences of 11 consecutive recessive bits on the bus. TXERRCNT is reset to zero. It counts in a manner where the internal counter

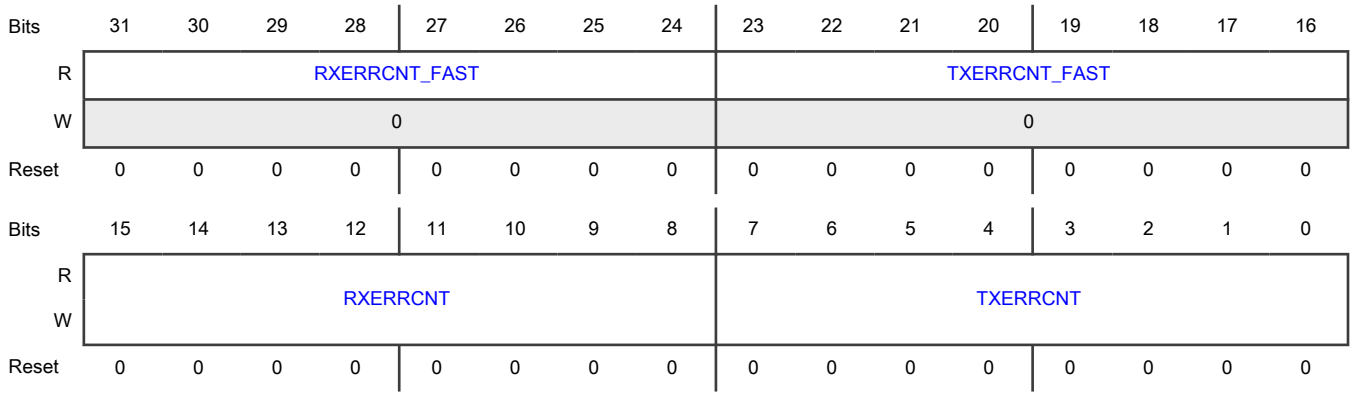
counts 11 such bits and then wraps around when incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, ESR1[FLTCONF] is updated to Error Active, and both error counters are reset to zero. Upon any instance of a dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value. The TXERRCNT_FAST counter is frozen during Bus Off.

- If only one node is operating during system startup, its TXERRCNT increases upon each attempted message transmission, as a result of acknowledge errors (indicated by ESR1[ACKERR]). After the transition to Error Passive state, TXERRCNT no longer increments upon acknowledge errors. The chip never goes into the Bus Off state.
- If RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected when being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to return to the Error Active state.
- TXERRCNT_FAST and RXERRCNT_FAST error counter values increment and decrement based on errors detected only in the data phase of CAN FD frames that have BRS = 1. These counters follow the same increment and decrement rules as TXERRCNT and RXERRCNT. These counters do not wrap around and get stuck at their maximum value (255). They stop counting and keep their values frozen when FlexCAN is in the Bus Off state. They are reset when FlexCAN leaves the Bus Off state and resume counting after FlexCAN returns to the Error Active state.
- When FlexCAN is in Pretended Networking mode, RXERRCNT and RXERRCNT_FAST keep counting errors, and error flags are stored. TXERRCNT and TXERRCNT_FAST preserve their values and do not change, because no transmission occurs in Pretended Networking mode. Error counters and error flags that changed values in Pretended Networking mode are updated in this register and in Error and Status 1 (ESR1) when FlexCAN returns to Normal mode. If FlexCAN is in Pretended Networking mode, the FAST error flags in ESR1 are not set.

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31-24 RXERRCNT_FAST	Receive Error Counter for Fast Bits Counts errors detected in the data phase of received CAN FD messages that have BRS = 1. This field is read-only except in Freeze mode, when the CPU can write an 8-bit zero value only.
23-16 TXERRCNT_FAST	Transmit Error Counter for Fast Bits Counts errors detected in the data phase of transmitted CAN FD messages that have BRS = 1. This field is read-only except in Freeze mode, when the CPU can write an 8-bit zero value only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 RXERRCNT	Receive Error Counter Counts all errors detected in received messages. This field is read-only except in Freeze mode, when the CPU can write to it.
7-0 TXERRCNT	Transmit Error Counter Counts all errors detected in transmitted messages. This field is read-only except in Freeze mode, when the CPU can write to it.

52.5.2.9 Error and Status 1 (ESR1)

Offset

Register	Offset
ESR1	20h

Function

Reports various error conditions detected in the reception and transmission of a CAN frame. This register provides status information about the chip, and is the source of some interrupts to the CPU. The reported error conditions are:

NOTE

Reading host can clear these fields.

- Errors detected in CAN frames of any format:
  - BIT1ERR
  - BIT0ERR
  - ACKERR
  - CRCERR
  - FRMERR
  - STFERR
- Errors detected in the data phase of CAN FD frames with the BRS bit set only:
  - BIT1ERR_FAST
  - BIT0ERR_FAST
  - CRCERR_FAST
  - FRMERR_FAST
  - STFERR_FAST

One or more error flags may report an error detected in a single CAN frame. To account for more error events occurring in subsequent frames when the CPU does not attempt to read this register, error reporting is cumulative.

Status flags:

- TXWRN
- RXWRN

- IDLE
- TX
- FLTCONF
- RX
- SYNCH

Interrupt flags:

- BOFFINT
- BOFFDONEINT
- ERRINT
- ERRINT_FAST
- WAKINT
- TWRNINT
- RWRNINT

The CPU should follow this procedure when servicing interrupt requests generated by these flags:

1. Read this register to capture all error condition and status flags. This action clears the respective flags that were set since the last read access.
2. Write 1 to clear the interrupt flag that triggered the interrupt request.
3. Write 1 to clear the ERROVR flag, if it is set.

Starting from all error flags cleared, a first error event sets either ERRINT or ERRINT_FAST (provided the corresponding mask bit is 1). If other error events in subsequent frames occur before the CPU serves the interrupt request, the ERROVR flag is set to indicate that errors from different frames have accumulated.

Table 348. CAN bus status

SYNCH	IDLE	TX	RX	FlexCAN state
0	0	0	0	Not synchronized to CAN bus
1	1	X	X	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BIT1ERR...	BIT0ERR...	0	CRCERR_...	FRMERR_...	STFERR_...	0		PTA	ATP	ERROVR	ERRINT_...	BOFFDON...	SYNCH	TWRNINT	RWRNINT
W									W1C	W1C	W1C	W1C	W1C		W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	WAKINT
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BIT1ERR_FAST	Fast Bit1 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit transmitted as recessive is received as dominant.
30 BIT0ERR_FAST	Fast Bit0 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit transmitted as dominant is received as recessive.
29 —	Reserved
28 CRCERR_FAST	Fast Cyclic Redundancy Check Error Flag Indicates that the receiver node has detected a CRC error in the CRC field of CAN FD frames that have BRS = 1. This error means that the calculated CRC is different from the received CRC. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
27	Fast Form Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRMERR_FAST	Indicates whether the receiver node has detected a form error in the data phase of CAN FD frames that have BRS = 1. This error means that a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A form error occurred since last read of this register.
26 STFERR_FAST	Fast Stuffing Error Flag Indicates that a stuffing error has been detected in the data phase of CAN FD frames that have BRS = 1. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.
25-24 —	Reserved
23 PTA	Passive to Active State If this field is 1, controller is transitioned from passive to active error state. If this field is 0, controller is not transitioned from passive to active error state. Interrupt signals are asserted if enabled from FLTCONF_IE when the controller is transitioned from passive to active error state. 0b - Does not transition 1b - Transitions
22 ATP	Active to Passive State If this field is 1, controller is transitioned from active to passive error state. If this field is 0, controller is not transitioned from active to passive error state. Interrupt signals are asserted if enabled from FLTCONF_IE when the controller is transitioned from active to passive error state. 0b - Does not transition 1b - Transitions
21 ERROVR	Error Overrun Flag Indicates that an error condition occurred when any error flag is already set. 0b - No overrun 1b - Overrun
20 ERRINT_FAST	Fast Error Interrupt Flag Indicates that at least one error flag detected in the data phase of CAN FD frames that have BRS = 1 (BIT1ERR_FAST, BIT0ERR_FAST, CRCERR_FAST, FRMERR_FAST, or STFERR_FAST) is set. If CTRL2[ERRMSK_FAST] = 1, an interrupt is generated to the CPU. 0b - No such occurrence. 1b - Error flag set in the data phase of CAN FD frames that have BRS = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 BOFFDONEINT	Bus Off Done Interrupt Flag Indicates whether ECR[TXERRCNT] has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If CTRL2[BOFFDONEMSK] = 1, an interrupt is generated to the CPU. 0b - No such occurrence 1b - FlexCAN module has completed Bus Off process.
18 SYNCH	CAN Synchronization Status Flag Indicates whether FlexCAN is synchronized to the CAN bus and able to participate in the communication process. FlexCAN sets and clears this flag. See the table in Error and Status 1 (ESR1). 0b - Not synchronized 1b - Synchronized
17 TWRNINT	TX Warning Interrupt Flag Indicates whether TX error counter changed from less than 96 to greater than or equal to 96. If MCR[WRNEN] = 1, this flag is set when the TXWRN flag transitions from 0 to 1, meaning that the TX error counters reached 96. If CTRL1[TWRNMSK] = 1, an interrupt is sent to the CPU. When MCR[WRNEN] = 0, this flag is masked. The CPU must clear this flag before writing 0 to MCR[WRNEN]. Otherwise, this flag is set when MCR[WRNEN] = 1 again. Writing 0 has no effect. This flag is not generated when in the Bus Off state. This flag is not updated during Freeze mode. When FlexCAN returns to Normal mode from Pretended Network mode (see Receive process in Pretended Networking mode), this bit is not updated. 0b - No such occurrence 1b - TX error counter changed from less than 96 to greater than or equal to 96.
16 RWRNINT	RX Warning Interrupt Flag Indicates whether the RX error counter changed from less than 96 to greater than or equal to 96. If MCR[WRNEN] = 1, this flag is set when the RXWRN flag transitions from 0 to 1, meaning that the RX error counters reached 96. If CTRL1[RWRNMSK] = 1, an interrupt is sent to the CPU. When MCR[WRNEN] = 0, this flag is masked. The CPU must clear this flag before writing 0 to MCR[WRNEN]. Otherwise, this flag is set when MCR[WRNEN] = 1 again. Writing 0 has no effect. This flag is not updated during Freeze mode. When FlexCAN returns to Normal mode from Pretended Network mode (see Receive process in Pretended Networking mode), this bit is updated to reflect the RX error counter state. 0b - No such occurrence 1b - RX error counter changed from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This flag is updated when FlexCAN returns to Normal mode from Pretended Network mode. **NOTE** A transmitter does not set this flag for an arbitration field or ACK slot. It is not set for a node sending a error passive flag that detects dominant bits. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
14 BIT0ERR	Bit0 Error Flag Indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. This field is updated when FlexCAN returns to Normal mode from Pretended Network mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error Flag Indicates whether the transmitter node has detected an acknowledge error. This error means that a dominant bit has not been detected during the ACK SLOT. This flag is updated when FlexCAN returns to Normal mode from Pretended Network mode. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error Flag Indicates whether the receiver node has detected a cyclic redundancy check (CRC) error either in a non-FD message or in the arbitration or data phase of a frame in CAN FD format. This error means that the calculated CRC is different from the received. This flag is updated when FlexCAN returns to Normal mode from Pretended Network mode. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
11 FRMERR	Form Error Flag Indicates whether a form error has been detected in a non-FD message or in the arbitration or data phase of an FD message by the receiver node. This error means that a fixed-form field contains at least one illegal bit. This flag is updated when FlexCAN returns to Normal mode from Pretended Network mode. After a read operation, the field's value clears to 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No error 1b - Error occurred since last read of this register.
10 STFERR	Stuffing Error Flag Indicates that a stuffing error has been detected in a non-FD message or in the arbitration or data phase of an FD message by the receiver node. This flag is updated when FlexCAN returns to Normal mode from Pretended Network mode. After a read operation, the field's value clears to 0. 0b - No error 1b - Error occurred since last read of this register.
9 TXWRN	TX Error Warning Flag Indicates when repetitive errors occur during message transmission. Only the value of ECR[TXERRCNT] affects this flag. This flag is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - TXERRCNT is 96 or greater.
8 RXWRN	RX Error Warning Flag Indicates when repetitive errors occur during message reception. Only the value of ECR[RXERRCNT] affects this flag. This flag is not updated during Freeze mode. Additionally, this flag is updated when FlexCAN returns to Normal mode from Pretended Networking mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - RXERRCNT is greater than or equal to 96.
7 IDLE	Idle Indicates whether CAN bus is in IDLE state. See Table 348. 0b - Not IDLE 1b - IDLE
6 TX	FlexCAN In Transmission Indicates whether FlexCAN is transmitting a message. See Table 348. 0b - Not transmitting 1b - Transmitting
5-4 FLTCONF	Fault Confinement State Indicates the confinement state of FlexCAN.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If CTRL1[LOM] = 1, after a delay that depends on the CAN bit timing, this field indicates Error Passive. The same delay affects the way that this field reflects an update to ECR register by the CPU. It may be necessary to wait up to one CAN bit time for coherence to be restored. Soft reset affects this field, but if CTRL1[LOM] = 1, its reset value lasts for only one CAN bit. After that time, this field reports Error Passive. 00b - Error Active 01b - Error Passive 1xb - Bus Off
3 RX	FlexCAN in Reception Flag Indicates whether FlexCAN is receiving a message. See the table in Error and Status 1 (ESR1). 0b - Not receiving 1b - Receiving
2 BOFFINT	Bus Off Interrupt Flag Indicates whether FlexCAN has entered Bus Off state. If CTRL1[BOFFMSK]) = 1, an interrupt is generated to the CPU. Writing 0 to this field has no effect. 0b - No such occurrence. 1b - FlexCAN module entered Bus Off state.
1 ERRINT	Error Interrupt Flag Indicates that at least one of the error flags (ESR1[BIT1ERR], ESR1[BIT0ERR], ESR1[ACKERR], ESR1[CRCERR], ESR1[FRMERR], or ESR1[STFERR]) is set. If the corresponding mask CTRL1[ERRMSK] = 1, an interrupt is generated to the CPU. Writing 0 to this field has no effect. 0b - No such occurrence. 1b - Indicates setting of any error flag in the Error and Status register.
0 WAKINT	Wake-up Interrupt Flag Generates an interrupt to the CPU when a recessive-to-dominant transition is detected on the CAN bus and MCR[WAKMSK] = 1. This field applies when FlexCAN is in low-power mode during Self Wake-up: • Doze mode • Stop mode When MCR[SLFWAK] = 0, this flag is masked. The CPU must clear this flag before disabling the field. Otherwise, it is set when MCR[SLFWAK] becomes 1 again. Writing 0 has no effect. 0b - No such occurrence. 1b - Indicates that a recessive-to-dominant transition was received on the CAN bus.

52.5.2.10 Interrupt Masks 1 (IMASK1)

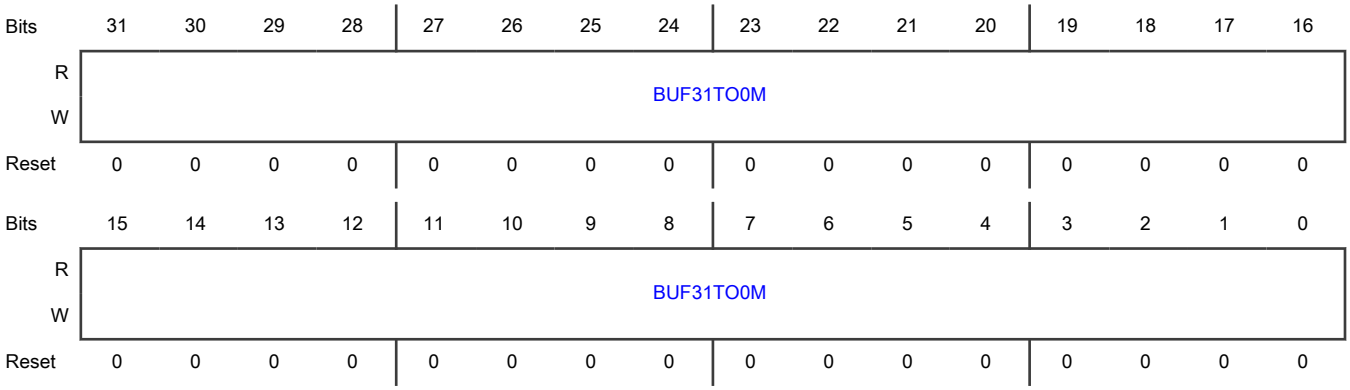
Offset

Register	Offset
IMASK1	28h

Function

Masks interrupt flags. This register allows any of the 32 message buffer interrupts to be enabled or disabled for MB31–MB0. It contains one interrupt mask bit per buffer. This configuration allows the CPU to determine which buffer generates an interrupt after a successful transmission or reception when the corresponding Interrupt Flags 1 (IFLAG1) flag is set.

Diagram



Fields

Field	Function
31-0 BUF31TO0M	Buffer MBi Mask Enables or disables the corresponding FlexCAN message buffer interrupt for MB31–MB0. **NOTE** If the corresponding Interrupt Flags 1 (IFLAG1) flag is set, writing 1 or 0 to a field in IMASK1 can set or clear an interrupt request. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

52.5.2.11 Interrupt Flags 1 (IFLAG1)

Offset

Register	Offset
IFLAG1	30h

Function

Contains the flags for the 32 message buffer interrupts for MB31–MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG1 bit. If the corresponding Interrupt Masks 1 (IMASK1) bit is set, an interrupt is generated. There is an exception when DMA for Legacy RX FIFO is enabled, as described below.

The BUF7I–BUF5I flags also represent Legacy FIFO interrupts when the Legacy RX FIFO is enabled. When MCR[RFEN] is 1 and MCR[DMA] is 0, the function of the eight least significant interrupt flags changes:

- BUF7I, BUF6I, and BUF5I indicate operating conditions of the Legacy FIFO.
- BUF4I–BUF1I fields are reserved.
- BUF0I empties the Legacy FIFO.

Before writing 1 to MCR[RFEN], the CPU must service the IFLAG flags set in the Legacy RX FIFO region; see Legacy RX FIFO. Otherwise, these IFLAG flags mistakenly show the related message buffers now belonging to Legacy FIFO as having contents to be serviced. When MCR[RFEN] is 0, the Legacy FIFO flags must be cleared. The same care must be taken when a CTRL2[RFFN] value is selected, extending Legacy RX FIFO filters beyond MB7. For example, when RFFN is 8h, Legacy RX FIFO filters occupy the MB23–MB0 range, and related IFLAG flags must be cleared.

When MCR[RFEN] and MCR[DMA] are 1 (DMA feature for Legacy RX FIFO enabled), the function of the eight least significant interrupt flags (BUF7I–BUF0I) changes to support DMA operation. BUF7I, BUF6I, and BUF4I–BUF1I are not used.

BUF5I indicates the operating condition of the Legacy FIFO, and BUF0I empties the Legacy FIFO. Moreover, BUF5I does not generate a CPU interrupt, but it does generate a DMA request. IMASK1 bits in the Legacy RX FIFO region are not considered when bit MCR[DMA] = 1. In addition, the CPU must not clear the BUF5I flag when DMA is enabled. Before writing 1 to MCR[DMA], the CPU must service the IFLAG flags set in the Legacy RX FIFO region. When MCR[DMA] is 0, the Legacy FIFO must be empty. Legacy FIFO must be disabled when MCR[FDEN] = 1.

Before updating MCR[MAXMB], the CPU must service the IFLAG1 flags whose MB value is greater than the MCR[MAXMB] to be updated. Otherwise, those flags remain set and are inconsistent with the number of message buffers available.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BUF31TO8I															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF31TO8I								BUF7I	BUF6I	BUF5I	BUF4TO1I				BUF0I
W	W1C								W1C	W1C	W1C	W1C				W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 BUF31TO8I	Buffer MBi Interrupt Flags the corresponding FlexCAN message buffer interrupt for MB31–MB8. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt or Legacy RX FIFO Overflow Flags the interrupt for MB7 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, this flag represents a Legacy RX FIFO overflow. In this case, the flag indicates that a message was lost because the Legacy RX FIFO is full. When the Legacy RX FIFO is full and a message buffer captures the message, this flag is not set. 0b - No occurrence of MB7 completing transmission or reception, or no FIFO overflow. 1b - MB7 completed transmission or reception, or FIFO overflow.
6 BUF6I	Buffer MB6 Interrupt or Legacy RX FIFO Warning Flags the interrupt for MB6 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, this flag represents a Legacy RX FIFO warning. In this case, the flag indicates when the number of unread messages within the Legacy RX FIFO is increased to five from four due to the reception of a new message. In other words, the Legacy RX FIFO is almost full. If this flag is cleared when there are more than four unread messages, it does not set again until the number of unread messages in the Legacy RX FIFO decreases to four or fewer. 0b - No occurrence of MB6 completing transmission or reception, or FIFO not almost full. 1b - MB6 completed transmission or reception, or FIFO almost full.
5 BUF5I	Buffer MB5 Interrupt or Frames available in Legacy RX FIFO Flags the interrupt for MB5 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). **NOTE** When a CPU write changes the value of MCR[RFEN], FlexCAN clears this flag. When MCR[RFEN] = 1, the BUF5I flag represents frames available in Legacy RX FIFO. In this case, the flag indicates that at least one frame is available to be read from the Legacy RX FIFO. When MCR[DMA] = 1, this flag generates a DMA request. The CPU must not clear this field by writing 1 to BUF5I. 0b - No occurrence of completed transmission or reception, or no frames available 1b - MB5 completed transmission or reception, or frames available
4-1 BUF4TO1I	Buffer MBi Interrupt or Reserved Flags the interrupts for MB4–MB1 when MCR[RFEN] = 0 (Legacy RX FIFO disabled).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When a CPU write changes the value of MCR[RFEN], FlexCAN clears these flags. When MCR[RFEN] = 1, the BUF4TO1I flags are reserved. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.
0 BUF0I	Buffer MB0 Interrupt or Clear Legacy FIFO bit Flags the interrupt for MB0 when MCR[RFEN] = 0 (Legacy RX FIFO disabled). If MCR[RFEN] = 1, this field is used to trigger the clear Legacy FIFO operation. This operation empties the Legacy FIFO contents. Before performing this operation, the CPU must service all Legacy FIFO-related IFLAG flags. When MCR[DMA] = 1, this operation also clears the BUF5I flag, aborting the DMA request. The clear Legacy FIFO operation occurs when the CPU writes 1 to BUF0I. This operation is only allowed in Freeze mode; the module blocks it in other conditions. 0b - MB0 has no occurrence of successfully completed transmission or reception. 1b - MB0 has successfully completed transmission or reception.

52.5.2.12 Control 2 (CTRL2)

Offset

Register	Offset
CTRL2	34h

Function

Complements Control 1 (CTRL1), providing control bits for memory write-access in Freeze mode. This register extends Legacy FIFO filter quantity, and adjusts the operation of internal FlexCAN processes such as matching and arbitration.

Soft reset does not affect the contents of this register.

Table 349 shows how the value of CTRL2[RFFN] determines the Legacy RX FIFO filter structure.

Table 349. Possible Legacy RX FIFO filter structures

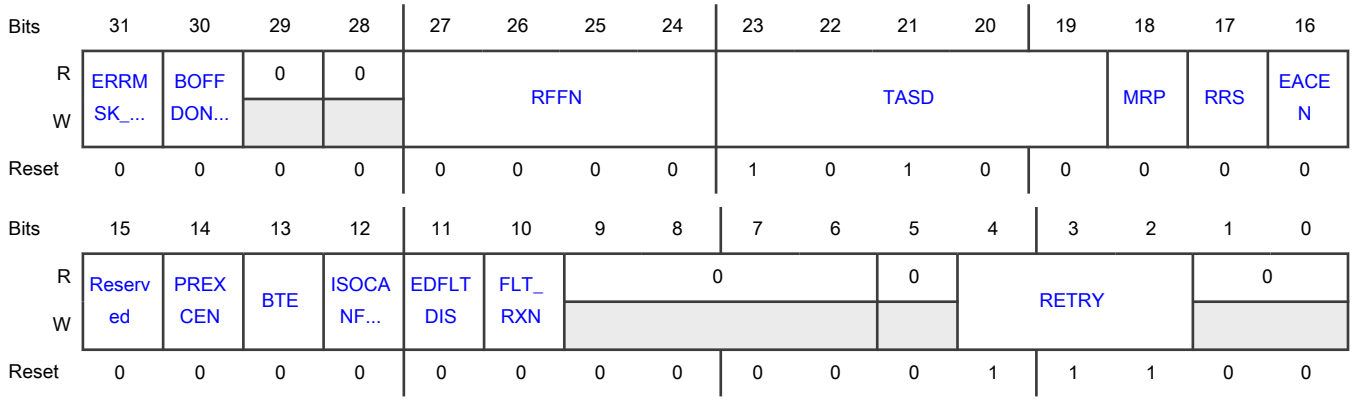
RFFN[3:0]	Number of Legacy RX FIFO filter elements	Message buffers occupied by Legacy RX FIFO and ID filter table	Remaining available message buffers	Legacy RX FIFO ID filter table elements affected by RX individual masks	Legacy RX FIFO ID filter table elements affected by Legacy RX FIFO global mask
0h	8	MB 0–7	MB 8–31	Elements 0–7	None
1h	16	MB 0–9	MB 10–31	Elements 0–9	Elements 10–15

Table continues on the next page...

Table 349. Possible Legacy RX FIFO filter structures (continued)

RFFN[3:0]	Number of Legacy RX FIFO filter elements	Message buffers occupied by Legacy RX FIFO and ID filter table	Remaining available message buffers	Legacy RX FIFO ID filter table elements affected by RX individual masks	Legacy RX FIFO ID filter table elements affected by Legacy RX FIFO global mask
2h	24	MB 0–11	MB 12–31	Elements 0–11	Elements 12–23
3h	32	MB 0–13	MB 14–31	Elements 0–13	Elements 14–31
4h	40	MB 0–15	MB 16–31	Elements 0–15	Elements 16–39
5h	48	MB 0–17	MB 18–31	Elements 0–17	Elements 18–47
6h	56	MB 0–19	MB 20–31	Elements 0–19	Elements 20–55
7h	64	MB 0–21	MB 22–31	Elements 0–21	Elements 22–63
8h	72	MB 0–23	MB 24–31	Elements 0–23	Elements 24–71
9h	80	MB 0–25	MB 26–31	Elements 0–25	Elements 26–79
Ah	88	MB 0–27	MB 28–31	Elements 0–27	Elements 28–87
Bh	96	MB 0–29	MB 30–31	Elements 0–29	Elements 30–95
Ch	104	MB 0–31	none	Elements 0–31	Elements 32–103

Diagram



Fields

Field	Function
31 ERRMSK_FAST	Error Interrupt Mask for Errors Detected in the Data Phase of Fast CAN FD Frames Enables the ESR1[ERRINT_FAST] interrupt. 0b - Disable 1b - Enable
30	Bus Off Done Interrupt Mask Enables the Bus Off Done interrupt, ESR1[BOFFDONEINT].

Table continues on the next page...

Table continued from the previous page...

Field	Function
BOFFDONEMSK	0b - Disable 1b - Enable
29 —	Reserved
28 —	Reserved
27-24 RFFN	Number of Legacy Receive FIFO Filters Defines the number of Receive Legacy FIFO filters, as shown in Table 349. The chip determines the maximum selectable number of filters. Do not program this field with values that cause the number of message buffers occupied by Legacy RX FIFO and Legacy RX FIFO ID Filter to exceed MCR[MAXMB]. MCR[MAXMB] defines the number of message buffers present. This field can only be written in Freeze mode; the module blocks it in other modes. Each group of eight filters occupies a memory space equivalent to two message buffers. The more filters are implemented, the fewer message buffers are available. The Legacy RX FIFO occupies the memory space originally reserved for MB5–MB0. This field should be programmed with a value corresponding to a number of filters less than the number of available memory words. The number of available words can be calculated as follows: (SETUP_MB - 6) × 4 Where SETUP_MB is the smaller of the parameter NUMBER_OF_MB and MCR[MAXMB]. The number of remaining message buffers available is: (SETUP_MB - 8) - (RFFN × 2) If the number of Legacy RX FIFO filters programmed through RFFN exceeds the SETUP_MB value (memory space available), the exceeding ones are not functional. **NOTE** • The number of the last remaining available message buffers is the smaller of (NUMBER_OF_MB - 1) and MCR[MAXMB]. • If RX Individual Mask registers are not enabled, the Legacy RX FIFO Global Mask affects all Legacy RX FIFO filters.
23-19 TASD	Transmission Arbitration Start Delay Indicates by how many CAN bits the transmission arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See TX arbitration start delay for details. This field can be written only in Freeze mode; the module blocks it in other modes.
18 MRP	Message Buffers Reception Priority Sets the priority for the matching process. This field can be written only in Freeze mode; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Matching starts from Legacy RX FIFO or Enhanced RX FIFO and continues on message buffers. 1b - Matching starts from message buffers and continues on Legacy RX FIFO or Enhanced RX FIFO.
17 RRS	Remote Request Storing Determines what the module does with a remote request. The remote request frame is submitted to a matching process. If this field is 1, the frame is stored in the corresponding message buffer in the same fashion as a data frame. No automatic remote response frame is generated. If this field is 0, an automatic remote response frame is generated if a message buffer with CODE = 1010b is found with the same ID. You can only write to this field in Freeze mode. The module blocks it in other modes. 0b - Generated 1b - Stored
16 EACEN	Entire Frame Arbitration Field Comparison Enable for RX Message Buffers Controls the comparison of IDE and RTR fields within RX message buffer filters with their corresponding bits in the incoming frame by the matching process. If enabled, the IDE and RTR fields of the RX message buffer are compared to their corresponding bits within the incoming frame (mask bits apply). If disabled, the IDE field of the RX message buffer filter is always compared and RTR is never compared despite mask bits. This field does not affect matching for Legacy RX FIFO or Enhanced RX FIFO. You can only write to this field in Freeze mode; the module blocks it in other modes. 0b - Disable 1b - Enable
15 —	Reserved When writing to this field, always write the reset value.
14 PREXCEN	Protocol Exception Enable Enables the protocol exception feature. You can only write to this field in Freeze mode. **NOTE** See Protocol exception event in the CAN Protocol standard (ISO 11898-1:2015) for details. 0b - Disabled 1b - Enabled
13 BTE	Bit Timing Expansion Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the use of Enhanced CAN Bit Timing Prescalers (EPRS), Enhanced Data Phase CAN Bit Timing (EDCBT), and Enhanced Nominal CAN Bit Timing (ENCBT) to configure the CAN bit timing segments, instead of using the bit timing fields of CAN Bit Timing (CBT), CAN FD Bit Timing (FDCBT), and Control 1 (CTRL1). If this field is 1: • CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] are read as zero. A write operation to these fields has no effect. • CBT[EPRESDIV], CBT[ERJW], CBT[EPROPSEG], CBT[EPSEG1], and CBT[EPSEG2], and the corresponding fields in CAN FD Bit Timing (FDCBT), are read as zero. A write operation to these fields has no effect. • ETDC[ETDCOFF], ETDC[ETDCEN], ETDC[ETDCFAIL], and ETDC[ETDCVAL] are used by FlexCAN instead of FDCTRL[TDCOFF], FDCTRL[TDCEN], FDCTRL[TDCFAIL], and FDCTRL[TDCVAL]. These fields are read as zero, and a write operation to them has no effect. • ETDC[TDMDIS] can be used to disable transceiver delay measurement. 0b - Disable 1b - Enable
12 ISOCANFDEN	ISO CAN FD Enable Enables the CAN FD protocol according to ISO specification (ISO 11898-1:2015) (see CAN FD ISO compliance). When disabled, FlexCAN operates using the non-ISO CAN FD protocol. You can only write to this field in Freeze mode. **NOTE** FlexCAN is able to transmit FD frame format according to CAN Protocol standard (ISO 11898-1:2015). 0b - Disable 1b - Enable
11 EDFLTDIS	Edge Filter Disable Disables the edge filter used during the Bus Integration state. When the Edge Filter is enabled, two consecutive nominal time quanta with dominant bus states are required to detect an edge that causes synchronization. When synchronization occurs, the counting of the sequence of 11 consecutive recessive bits is restarted. The edge filter prevents dominant pulses that are shorter than a nominal bit time (present during the data phase of an FD frame) from being mistaken for an idle condition. You can only write to this field in Freeze mode. **NOTE** See Bus Integration state in the CAN Protocol standard (ISO 11898-1:2015) for details. 0b - Enabled 1b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 FLT_RXN	Fault reaction Indicates fault reaction during the functional mode. Setting this bit makes the TX pin to recessive state. 0b - Fault reaction is disabled 1b - Fault reaction is enabled
9-6 —	Reserved
5 —	Reserved
4-2 RETRY	Number of Retransmission Requests The message is discarded in case of a re-transmission counter exhausted. **NOTE** You can only write to this field in Freeze mode. 000b - No retransmission 001b - Count of re-transmission attempts 010b - Count of re-transmission attempts 011b - Count of re-transmission attempts 100b - Count of re-transmission attempts 101b - Count of re-transmission attempts 110b - Count of re-transmission attempts 111b - Unlimited number of retransmission
1-0 —	Reserved

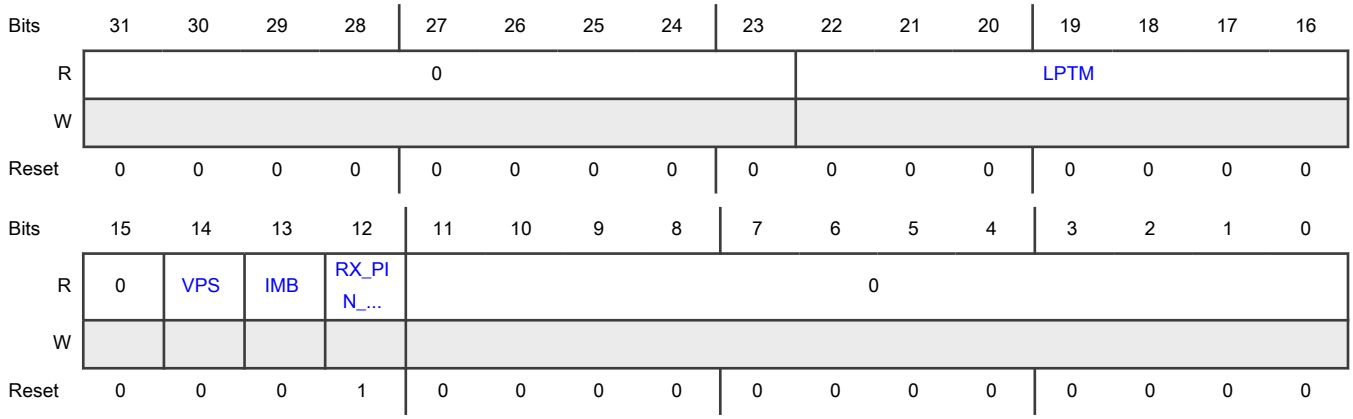
52.5.2.13 Error and Status 2 (ESR2)

Offset

Register	Offset
ESR2	38h

Function
Reports general status information.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 LPTM	Lowest Priority TX Message Buffer Indicates the lowest number inactive message buffer when ESR2[VPS] = 1 (see ESR2[IMB]). If no message buffer is inactive, the message buffer indicated depends on the value of CTRL1[LBUF]. If CTRL1[LBUF] = 0, the message buffer indicated is the one with the greatest arbitration value (see Highest-priority message buffer first). If CTRL1[LBUF] = 1, the message buffer indicated is the active TX message buffer with the highest number. If a TX message buffer is being transmitted, it is not considered in the LPTM calculation. If ESR2[IMB] is not 0 and a frame is transmitted successfully, the value of LPTM is updated with its message buffer number.
15 —	Reserved
14 VPS	Valid Priority Status Indicates whether the contents of ESR2[IMB] and ESR2[LPTM] are valid. It becomes 1 upon every complete TX arbitration process, unless the CPU writes to the Control and Status word of a message buffer already scanned. In other words, it is behind the TX Arbitration Pointer, during the TX arbitration process. If there is no inactive message buffer and only one TX message buffer that is being transmitted, this field remains 0. This field becomes 0 upon the start of every TX arbitration process or upon a write to the Control and Status word of any message buffer. **NOTE** No CPU write to the Control and Status of a message buffer that the abort mechanism blocks affects this field. When MCR[AEN] = 1, the abort code write to the Control and Status of an MB being transmitted (pending abort) is blocked. Any write attempt to a TX MB with its IFLAG flag set is also blocked. 0b - Invalid 1b - Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 IMB	Inactive Message Buffer Indicates whether any message buffer is inactive (CODE field is either 1000b or 0b) when ESR2[VPS] = 1. This field becomes 1 when: • A lowest-priority TX message buffer (ESR2[LPTM]) is found and it is inactive during arbitration. • This field is not 1, and a frame is transmitted successfully. This field always becomes 0 at the start of arbitration (see Arbitration process). If a message buffer is successfully transmitted and this field is 0 (no inactive message buffer), ESR2[VPS] and this field both become 1. The index related to the MB transmitted is loaded into ESR2[LPTM]. In this case, the value of ESR2[LPTM] is the number of the first inactive message buffer. 0b - Message buffer indicated by ESR2[LPTM] is not inactive. 1b - At least one message buffer is inactive.
12 RX_PIN_ST	RX Pin Status Indicates the current state of the RX pin. This bit is affected by soft reset. **NOTE** IP takes a few clock cycles to propagate the value of RX pin to this status bit. 0b - RX pin is in the dominant state 1b - RX pin is in a recessive state
11-0 —	Reserved

52.5.2.14 Cyclic Redundancy Check (CRCR)

Offset

Register	Offset
CRCR	44h

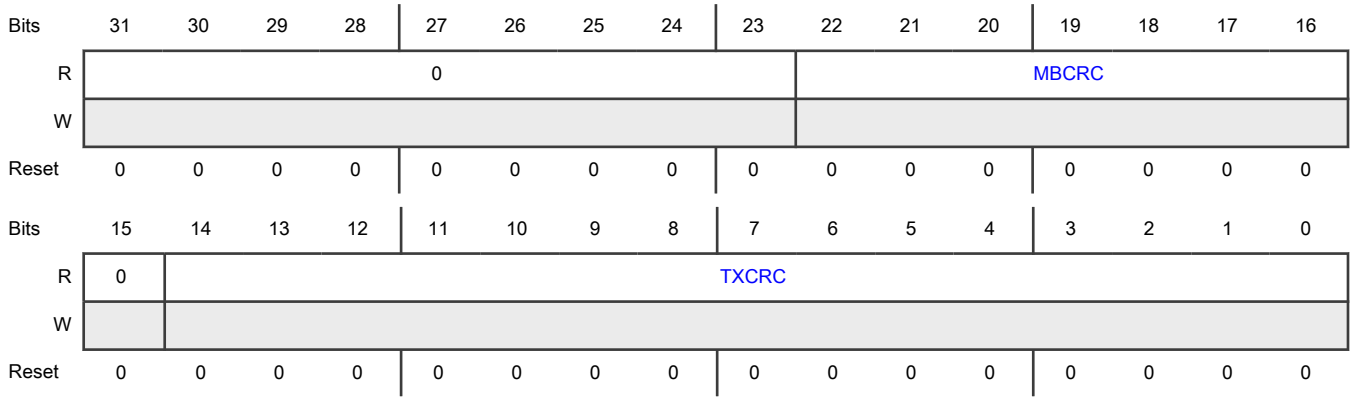
Function

Provides information about the CRC of transmitted messages for non-FD messages. This register only reports the 15 low-order bits of CRC calculations for messages in CAN FD format that require either 17 or 21 bits. For CAN FD format frames, you must use the CAN FD CRC (FDCRC). This register is updated at the same time that the TX interrupt flag is set.

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 MBCRC	CRC Message Buffer Indicates the number of the message buffer corresponding to the value in CRCR[TXCRC].
15 —	Reserved
14-0 TXCRC	Transmitted CRC value Indicates the CRC value of the last transmitted message for non-FD frames. For FD frames, CRC value is reported in CAN FD CRC (FDCRC).

52.5.2.15 Legacy RX FIFO Global Mask (RXFGMASK)

Offset

Register	Offset
RXFGMASK	48h

Function

Masks the Legacy RX FIFO ID filter table elements that do not have a corresponding RXIMR according to CTRL2[RFFN], when Legacy RX FIFO is enabled.

This register is located in RAM.

You can only write to this register in Freeze mode; the module blocks it in other modes.

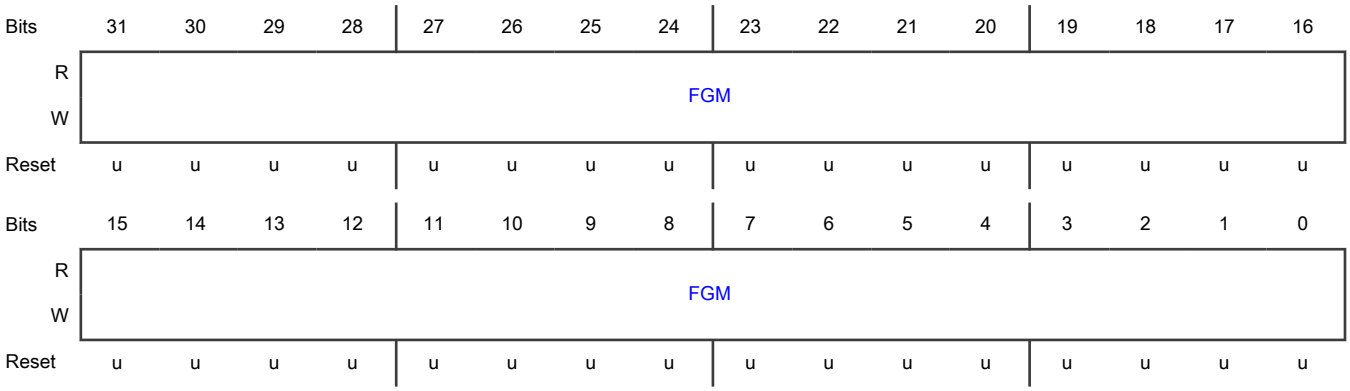
The following table shows how the FGM bits correspond to each IDAF field.

Table 350. Correspondence of Legacy RX FIFO global mask bits to IDF fields

Legacy RX FIFO ID filter table elements format (MCR[IDAM])	Identifier acceptance filter fields					
	RTR	IDE	RXIDA	RXIDB [1]	RXIDC [2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	—	—	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	—	FGM[29:16], FGM[13:0]		—
C	—	—		—	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

1. If MCR[IDAM] is equivalent to format B, only the 14 most significant bits of the identifier of the incoming frame are compared with the Legacy RX FIFO filter.
2. If MCR[IDAM] is equivalent to format C, only the eight most significant bits of the identifier of the incoming frame are compared with the Legacy RX FIFO filter.

Diagram



Fields

Field	Function
31-0 FGM	Legacy RX FIFO Global Mask Bits Masks the ID filter table elements bits in a perfect alignment. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

52.5.2.16 Legacy RX FIFO Information (RXFIR)

Offset

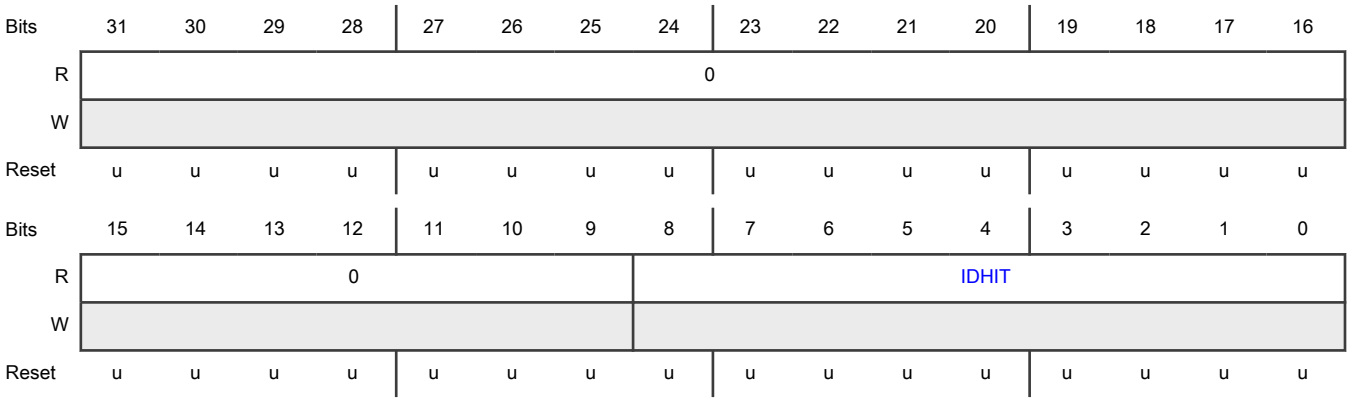
Register	Offset
RXFIR	4Ch

Function

Provides information about Legacy RX FIFO.

This register is the port through which the CPU accesses the output of the Legacy RXFIR FIFO located in RAM. FlexCAN writes to the Legacy RXFIR FIFO when a new message is moved into the Legacy RX FIFO. Also, its output is updated whenever the output of the Legacy RX FIFO is updated with the next message. See Legacy RX FIFO for instructions on reading this register.

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 IDHIT	Identifier Acceptance Filter Hit Indicator Indicates which Identifier Acceptance filter that the received message hit in the output of the Legacy RX FIFO. If multiple filters match the incoming message ID, the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only when IFLAG1[BUF5I] is set.

52.5.2.17 CAN Bit Timing (CBT)

Offset

Register	Offset
CBT	50h

Function

Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1). EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW are extended versions of CTRL1[PRESDIV], CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW] respectively.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

CBT[BTF] selects the use of the timing variables defined in this register.

When the CAN FD feature is enabled (MCR[FDEN] = 1), always write 1 to CBT[BTF].

Soft reset does not affect the contents of this register.

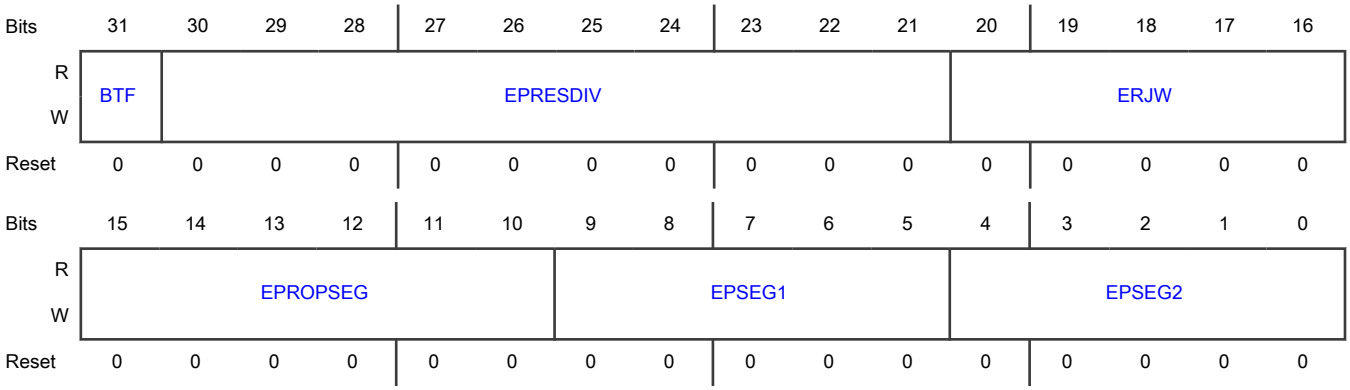
NOTE

Ensure that bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1:2015).

NOTE

If CTRL2[BTE] = 1, EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 are read as zero. A write operation to them has no effect.

Diagram



Fields

Field	Function
31 BTF	Bit Timing Format Enable Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW. These fields replace the CAN bit timing variables defined in Control 1 (CTRL1). This field can be written in Freeze mode only. 0b - Disable 1b - Enable
30-21 EPRESDIV	Extended Prescaler Division Factor Defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (EPRESDIV + 1)
20-16 ERJW	Extended Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time when CBT[BTF] = 1. Otherwise, it has no effect. It extends the CTRL1[RJW] value range.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = ERJW + 1. One Time Quantum = one Sclock period.
15-10 EPROPSEG	Extended Propagation Segment Defines the length of the propagation segment in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PROPSEG] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time Quanta. One Time Quantum = one Sclock period.
9-5 EPSEG1	Extended Phase Segment 1 Defines the length of phase segment 1 in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PSEG1] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.
4-0 EPSEG2	Extended Phase Segment 2 Defines the length of phase segment 2 in the bit time when CBT[BTF] = 1, otherwise it has no effect. It extends the CTRL1[PSEG2] value range. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

52.5.2.18 External Timer (ET)

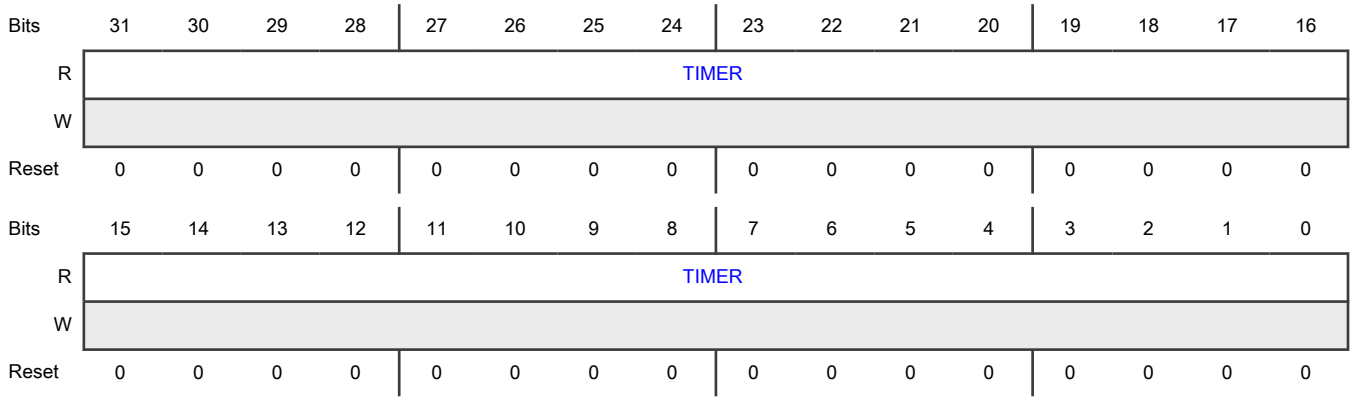
Offset

Register	Offset
ET	78h

Function

This register captures the value of external timer.

Diagram



Fields

Field	Function
31-0 TIMER	Timer Indicates the current value of the external timer. This field is affected by soft reset.

52.5.2.19 Fault Confinement Interrupt Enable (FLTCONF_IE)

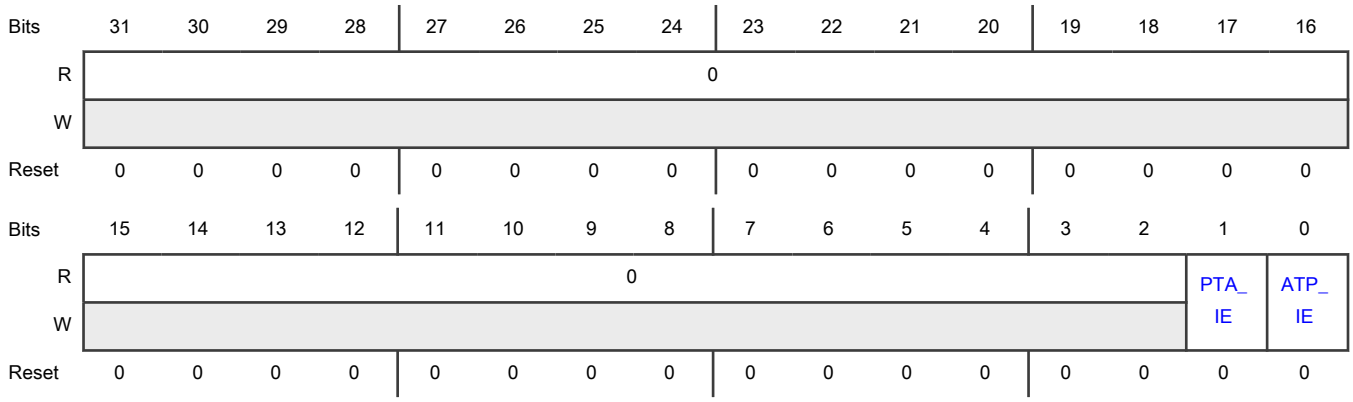
Offset

Register	Offset
FLTCONF_IE	7Ch

Function

Interrupt enables for active to passive and passive to active state transitions.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 PTA_IE	Passive to Active Interrupt Enable Enables passive to active error state interrupt. When this field is 1, passive to active error state interrupt is enabled. When this field is 0, passive to active error state interrupt is not enabled. 0b - Disable 1b - Enable
0 ATP_IE	Active to Passive Interrupt Enable Enables active to passive error state interrupt. When this field is 1, active to passive error state interrupt is enabled. When this field is 0, active to passive error state interrupt is not enabled. 0b - Disable 1b - Enable

52.5.2.20 Receive Individual Mask (RXIMR0 - RXIMR31)

Offset

For n = 0 to 31:

Register	Offset
RXIMRn	880h + (n × 4h)

Function

Stores the acceptance masks for ID filtering in RX message buffers and the Legacy RX FIFO.

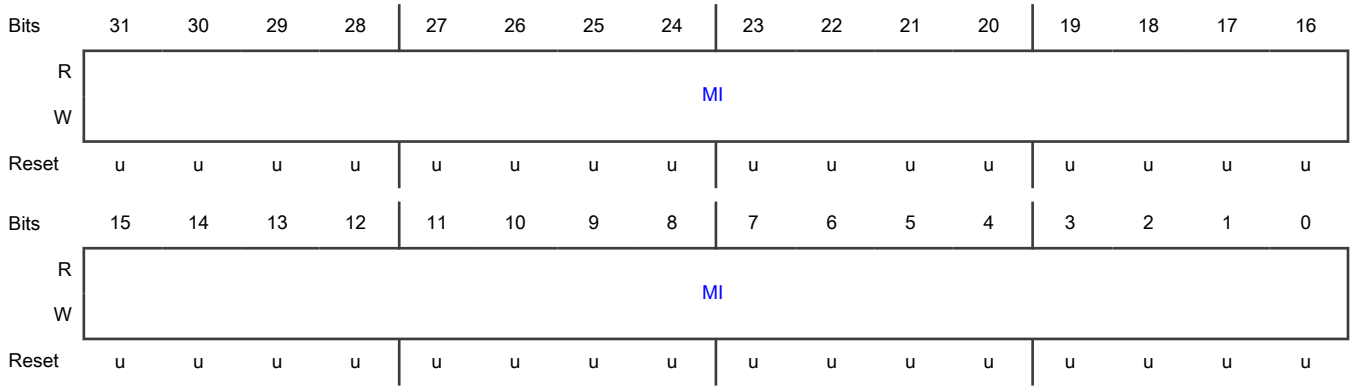
When the Legacy RX FIFO is disabled (MCR[RFEN] = 0), an individual mask is provided for each available RX message buffer on a one-to-one correspondence. When the Legacy RX FIFO is enabled (MCR[RFEN] = 1), an individual mask is provided for each Legacy RX FIFO ID filter table element on a one-to-one correspondence. This correspondence depends on the setting of CTRL2[RFFN] (see Legacy RX FIFO).

RXIMR0 stores the individual mask associated with either MB0 or ID filter table element 0. RXIMR1 stores the individual mask associated with either MB1 or ID filter table element 1, and so on.

The CPU can only access the RXIMR registers when the module is in Freeze mode; otherwise, the module blocks them. Reset does not affect these registers. They are located in RAM and must be explicitly initialized prior to any reception.

It is possible for the RXIMR memory region to be accessed as general-purpose memory. See Bus interface for more information.

Diagram



Fields

Field	Function
31-0 MI	Individual Mask Bits Masks the corresponding bit in both the message buffer filter and Legacy RX FIFO ID filter table element in distinct ways. For message buffer filters, see RX Message Buffers Global Mask (RXMGMASK). For Legacy RX FIFO ID filter table elements, see Legacy RX FIFO Global Mask (RXFGMASK). 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

52.5.2.21 Pretended Networking Control 1 (CTRL1_PN)

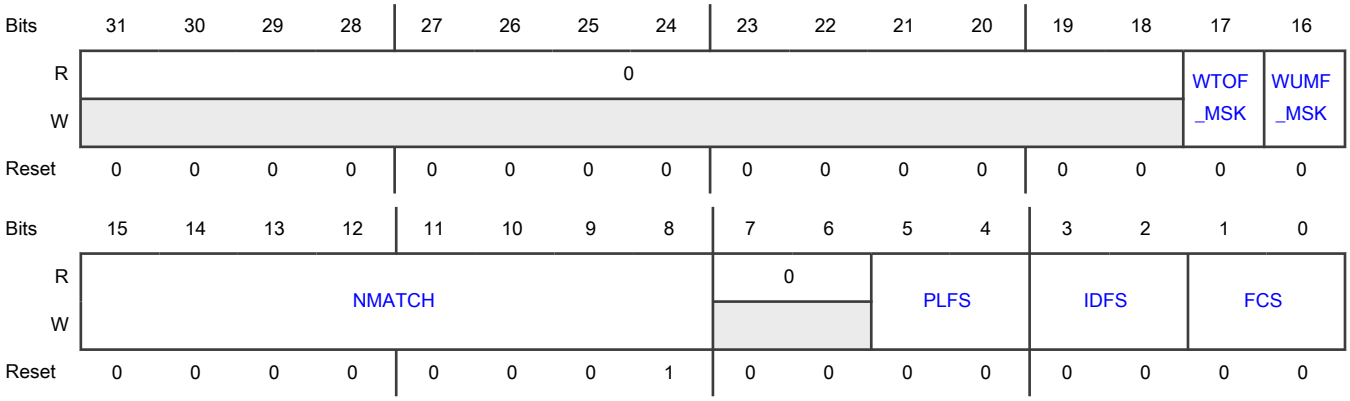
Offset

Register	Offset
CTRL1_PN	B00h

Function

Contains control bits for Pretended Networking mode filtering selection. Configure this register with the filter criteria to be used to receive wake-up messages. Fields in this register can be written in Freeze mode only, except for CTRL1_PN[WTOF_MSK] and CTRL1_PN[WUMF_MSK].

Diagram



Fields

Field	Function
31-18 —	Reserved
17 WTOF_MSK	Wake-up by Timeout Flag Mask Enables the generation of a wake-up event originated by a timeout. 0b - Disable 1b - Enable
16 WUMF_MSK	Wake-up by Matching Flag Mask Enables the generation of a wake-up event originated by a successful filtered RX message. 0b - Disable 1b - Enable
15-8 NMATCH	Number of Messages Matching the Same Filtering Criteria Defines the number of times a given message must match the predefined filtering criteria for ID or payload before generating a wake-up event. You can configure this quantity in the 1–255 range by using values 01h–FFh, respectively. 0000_0001b - Once 0000_0010b - Twice 1111_1111b - 255 times
7-6 —	Reserved
5-4 PLFS	Payload Filtering Selection Selects the level of payload filtering to be applied when FlexCAN is in Pretended Networking mode. When payload filtering is active, filtering does not accept remote messages (RTR = 1).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Match payload contents to an exact target value 01b - Match a payload value greater than or equal to a specified target value 10b - Match a payload value smaller than or equal to a specified target value 11b - Match upon a payload value within a range of values, inclusive
3-2 IDFS	ID Filtering Selection Selects the level of ID filtering to be applied when FlexCAN is in Pretended Networking mode. In ID filtering, if FLT_ID2_IDMASK[IDE_MSK] = 1 and FLT_ID2_IDMASK[RTR_MSK] = 1, the IDE and RTR bits are considered part of the reception filter. 00b - Match ID contents to an exact target value 01b - Match an ID value greater than or equal to a specified target value 10b - Match an ID value smaller than or equal to a specified target value 11b - Match an ID value within a range of values, inclusive
1-0 FCS	Filtering Combination Selection Selects the filtering criteria to be applied when FlexCAN is in Pretended Networking mode. See Receive process in Pretended Networking mode for more details. 00b - Message ID filtering only 01b - Message ID filtering and payload filtering 10b - Message ID filtering occurring a specified number of times 11b - Message ID filtering and payload filtering a specified number of times

52.5.2.22 Pretended Networking Control 2 (CTRL2_PN)

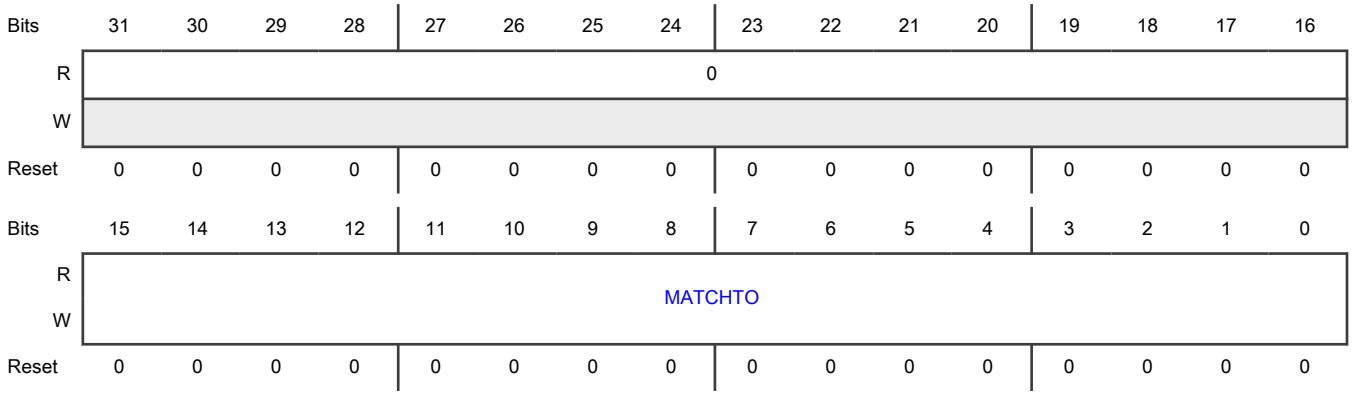
Offset

Register	Offset
CTRL2_PN	B04h

Function

Contains the configuration for the timeout value in Pretended Networking mode. You can only write to this register in Freeze mode.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 MATCHTO	Timeout for No Message Matching the Filtering Criteria Defines a timeout value that generates a wake-up event if MCR[PNET_EN] = 1. If the timeout counter reaches the target value when FlexCAN is in Pretended Networking mode, a wake-up event is generated. The timeout limit can be configured from 1 to 65535 to control an internal 16-bit incrementing timer to produce a trigger upon reaching this configured value. The internal timer is incremented based on periodic time ticks, where the period is 64 times the CAN Bit Time unit. Writing 0000h to this field disables the timeout.

52.5.2.23 Pretended Networking Wake-Up Match (WU_MTC)

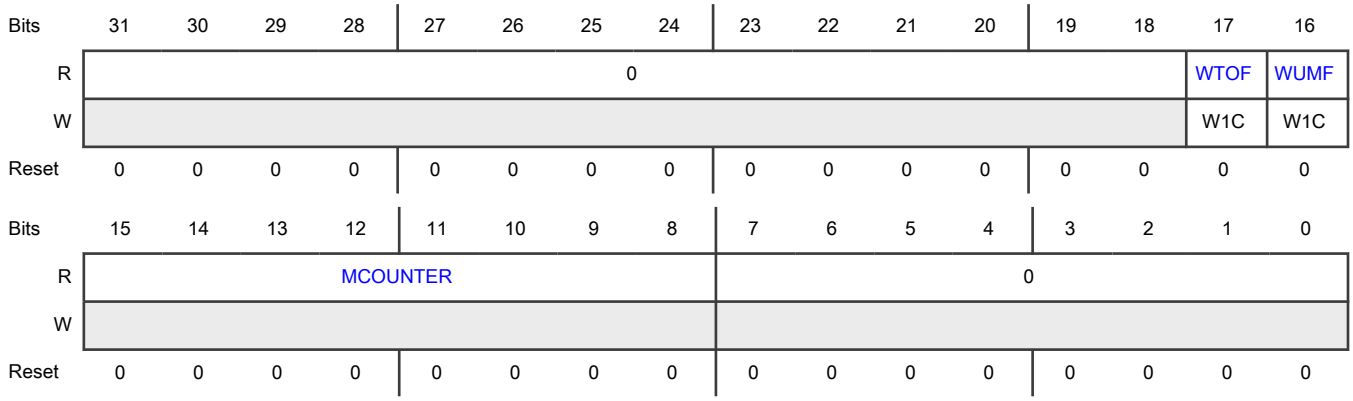
Offset

Register	Offset
WU_MTC	B08h

Function

Contains wake-up information related to the matching processes performed when FlexCAN receives frames in Pretended Networking mode.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 WTOF	Wake-up by Timeout Flag Bit Identifies whether FlexCAN has detected a timeout event during a time interval defined by CTRL2_PN[MATCHTO]. If CTRL1_PN[WTOF_MSK] = 1, this flag generates a wake-up event. 0b - No event detected 1b - Event detected
16 WUMF	Wake-up by Match Flag Identifies whether FlexCAN has detected a matching RX incoming message that meets the filtering criteria specified in Pretended Networking Control 1 (CTRL1_PN). If CTRL1_PN[WUMF_MSK] = 1, this flag generates a wake-up event. 0b - No event detected 1b - Event detected
15-8 MCOUNTER	Number of Matches in Pretended Networking Contains the number of times a given message has matched the predefined filtering criteria for ID or payload before a wake-up event. When FlexCAN enters Pretended Networking mode, this register is reset; soft reset does not affect it.
7-0 —	Reserved

52.5.2.24 Pretended Networking ID Filter 1 (FLT_ID1)

Offset

Register	Offset
FLT_ID1	B0Ch

Function

Contains FLT_ID1 target value, as well as IDE and RTR target values used to filter an incoming message ID. The FLT_ID1 is used for comparisons or as the lower limit value in an ID range detection. It can be written in Freeze mode only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	FLT_IDE	FLT_RTR	FLT_ID1												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLT_ID1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 FLT_IDE	ID Extended Filter Identifies whether the frame format is standard or extended. It is used as part of the ID reception filter. 0b - Standard 1b - Extended
29 FLT_RTR	Remote Transmission Request Filter Identifies whether the frame is remote. It is used as part of the ID reception filter. 0b - Reject remote frame (accept data frame) 1b - Accept remote frame
28-0 FLT_ID1	ID Filter 1 for Pretended Networking filtering Defines either the 29 bits of an extended frame format, considering all bits, or the 11 bits of a standard frame format, considering only the leftmost 11 bits.

52.5.2.25 Pretended Networking Data Length Code (DLC) Filter (FLT_DLC)

Offset

Register	Offset
FLT_DLC	B10h

Function

Contains the DLC inside a range of target values (FLT_DLC_LO and FLT_DLC_HI) used to filter an incoming message. The DLC range is used only for payload filtering. It can be written in Freeze mode only.

When a fixed quantity of data bytes is required, write the same value to FLT_DLC[FLT_DLC_LO] and FLT_DLC[FLT_DLC_HI]. See Receive process in Pretended Networking mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FLT_DLC_LO			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FLT_DLC_HI			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 FLT_DLC_LO	Lower Limit for Length of Data Bytes Filter Specifies the lower limit for the number of data bytes considered valid for payload comparison. It is used as part of payload reception filter.
15-4 —	Reserved
3-0 FLT_DLC_HI	Upper Limit for Length of Data Bytes Filter Specifies the upper limit for the number of data bytes considered valid for payload comparison. It is used as part of payload reception filter.

52.5.2.26 Pretended Networking Payload Low Filter 1 (PL1_LO)

Offset

Register	Offset
PL1_LO	B14h

Function

Contains the low-order bits of the target value used to filter incoming message payload for payload filter 1. It is used for comparisons or as the lower limit value in a payload range detection. It can be written in Freeze mode only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Data_byte_0								Data_byte_1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Data_byte_2								Data_byte_3							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 Data_byte_0	Data byte 0 Contains payload filter 1 low-order bits for PN payload filtering corresponding to data byte 0.
23-16 Data_byte_1	Data byte 1 Contains payload filter 1 low-order bits for PN payload filtering corresponding to data byte 1.
15-8 Data_byte_2	Data byte 2 Contains payload filter 1 low-order bits for PN payload filtering corresponding to data byte 2.
7-0 Data_byte_3	Data byte 3 Contains payload filter 1 low-order bits for PN payload filtering corresponding to data byte 3.

52.5.2.27 Pretended Networking Payload High Filter 1 (PL1_HI)

Offset

Register	Offset
PL1_HI	B18h

Function

Contains the high-order bits of the target value used to filter incoming message payload for payload filter 1. It is used either for comparisons or as the lower limit value in a payload range detection. It can be written in Freeze mode only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Data_byte_4								Data_byte_5							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Data_byte_6								Data_byte_7							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 Data_byte_4	Data byte 4 Contains payload filter 1 high-order bits for PN payload filtering corresponding to data byte 4.
23-16 Data_byte_5	Data byte 5 Contains payload filter 1 high-order bits for PN payload filtering corresponding to data byte 5.
15-8 Data_byte_6	Data byte 6 Contains payload filter 1 high-order bits for PN payload filtering corresponding to data byte 6.
7-0 Data_byte_7	Data byte 7 Contains payload filter 1 high-order bits for PN payload filtering corresponding to data byte 7.

52.5.2.28 Pretended Networking ID Filter 2 or ID Mask (FLT_ID2_IDMASK)

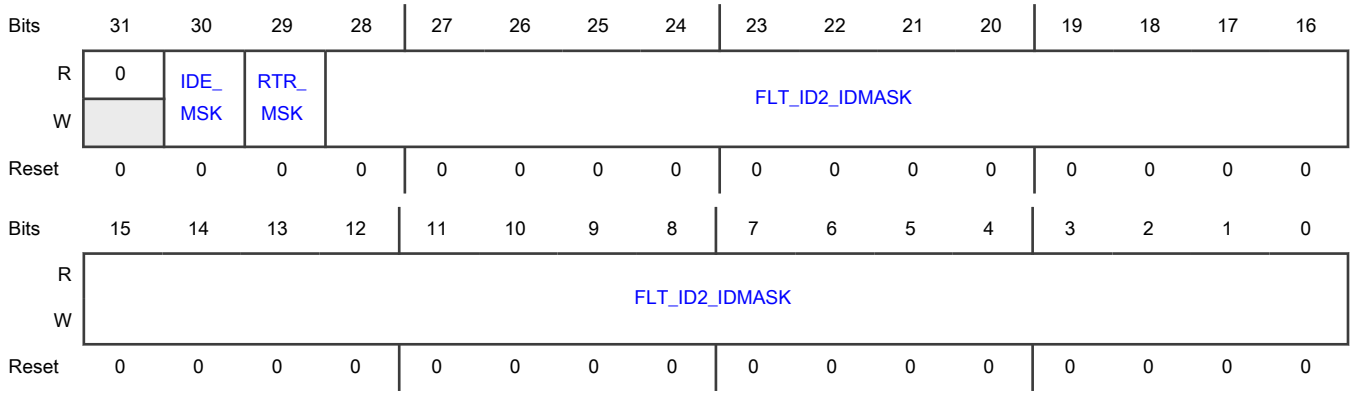
Offset

Register	Offset
FLT_ID2_IDMASK	B1Ch

Function

Contains FLT_ID2 target value used only as the upper limit value in ID range detection. When an exact ID filtering criterion is selected, this register stores the ID mask. IDE_MSK and RTR_MSK are used in both types of ID filtering (exact and range). These fields enable FLT_ID1[FLT_IDE] and FLT_ID1[FLT_RTR] to be used as part of the ID reception filter. This register can be written in Freeze mode only.

Diagram



Fields

Field	Function
31 —	Reserved
30 IDE_MSK	ID Extended Mask Indicates whether the frame format (standard or extended) is used as part of the ID reception filter. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.
29 RTR_MSK	Remote Transmission Request Mask Indicates whether the frame type (data or remote) is part of the ID reception filter. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.
28-0 FLT_ID2_IDMASK	ID Filter 2 for Pretended Networking Filtering or ID Mask Bits for Pretended Networking ID Filtering Defines filter values in range ID filtering: • Value in extended frame format (29 bits), considering FLT_ID2[28:0] • Value in standard frame format (11 bits), considering the FLT_ID2[28:18]. In this case, bits [17:0] are meaningless. Or, defines the mask values in exact ID filtering: • Values for extended frame format (29 bits), considering IDMASK[28:0] • Values for standard frame format (11 bits), considering IDMASK[28:18]. In this case, bits [17:0] are meaningless.

52.5.2.29 Pretended Networking Payload Low Filter 2 and Payload Low Mask (PL2_PLMASK_LO)

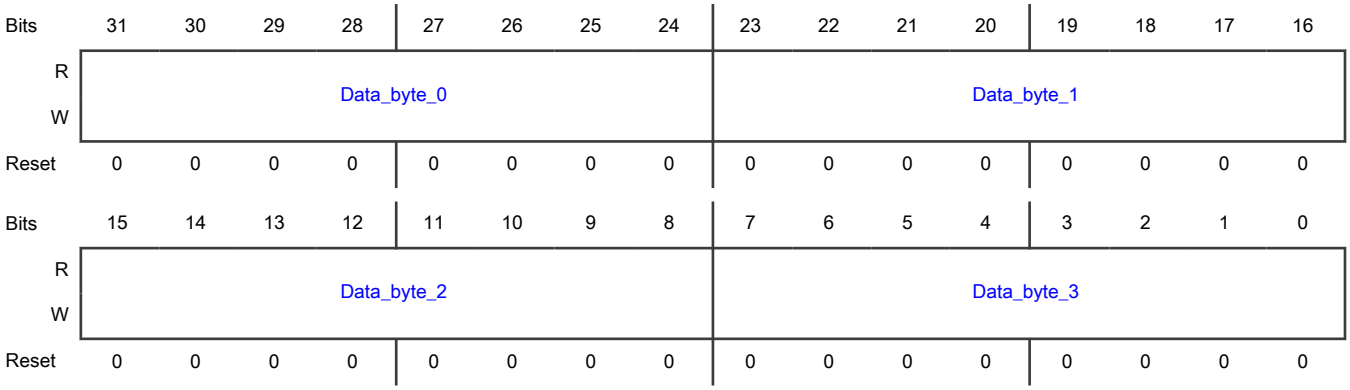
Offset

Register	Offset
PL2_PLMASK_LO	B20h

Function

Contains the low-order bits for Payload Filter 2, used only as the upper limit value in a payload range detection. Also, when an exact payload filtering criterion is selected, this register is used as payload mask for the low-order bits. Otherwise, this register is unused. It can be written in Freeze mode only.

Diagram



Fields

Field	Function
31-24 Data_byte_0	Data Byte 0 Contains payload filter 2 low-order bits, or Payload Mask low-order bits for PN payload filtering, corresponding to data byte 0
23-16 Data_byte_1	Data Byte 1 Contains payload filter 2 low-order bits, or Payload Mask low-order bits for PN payload filtering, corresponding to data byte 1
15-8 Data_byte_2	Data Byte 2 Contains payload filter 2 low-order bits, or Payload Mask low-order bits for PN payload filtering, corresponding to data byte 2
7-0 Data_byte_3	Data Byte 3 Contains payload filter 2 low-order bits, or Payload Mask low-order bits for PN payload filtering, corresponding to data byte 3

52.5.2.30 Pretended Networking Payload High Filter 2 and Payload High Mask (PL2_PLMASK_HI)

Offset

Register	Offset
PL2_PLMASK_HI	B24h

Function

Contains the high-order bits for the Payload Filter 2, used only as the upper limit value in a payload range detection. Also, when an exact payload filtering criterion is selected, this register is used as payload mask for the high-order bits. Otherwise, this register is unused. It can be written in Freeze mode only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data_byte_4								Data_byte_5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data_byte_6								Data_byte_7							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 Data_byte_4	Data Byte 4 Contains payload filter 2 high-order bits, or Payload Mask high-order bits for PN payload filtering, corresponding to data byte 4.
23-16 Data_byte_5	Data Byte 5 Contains payload filter 2 high-order bits, or Payload Mask high-order bits for PN payload filtering, corresponding to data byte 5.
15-8 Data_byte_6	Data Byte 6 Contains payload filter 2 high-order bits, or Payload Mask high-order bits for PN payload filtering, corresponding to data byte 6.
7-0 Data_byte_7	Data Byte 7 Contains payload filter 2 high-order bits, or Payload Mask high-order bits for PN payload filtering, corresponding to data byte 7.

52.5.2.31 Wake-Up Message Buffer (WMB0_CS - WMB3_CS)

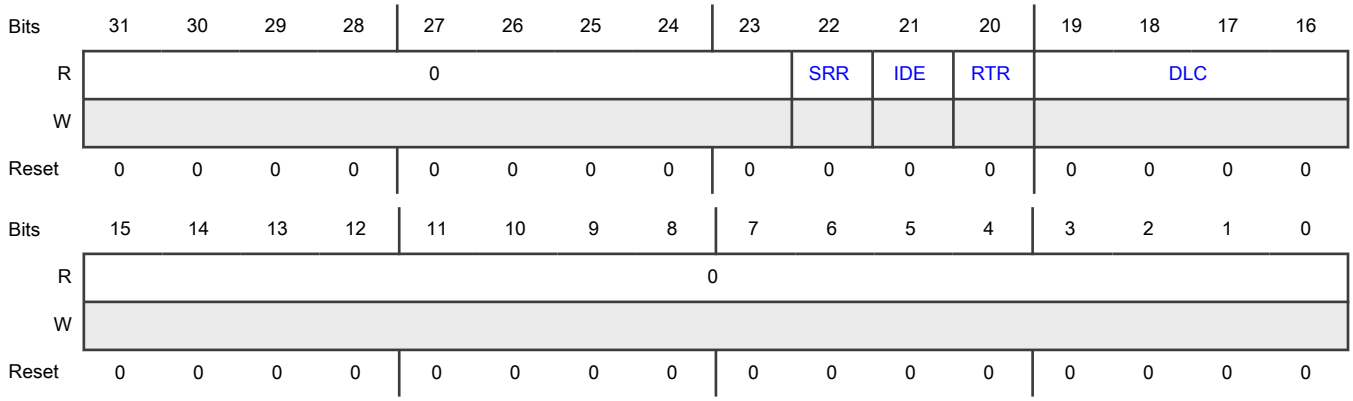
Offset

Register	Offset
WMB0_CS	B40h
WMB1_CS	B50h
WMB2_CS	B60h
WMB3_CS	B70h

Function

Stores the Control and Status information (IDE, RTR, and DLC fields) of an incoming RX message.

Diagram



Fields

Field	Function
31-23 —	Reserved
22 SRR	Substitute Remote Request Identifies whether the request is dominant or recessive. This field is used only in extended format. You must write 1 to this field for transmission buffers. The value of this field is stored with the value received on the CAN bus for receiving buffers. If FlexCAN receives this field as dominant, it is interpreted as an arbitration loss. 0b - Dominant 1b - Recessive
21 IDE	ID Extended Bit Identifies whether the frame format is standard or extended.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Standard 1b - Extended
20 RTR	Remote Transmission Request Identifies whether the frame is remote or data 0b - Data 1b - Remote
19-16 DLC	Length of Data in Bytes Contains the length (in bytes) of the RX data received when FlexCAN is in Pretended Networking mode. FlexCAN writes this field, copied from the DLC (Data Length Code) field of the received frame. The DLC field indicates which data bytes are valid.
15-0 —	Reserved

52.5.2.32 Wake-Up Message Buffer for ID (WMB0_ID - WMB3_ID)

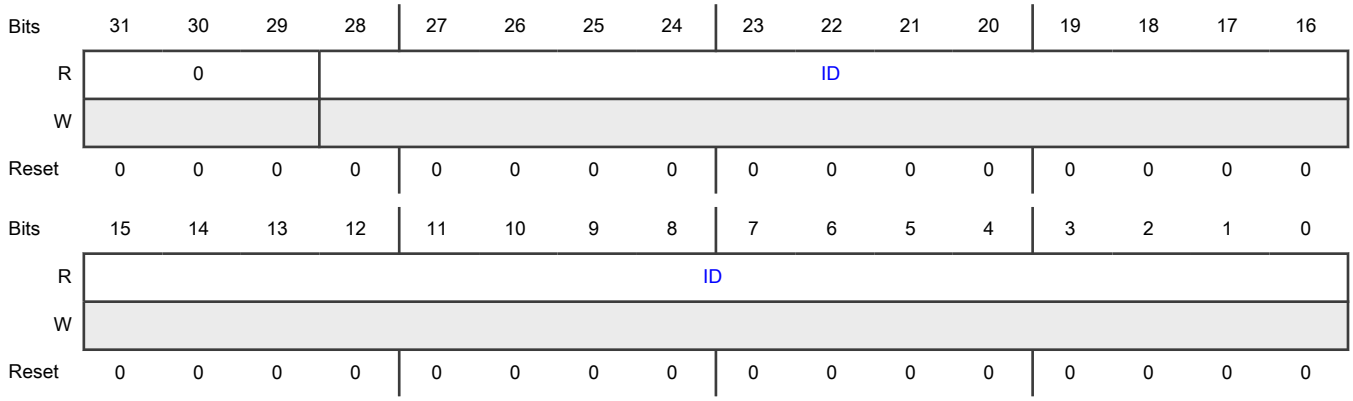
Offset

Register	Offset
WMB0_ID	B44h
WMB1_ID	B54h
WMB2_ID	B64h
WMB3_ID	B74h

Function

Stores the ID information of an incoming RX message.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-0 ID	Received ID in Pretended Networking Mode Stores the received ID, which is: • The 29 bits of the extended frame format (considering) • The 11 bits of the standard frame format (considering only; the remaining bits in the range are meaningless).

52.5.2.33 Wake-Up Message Buffer for Data 0–3 (WMB0_D03 - WMB3_D03)

Offset

Register	Offset
WMB0_D03	B48h
WMB1_D03	B58h
WMB2_D03	B68h
WMB3_D03	B78h

Function

Stores data bytes 0–3 of the payload information of an incoming RX message. The content of each register is cleared when the incoming matched message is either a remote frame (RTR = 1) or a data frame with DLC = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data_byte_0								Data_byte_1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data_byte_2								Data_byte_3							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 Data_byte_0	Data Byte 0 Contains received payload corresponding to data byte 0 in Pretended Networking mode
23-16 Data_byte_1	Data Byte 1 Contains received payload corresponding to data byte 1 in Pretended Networking mode
15-8 Data_byte_2	Data Byte 2 Contains received payload corresponding to data byte 2 in Pretended Networking mode
7-0 Data_byte_3	Data Byte 3 Contains received payload corresponding to data byte 3 in Pretended Networking mode

52.5.2.34 Wake-Up Message Buffer Register Data 4–7 (WMB0_D47 - WMB3_D47)

Offset

Register	Offset
WMB0_D47	B4Ch
WMB1_D47	B5Ch
WMB2_D47	B6Ch
WMB3_D47	B7Ch

Function

Stores the data bytes 4–7 of the payload information of an incoming RX message. The content of each register is cleared when the incoming matched message is either a remote frame (RTR = 1) or a data frame with DLC = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data_byte_4								Data_byte_5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data_byte_6								Data_byte_7							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 Data_byte_4	Data Byte 4 Contains received payload corresponding to data byte 4 in Pretended Networking mode
23-16 Data_byte_5	Data Byte 5 Contains received payload corresponding to data byte 5 in Pretended Networking mode
15-8 Data_byte_6	Data Byte 6 Contains received payload corresponding to data byte 6 in Pretended Networking mode
7-0 Data_byte_7	Data Byte 7 Contains received payload corresponding to data byte 7 in Pretended Networking mode

52.5.2.35 Enhanced CAN Bit Timing Prescalers (EPRS)

Offset

Register	Offset
EPRS	BF0h

Function

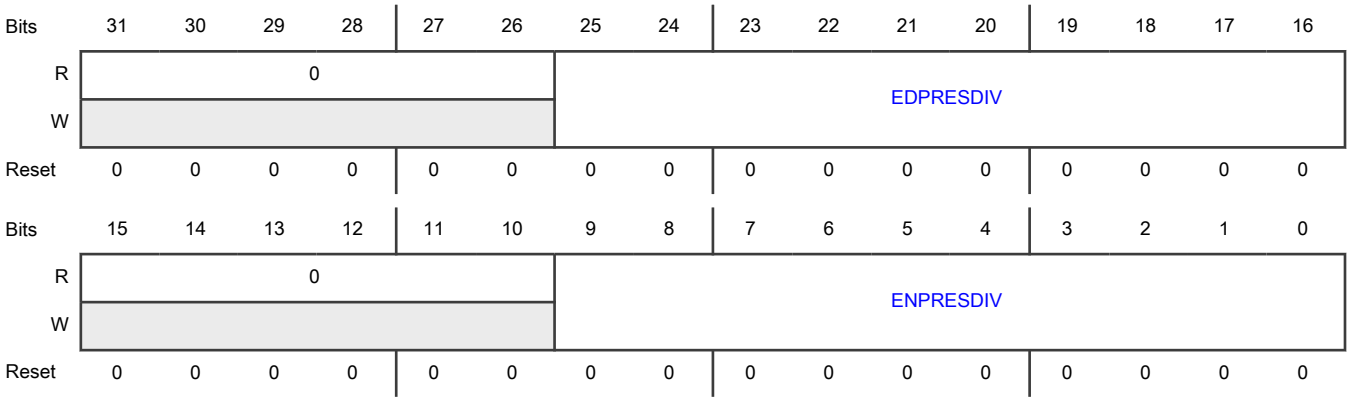
Defines the CAN bit timing prescalers for the nominal phase and data phase when CTRL2[BTE] = 1.

Used by the module only if CTRL2[BTE] = 1; otherwise a write operation has no effect and all fields are read as zero.

This register can be written only in Freeze mode; the module blocks it in other modes.

Soft reset does not affect the contents of this register.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 EDPRESDIV	Extended Data Phase Prescaler Division Factor Defines the ratio between the PE clock frequency and the Sclock frequency in the data phase of a CAN FD message when CTRL2[BTE] = 1. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. Sclock frequency = PE clock frequency ÷ (EDPRESDIV + 1) **NOTE** To minimize errors when processing FD frames, use the same value for this field and for EPRS[ENPRESDIV]. See the first note in CAN FD frames for details.
15-10 —	Reserved
9-0 ENPRESDIV	Extended Nominal Prescaler Division Factor Defines the ratio between the PE clock frequency and the Sclock frequency when CTRL2[BTE] = 1. Otherwise, it reads as 0 and a write operation has no effect. The Sclock period defines the time quantum of the CAN protocol in the nominal phase. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). Sclock frequency = PE clock frequency ÷ (ENPRESDIV + 1)

52.5.2.36 Enhanced Nominal CAN Bit Timing (ENCBT)

Offset

Register	Offset
ENCBT	BF4h

Function

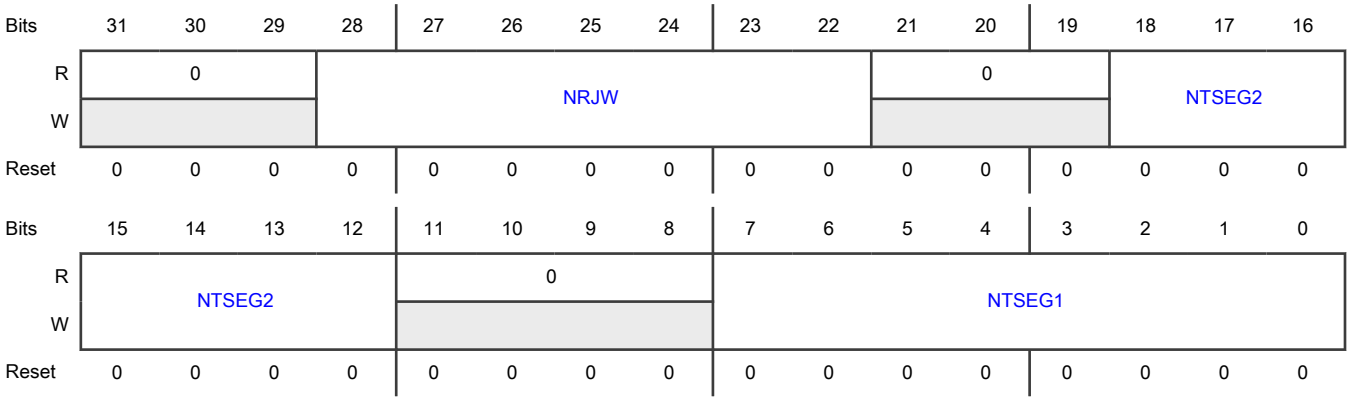
Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1) and CAN Bit Timing (CBT), to get higher CAN bit timing resolution.

This register is used by the module only if CTRL2[BTE] = 1. Otherwise, a write operation has no effect and all fields are read as zero.

Soft reset does not affect the contents of this register.

This register can be written only in Freeze mode; the module blocks it in other modes.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-22 NRJW	Nominal Resynchronization Jump Width Defines the maximum number of time quanta that one resynchronization can change a nominal bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. One time quantum = one Sclock period Nominal Resync Jump Width = NRJW + 1
21-19 —	Reserved
18-12 NTSEG2	Nominal Time Segment 2 Defines the length of Time Segment 2 in the nominal bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Nominal Time Segment 2 = (NTSEG2 + 1) × Time Quanta One time quantum = one Sclock period
11-8 —	Reserved
7-0 NTSEG1	Nominal Time Segment 1 Defines the length of Time Segment 1 in the bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Nominal Time Segment 1 = (NTSEG1 + 1) × Time Quanta One time quantum = one Sclock period

52.5.2.37 Enhanced Data Phase CAN Bit Timing (EDCBT)

Offset

Register	Offset
EDCBT	BF8h

Function

Provides an alternative way to store the data phase CAN bit timing variables described in CAN FD Bit Timing (FDCBT) to achieve higher CAN bit timing resolution.

This register is used by the module only if CTRL2[BTE] = 1; otherwise a write operation has no effect and all fields are read as zero.

Soft reset does not affect the contents of this register.

This register can be written only in Freeze mode; the module blocks it in other modes.

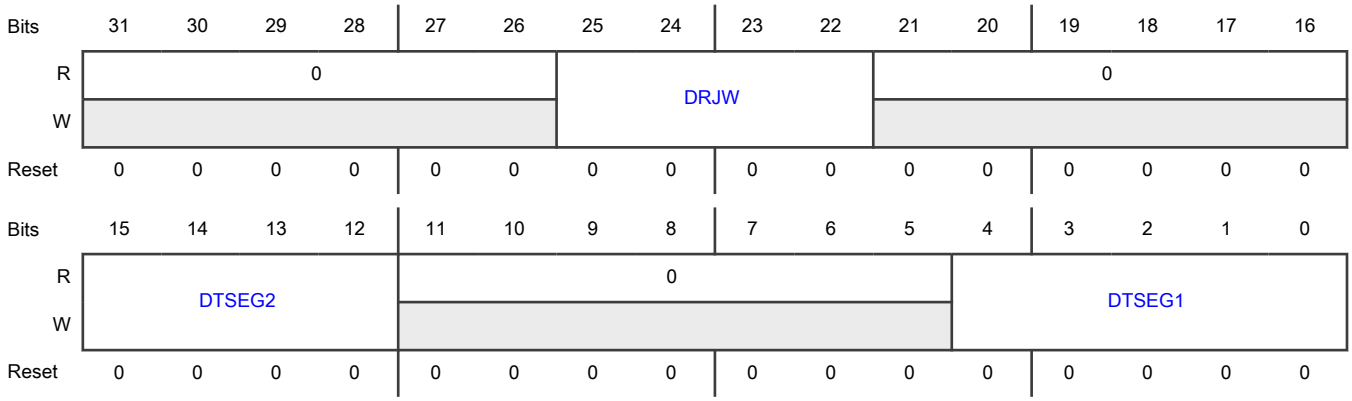
NOTE

Ensure that bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1:2015).

NOTE

DTSEG1 must be at least two time quanta.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-22 DRJW	Data Phase Resynchronization Jump Width Defines the maximum number of time quanta that one resynchronization can change a data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Resync Jump Width = DRJW + 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 —	Reserved
15-12 DTSEG2	Data Phase Time Segment 2 Defines the length of time segment 2 in the data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Time Segment 2 = (DTSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.
11-5 —	Reserved
4-0 DTSEG1	Data Phase Segment 1 Defines the length of time segment 1 in the data phase bit time when CTRL2[BTE] = 1. Otherwise, it has no effect. Data Phase Time Segment 1 = (NTSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.

52.5.2.38 Enhanced Transceiver Delay Compensation (ETDC)

Offset

Register	Offset
ETDC	BFCh

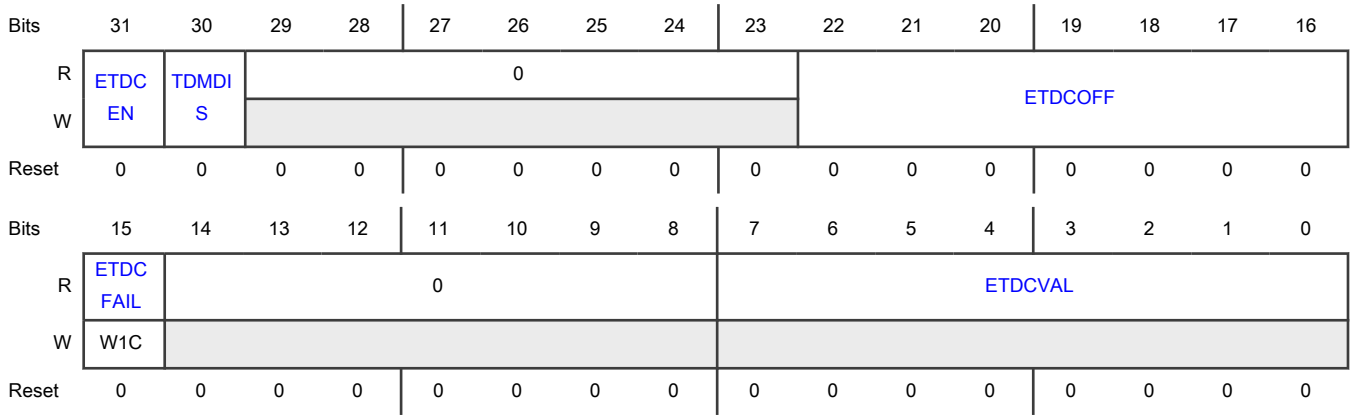
Function

Contains extended versions of FDCTRL[TDCOFF] and FDCTRL[TDCVAL]. This register is used by the module only if CTRL2[BTE] = 1. Otherwise, a write operation has no effect and all fields are read as zero.

NOTE

See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31 ETDCEN	Transceiver Delay Compensation Enable Enables the TDC feature. It can be written in Freeze mode only. **NOTE** See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1:2015) for details. **NOTE** TDC must be disabled when the Loop Back Mode is enabled. See CTRL1[LPB]. 0b - Disable 1b - Enable
30 TDMDIS	Transceiver Delay Measurement Disable Disables the transceiver delay measurement. When the TDC measurement is disabled, only ETDC[ETDCOFF] determines the secondary sample point position. If TCD measurement is enabled, the sum of the transceiver delay measurement plus the enhanced TDC offset determines the secondary sample point position. Soft reset does not affect this field. **NOTE** This bit can be enabled only if CTRL2[BTE] = 1. 0b - Enable 1b - Disable
29-23 —	Reserved
22-16	Enhanced Transceiver Delay Compensation Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
ETDCOFF	Contains the offset value to be added to the loop delay of the measured transceiver. This value defines the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for details on how the loop delay measurement is performed. This field can be written in Freeze mode only. Its value can be defined in protocol engine (PE) clock periods (CANCLK, see Protocol timing for more details). It must be smaller than the CAN bit duration in the data bit rate for proper operation. Do not write 0 to this field. **NOTE** If CTRL2[BTE] becomes 1 after a chip-level hard reset, this field is read as 1h.
15 ETDCFAIL	Transceiver Delay Compensation Fail Indicates whether the transceiver delay compensation (TDC) mechanism is out of range. In this case, it is unable to compensate the loop delay of the transceiver and successfully compare the delayed received bits to the transmitted ones. (See Transceiver delay compensation.) This field becomes 0 the first time FlexCAN detects the out of range condition. 0b - In range 1b - Out of range
14-8 —	Reserved
7-0 ETDCVAL	Enhanced Transceiver Delay Compensation Value Contains ETDC[ETDCOFF] added to the measured value of the transceiver loop delay in the latest transmitted CAN FD frame, with BRS = 1. The module only updates this field when ETDC[ETDCEN] = 1. Soft reset affects this field. **NOTE** If ETDC[TDMDIS] = 1, this field stores ETDC[ETDCOFF] only.

52.5.2.39 CAN FD Control (FDCTRL)

Offset

Register	Offset
FDCTRL	C00h

Function

Contains control bits for CAN FD operation. It also defines the data size of message buffers allocated in different partitions of RAM (memory blocks) as described in Table 351.

When an 8-byte payload is selected:

- Block R0 allocates MB0–MB31.
- Block R1 allocates MB32–MB63.

When a payload larger than eight bytes is selected, the maximum number of message buffers in a block is limited as described below.

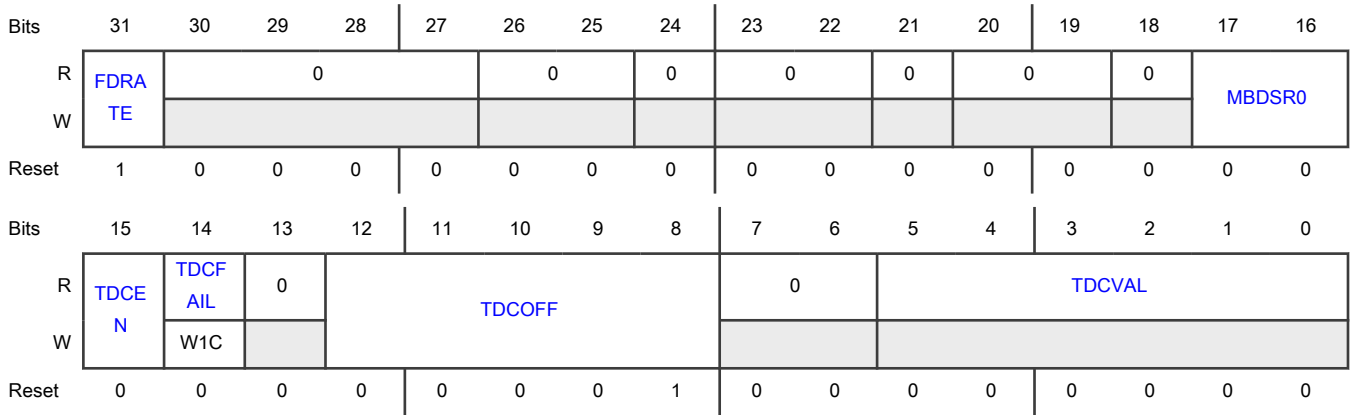
Table 351. Number of message buffers

Payload size	Maximum number of message buffers per RAM block
8 bytes	32
16 bytes	21
32 bytes	12
64 bytes	7

One memory block fits exactly 32 message buffers with an 8-byte payload. For other possible payload sizes, empty memory may exist between the last message buffer in a block and the beginning of the next block. This empty memory corresponds to less than one message buffer, and must not be used.

Soft reset does not affect the contents of this register.

Diagram



Fields

Field	Function
31 FDRATE	Bit Rate Switch Enable Enables the effect of the Bit Rate Switch (BRS bit) during the data phase of TX messages. When 1, if BRS = 1 in the TX message buffer, frames are transmitted with bit rate switching. When 0, frames are transmitted at a nominal rate, and the BRS bit in the TX MB has no effect. The CPU can write to this field at any time. However, its effect becomes active only under one of these conditions: • The CAN bus is in the Wait for Bus Idle state. • The CAN bus is in the Bus Idle state. • The CAN bus is in the Bus Off state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The current frame under reception or transmission reaches the interframe space. By writing 0 to FDCTRL[FDRATE], the CPU can force all bits in CAN FD messages to be transmitted at nominal bit rate. This transmission occurs regardless of the value in the BRS bit of the TX message buffers. 0b - Disable 1b - Enable
30-27 —	Reserved
26-25 —	Reserved
24 —	Reserved
23-22 —	Reserved
21 —	Reserved
20-19 —	Reserved
18 —	Reserved
17-16 MBDSR0	Message Buffer Data Size for Region 0 Selects the data size per message buffer for region R0 of message buffers allocated in RAM. This field can be written in Freeze mode only. 00b - 8 bytes 01b - 16 bytes 10b - 32 bytes 11b - 64 bytes
15 TDCEN	Transceiver Delay Compensation Enable Enables the TDC feature. It can be written in Freeze mode only. See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1:2015) for details. TDC must be disabled when Loopback mode is enabled (see CTRL1[LPB]).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If CTRL2[BTE] = 1, this field is read as 0 and a write operation has no effect. 0b - Disable 1b - Enable
14 TDCFAIL	Transceiver Delay Compensation Fail Indicates whether the Transceiver Delay Compensation (TDC) mechanism is out of range. In this case, the mechanism cannot compensate for the loop delay of the transceiver and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). The first time that FlexCAN detects the out-of-range condition, this field becomes 1. **NOTE** If CTRL2[BTE] = 1, this field is read as 0 and a write operation has no effect. 0b - In range 1b - Out of range
13 —	Reserved
12-8 TDCOFF	Transceiver Delay Compensation Offset Contains the offset value to be added to the loop delay of the measured transceiver. This value defines the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for details about loop delay measurement. This field can be written in Freeze mode only. Its value can be defined in Protocol Engine Clock periods (CANCLK, see Protocol timing for more details). The value must be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** If CTRL2[BTE] = 1, TDCOFF is read as 0 and a write operation has no effect. Do not write 0 to this field.
7-6 —	Reserved
5-0 TDCVAL	Transceiver Delay Compensation Value Contains the value of the transceiver loop delay measured from the transmitted EDL-to-R0 transition edge to the respective received one added to FDCTRL[TDCOFF]. This value is an integer multiple of the Protocol Engine Clock period (CANCLK). If CTRL2[BTE] = 1, this field is read as 0. See Protocol timing for details on the loop delay measurement.

52.5.2.40 CAN FD Bit Timing (FDCBT)

Offset

Register	Offset
FDCBT	C04h

Function

Stores the CAN bit timing variables used in the data phase of CAN FD messages when the FDCTRL[FDRATE] = 1, compatible with the CAN FD specification. Fields in this register define:

- The time quantum duration
- The number of time quanta per CAN bit
- The sample point position for the data bit rate portion of a CAN FD message with BRS = 1

Soft reset does not affect the contents of this register.

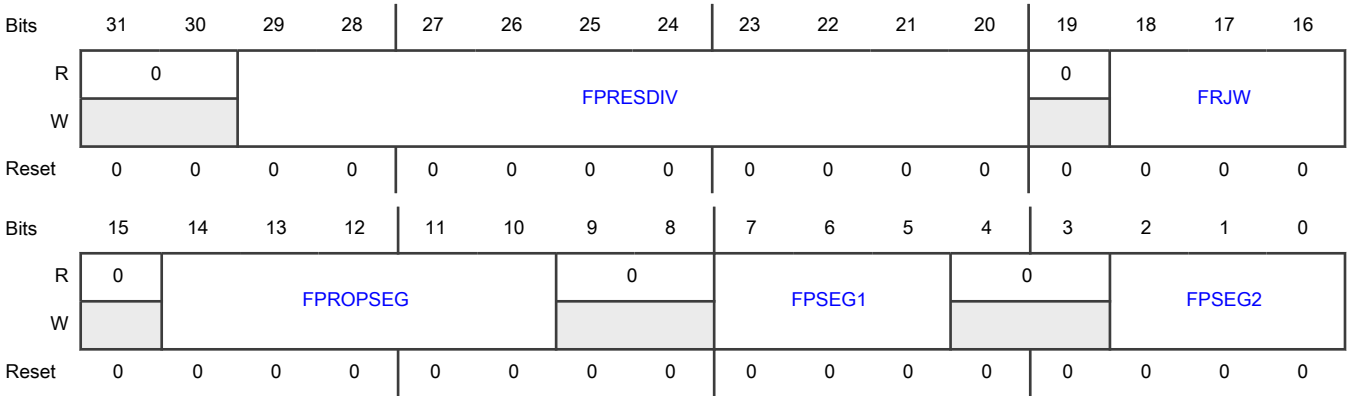
The sum of the Fast Propagation Segment (FPROPSEG) and Fast Phase Segment 1 (FPSEG1) must be at least two time quanta.

Ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1:2015).

NOTE

If CTRL2[BTE] = 1, this register is read as zero and a write operation has no effect.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 FPRESDIV	Fast Prescaler Division Factor Defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data bit rate portion of a CAN FD message with BRS = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. This field can be written only in Freeze mode; the module blocks it in other modes. Sclock frequency = PE clock frequency ÷ (FPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for this field and for CTRL1[PRESDIV] or CBT[EPRESDIV]. See the first note in CAN FD frames for details.
19 —	Reserved
18-16 FRJW	Fast Resync Jump Width Defines the maximum number of time quanta that one resynchronization can change a bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Resync Jump Width = FSJW + 1. One Time Quantum = one Sclock period.
15 —	Reserved
14-10 FPROPSEG	Fast Propagation Segment Defines the length of the propagation segment in the bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Propagation Segment Time = FPROPSEG × Time Quanta. One Time Quantum = one Sclock period.
9-8 —	Reserved
7-5 FPSEG1	Fast Phase Segment 1 Defines the length of phase segment 1 in the bit time in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes. Phase Segment 1 = (FPSEG1 + 1) × Time Quanta. One Time Quantum = one Sclock period.
4-3 —	Reserved
2-0 FPSEG2	Fast Phase Segment 2 Defines the length of phase segment 2 in the data bit rate portion of a CAN FD message with BRS = 1. This field can be written only in Freeze mode; the module blocks it in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Phase Segment 2 = (FPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

52.5.2.41 CAN FD CRC (FDCRC)

Offset

Register	Offset
FDCRC	C08h

Function

Provides information about the cyclic redundancy check (CRC) of transmitted messages.

FlexCAN uses different CRC polynomials for different frame formats.

- The CRC_15 polynomial is used for all frames in CAN format.
- The CRC_17 polynomial is used for frames in CAN FD format with a DATA FIELD up to 16 bytes.
- The CRC_21 polynomial is used for frames in CAN FD format with a DATA FIELD longer than 16 bytes.

Each polynomial shown below results in a Hamming distance of 6. This register is updated at the same time that the TX Interrupt flag is set.

$$\text{CRC\_15} = \text{C599h}: \ (x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1)$$

$$\text{CRC\_17} = \text{3685Bh}: \ (x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^{6} + x^{4} + x^{3} + x^{1} + 1)$$

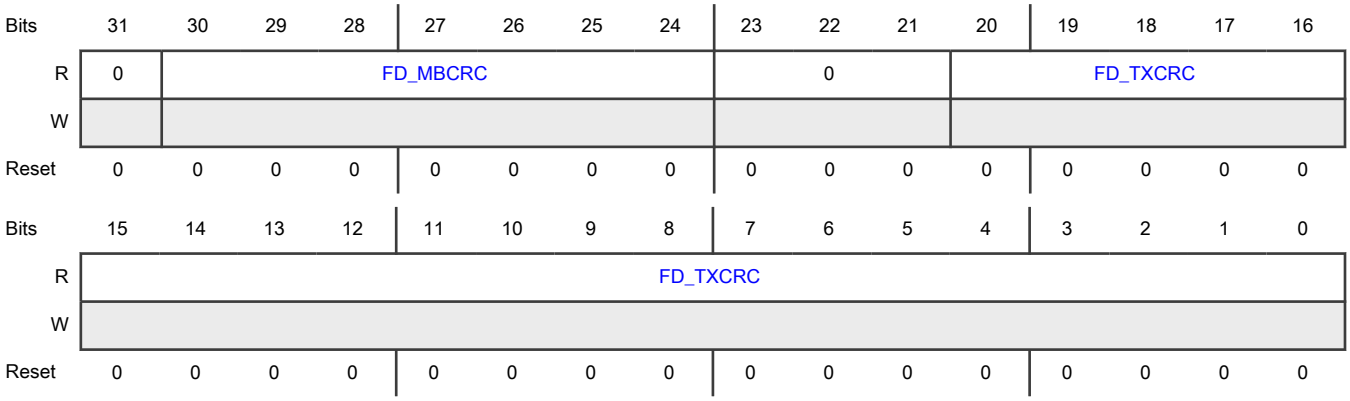
$$\text{CRC\_21} = \text{302899h}: \ (x^{21} + x^{20} + x^{13} + x^{11} + x^{7} + x^{4} + x^{3} + 1)$$

Equation 20. CRC polynomial used on CAN frame

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1:2015) for details.

Diagram



Fields

Field	Function
31 —	Reserved
30-24 FD_MBCRC	CRC Message Buffer Number for FD_TXCRC Indicates the number of the message buffer corresponding to the value in FDCRC[FD_TXCRC], for both FD and non-FD frames. It reports the same information as in CRCR[MBCRC].
23-21 —	Reserved
20-0 FD_TXCRC	Extended Transmitted CRC value Contains the CRC value calculated over the most recent transmitted message. Different CRC polynomials are used for different frame formats. For CRC_15 and CRC_17, the six most significant bits and the four most significant bits are reported as zeroes, respectively. For CRC_15, this field has the same content as Cyclic Redundancy Check (CRCR).

52.5.2.42 Enhanced RX FIFO Control (ERFCR)

Offset

Register	Offset
ERFCR	C0Ch

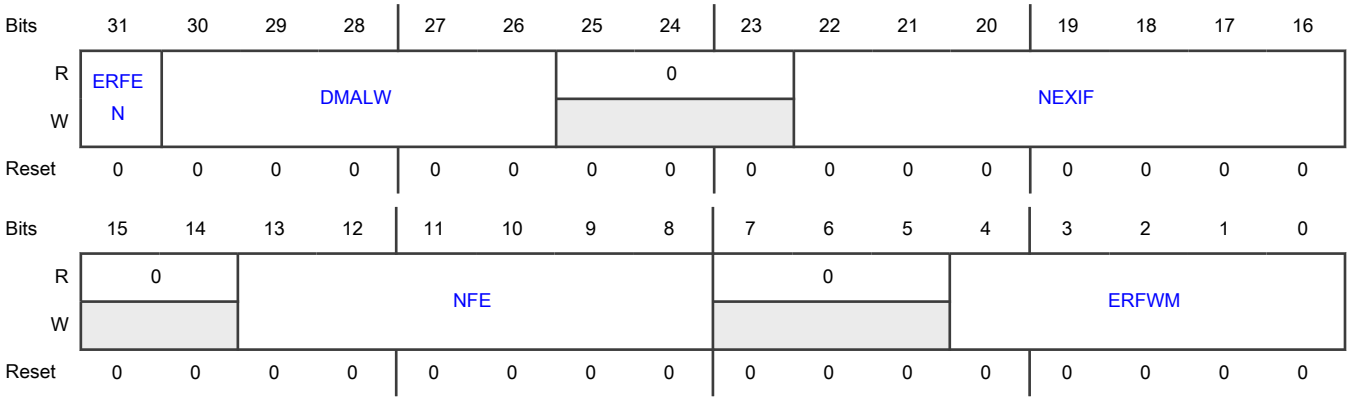
Function

Defines the Enhanced RX FIFO configuration.

This register can be written only in Freeze mode.

Soft reset does not affect any of the contents of this register.

Diagram



Fields

Field	Function
31 ERFEN	Enhanced RX FIFO enable Enables the Enhanced RX FIFO. **NOTE** If MCR[RFEN] = 1, do not write 1 to this field. 0b - Disable 1b - Enable
30-26 DMALW	DMA Last Word Defines the last DMA address for each Enhanced RX FIFO element. This table shows the number of elements and the last address for each Enhanced RX FIFO element according to the value of DMALW.

DMALW	Number of 32-bit words transferred	Last FIFO address
0	1	2000h
1	2	2004h
2	3	2008h
3	4	200Ch
4	5	2010h
5	6	2014h
6	7	2018h
7	8	201Ch
8	9	2020h

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr>
<tr><td>9</td><td>10</td><td>2024h</td></tr>
<tr><td>10</td><td>11</td><td>2028h</td></tr>
<tr><td>11</td><td>12</td><td>202Ch</td></tr>
<tr><td>12</td><td>13</td><td>2030h</td></tr>
<tr><td>13</td><td>14</td><td>2034h</td></tr>
<tr><td>14</td><td>15</td><td>2038h</td></tr>
<tr><td>15</td><td>16</td><td>203Ch</td></tr>
<tr><td>16</td><td>17</td><td>2040h</td></tr>
<tr><td>17</td><td>18</td><td>2044h</td></tr>
<tr><td>18</td><td>19</td><td>2048h</td></tr>
</table>
NOTE
Undefined DMALW values in the table are reserved and must not be used.
</td></tr>
<tr><td>25-23
—</td><td>Reserved</td></tr>
<tr><td>22-16
NEXIF</td><td>Number of Extended ID Filter Elements
Defines the number of extended ID filter elements used during the Enhanced RX FIFO matching process.
The value of this field must be less than or equal to NFE + 1.
The number of standard ID filter elements is 2 × (NFE - NEXIF + 1).
This table shows the number of extended ID filters and standard ID filters available for Enhanced RX FIFO if all filter elements are used.
<table>
<tr><th>NEXIF</th><th>NFE</th><th>Number of Extended ID filter elements</th><th>Number of Standard ID filter elements</th></tr>
<tr><td>0</td><td>15</td><td>0</td><td>32</td></tr>
<tr><td>1</td><td>15</td><td>1</td><td>30</td></tr>
<tr><td>2</td><td>15</td><td>2</td><td>28</td></tr>
<tr><td>3</td><td>15</td><td>3</td><td>26</td></tr>
<tr><td>4</td><td>15</td><td>4</td><td>24</td></tr>
<tr><td>5</td><td>15</td><td>5</td><td>22</td></tr>
<tr><td>6</td><td>15</td><td>6</td><td>20</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>NEXIF</th><th>NFE</th><th>Number of Extended ID filter elements</th><th>Number of Standard ID filter elements</th></tr>
<tr><td>7</td><td>15</td><td>7</td><td>18</td></tr>
<tr><td>8</td><td>15</td><td>8</td><td>16</td></tr>
<tr><td>9</td><td>15</td><td>9</td><td>14</td></tr>
<tr><td>10</td><td>15</td><td>10</td><td>12</td></tr>
<tr><td>11</td><td>15</td><td>11</td><td>10</td></tr>
<tr><td>12</td><td>15</td><td>12</td><td>8</td></tr>
<tr><td>13</td><td>15</td><td>13</td><td>6</td></tr>
<tr><td>14</td><td>15</td><td>14</td><td>4</td></tr>
<tr><td>15</td><td>15</td><td>15</td><td>2</td></tr>
<tr><td>16</td><td>15</td><td>16</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>15-14
—</td><td>Reserved</td></tr>
<tr><td>13-8
NFE</td><td>Number of Enhanced RX FIFO Filter Elements
Defines the total number of filter elements used during the enhanced RX FIFO matching process according to the table.
<table>
<tr><th>NFE</th><th>Maximum number of extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td>4</td></tr>
<tr><td>2</td><td>3</td><td>6</td></tr>
<tr><td>3</td><td>4</td><td>8</td></tr>
<tr><td>4</td><td>5</td><td>10</td></tr>
<tr><td>5</td><td>6</td><td>12</td></tr>
<tr><td>6</td><td>7</td><td>14</td></tr>
<tr><td>7</td><td>8</td><td>16</td></tr>
<tr><td>8</td><td>9</td><td>18</td></tr>
<tr><td>9</td><td>10</td><td>20</td></tr>
<tr><td>10</td><td>11</td><td>22</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>NFE</th><th>Maximum number of extended ID filter elements (NEXIF = NFE + 1)</th><th>Maximum number of standard ID filter elements (NEXIF = 0)</th></tr>
<tr><td>11</td><td>12</td><td>24</td></tr>
<tr><td>12</td><td>13</td><td>26</td></tr>
<tr><td>13</td><td>14</td><td>28</td></tr>
<tr><td>14</td><td>15</td><td>30</td></tr>
<tr><td>15</td><td>16</td><td>32</td></tr>
</table>
</td></tr>
<tr><td>7-5
—</td><td>Reserved</td></tr>
<tr><td>4-0
ERFWM</td><td>Enhanced RX FIFO Watermark
Defines the minimum number of CAN messages stored in the Enhanced RX FIFO. When that number is reached, ERFSR[ERFWMI] becomes 1.
Minimum number of CAN messages = ERFWM + 1.
NOTE
If MCR[DMA] = 1, write 0h to this field.</td></tr>
</table>

52.5.2.43 Enhanced RX FIFO Interrupt Enable (ERFIER)

Offset

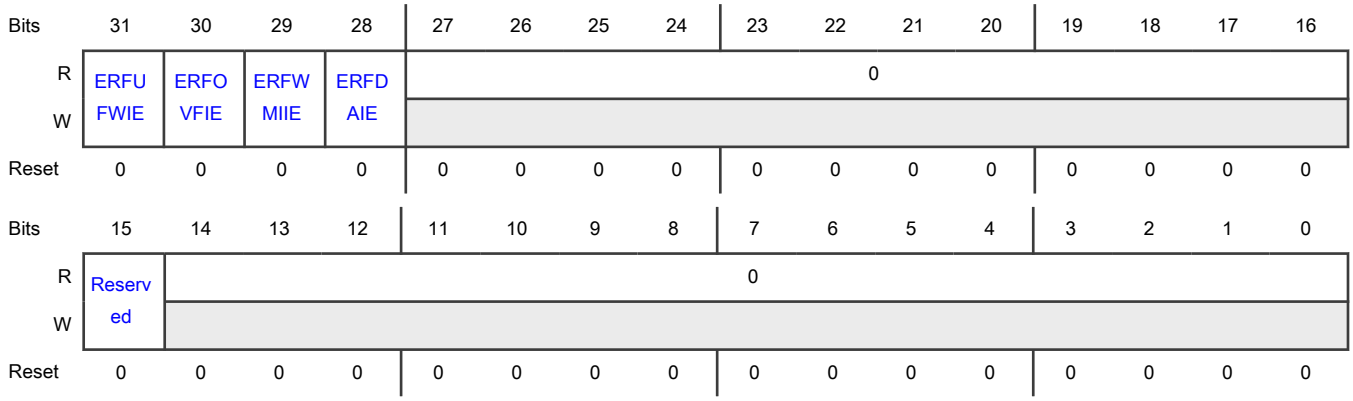
Register	Offset
ERFIER	C10h

Function

Contains the interrupt enables for the Enhanced RX FIFO.

Soft reset does not affect this register.

Diagram



Fields

Field	Function
31 ERFUFWIE	Enhanced RX FIFO Underflow Interrupt Enable Enables interrupt for ERFSR[ERFUFW]. 0b - Disable 1b - Enable
30 ERFOVFIE	Enhanced RX FIFO Overflow Interrupt Enable Enables interrupt for ERFSR[ERFOVF]. 0b - Disable 1b - Enable
29 ERFWMIIE	Enhanced RX FIFO Watermark Indication Interrupt Enable Enables interrupt for ERFSR[ERFWMI]. 0b - Disable 1b - Enable
28 ERFDAIE	Enhanced RX FIFO Data Available Interrupt Enable Enables interrupt for ERFSR[ERFDA]. 0b - Disable 1b - Enable
27-16 —	Reserved
15 —	Reserved
14-0 —	Reserved

52.5.2.44 Enhanced RX FIFO Status (ERFSR)

Offset

Register	Offset
ERFSR	C14h

Function

Contains the status fields of the Enhanced RX FIFO including error indications and a clear FIFO field.

Soft reset does not affect this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERFUFW	ERFOVF	ERFWMI	ERFDA	0	0									ERFE	ERFF
W	W1C	W1C	W1C	W1C	ERFCLR											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	0									ERFEL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ERFUFW	Enhanced RX FIFO Underflow Flag Indicates whether an underflow condition occurred in the enhanced RX FIFO. If ERFIER[ERFUFWIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Underflow
30 ERFOVF	Enhanced RX FIFO Overflow Flag Indicates whether an overflow condition occurred in the Enhanced RX FIFO. If ERFIER[ERFOVFIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Overflow
29	Enhanced RX FIFO Watermark Indication Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERFWMI	Indicates whether the number of messages available in the Enhanced RX FIFO is greater than the watermark defined in ERFCR[ERFWM]. If ERFIER[ERFWMIIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - Number of messages in FIFO is greater than the watermark
28 ERFDA	Enhanced RX FIFO Data Available Flag Indicates whether there is at least one message stored in the ERX FIFO. If ERFIER[ERFDAIE] = 1, this field generates an interrupt. 0b - No such occurrence 1b - At least one message stored in Enhanced RX FIFO
27 ERFCLR	Enhanced RX FIFO Clear Writing 1 to this field during Freeze mode resets the internal FIFO pointers, but the FIFO content stored in RAM is not changed. Writing to this field outside Freeze mode, or writing 0 to this field, has no effect. 0b - No effect 1b - Clear enhanced RX FIFO content
26-18 —	Reserved
17 ERFE	Enhanced RX FIFO Empty Flag Indicates whether Enhanced RX FIFO is empty. 0b - Not empty 1b - Empty
16 ERFF	Enhanced RX FIFO Full Flag Indicates whether enhanced RX FIFO is full. 0b - Not full 1b - Full
15 —	Reserved
14-6 —	Reserved
5-0	Enhanced RX FIFO Elements

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERFEL	Indicates the number of CAN messages stored in the Enhanced RX FIFO.

52.5.2.45 Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL31)

Offset

For n = 0 to 31:

Register	Offset
ERFFELn	3000h + (n × 4h)

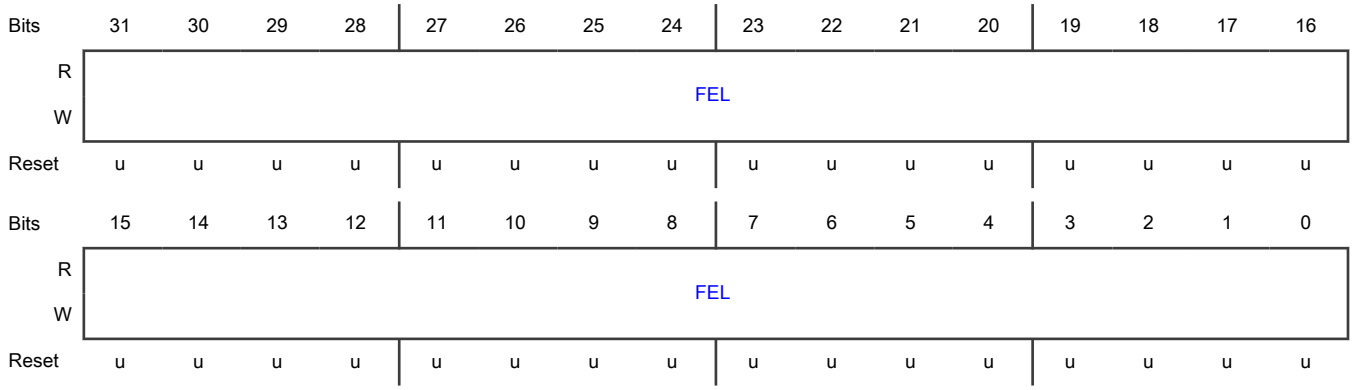
Function

Stores the filter elements of the Enhanced RX FIFO.

For standard ID filtering, each ERFFEL register stores one filter element. For extended ID filtering, each pair of ERFFEL registers stores one filter element.

ERFFEL registers can be written only in Freeze mode; otherwise, the module blocks them. Reset does not affect these registers. They are located in RAM and must be explicitly initialized prior to any reception.

Diagram



Fields

Field	Function
31-0 FEL	Filter Element Bits Stores filter elements. Each filter element is used during the match process. If the matching criteria are met, a message is stored in the Enhanced RX FIFO.

52.5.3 Message buffer structure

The message buffer structure used by FlexCAN is represented in the following figure. Both extended (29-bit identifier) and standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual message

buffer is 16, 24, 40, or 72 bytes, depending on the quantity of data bytes allocated for the message payload: 8, 16, 32, or 64 data bytes, respectively.

The memory area 80h–27Fh is used by the message buffers. When CAN FD is enabled, the exact address for each message buffer depends on the size of its payload. See FlexCAN memory partition for CAN FD.

Table 352. Message buffer structure example with 64-byte payload

0h	EDL	BRS	ESI		CODE		SRR	IDE	RTR	DLC	TIMESTAMP
4h	PRIO			ID (standard/extended)						ID (extended)	
8h	Data byte 0				Data byte 1					Data byte 2	Data byte 3
Ch	Data byte 4				Data byte 5					Data byte 6	Data byte 7
10h	Data byte 8				Data byte 9					Data byte 10	Data byte 11
14h	Data byte 12				Data byte 13					Data byte 14	Data byte 15
18h	Data byte 16				Data byte 17					Data byte 18	Data byte 19
1Ch	Data byte 20				Data byte 21					Data byte 22	Data byte 23
20h	Data byte 24				Data byte 25					Data byte 26	Data byte 27
24h	Data byte 28				Data byte 29					Data byte 30	Data byte 31
28h	Data byte 32				Data byte 33					Data byte 34	Data byte 35
2Ch	Data byte 36				Data byte 37					Data byte 38	Data byte 39
30h	Data byte 40				Data byte 41					Data byte 42	Data byte 43
34h	Data byte 44				Data byte 45					Data byte 46	Data byte 47
38h	Data byte 48				Data byte 49					Data byte 50	Data byte 51
3Ch	Data byte 52				Data byte 53					Data byte 54	Data byte 55
40h	Data byte 56				Data byte 57					Data byte 58	Data byte 59
44h	Data byte 60				Data byte 61					Data byte 62	Data byte 63
		= Unimplemented or reserved									

Table 353. Field descriptions

Mnemonic	Field	Description
EDL	Extended Data Length	Distinguishes between CAN format and CAN FD format frames. EDL must not be 1 for message buffers configured to RANSWER with code field 1010b (see Table 354).
BRS	Bit Rate Switch	Defines whether the bit rate is switched inside a CAN FD format frame.
ESI	Error State Indicator	Indicates whether the transmitting node is error-active or error-passive.
CODE	Message Buffer Code	Can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 354 and Table 355. See Functional description.

Table 354. Message buffer code for RX buffers

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
0000b: INACTIVE. Message buffer is not active.	INACTIVE	—	—	—	Message buffer does not participate in the matching process.
0100b: EMPTY. Message buffer is active and empty.	EMPTY	—	FULL	—	When a frame is received successfully (after Move-in), CODE is automatically updated to FULL.
0010b: FULL. Message buffer is full.	FULL	Yes	FULL	—	The act of reading the Control and Status word followed by unlocking the message buffer (SRV) does not make CODE return to EMPTY. It remains FULL. If a new frame is moved to the message buffer after the message buffer is serviced, the code remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	—	If the message buffer is FULL and a new frame is moved to this message buffer before the CPU services it, CODE is automatically updated to OVERRUN. See Matching process for details about overrun behavior.
0110b: OVERRUN. Message buffer is being overwritten into a full buffer.	OVERRUN	Yes	FULL	—	If CODE indicates OVERRUN and the CPU has serviced the message buffer, when a new frame is moved to the message buffer, CODE returns to FULL.
		No	OVERRUN	—	If CODE already indicates OVERRUN, and another new frame must be moved, the message buffer is overwritten

Table continues on the next page...

Table 354. Message buffer code for RX buffers (continued)

CODE description	RX code BEFORE RX new frame	SRV[1]	RX code AFTER successful reception[2]	RRS[3]	Comment
					again, and CODE remains OVERRUN. See Matching process for details about overrun behavior.
1010b: RANSWER[4]. A frame was configured to recognize a Remote Request frame and transmit a Response frame in return.[5]	RANSWER	—	TANSWER (1110b)	0	A Remote Answer was configured to recognize a Remote Request frame received. After that, a message buffer is set to transmit a response frame. CODE is automatically changed to TANSWER (1110b). See Matching process for details. If CTRL2[RRS] = 0, transmit a response frame when a remote request frame with the same ID is received.
		—	—	1	This code is ignored during matching and arbitration process. See Matching process for details.
CODE[0] = 1: BUSY. FlexCAN is updating the contents of the message buffer. The CPU must not access the message buffer.	BUSY[6]	—	FULL	—	Indicates that the message buffer is being updated. It automatically becomes 0 and does not interfere with the next CODE.
		—	OVERRUN	—	

1. SRV: Serviced message buffer. Message buffer was read and unlocked by reading TIMER or other message buffer.
2. A frame is considered a successful reception after the frame is moved to a message buffer (move-in process). See Move-in.
3. Remote Request Stored field. See Control 2 (CTRL2).
4. Code 1010b is not considered TX and a message buffer with this code should not be aborted.
5. Code 1010b must be used in message buffers configured in CAN FD format, with EDL = 1.
6. For TX message buffers, the BUSY bit should be ignored upon read, except when MCR[AEN] = 1. If this field is 1, the corresponding message buffer does not participate in the matching process.

Table 355. Message buffer code for TX buffers

CODE Description	TX Code BEFORE TX frame	MB RTR	TX Code AFTER successful transmission	Comment
1000b: INACTIVE. Message buffer is not active.	INACTIVE	—	—	Message buffer does not participate in arbitration process.
1001b: ABORT. Message buffer is aborted.	ABORT	—	—	Message buffer does not participate in arbitration process.
1100b: DATA. Message buffer is a TX data frame (MB RTR must be 0).	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the message buffer automatically returns to the INACTIVE state.
1100b: REMOTE. Message buffer is a Transmit Remote Request frame (MB RTR must be 1).	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the message buffer automatically becomes an RX Empty message buffer with the same ID.
1110b: TANSWER. Message buffer is a Transmit Response frame from an incoming Remote Request frame.	TANSWER	—	RANSWER	This intermediate code is automatically written to the message buffer by the CHI as a result of a match to a Remote Request frame. The Remote Response frame is transmitted unconditionally once, then the code automatically returns to RANSWER (1010b). The CPU can also write this code with the same effect. The Remote Response frame can be a data frame or another remote request frame, depending on the value of RTR. See Matching process and Arbitration process for details.

Table 356. RX and TX message buffer field descriptions

Mnemonic	Field	Description
SRR	Substitute Remote Request	Fixed recessive bit, used only in extended format. Write 1 to SRR for transmission (TX Buffers). SRR is stored with the value received on the CAN

Table continues on the next page...

Table 356. RX and TX message buffer field descriptions (continued)

Mnemonic	Field	Description
		bus for RX receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, it is interpreted as an arbitration loss. 1: Recessive value is compulsory for transmission in extended format frames. 0: Dominant is not a valid value for transmission in extended format frames.
IDE	ID Extended Bit	Identifies whether the frame format is standard or extended. 1: Frame format is extended 0: Frame format is standard
RTR	Remote Transmission Request	Affects the behavior of remote frames and is part of the reception filter. See Table 354, Table 355, and CTRL2[RRS]. If FlexCAN transmits this field as 1 (recessive) and receives it as 0 (dominant), it is interpreted as an arbitration loss. If this field is transmitted as 0 (dominant) and it is received as 1 (recessive), FlexCAN treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission. 1: If message buffer is TX, indicates that the current message buffer may have a Remote Request frame to be transmitted. If the message buffer is RX, incoming remote request frames may be stored. 0: Indicates that the current message buffer has a Data frame to be transmitted. In an RX message buffer, it may be considered in matching processes. **NOTE** When configuring CAN FD frames, this field must be 0.
DLC	Data Length Code	Indicates the length (in bytes) of the RX or TX data, which is located in offset 8h–Fh of the message buffer space (see Table 352). In reception, this field is written by FlexCAN, copied from the DLC field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 357).
TIMESTAMP	Free-Running Counter Timestamp	Provides a copy of the Free-Running Timer, captured for TX and RX frames when the beginning of the Identifier field appears on the CAN bus.
PRIO	Local priority	Used only when MCR[LPRIOEN] = 1, and only makes sense for transmit message buffers. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.
ID	Frame Identifier	In standard frame format, only the 11 most significant bits are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.

Table continues on the next page...

Table 356. RX and TX message buffer field descriptions (continued)

Mnemonic	Field	Description
DATA BYTE 0–63	Data Field	Up to 64 bytes can be used for a data frame, depending on the size of payload selected for the message buffers. For RX frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC, as shown in Table 357.

Table 357. DATA BYTE validity

DLC	Valid data bytes
0	None
1	DATA BYTE 0
2	DATA BYTE 0–1
3	DATA BYTE 0–2
4	DATA BYTE 0–3
5	DATA BYTE 0–4
6	DATA BYTE 0–5
7	DATA BYTE 0–6
8	DATA BYTE 0–7
9	DATA BYTE 0–11
10	DATA BYTE 0–15
11	DATA BYTE 0–19
12	DATA BYTE 0–23
13	DATA BYTE 0–31
14	DATA BYTE 0–47
15	DATA BYTE 0–63

52.5.4 FlexCAN memory partition for CAN FD

When CAN FD is enabled, FlexCAN RAM can be partitioned into blocks of 512 bytes each. Each block can accommodate a number of message buffers depending on the configuration provided by FDCTRL[MBDSR*n*] as shown in Table 358.

Table 358. RAM partition

RAM block	Number of MBs with 8 bytes (default range)	Size control field in FDCTRL	Number of MBs of different sizes, per block
0	0 to 31	MBDSR0	MBDSR0 = 00, 32 MBs with 8-byte payload MBDSR0 = 01, 21 MBs with 16-byte payload MBDSR0 = 10, 12 MBs with 32-byte payload MBDSR0 = 11, 7 MBs with 64-byte payload

Payload sizes of 16, 32, or 64 bytes may be configured in some or all of RAM blocks. In those cases, the total number of MBs and their respective number order may differ from the default configuration of 8 bytes. Consider an example where:

- Block0 is configured to an 8-byte payload
- Block1 is configured to a 16-byte payload

In this case, Table 359 indicates how the message buffers are arranged in RAM.

Table 359. RAM partition example

RAM block	Payload size	Number of MBs in the RAM block	Message buffer range
0	FDCTRL[MBDSR0] = 00, 8-byte payload	32	0 to 31

52.5.5 FlexCAN message buffer memory map

The FlexCAN memory buffers are allocated in memory according to the tables below.

Table 360. 8-byte message buffers

Address offset (hex)	MBDSR = b00 8-byte payload
0080	MB0
0090	MB1
00A0	MB2
00B0	MB3
00C0	MB4
00D0	MB5
00E0	MB6
00F0	MB7
0100	MB8
0110	MB9
0120	MB10
0130	MB11
0140	MB12
0150	MB13
0160	MB14
0170	MB15
0180	MB16
0190	MB17
01A0	MB18

Table continues on the next page...

Table 360. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
01B0	MB19
01C0	MB20
01D0	MB21
01E0	MB22
01F0	MB23
0200	MB24
0210	MB25
0220	MB26
0230	MB27
0240	MB28
0250	MB29
0260	MB30
0270	MB31

Table 361. 16-byte message buffers

Address offset (hex)	MBDSR = b01 16-byte payload
0080	MB0
0098	MB1
00B0	MB2
00C8	MB3
00E0	MB4
00F8	MB5
0110	MB6
0128	MB7
0140	MB8
0158	MB9
0170	MB10
0188	MB11
01A0	MB12
01B8	MB13

Table continues on the next page...

Table 361. 16-byte message buffers (continued)

Address offset (hex)	MBDSR = b01 16-byte payload
01D0	MB14
01E8	MB15
0200	MB16
0218	MB17
0230	MB18
0248	MB19
0260	MB20

Table 362. 32-byte message buffers

Address offset (hex)	MBDSR = b10 32-byte payload
0080	MB0
00A8	MB1
00D0	MB2
00F8	MB3
0120	MB4
0148	MB5
0170	MB6
0198	MB7
01C0	MB8
01E8	MB9
0210	MB10
0238	MB11

Table 363. 64-byte message buffers

Address offset (hex)	MBDSR = b11 64-byte payload
0080	MB0
00C8	MB1
0110	MB2
0158	MB3

Table continues on the next page...

Table 363. 64-byte message buffers (continued)

Address offset (hex)	MBDSR = b11 64-byte payload
01A0	MB4
01E8	MB5
0230	MB6

52.5.6 Legacy RX FIFO structure

When MCR[RFEN] = 1, the memory area 80h–DCh (which is normally occupied by MBs 0–5) is used by the reception Legacy FIFO engine.

The region 80h–8Ch contains the output of the Legacy RX FIFO, which the CPU must read as a message buffer. This output contains the oldest message that has been received but not yet read. The region 90h–DCh is reserved for internal use of the Legacy RX FIFO engine.

An additional memory area, which starts at E0h and may extend up to 2DCh (normally occupied by MBs 6–37) depending on the value of CTRL2[RFFN], contains the ID filter table (configurable from 8 to 128 table elements) that specifies filtering criteria for accepting frames into the Legacy RX FIFO.

Out of reset, the ID filter table flexible memory area defaults to E0h and extends only to FCh, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Legacy RX FIFO data structure.

Table 364. Legacy RX FIFO structure

80h	IDHIT		SRR	IDE	RTR	DLC	TIMESTAMP
84h		ID standard				ID extended	
88h	Data byte 0		Data byte 1			Data byte 2	Data byte 3
8Ch	Data byte 4		Data byte 5			Data byte 6	Data byte 7
90h–DCh	Reserved						
E0h	ID filter table element 0						
E4h	ID filter table element 1						
E8h–2D4h	ID filter table elements 2 to 125						
2D8h	ID filter table element 126						
2DCh	ID filter table element 127						
		= Unimplemented or reserved					

Each ID filter table element occupies an entire 32-bit word. One, two, or four Identifier Acceptance Filters (IDAF) can compound each element, depending on MCR[IDAM]. The following tables show the IDAF indexing.

Table 365 shows the three different formats of the ID table elements. All elements of the table must have the same format. See Legacy RX FIFO for more information.

Table 365. ID table structure

A	RTR	IDE	RXIDA				
B	RTR	IDE	RXIDB_0	RTR	IDE	RXIDB_1	
C	RXIDC_0		RXIDC_1	RXIDC_2		RXIDC_3	
		= Unimplemented or Reserved					

Table 366. Field descriptions

Mnemonic	Field	Description
RTR	Remote Frame	Specifies whether remote frames are accepted into the Legacy FIFO if they match the target ID. 1: Remote frames can be accepted and data frames are rejected. 0: Remote frames are rejected and data frames can be accepted.
IDE	Extended Frame	Specifies whether extended or standard frames are accepted into the Legacy FIFO if they match the target ID. 1: Extended frames can be accepted and standard frames are rejected. 0: Extended frames are rejected and standard frames can be accepted.
RXIDA	RX Frame Identifier (Format A)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, only the 11 most significant bits (to) are used for frame identification. In the extended frame format, all bits are used.
RXIDB_0, RXIDB_1	RX Frame Identifier (Format B)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (to and to) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.
RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3	RX Frame Identifier (Format C)	Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.
IDHIT	Identifier Acceptance Filter Hit Indicator	Indicates which identifier acceptance filter the received message in the output of the Legacy RX FIFO hit. See Legacy RX FIFO for more information.

52.5.7 Enhanced RX FIFO structure

When ERFCR[ERFEN] = 1, the Enhanced RX FIFO is enabled. The region 2000h–2048h contains the output of the Enhanced RX FIFO, which the CPU must read as a message buffer. This output contains the oldest message that has been received but not yet read.

Table 367. Enhanced RX FIFO structure

2000h	EDL	BRS	ESI	Reserved	SRR	IDE	RTR	DLC	TIMESTAMP LEGACY
2004h	Reserved			ID (standard/extended)					ID (extended)

Table continues on the next page...

Table 367. Enhanced RX FIFO structure (continued)

2008h	Data byte 0	Data byte 1	Data byte 2	Data byte 3
200Ch	Data byte 4	Data byte 5	Data byte 6	Data byte 7
2010h	Data byte 8	Data byte 9	Data byte 10	Data byte 11
2014h	Data byte 12	Data byte 13	Data byte 14	Data byte 15
2018h	Data byte 16	Data byte 17	Data byte 18	Data byte 19
201Ch	Data byte 20	Data byte 21	Data byte 22	Data byte 23
2020h	Data byte 24	Data byte 25	Data byte 26	Data byte 27
2024h	Data byte 28	Data byte 29	Data byte 30	Data byte 31
2028h	Data byte 32	Data byte 33	Data byte 34	Data byte 35
202Ch	Data byte 36	Data byte 37	Data byte 38	Data byte 39
2030h	Data byte 40	Data byte 41	Data byte 42	Data byte 43
2034h	Data byte 44	Data byte 45	Data byte 46	Data byte 47
2038h	Data byte 48	Data byte 49	Data byte 50	Data byte 51
203Ch	Data byte 52	Data byte 53	Data byte 54	Data byte 55
2040h	Data byte 56	Data byte 57	Data byte 58	Data byte 59
2044h	Data byte 60	Data byte 61	Data byte 62	Data byte 63
IH_OFF	Reserved			ID HIT
204Ch	11 Enhanced FIFO Elements (Reserved)			
...				
238Ch				

NOTE

ID HIT offset change dynamically according to data length code (DLC) as shown in Table 368.

Table 368. ID HIT offset

Data Length Code (DLC)	ID HIT offset (IH_OFF)
0	2008h
1–4	200Ch
5–8	2010h
9	2014h
10	2018h
11	201Ch
12	2020h
13	2028h
14	2038h
15	2048h

Table 369. Field descriptions

Mnemonic	Field	Description
EDL	Extended Data Length	Distinguishes between classical CAN format and CAN FD format frames. 0: Classical CAN frame format 1: CAN FD frame format
BRS	Bit Rate Switch	Defines whether the bit rate is switched inside a CAN FD format frame. 0: Bit rate is not switched in a CAN FD frame. 1: Bit rate is switched in a CAN FD frame.
ESI	Error State Indicator	Indicates whether the transmitting node is error-active or error-passive. This field is meaningful only if EDL = 1. 0: Error-active 1: Error-passive
SRR	Substitute Remote Request	Fixed recessive bit, used only in extended format. Transmitting nodes always send it as recessive and receiving nodes can receive it as either recessive or dominant. If FlexCAN receives this bit as dominant, it is interpreted as an arbitration loss.
IDE	ID Extended Bit	Identifies whether the frame format is standard or extended. 0: Standard 1: Extended
RTR	Remote Frame	Identifies whether the current frame is a data frame or a remote request. 0: Data frame 1: Remote request
DLC	Data Length Code	Defines the number of bytes in the data field of a CAN frame (Data byte 0 to Data byte 63). When RTR = 1, the frame is a remote request and does not include the data field, regardless of the DLC field. See Table 357 for more details.
TIMESTAMP	16-bit Timestamp	Provides a copy of the Free-Running Timer, captured during the CAN frame.
ID	Frame Identifier	In base frame format, only the 11 most significant bits are used for frame identification. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification.
DATA BYTE 0–63	Data Field	Up to 64 bytes can be stored in the data field.
IDHIT	Identifier Acceptance Filter Hit Indicator	Indicates which Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL31) the received message in the output of the Enhanced RX FIFO hit. For each filter region, standard-ID filter space, and extended-ID filter space, there is an independent index starting from zero. Table 370 shows how FlexCAN writes IDHIT according to each filter element.

Table 370. IDHIT for Enhanced RX FIFO

Enhanced RX FIFO filter element - ERFFEL	IDHIT value	Filter element type
ERFFEL0	0	Extended-ID
ERFFEL1	1	Extended-ID
. . .	. . .	Extended-ID
ERFFEL(m-1)	m-1	Extended-ID
ERFFEL(m)	0	Standard-ID
ERFFEL(m+1)	1	Standard-ID
. . .	. . .	Standard-ID
ERFFEL(2n-m+1)	2x(n-m)+1	Standard-ID

NOTE

Where m = NEXIF and n = NFE. If NEXIF = 0, only standard-ID filter elements exist. If NEXIF > NFE, only extended-ID filter elements exist.

52.6 Initialization and application information

52.6.1 FlexCAN initialization sequence

For any configuration change or initialization, you must put FlexCAN into Freeze mode (see Freeze mode). The module must be initialized after every reset. FlexCAN memory must be initialized before switching to functional mode.

The following is a generic initialization sequence applicable to FlexCAN:

1. Initialize Module Configuration (MCR).
    a. Enable the individual filtering per message buffer and reception queue features by writing 1 to MCR[IRMQ].
    b. Enable the warning interrupts by writing 1 to MCR[WRNEN].
    c. If required, disable frame self-reception by writing 1 to MCR[SRXDIS].
    d. Enable the Legacy RX FIFO by writing 1 to MCR[RFEN] or enable the Enhanced RX FIFO by writing 1 to ERFCR[ERFEN].
    e. If Legacy RX FIFO or Enhanced RX FIFO is enabled and DMA is required, write 1 to MCR[DMA].
    f. If Pretended Networking mode is required, write 1 to MCR[PNET_EN].
    g. Enable the abort mechanism by writing 1 to MCR[AEN].
    h. Enable the local priority feature by writing 1 to MCR[LPRIOEN].
2. Initialize Control 1 (CTRL1) and CAN FD Bit Timing (FDCBT). Optionally initialize CAN Bit Timing (CBT).
    a. Determine the bit timing parameters: CTRL1[PROPSEG], CTRL1[PSEG1], CTRL1[PSEG2], and CTRL1[RJW].

b. Optionally determine the bit timing parameters: CBT[EPROPSEG], CBT[EPSEG1], CBT[EPSEG2], and CBT[ERJW].

c. Determine the CAN FD bit timing parameters: FDCBT[FPROPSEG], FDCBT[FPSEG1], FDCBT[FPSEG2], and FDCBT[FRJW].

d. Determine the bit rate by programming CTRL1[PRESDIV] and optionally programming CBT[EPRESDIV].

e. Determine the CAN FD bit rate by programming FDCBT[FPRESDIV].

f. Determine the internal arbitration mode by programming CTRL1[LBUF].

3. Initialize the message buffers. (See Message buffer structure for message buffer details.)

   a. The control and status word of all message buffers must be initialized.

   b. If RX FIFO is enabled, the ID filter table must be initialized.

   c. Other entries in each message buffer should be initialized as required.

4. Initialize Receive Individual Mask (RXIMR0 - RXIMR31).

5. Write 1 to required interrupt mask bits in:

   - IMASK registers (for all message buffer interrupts)
   - Module Configuration (MCR) (for wake-up interrupt)
   - Control 1 (CTRL1) and Control 2 (CTRL2) (for Bus Off and Error interrupts)

6. If Pretended Networking mode is enabled, configure the necessary registers for selective wake-up.

7. Write 0 to MCR[HALT].

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

52.7 Glossary

Active message buffer	A message buffer is active if it can participate in the current matching or arbitration process.
Bus interface unit	FlexCAN submodule responsible for the interface to the CPU.
Bus off	See CAN Specification.[3]
CAN	Controller Area Network, a serial communication protocol defined in CAN Specification[3] and ISO International Standard.[4]
CHI	Controller-host interface, a FlexCAN submodule responsible for message buffer matching and arbitration algorithms.
CRC	Cyclic redundancy check
Dominant bit	A dominant bit wins the arbitration on the CAN bus. It is transmitted as 0.
Doze mode	A system low-power mode where the CPU bus remains active and a global Doze mode request is sent to all peripherals asking them to enter low power mode.
Error active	See CAN Specification.[3]
Error frame	See CAN Specification.[3]
Error passive	See CAN Specification.[3]
Form error	See CAN Specification.[3]

[3] Controller Area Network - CAN Specification Version 2.0 Part A, Part B, Robert Bosch GmbH, 1991.
[4] ISO International Standard - ISO 11898 First Edition 1993 Road Vehicles - Interchange of Digital Information - Controller Area Network (CAN) for high-speed Communication.

Hard reset	A reset coming from an external pin and/or following power-on. It resets everything.
Idle	See CAN Specification.[3]
Information processing time	See CAN Specification.[3]
Intermission	See CAN Specification.[3]
Matching elements	Data used in the matching process, such as ID, filter, mask, etc.
MB	See Message buffer.
Message buffer	Internal FlexCAN data structure containing bytes received from, or to be transmitted to, the CAN line, as well as information about this data.
Overload frame	See CAN Specification.[3]
Phase buffer segment	See CAN Specification.[3]
Recessive bit	A recessive bit loses the arbitration on the CAN bus. It is transmitted as 1.
Sclock	Serial clock. Obtained by dividing the clock feeding the CAN engine (oscillator or bus clock) by a prescaler factor. The Sclock period defines the time quantum for CAN protocol timing.
Soft reset	Global reset typically used by peripherals to reinitialize some of its registers, but not all of them.
Stop mode	A system low-power mode in which all chip clocks are stopped for maximum power savings.
Stuffing error	See CAN Specification.[3]
Time quantum	This is equal to the Sclock period. It is the minimum time period used to compose the CAN protocol bit timing.

Chapter 53
Flexible I/O (FlexIO)

53.1 Chip-specific FLEXIO information

Table 371. Reference links to related information

Topic	Related module	Reference
Full description	FLEXIO	FLEXIO
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

53.1.1 Module instances

This device has one FlexIO module, FLEXIO0.

53.2 Overview

Flexible I/O (FlexIO) is a highly configurable module providing a wide range of functionality including:

- Emulation of various serial or parallel communication protocols
- Flexible 16-bit timers with support for various trigger, reset, enable, and disable conditions
- Programmable logic blocks allowing the implementation of digital logic functions on-chip, and configurable interaction of internal and external modules
- Programmable state machine for offloading basic system control functions from CPU

53.2.1 Block Diagram

The following diagram gives a high-level overview of the FlexIO timers and shifters configuration.

FlexIO uses shifters, timers, and external triggers to shift data into or out of the FlexIO. Timers control the timing of the data shift, as shown in the block diagram. The timers can be configured to use generic timer functions, external triggers, or various other conditions to determine this control.

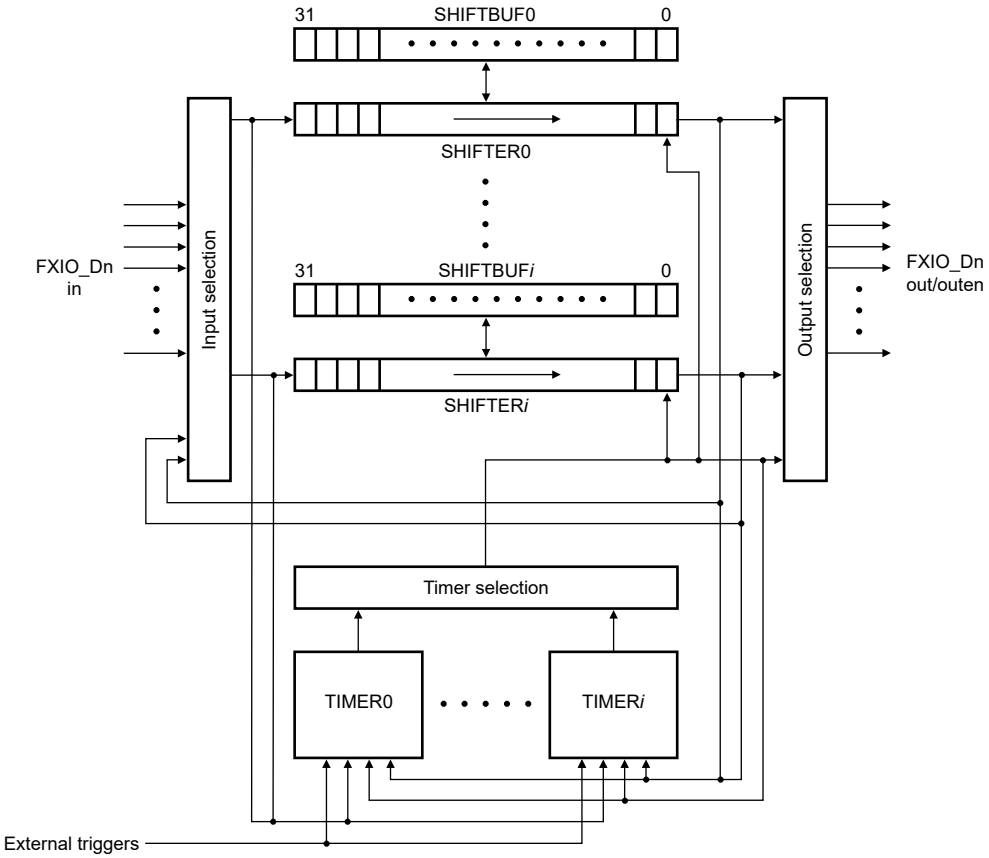


Figure 202. FlexIO block diagram

53.2.2 Features

FlexIO can support a wide range of protocols including, but not limited to:

- UART
- I2C
- SPI
- I2S
- Camera IF
- Motorola 68K or Intel 8080 bus
- PWM or waveform generation
- Input-capture (pulse edge interval measurement), such as SENT

FlexIO includes the following features:

- Array of 32-bit shift registers with transmit, receive, data match, logic, and state modes
- Double-buffered shifter operation for continuous data transfer
- Shifter concatenation to support large transfer sizes
- Automatic start and stop bit generation
- 1, 2, 4, 8, 16, or 32 multi-bit shift widths for parallel interface support

- Interrupt, DMA, or polled transmit and receive operation
- Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during stop modes
- Highly flexible 16-bit timers with support for various internal or external trigger, reset, enable, and disable conditions
- Programmable logic mode for integrating external digital logic functions on-chip or combining pin, shifter, or timer functions to generate complex outputs
- Programmable state machine for offloading basic system control functions from CPU with support for up to eight states, eight outputs, and three selectable inputs per state
- Integrated general purpose input/output registers and pin rising or falling edge interrupts to simplify software support

53.3 Functional description

53.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of the FlexIO. The timing of shift, load and store events are controlled by the Timer assigned to the Shifter via the SHIFTCTL[TIMSEL] register. The Shifters are designed to support either DMA, interrupt or polled operation. The following block diagram provides a detailed view of the Shifter microarchitecture.

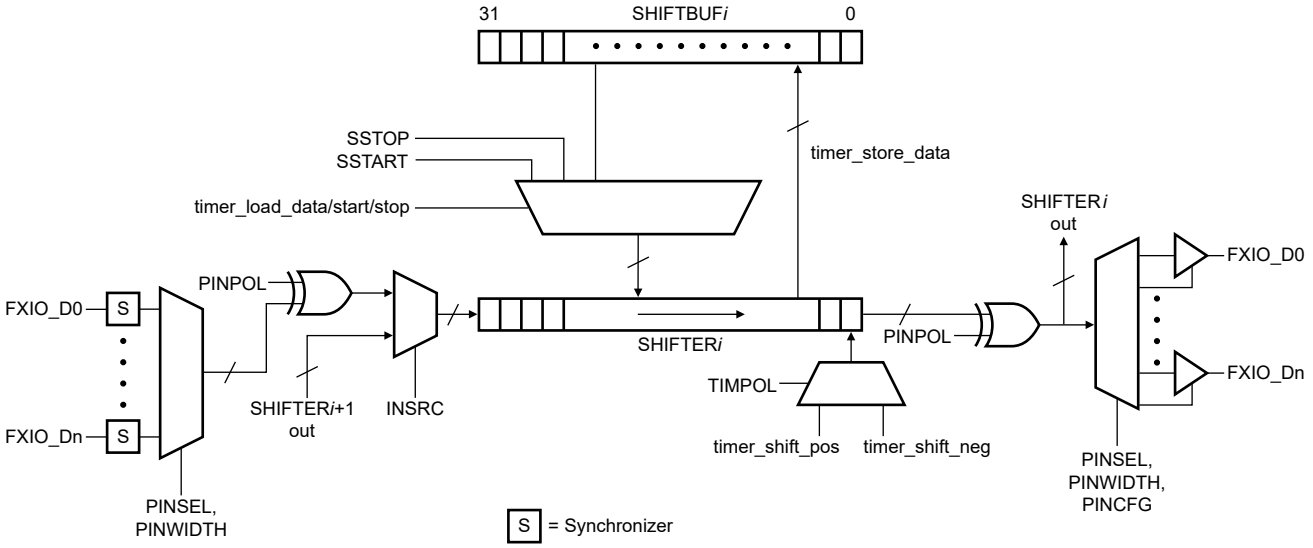


Figure 203. Shifter Microarchitecture

Transmit Mode

When configured for Transmit mode (SHIFTCTL[SMOD]=Transmit), the shifter loads data from the SHIFTBUF register and shift data out when a load event is signalled by the assigned Timer. An optional start/stop bit can also be automatically loaded before/ afterSHIFTBUFdata by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

NOTE

The shifter immediately loads a stop bit when the Shifter is initially configured for Transmit mode if a stop bit is enabled.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has been loaded from the SHIFTBUF register into the Shifter or when the Shifter is initially configured into Transmit mode. The flag clears when new data has been written into the SHIFTBUF register or a logic 1 is written to this flag. In transmit mode, any write of the SHIFTBUF

register clears the corresponding Shifter Status Flag. It does not matter what is writing or the state of the DMA/interrupt enables, the flag is cleared. How the flag is set/cleared for each mode is documented in the SSF register description.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). The flag can be cleared by writing it with logic 1.

Receive Mode

When configured for Receive mode (SHIFTCTL[SMOD]=Receive), the shifter shifts data in and store data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has been stored into the SHIFTBUF register from the Shifter. The flag clears when the data has been read from theSHIFTBUF register or a logic 1 is written to this flag. Any read of the SHIFTBUF register clears the corresponding Shifter Status Flag when Shifter is configured in Receive mode. It does not matter what is reading or the state of the DMA/interrupt enables, the flag is cleared. How the flag is set/clear for each mode is documented in the SSF register description.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store data into a fullSHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

Match Store Mode

When configured for Match Store mode (SHIFTCTL[SMOD]=Match Store), the shifter shifts data in, check for a match result and store matched data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs and matched data has been stored into the SHIFTBUF register from the Shifter. The flag clears when the matched data has been read from the SHIFTBUF register or a logic 1 is written to this flag. Any read of the SHIFTBUF register clears the corresponding Shifter Status Flag when Shifter is configured in Match Store mode. It does not matter what is reading or the state of the DMA/interrupt enables, the flag is cleared. How the flag is set/clear for each mode is documented in the SSF register description.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

Match Continuous Mode

When configured for Match Continuous mode (SHIFTCTL[SMOD]=Match Continuous), the shifter shifts data in and continuously check for a match result whenever a shift event is signalled by the assigned Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when a match occurs. The flag clears automatically as soon as there is no longer a match between Shifter data and SHIFTBUF register. **The flag cannot be cleared by reading the SHIFTBUF register.**

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when a match occurs. The flag clears when there is a read from the SHIFTBUF register or it written with logic 1.

State Mode

Using State mode enables the user to implement any state machine with up to 8 states, 8 outputs and 3 selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU, which could potentially remain in a low power mode.

When configured for State mode (SHIFTCTL[SMOD]=State), the SHIFTBUF register is used to drive the output and compute next state values when the Shifter has been selected by the current state pointer (SHIFTSTATE[STATE]). The following diagram provides a detailed view of Shifter microarchitecture when configured for State mode.

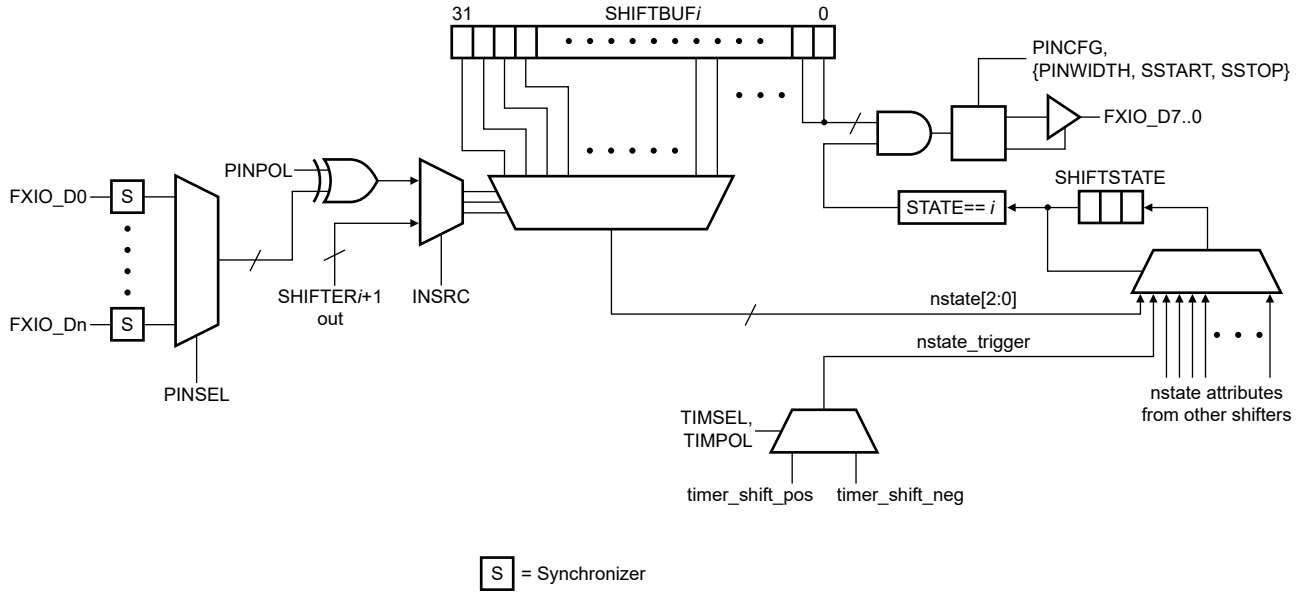


Figure 204. State Microarchitecture

When Shifter i has been selected by the current state pointer, output pins FXIO_D[7:0] are driven by SHIFTBUFi[31:24] using the configuration set by SHIFTCTLi[PINCFG]. When set, SHIFTCFGi{PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} are respectively used to disable the output drive on pins FXIO_D[7:0] for state machine applications which require less than 8 output pins.

The next state value is computed using the 3 input pins selected by SHIFTCTLi[PINSEL] together with SHIFTBUFi[23:0].

NOTE

Each state could potentially use a different set of 3 input pins.

The following table details how the next state value is computed when the current state pointer is pointing to Shifter i.

Table 372. Next State computation for SHIFTSTATE[STATE]=i

FXIO_D[PINSEL+2]	FXIO_D[PINSEL+1]	FXIO_D[PINSEL]	Next State Value
0	0	0	SHIFTBUFi[2:0]
0	0	1	SHIFTBUFi[5:3]
0	1	0	SHIFTBUFi[8:6]
0	1	1	SHIFTBUFi[11:9]
...	...	...	...
1	1	1	SHIFTBUFi[23:21]

Note that other shifters/timers could potentially be configured to drive the input pins of a given state, allowing the user to create complex combinations of shifters/timers as desired e.g. the output of a Shifter configured for logic mode could potentially be used to drive a state machine input.

The next state transition is triggered using the Timer output selected by SHIFTCTLi[TIMSEL] with polarity controlled by SHIFTCTLi[TIMPOL]. Note that each state could potentially use a different Timer to trigger each next state transition, allowing a variety of internal/external trigger sources and clocking configurations to be used (see Timer section for more detail).

The current state pointer defaults to Shifter 0 at reset, however it can be written by the user to select a different Shifter for the initial state. If the current state pointer selects a Shifter which is not configured for State mode, then outputs are not driven and a next state transition is never triggered.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set whenever the Shifter has been selected by the current state pointer. The flag clears when the current state pointer is updated to a different Shifter.

Logic Mode

Using logic mode enables the user to implement a small amount of programmable digital logic within a FlexIO Shifter.

When configured for Logic mode (SHIFTCTL[SMOD]=Logic), theSHIFTBUF register is used to implement a 5-input, 32-bit programmable logic look-up table. The following diagram provides a detailed view of Shifter microarchitecture when configured for Logic mode.

The SHIFTBUFi configures the lookup table for the 4 pin inputs and the SHIFTERi can optionally be used to configure a feedback or delayed pin source as the 5th input to the look up table.

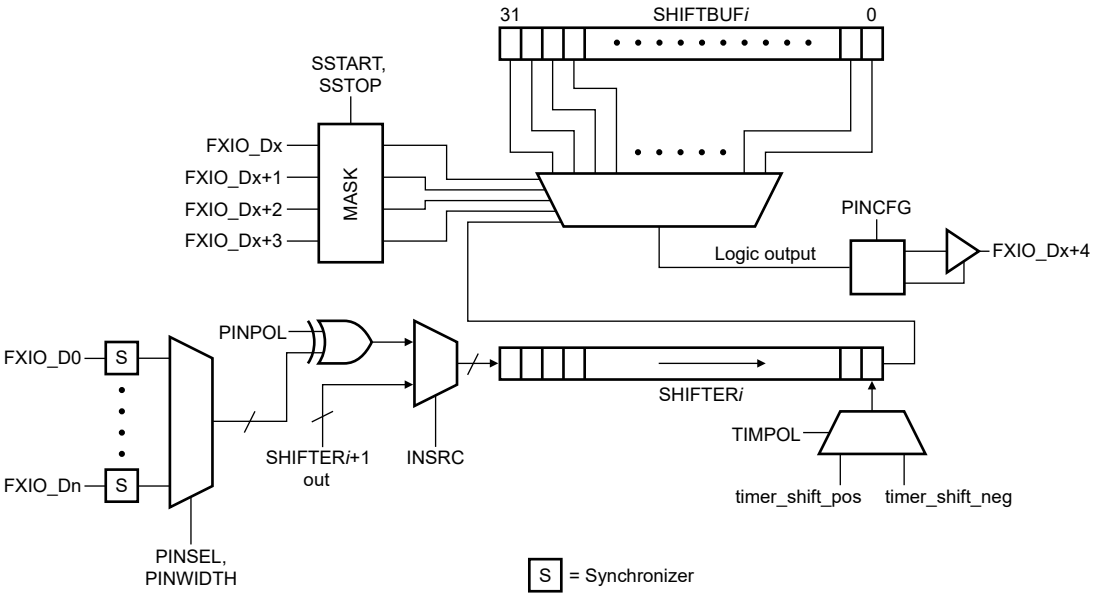


Figure 205. Logic Microarchitecture

The look-up table is driven using 4 pin inputs (maskable using SHIFTCFG[SSTOP] and SHIFTCFG[SSTART]) plus 1 input from the internal shifter and can be configured to drive an output pin using the SHIFTCTL[PINCFG] field. Pin inputs and outputs are fixed for each logic look-up table and are not selectable. The following table lists the logic output value selected by the look-up table for Shifter 'i'.

Table 373. Logic Look-up table for Shifter 'i'

SHIFTERi[0]	FXIO_D[x+3][1]	FXIO_D[x+2]	FXIO_D[x+1]	FXIO_D[x]	Logic Output to FXIO_D[x+4]
0	0	0	0	0	SHIFTBUFi[0]
0	0	0	0	1	SHIFTBUFi[1]
0	0	0	1	0	SHIFTBUFi[2]
0	0	0	1	1	SHIFTBUFi[3]

Table continues on the next page...

Table 373. Logic Look-up table for Shifter 'i' (continued)

...	...	...	...	...	...
1	1	1	1	1	SHIFTBUFi[31]

1. for Shifter i=0...3, x=i

   for Shifter i=4...7, x=i+4

To minimize output glitches, SHIFTCFG[SSTOP] and SHIFTCFG[SSTART] can be used to mask unused input pins. When set, {SSTOP[1:0], SSTART[1:0]} mask FXIO_D[x+3]...FXIO_D[x] inputs respectively, so that any transitions on these pins does not cause the logic output to glitch.

Note that other shifters/timers could potentially be configured to drive the input pins of a given look-up table, allowing the user to concatenate look-up tables or create complex combinations of shifters/timers as desired.

SHIFTCFG[PWIDTH] controls the number of delay stages introduced by the internal shifter input (SHIFTERi[0]). For example, when configured for 1-bit shift (PWIDTH=0), the internal shifter introduces a 32 Shift clock delay before passing its input (selected by SHIFTCTL[PINSEL]) to the look-up table. When configured for 32-bit shift (PWIDTH=16...31), the internal shifter introduces a 1 Shift clock delay to its input.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set whenever the output pin allocated to the logic look-up table has a value of 1 (after being synchronized to the FlexIO clock). The flag clears when the output pin has a value of 0. This also allows the SSF flag to be used as a trigger to a Timer if desired.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts are set when the output pin allocated to the logic look-up table has asserted. The flag can be cleared by writing it with logic 1.

The logic mode input pins (including pins driven by other shifters/timers) are first synchronized to the FlexIO functional clock before they are input to the programmable logic look-up table.

53.3.2 Timer Operation

The FlexIO 16-bit timers control the loading, shifting, and storing of the shift registers. The counters load the contents of the compare register and decrement down to zero on the FlexIO clock. The counters can perform generic timer functions such as generating a clock or select output or a PWM waveform. Timers can be configured to:

- enable in response to a trigger, pin, or shifter condition,
- decrement always or only on a trigger or pin edge,
- reset in response to a trigger or pin condition, or
- disable on a trigger or pin condition or on a timer compare.

Timers can optionally include a start condition and a stop condition.

Although each timer operates independently, a timer can be configured to enable or disable at the same time as the previous timer (for example, timer1 can enable or disable at the same time as timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently and can be configured to be a timer output, shifter status flag, pin input, or an external trigger input. The trigger configuration is separate from the pin configuration which can be configured for input, output data, or output enable. See the chip-specific FlexIO information for details on the external trigger connections.

The Timer Configuration Register (TIMCFGn) should be configured before setting the Timer Mode (TIMOD).

Timer 8-bit Baud Counter mode

In 8-bit Baud Counter Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the baud rate of the shift clock and the upper 8-bits are used to configure the number of shift clock edges in the transfer. When the lower 8-bits decrement to zero, the timer output is toggled and the lower 8-bits reload from the compare register. The upper 8-bits decrement when the lower 8-bits equal zero and decrement.

Note that a timer reset event in 8-bit Baud Counter Mode only resets the lower 8-bit counter, the upper 8-bit counter is not affected and can decrement if the timer reset is configured to update the state of the timer output, and the timer output toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

Timer 8-bit High PWM Mode

In 8-bit High PWM Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the timer output high period and the upper 8-bits are used to configure the timer output low period. The lower 8-bits decrement when the output is high. When the lower 8-bits equal zero and decrement, the timer output is cleared and the lower 8-bits are reloaded from the compare register. The upper 8-bits decrement when the output is low. When the upper 8-bits equal zero and decrement, the timer output is set and the upper 8-bits are reloaded from the compare register.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

Timer 16-bit Counter Mode

In 16-bit Counter Mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (eg: TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (eg: TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer compare event.

Timer 16-bit Counter Disable Mode

In 16-bit Counter Disable Mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (eg: TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (eg: TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer disable event.

Timer 8-bit Word Counter Mode

In 8-bit Word Counter Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the number of shift clock edges in each word and the upper 8-bits are used to configure the number of words in the transfer. When the lower 8-bits decrement to zero, the timer output is toggled and the lower 8-bits reload from the compare register. The upper 8-bits only decrement when the lower 8-bits equal zero and decrement.

A timer compare event occurs when the lower 8-bits equal zero and decrements. The timer status flag is set when the upper 8-bits equal zero and decrements.

Timer 8-bit Low PWM Mode

In 8-bit Low PWM Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the timer output low period and the upper 8-bits are used to configure the timer output high period. The lower 8-bits decrement when the output is low. When the lower 8-bits equal zero and decrement, the timer output is set and the lower 8-bits are reloaded from the compare register. The upper 8-bits decrement when the output is high. When the upper 8-bits equal zero and decrement, the timer output is cleared and the upper 8-bits are reloaded from the compare register.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

Timer 16-bit Input Capture Mode

In 16-bit Input Capture Mode, the 16-bit counter uses a fixed initial value of 0xFFFF to load the counter. Whenever a timer counter disable condition is detected (as configured by TIMDIS), the timer status flag is set and the inverted timer counter value is stored in the timer compare register, then the timer counter is immediately restarted from 0xFFFF. The timer output is always output on the FlexIO trigger outputs (which can also be used by other timers as an input trigger) and can also be output to a pin (PINCFG/PINPOL/PINSEL) or a shifter.

If the timer status flag is already set, then the input capture event is missed. When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads with 0xFFFF. The timer output can be used to count the number of counter overflows between two input capture events. Input capture mode is not intended to be used to control shift registers, it is to enable software to count the number of bits in a variable length transfer or to monitor the period or frequency of an input.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer disable event. Whenever the status flag asserts, the inverted timer counter value is stored in the timer compare register.

Timer Enable and Start Bit

When the TIMOD is configured for the desired mode, and the condition configured by timer enable (TIMENA) is detected then the following events occur. When ONETIM is set, then the timer status flag must also be clear to generate a timer enable event, otherwise the timer enable event is blocked. This can be used to enforce software intervention after each timer iteration.

- Timer counter loads the current value of the compare Register and start decrementing as configured by TIMDEC.
- Timer output may update to its initial state depending on the TIMOUT configuration. Shifters that are controlled by this timer do not see this as a rising edge on the timer shift clock.
- Transmit shifters controlled by this timer either output their start bit value, or load the shift register from the shift buffer and output the first bit, as configured by SSTART.

If the Timer start bit is enabled, the timer counter reloads with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMOUT=1), a shifter that is configured to shift on falling edge and load on the first shift does not load correctly.

Timer Decrement and Reset

The Timer then generates the timer output and timer shift clock depending on the TIMOD and TIMDEC fields. The shifter clock is either equal to the timer output (when TIMDEC != 10 or 11) or equal to the decrement clock (when TIMDEC = 10 or 11). When TIMDEC is configured to decrement from a pin or trigger, the timer decrements on both rising and falling edges.

If a Timer is configured to decrement on the FlexIO functional clock divided by 16 or 256 (when TIMDEC = 100 or 101), then a common prescaler that is shared by all Timers is used to generate the two divide ratios. This prescaler is reset when all Timers are either idle or configured to not use the prescaler (TIMDEC != 100 or 101).

When the Timer is configured to reset as configured in the TIMRST field then the Timer counter loads the current value of the compare register again. The timer output and timer shift clock can be configured to update on timer reset, as configured by TIMOUT. This can result in a timer shift clock edge if the timer output toggles as a result of the timer reset. In 8-bit Baud Counter mode this would also decrement the upper 8-bits of the counter.

In general, when the timer counter decrements to zero a timer compare event is triggered. The timer compare event causes the timer counter to load the contents of the timer compare register, the timer output to toggle, any configured transmit shift registers to load, and any configured receive shift registers to store. Depending on the timer mode, the timer status flag may also be set.

Timer Disable and Stop Bit

When the is Timer is configured to add a stop bit on each compare, the following additional events occur.

- Transmit shifters controlled by this timer output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SSTOP.

- On the first rising edge of the shifter clock after the compare, the timer counter reloads the current value of the Compare Register.

Transmit shifters must be configured to load on the first shift when the timer is configured to insert a stop bit on each compare.

When the condition configured by timer disable (TIMDIS) is detected, the following events occur.

- Timer counter reloads the current value of the Compare Register and start decrementing as configured by TIMDEC.
- Timer output clears. Shifters that are controlled by this timer do not see this as a falling edge on the timer shift clock, but can generate a shift event if the timer shift clock would otherwise generate one.
- Transmit shifters controlled by this timer output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SSTOP.

If the timer stop bit is enabled, the timer counter continues decrementing until the next rising edge of the shift clock is detected, at which point it finishes. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit, but does not generate shift events.

A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). When ONETIM is set, the timer status flag must be clear before the next timer enable condition is detected. Receive shift registers with stop bit enabled store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge while the timer is in the stop state condition. If there is no configured edge between the timer disable and the next rising edge of the shift clock then the final store and verify do not occur.

53.3.3 Pin operation

The pin configuration for each timer and shifter can be configured to use any FlexIO pin with either polarity. Each timer and shifter can be configured as an input, output data, output enable or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity, since the output enable assertion would cause logic zero to be output on the pin) or to control the enable on the bidirectional output. Any timer or shifter could be configured to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

Parallel Interface

Shifters can be configured to use multiple FlexIO pins in parallel using the SHIFTCFG[PWIDTH] field. PWIDTH is used to configure the following settings of a shifter:

1. Number of bits shifted per Shift clock.
2. Number of pins driven by the shifter per Shift clock (only on shifters supporting parallel transmit i.e. SHIFTER0, SHIFTER4).
3. Number of pins sampled by the shifter per Shift clock (only on Shifter supporting parallel receive i.e. SHIFTER3, SHIFTER7).

When configured for parallel shift, either 4, 8, 16 or 32-bits can be shifted on every Shift clock. If an adjacent shifter is selected as the input source (SHIFTCFG[INSRC]=1), the least significant 4, 8, 16 or 32-bits from the adjacent shifter is sampled on each Shift clock.

For shifters supporting parallel receive (SHIFTER3, SHIFTER7), the shifter can be configured to sample multiple pins (SHIFTCFG[INSRC]=0), with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant bits are masked with 0 e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[31:24] samples {0,0,FXIO_D[12:7]} on each Shift clock.

For shifters supporting parallel transmit (SHIFTER0, SHIFTER4), the shifter can be configured to drive multiple pins using SHIFTCTL[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant pins are not driven e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[5:0] drives only FXIO_D[12:7] on each Shift clock.

Pin Synchronization

When configuring a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized to the FlexIO clock before the signal is used by a timer or shifter. This introduces a small latency of between 0.5 to 1.5 FlexIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FlexIO clock cycles.

If an input is used by more than one timer or shifter then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

Note that FlexIO pins are also connected internally, configuring a FlexIO shifter or timer to output data on an unused pin makes an internal connection that allows other shifters and timer to use this pin as an input. This allows a shifter output to be used to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized to the FlexIO clock and therefore incurs a 1 cycle latency.

So when using a Pin input as a Timer Trigger, Timer Clock or Shifter Data Input, the following synchronization delays occur:

1. 0.5 to 1.5 FlexIO clock cycles for external pin
2. 1 FlexIO clock cycle for an internally driven pin

For timing considerations such as output valid time and input setup time for specific applications (SPI Master, SPI Slave, I2C Master, I2S Master, I2S Slave) refer to the FlexIO Application Information Section.

Pin Override

The state of any FlexIO pin can be changed at any time by software. The Pin Output Enable register configures any pin as an output and drives that pin with the value in the Pin Output Data Register.

Alias registers for the Pin Output Enable/Data Registers also exist, writing a logic one to an alias register updates the corresponding register bits in both the Pin Output Enable Register and the Pin Output Data Register as follows.

- Pin Output Disable Register clears Output Enable Register and clear Output Data Register.
- Pin Output Clear Register sets Output Enable Register and clear Output Data Register.
- Pin Output Set Register sets Output Enable Register and set Output Data Register.
- Pin Output Toggle Register sets Output Enable Register and toggle Output Data Register.

Pin Interrupt

The state of any FlexIO pin can be read by software at any time. Software can also configure any pin to set a status flag when either a rising and/or falling edge is detected on that pin. The pin status flag can also be configured to generate an interrupt.

53.3.4 Low power modes

The FlexIO remains functional during low power modes, provided the Doze Enable field (CTRL[DOZEN]) is clear and the FlexIO functional clock remains enabled.

The exception to this is in Deep Sleep mode. In this case, the Doze Enable (CTRL[DOZEN]) field is ignored and the FlexIO waits for all timers to complete any pending operation before acknowledging Deep Sleep mode entry.

53.3.5 Debug mode

The FlexIO remains functional in debug mode, provided the Debug Enable field (CTRL[DBGE]) is set.

53.3.6 Clocks

Functional clock

The FlexIO functional clock is asynchronous to the bus clock and can remain enabled in low power modes. The FlexIO functional clock must be enabled before accessing any FlexIO registers. Provided the FlexIO functional clock is at least equal to the bus clock, the CTRL[FASTACC] field can be set to support fast register accesses.

Bus clock

The bus clock is only used for bus accesses to the control and configuration registers.

53.3.7 Reset

Chip reset

The logic and registers for the FlexIO are reset to their default state on chip reset.

Software reset

The FlexIO implements a software reset field in its Control Register. CTRL[SWRST] resets all logic and registers to their default state, except for the CTRL itself.

53.3.8 Interrupts and DMA requests

The following table shows the status flags that generate the FlexIO interrupt and DMA requests.

Table 374. FlexIO interrupts and DMA requests

Flag	Description	Interrupt	DMA request	Low power wakeup
SSF	Shifter Status Flag.	Y	Y	Y
SEF	Shifter Error Flag.	Y	N	Y
TSF	Timer Status Flag.	Y	Y	Y
PSF	Pin Status Flag.	Y	N	Y
ETSF	External Trigger Status Flag.	Y	N	Y

53.3.9 Peripheral triggers

The connection between the FlexIO peripheral triggers and other peripherals is device-specific.

Output triggers

Each FlexIO timer generates an output trigger equal to the timer output. The output trigger is not affected by the timer pin polarity configuration.

Input trigger

FlexIO supports multiple external trigger inputs that can be used to trigger one or more FlexIO timers. If a rising edge is detected on an external trigger when FlexIO is enabled, then the external trigger status flag is set. The external triggers are synchronized to the FlexIO functional clock and must assert for at least two cycles of the FlexIO functional clock to be sampled correctly.

53.4 Application information

This section provides examples for a variety of FlexIO module applications. See the register descriptions for further details.

53.4.1 UART transmit

UART transmit can be supported using one Timer, one Shifter, and one Pin (two Pins if supporting CTS). The start and stop bit insertion is handled automatically and multiple transfers are supported using DMA controller. The timer status flag is used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention. Before transmitting a break or idle character, the SSTART and SSTOP fields must be altered to transmit the required state, and the data to transmit must equal FFh or 00h. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the number of bits to transmit). When performing byte writes to SHIFTBUFn (or SHIFTBUFBIS for transmitting MSB first), the rest of the register remains unaltered. This allows an address mark bit or additional stop bit to remain undisturbed.

NOTE

FlexIO does not support automatic insertion of parity bits.

Table 375. UART transmit configuration

Register	Value	Comments
SHIFTCFGn	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0003_0002h	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open-drain by setting PINPOL=1h and PINCFG=1h.
TIMCMPn	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0000_2222h	Configure start bit, stop bit, enable on trigger asserted and disable on compare. Can support CTS by configuring TIMEN=3h.
TIMCTLn	01C0_0001h	Configure dual 8-bit counter using Shifter 0 status flag as inverted internal trigger source. Supports CTS by configuring PINSEL=1h (for Pin 1) and PINPOL=1h.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBBS[7:0] register instead.

The following table shows an alternative configuration that supports slower baud rates. This configuration requires two Timers.

Table 376. UART transmit configuration for slow baud rate

Register	Value	Comments
SHIFTCFGn	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0003_0002h	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Invert output data by setting PINPOL. Support open-drain by setting PINPOL=1h and PINCFG=1h.
TIMCMPn	0000_000Fh	Configure for 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0030_2622h	Configure start bit, stop bit, enable on trigger rising edge, decrement on trigger and disable on compare.
TIMCTLn	0740_0003h	Configure 16-bit counter using Timer 1 output as internal trigger source.
TIMCMP(n+1)	0000_0001h	Configure baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (baud rate divider / 2) - 1.
TIMCFG(n+1)	0000_1200h	Configure enable on trigger asserted and disable on Timer 0 disable. Can support CTS by configuring TIMEN=3h.
TIMCTL(n+1)	01C0_0003h	Configure 16-bit counter using Shifter 0 status flag as inverted internal trigger source. Support CTS by configuring PINSEL=1h (for Pin 1) and PINPOL=1h.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Support MSB first transfer by writing to SHIFTBUFBBS[7:0] register instead.

53.4.2 UART receive

UART receive can be supported using one Timer, one Shifter, and one Pin (two Timers and two Pins if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers are supported using the DMA controller. The timer status flag is used to indicate when the stop bit of each word is received.

Triple voting of the received data is not supported by FlexIO; data is sampled only once in the middle of each bit. A separate timer can be used to implement a glitch filter on the incoming data, and a different Timer can be used to detect an idle line of programmable length. Break characters cause the error flag to set and the shifter buffer register returns 00h.

NOTE

FlexIO does not support automatic verification of parity bits.

Table 377. UART receiver configuration

Register	Value	Comments
SHIFTCFGn	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0080_0001h	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0204_2422h	Configure start bit, stop bit, enable on pin posedge and disable on compare. Enable resynchronization to received data with TIMOUT=2h and TIMRST=4h.
TIMCTLn	0000_0081h	Configure dual 8-bit counter using inverted Pin 0 input.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0]. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS[7:0] register instead.

The UART receiver with RTS configuration uses a second Timer to generate the RTS output. The RTS asserts when the start bit is detected and negates when the data is read from the shifter buffer register. If no start bit is detected while the RTS is asserted, the received data is ignored.

Table 378. UART receiver with RTS configuration

Register	Value	Comments
SHIFTCFGn	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0080_0001h	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Invert input data by setting PINPOL.
TIMCMPn	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0204_2522h	Configure start bit, stop bit, enable on pin posedge with trigger asserted and disable on compare. Enable resynchronization to received data with TIMOUT=2h and TIMRST=4h.

Table continues on the next page...

Table 378. UART receiver with RTS configuration (continued)

Register	Value	Comments
TIMCTLn	02C0_0081h	Configure dual 8-bit counter using inverted Pin 0 input. Trigger is internal using inverted Pin 1 input.
TIMCMP(n+1)	0000_FFFFh	Never compare.
TIMCFG(n+1)	0030_6100h	Enable on Timer N enable and disable on trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(n+1)	0143_0003h	Configure 16-bit counter and output on Pin 1. Trigger is internal using Shifter 0 flag.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0]. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS[7:0] register instead.

53.4.3 SPI Leader

SPI Leader mode can be supported using two Timers, two Shifters, and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of one clock cycle between the follower select negating and before the next transfer. Writing to the transmit buffer by either core or DMA is used to initiate each transfer.

NOTE

Due to synchronization delays, the setup time for the serial input data is 1.5 FlexIO clock cycles. This means the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 379. SPI leader (CPHA=0) configuration

Register	Value	Comments
SHIFTCFGn	0000_0000h	Start and stop bit disabled.
SHIFTCTLn	0083_0002h	Configure transmit using Timer 0 on negedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(n+1)	0000_0101h	Configure receive using Timer 0 on posedge of clock with input data on Pin 1.
TIMCMPn	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1.

Table continues on the next page...

Table 379. SPI leader (CPHA=0) configuration (continued)

Register	Value	Comments
		Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTLn	01C3_0201h	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(n+1)	0000_FFFFh	Never compare.
TIMCFG(n+1)	0000_1100h	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0003_0383h	Configure 16-bit counter (never compare) using inverted Pin 3 output (as follower select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBIS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS register instead.

Table 380. SPI leader (CPHA=1) configuration

Register	Value	Comments
SHIFTCFGn	0000_0021h	Start bit loads data on first shift.
SHIFTCTLn	0003_0002h	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(n+1)	0080_0101h	Configure receive using Timer 0 on negedge of clock with input data on Pin 1.
TIMCMPn	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1.

Table continues on the next page...

Table 380. SPI leader (CPHA=1) configuration (continued)

Register	Value	Comments
		Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTLn	01C3_0201h	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock. Set TIMDIS=3 to keep follower select asserted for as long as there is data in the transmit buffer.
TIMCMP(n+1)	0000_FFFFh	Never compare.
TIMCFG(n+1)	0000_1100h	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0003_0383h	Configure 16-bit counter (never compare) using inverted Pin 3 output (as follower select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBIS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from SHIFTBUFBIS register instead.

53.4.4 SPI follower

SPI Follower mode can be supported using one Timer, two Shifters, and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The transmit data must be written to the transmit buffer register before the external follower select asserts, otherwise the shifter error flag is set.

NOTE

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles. This means the maximum baud rate is divide by 6 of the FlexIO clock frequency.

Table 381. SPI follower (CPHA=0) configuration

Register	Value	Comments
SHIFTCFGn	0000_0000h	Start and stop bit disabled.
SHIFTCTLn	0083_0002h	Configure transmit using Timer 0 on falling edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(n+1)	0000_0101h	Configure receive using Timer 0 on rising edge of shift clock with input data on Pin 1.
TIMCMPn	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0120_0600h	Configure enable on trigger rising edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTLn	06C0_0203h	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (follower select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to SHIFTBUFBIS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from the SHIFTBUFBIS register instead.

Table 382. SPI follower (CPHA=1) configuration

Register	Value	Comments
SHIFTCFGn	0000_0001h	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTLn	0003_0002h	Configure transmit using Timer 0 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(n+1)	0080_0101h	Configure receive using Timer 0 on falling edge of shift clock with input data on Pin 1.

Table continues on the next page...

Table 382. SPI follower (CPHA=1) configuration (continued)

Register	Value	Comments
TIMCMPn	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0120_6602h	Configure start bit, enable on trigger rising edge, disable on trigger falling edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTLn	06C0_0203h	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (follower select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports MSB first transfer by writing to the SHIFTBUFBIS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUF. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports MSB first transfer by reading from SHIFTBUFBIS register instead.

53.4.5 I2C controller

I2C Controller mode can be supported using two Timers, two Shifters, and two Pins. One timer is used to generate the SCL output and one timer is used to control the shifters. The two shifters are used to transmit and receive for every word when receiving the transmitter must transmit FFh to 3-state the output. FlexIO inserts a stop bit after every word to generate and verify the ACK or NACK. FlexIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (START to Repeated START or STOP), so the compare register must be programmed with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin is equal to output. However, this increases both the clock high and clock low periods by at least 1 FlexIO clock cycle each. The second timer uses the SCL input pin to control the transmit and receive shift registers. This enforces an SDA data hold time by an extra 2 FlexIO clock cycles.

Both the transmit and receive shifters must be serviced for each word in the transfer. The transmit shifter must transmit FFh when receiving and the receive shifter returns the data present on the SDA pin. The transmit shifter loads one additional word on the last falling edge of SCL pin. If generating a STOP condition or a repeated START condition, this word must be 00h or FFh, respectively. During the last word of a controller-receiver transfer, you must set the transmit SSTOP bit to generate a NACK.

The receive shift register asserts an error interrupt if a NACK is detected, but you are responsible for generating the STOP or repeated START condition. If a NACK is detected during master-transmit, the interrupt routine must immediately write 00h (if generating STOP) or FFh (if generating repeated START) to the transmit shifter register. You must wait for the next rising edge on SCL before disabling both timers. The transmit shifter should be disabled after waiting the setup delay for a repeated START or STOP condition.

NOTE

Due to synchronization delays, the data valid time for the transmit output is two FlexIO clock cycles. This means the maximum baud rate is divide by 6 of the FlexIO clock frequency.

To guarantee SDA hold time, the I2C controller data valid is delayed two cycles because the clock output is passed through a synchronizer before clocking the transmit or receive shifter. Because the SCL output is synchronous with FlexIO clock, the synchronization delay is one cycle and then one cycle to generate the output.

Table 383. I2C controller configuration

Register	Value	Comments
SHIFTCFGn	0000_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTLn	0101_0082h	Configure transmit using Timer 1 on rising edge of clock with inverted output enable (open-drain output) on Pin 0.
SHIFTCFG(n+1)	0000_0020h	Start bit disabled and stop bit enabled (logic 0) for ACK or NACK detection.
SHIFTCTL(n+1)	0180_0001h	Configure receive using Timer 1 on falling edge of clock with input data on Pin 0.
TIMCMPn	0000_2501h	Configure 2 word transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of words x 18) + 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0102_2222h	Configure start bit, stop bit, enable on trigger high, disable on compare, reset if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTLn	01C1_0101h	Configure dual 8-bit counter using Pin 1 output enable (SCL open-drain), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0000_000Fh	Configure 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0020_1112h	Enable when Timer 0 is enabled, disable when Timer 0 is disabled. Enable start bit and stop bit at end of each word and decrement on pin input.
TIMCTL(n+1)	01C0_0183h	Configure 16-bit counter using inverted Pin 1 input (SCL).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0]. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0]. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

53.4.6 I2S controller

I2S Controller mode can be supported using two Timers, two Shifters, and four Pins. One timer is used to generate the bit clock and control the shifters and one timer is used to generate the frame sync. FlexIO waits for the first write to the transmit data buffer before enabling bit clock and frame sync generation. Data transfers are supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FlexIO clock frequency. The initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure that the frame sync is generated one clock cycle before the first output data.

NOTE

Due to synchronization delays, the setup time for the receiver input is 1.5 FlexIO clock cycles. This means the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 384. I2S controller configuration

Register	Value	Comments
SHIFTCFGn	0000_0001h	Load transmit data on first shift and stop bit disabled.
SHIFTCTLn	0003_0002h	Configure transmit using Timer 0 on rising edge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(n+1)	0080_0101h	Configure receive using Timer 0 on falling edge of clock with input data on Pin 1.
TIMCMPn	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0000_0202h	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTLn	01C3_0281h	Configure dual 8-bit counter using inverted Pin 2 output (bit clock), with Shifter 0 flag as the inverted trigger. Clear PINPOL to invert the polarity of the output shift clock.
TIMCMP(n+1)	0000_007Fh	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (number of bits x baud rate divider) - 1.
TIMCFG(n+1)	0000_0100h	Enable when Timer 0 is enabled and never disable.
TIMCTL(n+1)	0003_0383h	Configure 16-bit counter using inverted Pin 3 output (as frame sync). Clear

Table continues on the next page...

Table 384. I2S controller configuration (continued)

Register	Value	Comments
		PINPOL to invert the polarity of the output frame sync
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports LSB first transfer by reading from SHIFTBUF register instead.

53.4.7 I2S target

I2S Target mode can be supported using three Timers, two Shifters, and four Pins. For single transmit and single receive, other combinations of transmit and receive are possible.

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag is set.

NOTE

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles. This means the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The output valid time of I2S follower is max 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization, plus 1 cycle to output the data

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until rising edge of bit clock (when the frame sync is normally sampled). Timer 0 detects rising edge of bit clock with Timer 2 output asserted and asserts output for length of frame. Timer 1 detects falling edge of bit clock with Timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 385. I2S follower configuration

Register	Value	Comments
SHIFTCFGn	0000_0000h	Start and stop bit disabled.
SHIFTCTLn	0103_0002h	Configure transmit using Timer 1 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(n+1)	0180_0101h	Configure receive using Timer 1 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0000_007Fh	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] = (number of bits x 4) - 1.

Table continues on the next page...

Table 385. I2S follower configuration (continued)

Register	Value	Comments
TIMCFGn	0020_2500h	Configure enable on pin rising edge (inverted bit clock) with trigger high (Timer 2) and disable on compare. Initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTLn	0B40_0203h	Configure 16-bit counter using Pin 2 input (bit clock), with Timer 2 output as the trigger.
TIMCMP(n+1)	0000_003Fh	Configure 32-bit transfers. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0020_2500h	Configure enable on pin (bit clock) rising edge with trigger (Timer 0) high and disable on compare. Initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTL(n+1)	0340_0283h	Configure 16-bit counter using inverted Pin 2 input (bit clock), with Timer 0 output as the trigger.
TIMCMP(n+2)	0000_0000h	Compare on zero (first edge).
TIMCFG(n+2)	0020_6400h	Configure enable on inverted pin (frame sync) rising edge and disable on trigger falling edge (bit clock). Initial clock state is logic 1 and decrement on inverted pin input (frame sync).
TIMCTL(n+2)	04C0_0383h	Configure 16-bit counter using inverted Pin 3 input (frame sync), with Pin 2 inverted input (bit clock) as the trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Supports LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Supports LSB first transfer by reading from SHIFTBUF register instead.

53.4.8 SENT receiver

SENT receiver can be supported by using one timer and one pin. The timer is configured as Input-Capture mode, and captures the counter value at falling edge of the pin input. After the counter value is captured, the counter is automatically restarted from counter value 0. Therefore, the captured value always indicates the period between previous falling edge and current falling edge.

The CPU interrupt or DMA trigger can be configured at each capture of the counter. The entire SENT frame decoding with the latest tick width adjustment is performed by CPU software.

Table 386. SENT receiver configuration

Register	Value	Comments
TIMCMPn	-	Stores counter value at the falling edge of the pin.
TIMCFGn	0x0000_6000	Timer is always enabled. Timer is disabled on trigger falling edge, decrement counter on FlexIO clock.
TIMCTLn	0xnn40_0007	Single 16-bit input capture mode. Timer pin input and output are selected by PINSEL. Timer pin output disabled, internal trigger selected, select pin nn as trigger.

53.4.9 Camera interface

Camera interface can be supported using one Timer, one or more Shifters, and multiple pins. Multiple transfers can be supported using DMA controller.

The example below describes the FlexIO configuration for interfacing to an 8-bit CMOS sensor with PCLK, VSYNC, HREF, and D[7:0] outputs. The example uses a 128-bit buffer to capture 16-pixels of image data before interrupt or DMA transfer. You can use a bigger or smaller buffer depending on system DMA performance and FlexIO resource usage by other applications.

NOTE

You can use additional timers to track the number of pixels per row and number of rows per frame, or HREF or VSYNC can be assigned as GPIO interrupts for software tracking.

Table 387. Camera interface configuration for 8-bit CMOS sensor

Register	Value	Comments
SHIFTCFGn...n+2[1]	0007_0100h	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFGn+3	0007_0000h	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTLn...n+3	0080_0001h	Configure receive using Timer 0 on negedge of clock.
TIMCMPn	0000_001Fh	Configure 16-pixel (8 bits/pixel x 16 pixels = 128-bits) transfer. Set TIMCMP[15:0] = (number of pixels x 2) - 1.
TIMCFGn	0120_6600h	Configure enable on trigger (HREF) rising edge and disable on trigger falling edge. Initial Shift clock state is logic 0 and decrement on PCLK input.
TIMCTLn	12C0_0803h	Configure 16-bit counter using FXIO_D[8] input (PCLK), with

Table continues on the next page...

Table 387. Camera interface configuration for 8-bit CMOS sensor (continued)

Register	Value	Comments
		FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUFn...n+3	Data to receive	Received data can be read from SHIFTBUFn...n+3. Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

1. n=0 or 4

53.4.10 Motorola 68K and Intel 8080 bus interface

The Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by Smart/Asynchronous LCD controllers. With GPIO, FlexIO can drive these interfaces using one Timer and one Shifter. Additional Shifters can be used to support large transfers via the DMA controller.

The table below provides an example of how to drive a 16-bit 68K or 8080 bus. For an 8080 bus, two GPIOs are used to drive the nCS and RS pins. For a 68K bus, an additional GPIO is required to drive the RDWR pin.

Table 388. Motorola 68K and Intel 8080 write configuration

Register	Value	Comments
SHIFTCFG0...7	000F_0100h	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0003_0002h	Configure transmit using Timer 0 on posedge of clock with data output to FXIO_D[15:0].
SHIFTCTL1...7	0000_0002h	Configure transmit using Timer 0 on posedge of clock.
TIMCMP0	0000_0101h (1-beat) 0000_1F01h (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0000_2200h	Configure enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	01C3_1001h (Motorola 68K, 1-beat) 1DC3_1001h (Motorola 68K, 16-beats) 01C3_1081h (Intel 8080, 1-beat) 1DC3_1081h (Intel 8080, 16-beats)	Configure dual 8-bit counter using FXIO_D[16] output (EN pin for 68K, nWR pin for 8080), with Shifter 0 (1-beat) or Shifter 7 (16-beats) flag as the inverted trigger.
SHIFTBUF0...7	Data to transmit	Transmit data can be written to SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats) to initiate a transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table 389. Motorola 68K 8080 Read Configuration

Register	Value	Comments
SHIFTCFG0...6	000F_0100h	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCFG7	000F_0000h	Configure 16-bit parallel shift in from pin.
SHIFTCTL0...7	0080_0001h	Configure receive using Timer 0 on negedge of clock with data input from FXIO_D[15:0].
TIMCMP0	0000_0101h (1-beat) 0000_1F01h (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0000_2220h	Configure stop_bit. Enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	1DC3_1001h (Motorola 68K, 1 beat) 01C3_1001h (Motorola 68K, 16 beats) 1DC3_1181h (Intel 8080, 1 beat) 01C3_1181h (Intel 8080, 16 beats)	Configure dual 8-bit counter using either FXIO_D[16] output (EN pin for 68K) or FXIO_D[17] output (nRD pin for 8080), with Shifter 7 flag (1-beat) or Shifter 0 flag (16-beats) as the inverted trigger.
SHIFTBUF0...7	Data received	Received data can be read from SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats). Use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

In general, any operation to a 68K or 8080 bus follower begins with a register write cycle followed by one or more data read or write cycles. To accomplish this, the following program flow must be used:

1. Configure FlexIO with 1-beat write configuration
2. Configure GPIO to assert nCS, RS pins (and deassert RDWR pin for 68K)
3. Write register index data to SHIFTBUF0[15:0]
4. Configure GPIO to deassert RS pin (and assert RDWR pin for 68K data read)
5. Configure FlexIO with desired read or write configuration (1 or 16-beats)
6. Use the Shifter Status Flag to trigger interrupt or DMA driven data transfers to and from SHIFTBUF registers
7. Configure GPIO to deassert nCS pin

53.4.11 Low-power state machine

The table below shows an example of a hypothetical state machine in order to illustrate the flexibility allowed in Shifter State mode.

In this example, FlexIO waits for the FXIO_D[2] pin to assert and then drives a complementary square wave output at a frequency of FLEXIO_CLK/131072 on the FXIO_D[1:0] pins while the comparator output is asserted. This assumes that the comparator isthat connected to external trigger 15. See the chip-specific FlexIO information for actual FlexIO trigger mappings. Throughout this operation, the CPU can be kept in a STOP or VLPS mode by clearing the CTRL[DOZEN] field and ensuring the FLEXIO_CLK is enabled. The state diagram below shows the states and transitions implemented by this example.

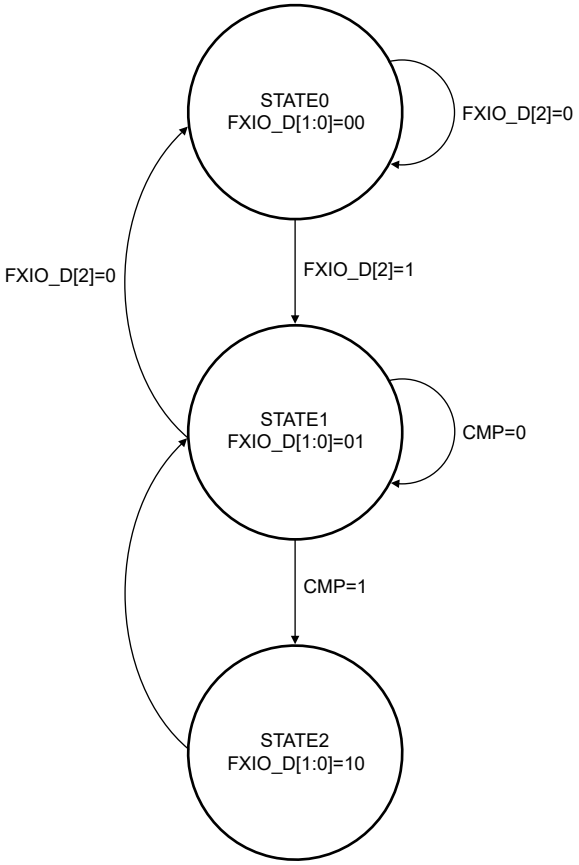


Figure 206. State diagram

Table 390. State machine configuration

Register	Value	Comments
SHIFTCFG0...2	0000_0003h	Enable FXIO_D[1:0] as outputs.
SHIFTCTL0	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF0	0020_8208h	State0: Drive FXIO_D[1:0]=00, transition to State0 if FXIO_D[2]=0, State1 if FXIO_D[2]=1.
SHIFTCTL1	0000_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output high to trigger next state.
SHIFTBUF1	0140_8408h	State1: Drive FXIO_D[1:0]=01, transition to State0 if FXIO_D[2]=0, State1 if CMP=0, State2 if CMP=1 (FXIO_D[3]=1)

Table continues on the next page...

Table 390. State machine configuration (continued)

Register	Value	Comments
SHIFTCTL2	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF2	0224_9249h	State2: Drive FXIO_D[1:0]=10, transition to State1 when Timer0 output low
TIMCMP0	0000_FFFFh	Configure baud rate of divide by 131072 of the FlexIO clock. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0000_0000h	Configure timer always enabled.
TIMCTL0	0000_0003h	Configure single 16-bit counter.
TIMCFG1	0010_7600h	Configure timer enabled on trigger rising edge, disabled on trigger falling edge, decrement on trigger edge.
TIMCTL1	0F03_0303h	Configure timer output enabled on FXIO_D[3], external trigger 15 (comparator output) selected.

53.4.12 Keypad interface

The keypad interface can support a 3 × 4 keypad matrix using three timers and three shifters, although a larger matrix can be supported using additional shifters. The configuration is designed for four columns configured as active low open-drain pins and three rows configured as input pins with pull-up resistors enabled.

One shifter is configured in logic mode to assert its output when any of the row inputs are low, indicating a key is pressed. Use a timer to filter the shifter output to ensure that the key is pressed for a minimum amount of time before performing the column scan.

A different shifter is configured for parallel transmit. Use this shifter to scan each column when a keypress is detected. When not scanning, the shifter output is configured to assert all active low open-drain column outputs to detect any keypress. Use a dedicated timer to control the transmit shifter.

The last shifter is configured for parallel receive. Use this shifter to capture the result of the column scanning for software to decode which key (or keys) was pressed. This configuration captures the state of both row and column pins for each scan, although the row state can also be deduced by the shift order. Use a dedicated timer to control the receive shifter, which shifts at half the frequency of the transmit shifter.

When the result of the key scan is available in the receive shifter register, FlexIO continues to monitor the row inputs and can trigger multiple scans from the one keypress. To support debouncing, you can decide how many consecutive scans should be considered a single keypress.

Because the pins used in logic mode are fixed per shifter and the shifters that support parallel shifts are limited, this configuration is restricted in what pins and shifters it can use. Increasing the matrix beyond four-row inputs requires multiple shifters in logic mode. Increasing beyond four-column outputs requires concatenating transmit shifters to create a larger shift register.

Table 391. Keypad interface configuration

Register	Value	Comments
SHIFTCFG0	0003_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1), configured for 4-bit shift.
SHIFTCTL0	0101_0402h	Configure transmit using Timer 1 on rising edge of clock generating open-drain output on Pins [7:4] (column outputs).
SHIFTBUF0	0804_0201h	Static data containing the column scan pattern, each column is scanned one-hot with dead time in between.
SHIFTCFG1	0000_0020h	In logic mode, mask input 3.
SHIFTCTL1	0000_0007h	Configure logic mode.
SHIFTBUF1	07FF_07FFh	Static data configuring logic mode LUT. Output asserts if pins [3:1] are logic 0.
SHIFTCFG3	0007_0000h	Configured for 8-bit shift.
SHIFTCTL3	0280_0001h	Configure receive using Timer 2 on falling edge of clock with input data on Pins [7:0] (both rows and columns, pin 0 is don't care).
TIMCMP0	0000_00ffh	Configures pre-scanning glitch filter to 256 FlexIO clock cycles. For different filter cycles, set TIMCMP[15:0] = (filter cycles) - 1.
TIMCFG0	0103_6600h	Configure enable on trigger rising edge and disable on trigger falling edge. Initial clock state is logic 0 and is not affected by reset.
TIMCTL0	0540_0003h	Configure 16-bit counter using Shifter 1 flag (logic state) as trigger.
TIMCMP1	0000_0F3Fh	Configure 8 shifts (twice for each column) at column scan rate of divide by 128. For different scan frequency, set TIMCMP[7:0] = (scan divider / 2) - 1.
TIMCFG1	0020_2622h	Enable on trigger rising edge, disable on compare. Start and stop bits are enabled.
TIMCTL1	0340_0001h	Configure dual 8-bit counter using Timer 0 output as the trigger.
TIMCMP2	0000_0801h	Configure 4 shifts at half the frequency of Timer 1 trigger.
TIMCFG2	0110_2100h	Enable on Timer 1 enable, disable on compare, decrement on trigger input

Table continues on the next page...

Table 391. Keypad interface configuration (continued)

Register	Value	Comments
		with output initially negated, and not affected by reset
TIMCTL2	0740_0001h	Configure dual 8-bit counter using Timer 1 output as the trigger.
SHIFTBUF3	Keypad Scan Result	Keypad scan result can be read from SHIFTBUF3. Each byte is the result of one scan with both row and column pin state from the scan (pin 0 is not used). Use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

53.5 FlexIO Signal Descriptions

Table 392.

Signal	Description	I/O
FXIO_Dn (n=0...31)	Bidirectional FlexIO Shifter and Timer pin inputs/outputs	I/O

53.6 Memory Map and Registers

53.6.1 FLEXIO register descriptions

NOTE

An invalid register access results in a bus error. Invalid register accesses include reading a write-only register, writing a read-only register, or accessing an invalid address.

53.6.1.1 FLEXIO memory map

FLEXIO0 base address: 4003_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0201_0003h
4h	Parameter Register (PARAM)	32	R	0420_0808h
8h	FlexIO Control Register (CTRL)	32	RW	0000_0000h
Ch	Pin State Register (PIN)	32	R	0000_0000h
10h	Shifter Status Register (SHIFTSTAT)	32	RW	0000_0000h
14h	Shifter Error Register (SHIFTERR)	32	RW	0000_0000h
18h	Timer Status Register (TIMSTAT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable Register (TIMIEN)	32	RW	0000_0000h
30h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	0000_0000h
38h	Timer Status DMA Enable (TIMERSDEN)	32	RW	0000_0000h
40h	Shifter State Register (SHIFTSTATE)	32	RW	0000_0000h
48h	Trigger Status Register (TRGSTAT)	32	RW	0000_0000h
4Ch	External Trigger Interrupt Enable Register (TRIGIEN)	32	RW	0000_0000h
50h	Pin Status Register (PINSTAT)	32	RW	0000_0000h
54h	Pin Interrupt Enable Register (PINIEN)	32	RW	0000_0000h
58h	Pin Rising Edge Enable Register (PINREN)	32	RW	0000_0000h
5Ch	Pin Falling Edge Enable Register (PINFEN)	32	RW	0000_0000h
60h	Pin Output Data Register (PINOUTD)	32	RW	0000_0000h
64h	Pin Output Enable Register (PINOUTE)	32	RW	0000_0000h
68h	Pin Output Disable Register (PINOUTDIS)	32	W	0000_0000h
6Ch	Pin Output Clear Register (PINOUTCLR)	32	W	0000_0000h
70h	Pin Output Set Register (PINOUTSET)	32	W	0000_0000h
74h	Pin Output Toggle Register (PINOUTTOG)	32	W	0000_0000h
80h - 9Ch	Shifter Control N Register (SHIFTCTL0 - SHIFTCTL7)	32	RW	0000_0000h
100h - 11Ch	Shifter Configuration N Register (SHIFTCFG0 - SHIFTCFG7)	32	RW	0000_0000h
200h - 21Ch	Shifter Buffer N Register (SHIFTBUF0 - SHIFTBUF7)	32	RW	0000_0000h
280h - 29Ch	Shifter Buffer N Bit Swapped Register (SHIFTBUFBIS0 - SHIFTBUFBIS7)	32	RW	0000_0000h
300h - 31Ch	Shifter Buffer N Byte Swapped Register (SHIFTBUFBYS0 - SHIFTBUFBYS7)	32	RW	0000_0000h
380h - 39Ch	Shifter Buffer N Bit Byte Swapped Register (SHIFTBUFBBS0 - SHIFTBUFBBS7)	32	RW	0000_0000h
400h - 41Ch	Timer Control N Register (TIMCTL0 - TIMCTL7)	32	RW	0000_0000h
480h - 49Ch	Timer Configuration N Register (TIMCFG0 - TIMCFG7)	32	RW	0000_0000h
500h - 51Ch	Timer Compare N Register (TIMCMP0 - TIMCMP7)	32	RW	0000_0000h
680h - 69Ch	Shifter Buffer N Nibble Byte Swapped Register (SHIFTBUFNBS0 - SHIFTBUFNBS7)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
700h - 71Ch	Shifter Buffer N Half Word Swapped Register (SHIFTBUFHWS0 - SHIFTBUFHWS7)	32	RW	0000_0000h
780h - 79Ch	Shifter Buffer N Nibble Swapped Register (SHIFTBUFNIS0 - SHIFTBUFNIS7)	32	RW	0000_0000h
800h - 81Ch	Shifter Buffer N Odd Even Swapped Register (SHIFTBUFOES0 - SHIFTBUFOES7)	32	RW	0000_0000h
880h - 89Ch	Shifter Buffer N Even Odd Swapped Register (SHIFTBUFEOS0 - SHIFTBUFEOS7)	32	RW	0000_0000h
900h - 91Ch	Shifter Buffer N Halfword Byte Swapped Register (SHIFTBUFHBS0 - SHIFTBUFHBS7)	32	RW	0000_0000h

53.6.1.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

Contains the version of the FlexIO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-24	Major Version Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
MAJOR	This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read-only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number This read-only field returns the feature set number. 0000_0000_0000_0000b - Standard features implemented. 0000_0000_0000_0001b - Supports state, logic, and parallel modes. 0000_0000_0000_0010b - Supports pin control registers. 0000_0000_0000_0011b - Supports state, logic, and parallel modes; plus pin control registers.

53.6.1.3 Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains the number of shifters, timers, pins, and triggers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRIGGER								PIN							
W																
Reset	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER								SHIFTER							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-24	Trigger Number

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIGGER	Number of external triggers implemented.
23-16 PIN	Pin Number Number of Pins implemented.
15-8 TIMER	Timer Number Number of Timers implemented.
7-0 SHIFTER	Shifter Number Number of Shifters implemented.

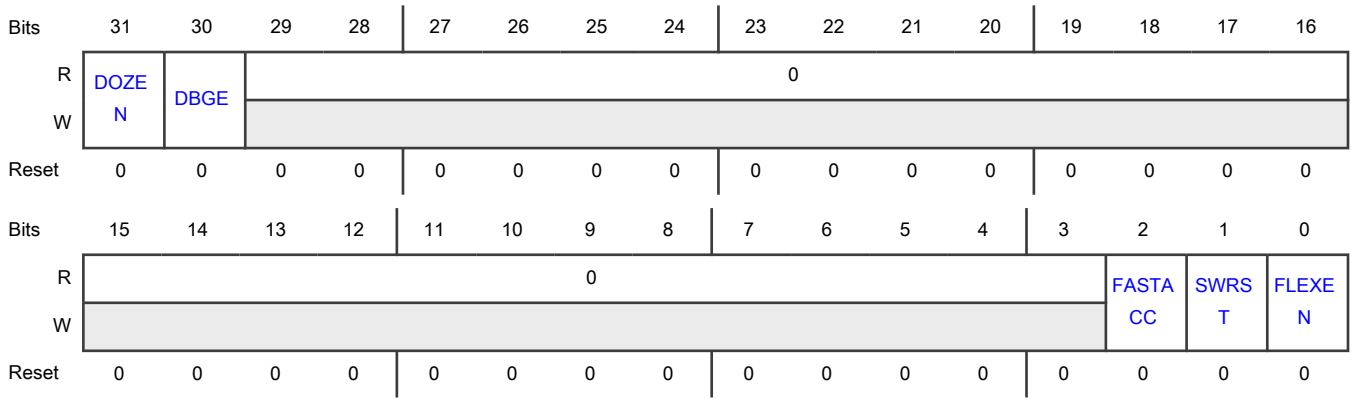
53.6.1.4 FlexIO Control Register (CTRL)

Offset

Register	Offset
CTRL	8h

Function
Control register.

Diagram



Fields

Field	Function
31 DOZEN	Doze Enable Disables FlexIO operation in Doze modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FlexIO enabled in Doze modes. 1b - FlexIO disabled in Doze modes.
30 DBGE	Debug Enable Enables FlexIO operation in Debug mode. 0b - FlexIO is disabled in debug modes. 1b - FlexIO is enabled in debug modes
29-3 —	Reserved
2 FASTACC	Fast Access Enables fast register accesses to FlexIO registers, but requires the FlexIO functional clock to be at least equal to the frequency of the bus clock. 0b - Configures for normal register accesses to FlexIO 1b - Configures for fast register accesses to FlexIO
1 SWRST	Software Reset The FlexIO Control Register is not affected by the software reset, all other logic in the FlexIO is affected by the software reset and all other register accesses are ignored until this bit is cleared. This register bit remains set until cleared by software, and the reset has cleared in the FlexIO clock domain. 0b - Software reset is disabled 1b - Software reset is enabled, all FlexIO registers except the Control Register are reset.
0 FLEXEN	FlexIO Enable Determines if the FlexIO is enabled or disabled. 0b - FlexIO module is disabled. 1b - FlexIO module is enabled.

53.6.1.5 Pin State Register (PIN)

Offset

Register	Offset
PIN	Ch

Function
Reports status of the Pin Data Input.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PDI	Pin Data Input Returns the input data on each of the FlexIO pins.

53.6.1.6 Shifter Status Register (SHIFTSTAT)

Offset

Register	Offset
SHIFTSTAT	10h

Function

Shifter status flags.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 SSF	Shifter Status Flag The shifter status flag is updated when one of the following events occurs: For SMOD=Receive, the status flag is set when SHIFTBUF has been loaded with data from Shifter (SHIFTBUF is full), and the status flag is cleared when SHIFTBUF register is read. For SMOD=Transmit, the status flag is set when SHIFTBUF data has been transferred to the Shifter (SHIFTBUF is empty) or when initially configured for SMOD=Transmit, and the status flag is cleared when the SHIFTBUF register is written. For SMOD=Match Store, the status flag is set when a match has occurred between SHIFTBUF and Shifter, and the status flag is cleared when the SHIFTBUF register is read. For SMOD=Match Continuous, returns the current match result between the SHIFTBUF and Shifter. The status flag cannot be cleared by reading SHIFTBUF. For SMOD=State, the status flag for a shifter sets when it is selected by the current state pointer. For SMOD=Logic, returns the current value of the programmable logic block output. The status flag can also be cleared by writing a logic one to the flag for all modes except Match Continuous/ State /Logic. 0b - Status flag is clear. 1b - Status flag is set.

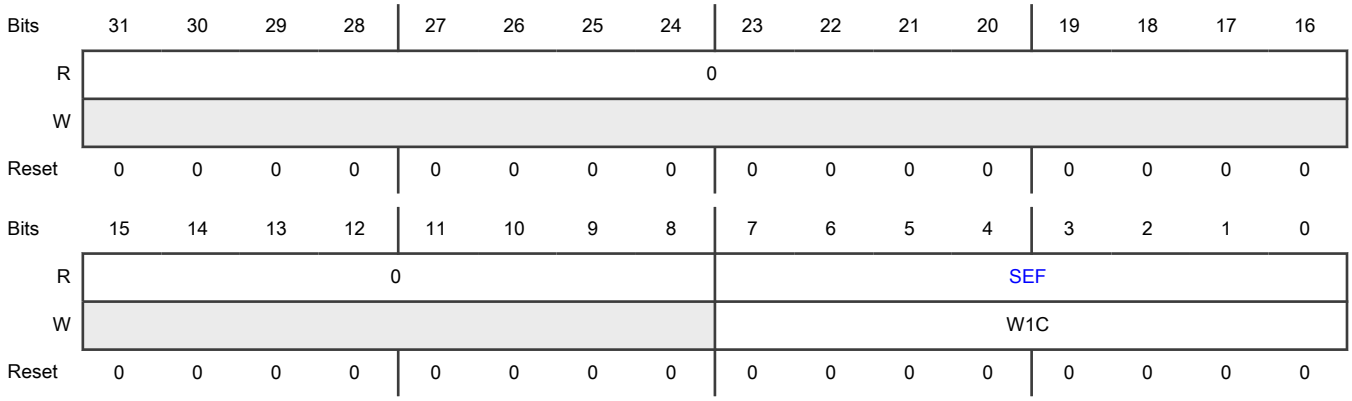
53.6.1.7 Shifter Error Register (SHIFTERR)

Offset

Register	Offset
SHIFTERR	14h

Function
This register reports shifter error flags. Write one to clear.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEF	Shifter Error Flags The shifter error flag is set when one of the following events occurs: For SMOD=Receive, indicates Shifter was ready to store new data into SHIFTBUF before the previous data was read from SHIFTBUF (SHIFTBUF Overrun), or indicates that the received start or stop bit does not match the expected value. For SMOD=Transmit, indicates Shifter was ready to load new data from SHIFTBUF before new data had been written into SHIFTBUF (SHIFTBUF Underrun). For SMOD=Match Store, indicates a match event occurred before the previous match data was read from SHIFTBUF (SHIFTBUF Overrun). For SMOD=Match Continuous, the error flag is set when a match has occurred between SHIFTBUF and Shifter. For SMOD=Logic, the error flag is set when the output of the programmable logic block has asserted. Can be cleared by writing logic one to the flag. For SMOD=Match Continuous, can also be cleared when the SHIFTBUF register is read. 0b - Shifter Error Flag is clear. 1b - Shifter Error Flag is set.

53.6.1.8 Timer Status Register (TIMSTAT)

Offset

Register	Offset
TIMSTAT	18h

Function

This register reports timer status flags. Write one to clear.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 TSF	Timer Status Flags The timer status flag sets depending on the timer mode, and can be cleared by writing logic one to the flag. In 8-bit baud counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit high PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit counter mode, the timer status flag is set when the 16-bit counter equals zero and decrements. In 16-bit counter disable mode, the timer status flag is set when a timer disable event is detected. In 8-bit word counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit low PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit input capture mode, the timer status flag is set when a timer disable event is detected and the timer status flag is clear. In this mode, the timer compare register should only be read when the timer status flag is set. 0b - Timer Status Flag is clear. 1b - Timer Status Flag is set.

53.6.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

Offset

Register	Offset
SHIFTSIEN	20h

Function
Enables for Shifter Status Interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSIE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when corresponding SSF is set. 0b - Shifter Status Flag interrupt disabled. 1b - Shifter Status Flag interrupt enabled.

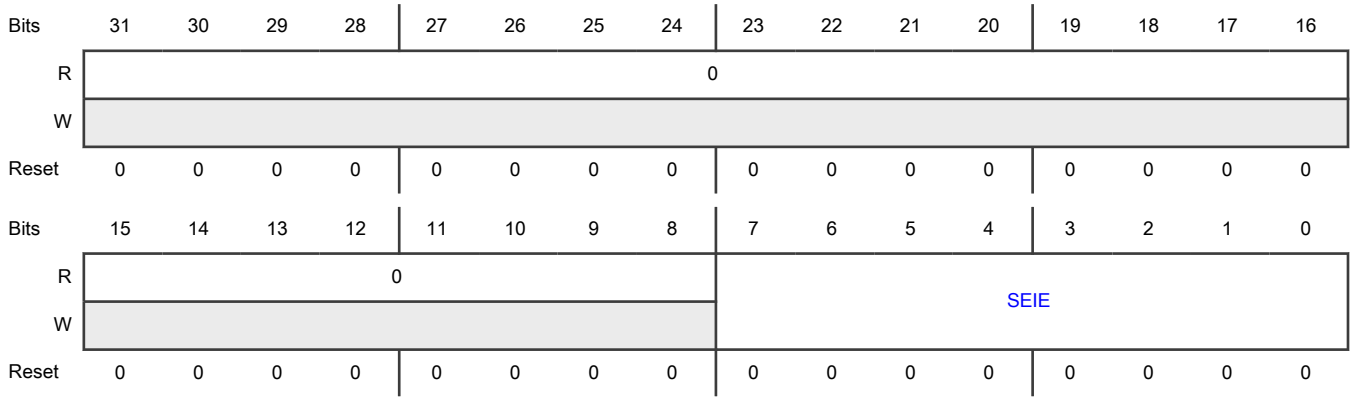
53.6.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

Offset

Register	Offset
SHIFTEIEN	24h

Function
Enables for Shifter Error Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when corresponding SEF is set. 0b - Shifter Error Flag interrupt disabled. 1b - Shifter Error Flag interrupt enabled.

53.6.1.11 Timer Interrupt Enable Register (TIMIEN)

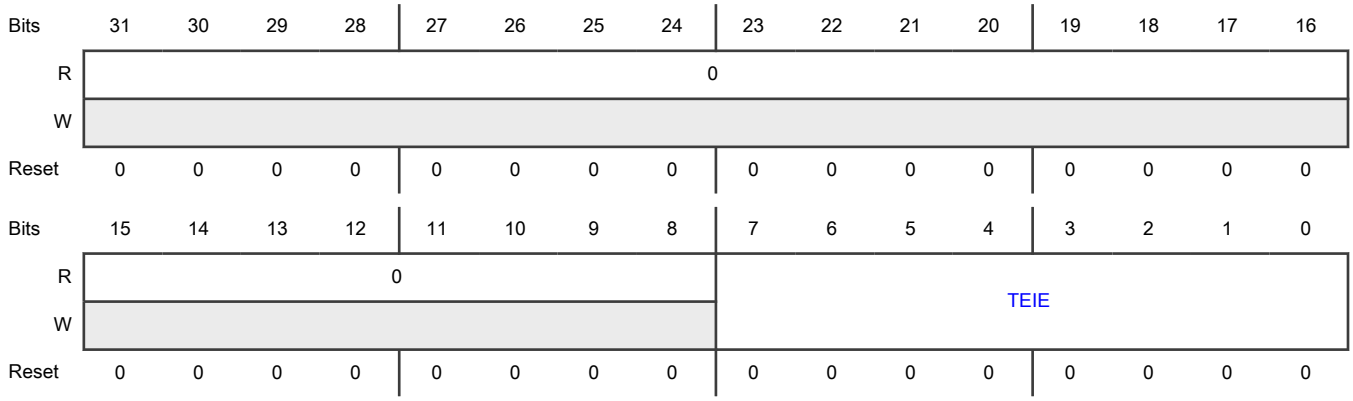
Offset

Register	Offset
TIMIEN	28h

Function

Enables for Timer Status Interrupts.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when corresponding TSF is set. 0b - Timer Status Flag interrupt is disabled. 1b - Timer Status Flag interrupt is enabled.

53.6.1.12 Shifter Status DMA Enable (SHIFTSDEN)

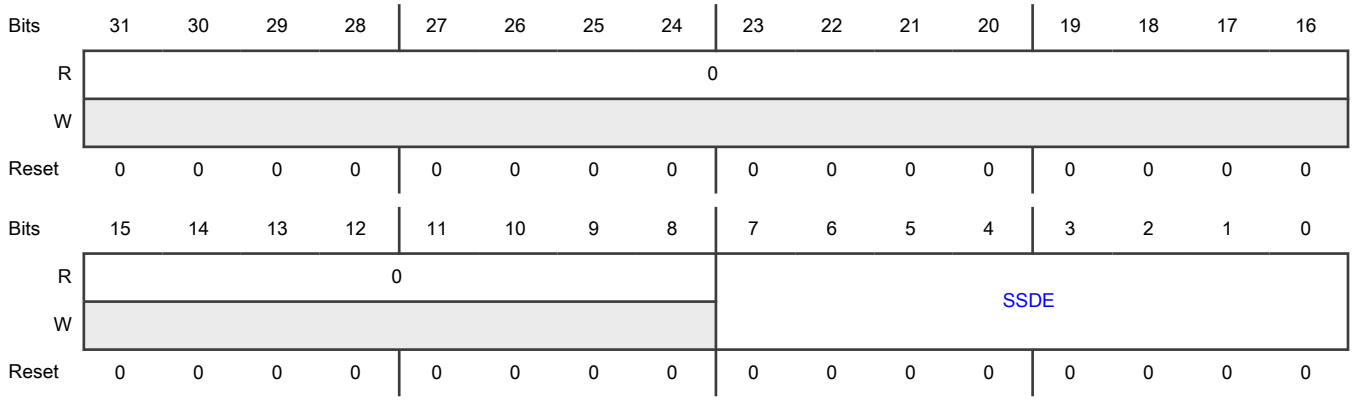
Offset

Register	Offset
SHIFTSDEN	30h

Function

Enables for Shifter DMA requests.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when corresponding SSF is set. 0b - Shifter Status Flag DMA request is disabled. 1b - Shifter Status Flag DMA request is enabled.

53.6.1.13 Timer Status DMA Enable (TIMERSDEN)

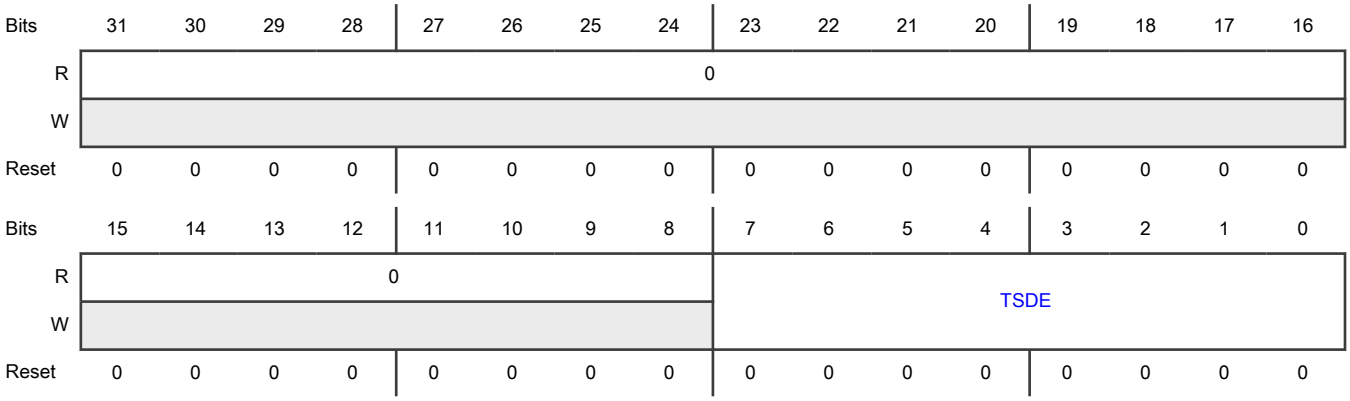
Offset

Register	Offset
TIMERSDEN	38h

Function

Enables for DMA requests when Timer Status Flag is set.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 TSDE	Timer Status DMA Enable Enables DMA request generation when corresponding TSF is set. When the Timer Status DMA request is enabled, reading or writing a Timer Compare Register clears the corresponding Timer Status Register. The DMA must therefore read or write the Timer Compare Register as part of the DMA transfer, otherwise the DMA request remains asserted. 0b - Timer Status Flag DMA request is disabled. 1b - Timer Status Flag DMA request is enabled.

53.6.1.14 Shifter State Register (SHIFTSTATE)

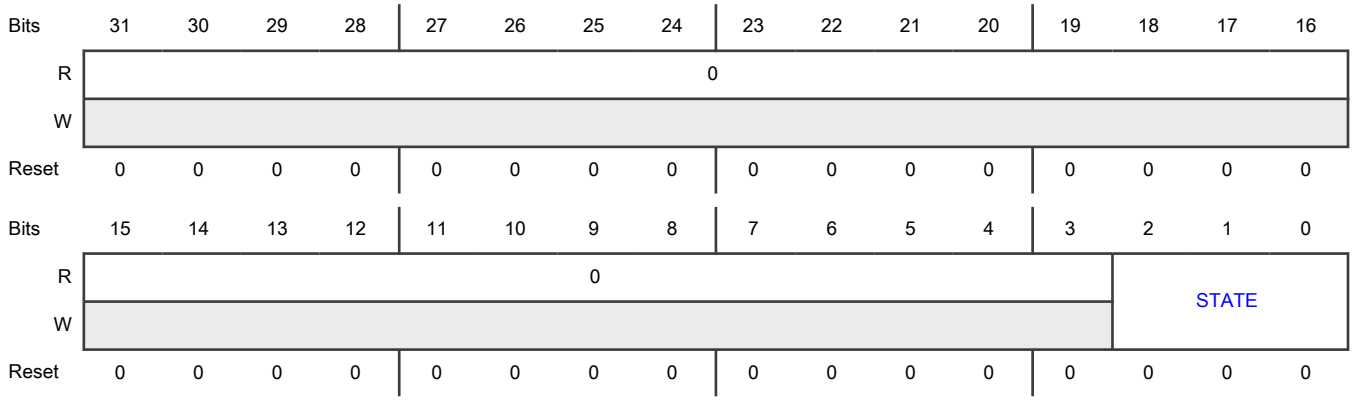
Offset

Register	Offset
SHIFTSTATE	40h

Function

Contains the pointer to keep track of the current shifter.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 STATE	Current State Pointer The current state field maintains a pointer to keep track of the current Shifter (configured for State mode) enabled to drive outputs and compute the next state. Reading this register when the state pointer is updating can result in the incorrect state returned. This field is writable and the value written overrides the current state.

53.6.1.15 Trigger Status Register (TRGSTAT)

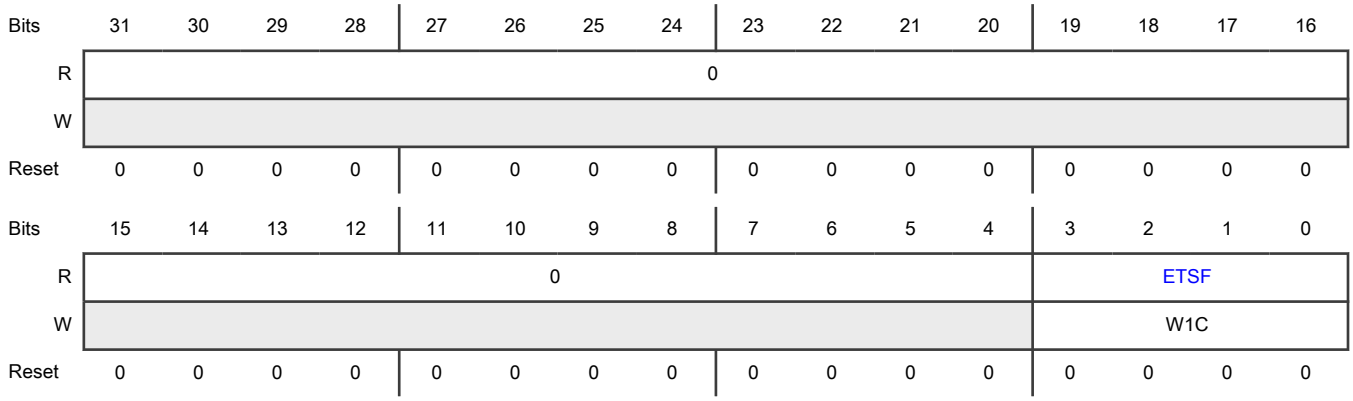
Offset

Register	Offset
TRGSTAT	48h

Function

External Trigger Status Flags. Write one to clear.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 ETSF	External Trigger Status Flags The external trigger status flag is set when a rising edge is detected on the corresponding external trigger input. 0b - External Trigger Status Flag is clear 1b - External Trigger Status Flag is set

53.6.1.16 External Trigger Interrupt Enable Register (TRIGIEN)

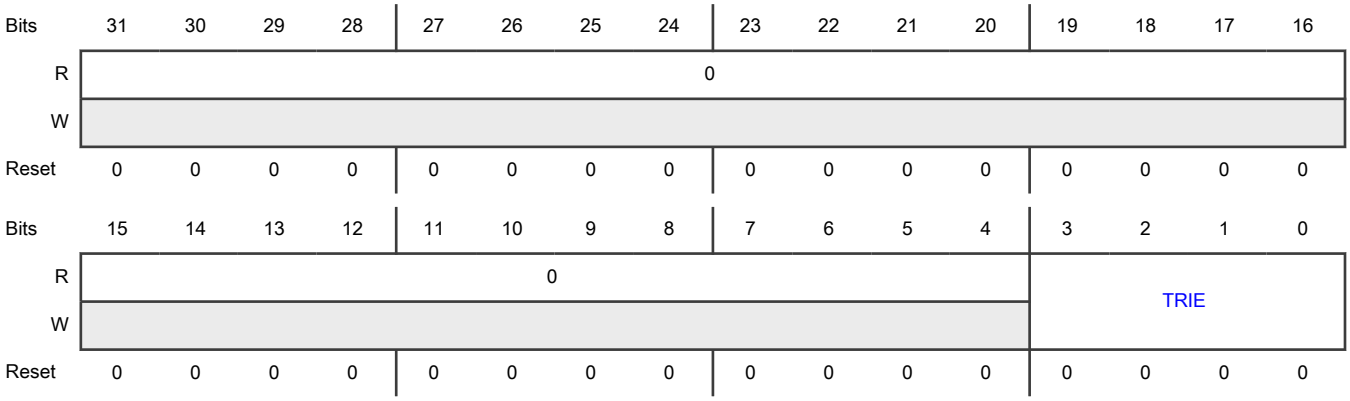
Offset

Register	Offset
TRIGIEN	4Ch

Function

Enables for external trigger interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TRIE	External Trigger Interrupt Enable Enables interrupt generation when corresponding ETSF is set. 0b - External Trigger Status Flag interrupt is disabled. 1b - External Trigger Status Flag interrupt is enabled.

53.6.1.17 Pin Status Register (PINSTAT)

Offset

Register	Offset
PINSTAT	50h

Function

Pin Status Flags. Write one to clear.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PSF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSF	Pin Status Flags The pin status flag is set when a rising edge (if configured) or falling edge (if configured) is detected on the corresponding pin, as configured by pin. 0b - Pin Status Flag is clear 1b - Pin Status Flag is set

53.6.1.18 Pin Interrupt Enable Register (PINIEN)

Offset

Register	Offset
PINIEN	54h

Function

Enables for pin status interrupts.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PSIE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PSIE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PSIE	Pin Status Interrupt Enable Enables interrupt generation when corresponding PSF is set. 0b - Pin Status Flag interrupt is disabled. 1b - Pin Status Flag interrupt is enabled.

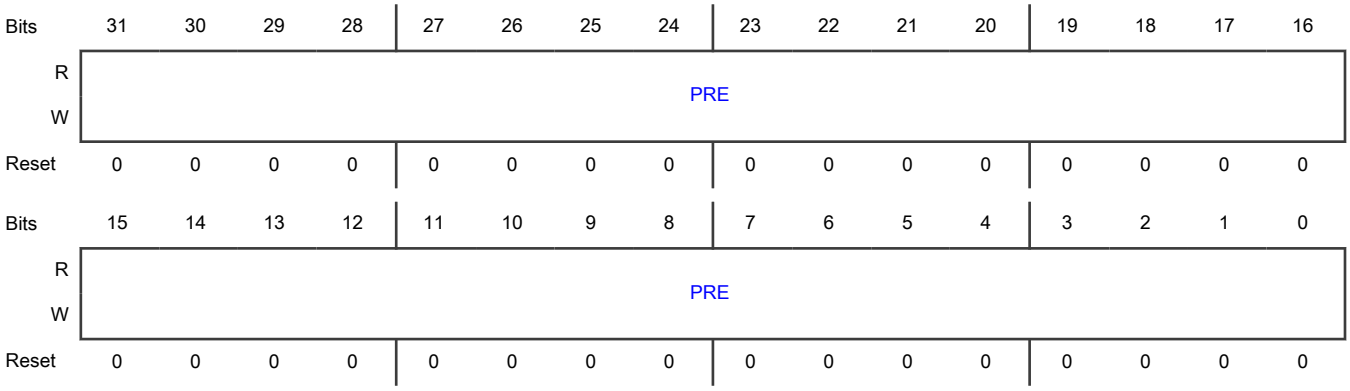
53.6.1.19 Pin Rising Edge Enable Register (PINREN)

Offset

Register	Offset
PINREN	58h

Function
Enables for pin status flag on rising edge.

Diagram



Fields

Field	Function
31-0 PRE	Pin Rising Edge When set, the pin status flag is set whenever a rising edge is detected on the pin. 0b - Pin Status Flag does not set when a pin rising edge is detected. 1b - Pin Status Flag does set when a pin rising edge is detected.

53.6.1.20 Pin Falling Edge Enable Register (PINFEN)

Offset

Register	Offset
PINFEN	5Ch

Function
Enables for pin status flag on falling edge.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PFE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PFE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PFE	Pin Falling Edge When set, the pin status flag is set whenever a falling edge is detected on the pin. 0b - Pin Status Flag does not set when a pin falling edge is detected. 1b - Pin Status Flag does set when a pin falling edge is detected.

53.6.1.21 Pin Output Data Register (PINOUTD)

Offset

Register	Offset
PINOUTD	60h

Function
Data Out when direct pin output in enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OUTD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OUTD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTD	Output Data Configures the value driven on the corresponding pin when direct pin output is enabled. 0b - Logic zero is driven on the pin when direct pin output is enabled. 1b - Logic one is driven on the pin when direct pin output is enabled.

53.6.1.22 Pin Output Enable Register (PINOUTE)

Offset

Register	Offset
PINOUTE	64h

Function
Enables for pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OUTE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OUTE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTE	Output Enable Enables direct output on the corresponding pin. 0b - Pin is controlled by timer/shifter configuration. 1b - Pin is an output and driven with value of PINOUTD register.

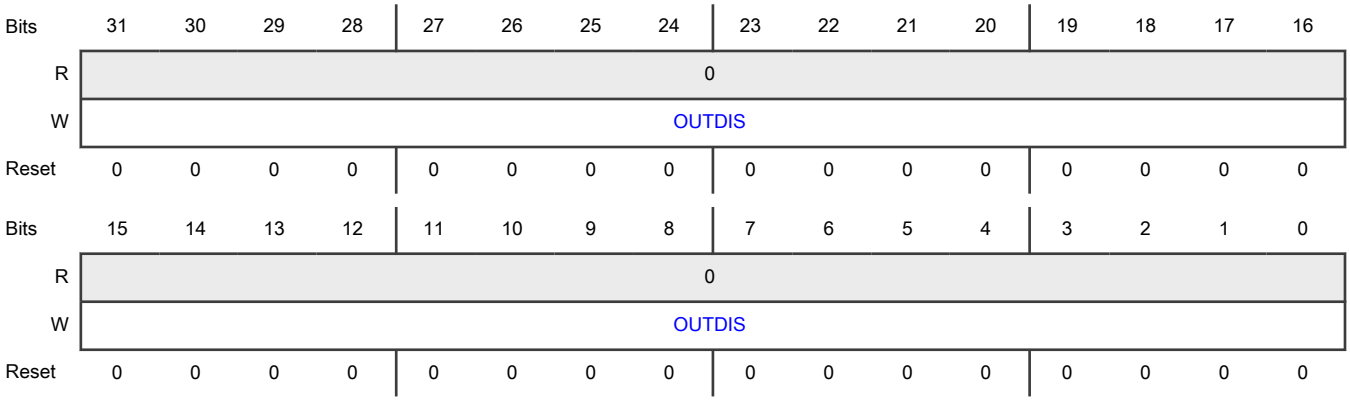
53.6.1.23 Pin Output Disable Register (PINOUTDIS)

Offset

Register	Offset
PINOUTDIS	68h

Function
Disable for pin output.

Diagram



Fields

Field	Function
31-0 OUTDIS	Output Disable Configures corresponding pins to disable direct output. 0b - No effect. 1b - Corresponding PINOUTD and PINOUTE bits are cleared.

53.6.1.24 Pin Output Clear Register (PINOUTCLR)

Offset

Register	Offset
PINOUTCLR	6Ch

Function
Clears pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTCLR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTCLR	Output Clear Configures corresponding pins to output zero. 0b - No effect. 1b - Corresponding PINOUTD is cleared and PINOUTE is set.

53.6.1.25 Pin Output Set Register (PINOUTSET)

Offset

Register	Offset
PINOUTSET	70h

Function
sets pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTSET															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTSET	Output Set Configures corresponding pins to output logic one. 0b - No effect. 1b - Corresponding PINOUTD is set and PINOUTE is set.

53.6.1.26 Pin Output Toggle Register (PINOUTTOG)

Offset

Register	Offset
PINOUTTOG	74h

Function
Toggles pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	OUTTOG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	OUTTOG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 OUTTOG	Output Toggle Configures corresponding pins to toggle. 0b - No effect. 1b - Corresponding PINOUTD is inverted and PINOUTE is set.

53.6.1.27 Shifter Control N Register (SHIFTCTL0 - SHIFTCTL7)

Offset

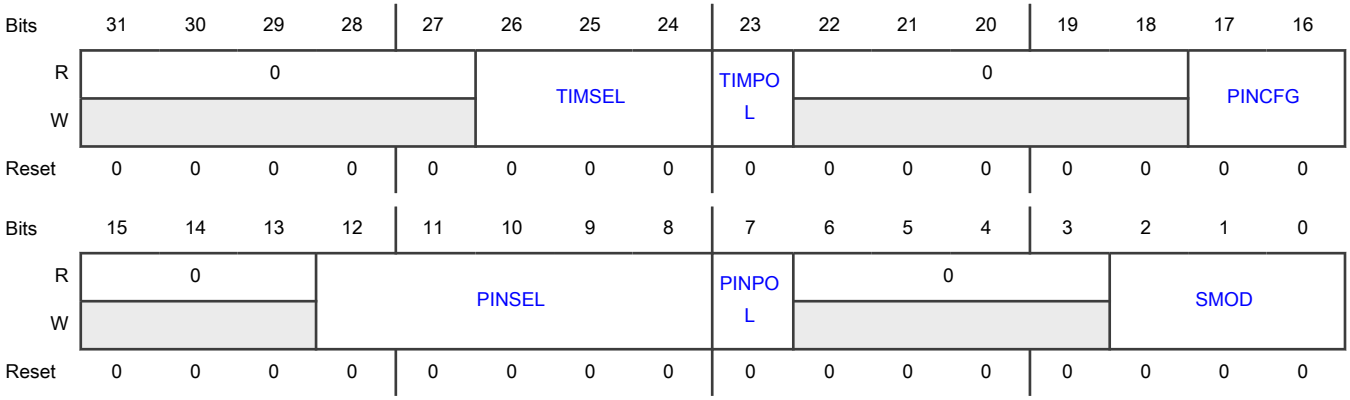
For a = 0 to 7:

Register	Offset
SHIFTCTLa	80h + (a × 4h)

Function

Control for Shift Register N

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 TIMSEL	Timer Select Selects which Timer is used for controlling the logic/shift register and generating the Shift clock. TIMSEL=i selects TIMERi.
23	Timer Polarity

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIMPOL	Determines the shift occurs on the positive edge or negative edge of the shift clock. 0b - Shift on posedge of Shift clock 1b - Shift on negedge of Shift clock
22-18 —	Reserved
17-16 PINCFG	Shifter Pin Configuration For pins configured as an output (PINCFG=11), this field takes effect when the register is written. **NOTE** When initially configuring PINCFG=0x11, FlexIO may briefly drive the pin low. To avoid this, software can configure PINCFG=0x10 along with the rest of the control register and then perform a subsequent write to set PINCFG=0x11. 00b - Shifter pin output disabled 01b - Shifter pin open drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	Reserved
12-8 PINSEL	Shifter Pin Select Selects which pin is used by the Shifter input or output. PINSEL=i selects the FXIO_Di pin. For pins configured as an output (PINCFG=11), this field takes effect when the register is written.
7 PINPOL	Shifter Pin Polarity For pins configured as an output (PINCFG=11), this field takes effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6-3 —	Reserved
2-0 SMOD	Shifter Mode Configures the mode of the Shifter. 000b - Disabled. 001b - Receive mode. Captures the current Shifter content into the SHIFTBUF on expiration of the Timer. 010b - Transmit mode. Load SHIFTBUF contents into the Shifter on expiration of the Timer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Reserved. 100b - Match Store mode. Shifter data is compared to SHIFTBUF content on expiration of the Timer. 101b - Match Continuous mode. Shifter data is continuously compared to SHIFTBUF contents. 110b - State mode. SHIFTBUF contents are used for storing programmable state attributes. 111b - Logic mode. SHIFTBUF contents are used for implementing programmable logic lookup table.

53.6.1.28 Shifter Configuration N Register (SHIFTCFG0 - SHIFTCFG7)

Offset

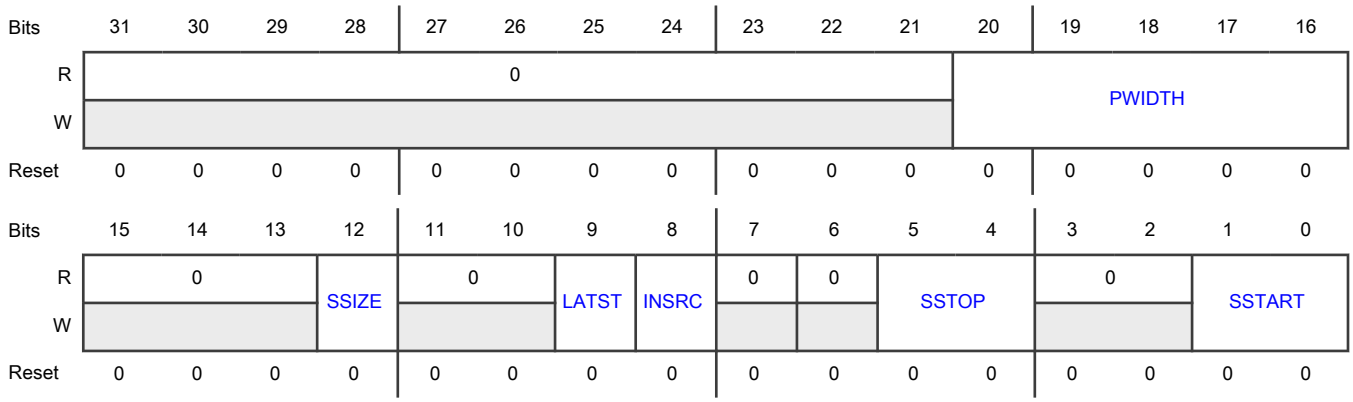
For a = 0 to 7:

Register	Offset
SHIFTCFGa	100h + (a × 4h)

Function

Contains configuration bit fields for Shifter Register N.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16	Parallel Width

Table continues on the next page...

Table continued from the previous page...

Field	Function
PWIDTH	For all Shifters, this register field configures the number of bits to be shifted on each Shift clock as follows: 1-bit shift for PWIDTH=0 2-bit shift for PWIDTH=1 4-bit shift for PWIDTH=2...3 8-bit shift for PWIDTH=4...7 16-bit shift for PWIDTH=8...15 32-bit shift for PWIDTH=16...31 For Shifters which support parallel transmit (SHIFTER0, SHIFTER4, ...) or parallel receive (SHIFTER3, SHIFTER7, ...), this register field, together with SHIFTCTL[PINSEL], also selects the pins to be driven or sampled on each Shift clock as follows: • FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL] Shifters which do not support parallel transmit or parallel receive only support parallel shift when INSRC=1. For SMOD=State, this field is used to disable state outputs. See State Mode for more details.
15-13 —	Reserved
12 SSIZE	Shifter Size Configures the size of the Shift Registers. A 24-bit shift register shifts data into bits [23:0] only and does not update bits [31:24] during shift operations. When shift register is configured for 24-bit, configuring PWIDTH=8..15 performs a 12-bit shift and PWIDTH=16..31 performs a 24-bit shift. 0b - Shift register is 32-bit. 1b - Shift register is 24-bit.
11-10 —	Reserved
9 LATST	Late Store Configures what happens when a receive or match shift register is configured to both shift and store on the same cycle. 0b - Shift register stores the pre-shift register state. 1b - Shift register stores the post-shift register state.
8 INSRC	Input Source Selects the input source for the shifter. Configuring INSRC=1 is not supported for the last shifter. 0b - Pin 1b - Shifter N+1 Output

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5-4 SSTOP	Shifter Stop bit For SMOD=Transmit, this field allows automatic stop bit insertion if the selected timer has also enabled a stop bit. For SMOD=Receive or Match Store, this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Stop bit disabled for transmitter/receiver/match store 01b - Stop bit disabled for transmitter/receiver/match store, receiver/match store will store receive data on the configured shift edge when timer in stop condition 10b - Transmitter outputs stop bit value 0 on store, receiver/match store sets error flag if stop bit is not 0, receiver/match store will also store receive data on the configured shift edge when timer in stop condition 11b - Transmitter outputs stop bit value 1 on store, receiver/match store sets error flag if stop bit is not 1, receiver/match store will also store receive data on the configured shift edge when timer in stop condition
3-2 —	Reserved
1-0 SSTART	Shifter Start bit For SMOD=Transmit, this field allows automatic start bit insertion if the selected timer has also enabled a start bit. For SMOD=Receive or Match Store, this field allows automatic start bit checking if the selected timer has also enabled a start bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on enable 01b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on first shift 10b - Transmitter outputs start bit value 0 before loading data on first shift, receiver/match store sets error flag if start bit is not 0 11b - Transmitter outputs start bit value 1 before loading data on first shift, receiver/match store sets error flag if start bit is not 1

53.6.1.29 Shifter Buffer N Register (SHIFTBUF0 - SHIFTBUF7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFa	200h + (a × 4h)

Function

Contains Shift buffer data.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Shift buffer data is used for various functions depending on the SHIFTCTL0[SMOD] setting: For SMOD=Receive, Shifter data is transferred into SHIFTBUF at the expiration of Timer. The SHIFTBUF register should only be read when the corresponding shifter status flag is set, indicating new Shifter data is available. For SMOD=Transmit, SHIFTBUF data is transferred into the Shifter before the Timer begins. For SMOD=Match Store, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). The Match is checked when the Timer expires and Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. The SHIFTBUF register should only be read when the corresponding shifter status flag is set, indicating new Shifter data is available. For SMOD=Match Continuous, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). For SMOD=Logic, SHIFTBUF[31:0] is used to implement a 5-input, 32-bit programmable logic look-up table. See Logic Mode for more details. For SMOD=State, SHIFTBUF[31:24] is used to drive the output value when this shifter is selected by the current state pointer and SHIFTBUF[23:0] is used to configure the value of the next state transition. See State Mode for more detail.

53.6.1.30 Shifter Buffer N Bit Swapped Register (SHIFTBUFBIS0 - SHIFTBUFBIS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFBISa	280h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are bit swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped. Reads return SHIFTBUF[0:31].

53.6.1.31 Shifter Buffer N Byte Swapped Register (SHIFTBUFBYS0 - SHIFTBUFBYS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFBYSa	300h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are byte swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are byte swapped. Reads return { SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24] }.

53.6.1.32 Shifter Buffer N Bit Byte Swapped Register (SHIFTBUFBBS0 - SHIFTBUFBBS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFBBSa	380h + (a × 4h)

Function

Contains SHIFTBUF register data, but is bit swapped within each byte.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped within each byte. Reads return { SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7] }.

53.6.1.33 Timer Control N Register (TIMCTL0 - TIMCTL7)

Offset

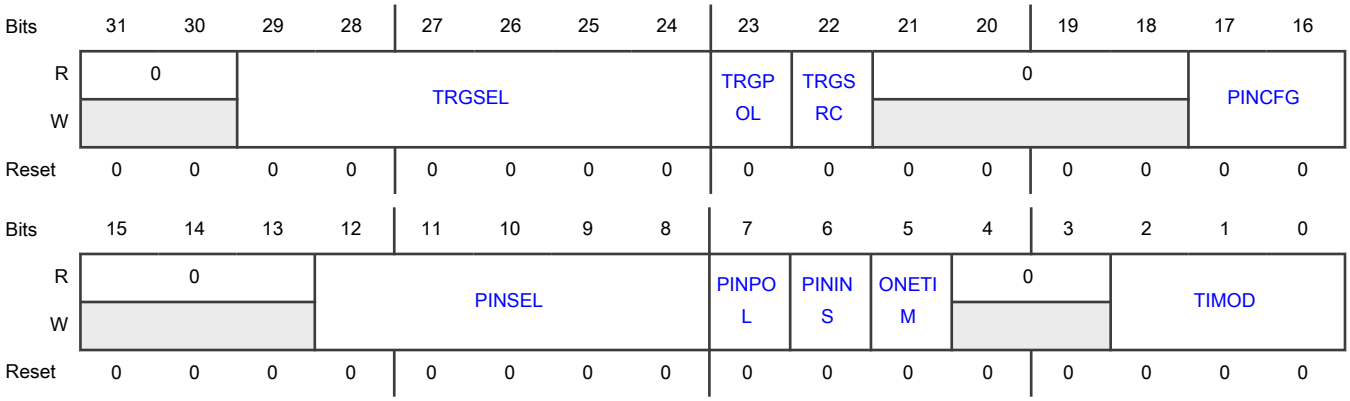
For a = 0 to 7:

Register	Offset
TIMCTLa	400h + (a × 4h)

Function

Timer Control for Timer N.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 TRGSEL	Trigger Select The valid values for TRGSEL depend on the FLEXIO_PARAM register. • When TRGSRC = 1, the valid values for N depend on PIN, TIMER, SHIFTER fields in the FLEXIO_PARAM register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• When TRGSRC = 0, the valid values for N depend on TRIGGER field in FLEXIO_PARAM register. Refer to the chip-specific FlexIO information for external trigger selection. **NOTE** For a pin, N=0 to 31. For a Shifter, N=0 to 7. For a Timer, N=0 to 7. When TRGSRC = 0 the trigger selection is configured as follows: • N - External trigger N input When TRGSRC = 1 the internal trigger can be configured to select an input pin as follows: • 2*N - Pin N input When TRGSRC = 1 the internal trigger can be configured to select a shifter/timer signal as follows: • 4*N+1 - Shifter N status flag • 4*N+3 - Timer N trigger output In other words, the expanded internal trigger selection (TRGSRC = 1) is as follows: • 0000 = Pin 0 • 0001 = Shifter 0 Flag • 0010 = Pin 1 • 0011 = Timer 0 Trigger • 0100 = Pin 2 • 0101 = Shifter 1 Flag • 0110 = Pin 3 • 0111 = Timer 1 Trigger • ... • This continues up to the Pin 31, Shifter 7 and Timer 7.
23 TRGPOL	Trigger Polarity Determines if the trigger is active high or active low. 0b - Trigger active high 1b - Trigger active low
22 TRGSRC	Trigger Source Determines if the trigger source is external or internal. 0b - External trigger selected 1b - Internal trigger selected
21-18 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-16 PINCFG	Timer Pin Configuration Configures the direction of the Timer pin. For pins configured as an output (PINCFG=11), this field takes effect when the register is written. **NOTE** When initially configuring PINCFG=0x11, FlexIO may briefly drive the pin low. To avoid this, software can configure PINCFG=0x10 along with the rest of the control register and then perform a subsequent write to set PINCFG=0x11. 00b - Timer pin output disabled 01b - Timer pin open drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output
15-13 —	Reserved
12-8 PINSEL	Timer Pin Select Selects which pin is used by the Timer input or output. PINSEL=i selects the FXIO_Di pin. For pins configured as an output (PINCFG=11), this field takes effect when the register is written.
7 PINPOL	Timer Pin Polarity For pins configured as an output (PINCFG=11), this field takes effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6 PININS	Timer Pin Input Select When set, the timer input pin is a different pin from the timer output pin. PINSEL must select an even-numbered pin when this bit is set, so the output pin is even-numbered and input pin is odd-numbered. 0b - Timer pin input and output are selected by PINSEL. 1b - Timer pin input is selected by PINSEL+1, timer pin output remains selected by PINSEL.
5 ONETIM	Timer One Time Operation When set, configures the timer to perform a single enable/disable iteration, after which software must clear the timer status flag for the timer to be enabled again. 0b - The timer enable event is generated as normal. 1b - The timer enable event is blocked unless timer status flag is clear.
4-3 —	Reserved
2-0	Timer Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIMOD	In 8-bit baud counter mode, the lower 8-bits of the counter and compare register are used to configure the baud rate of the timer shift clock, and the upper 8-bits are used to configure the shifter bit count. In 8-bit PWM high mode, the lower 8-bits of the counter and compare register are used to configure the high period of the timer shift clock, and the upper 8-bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit counter mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. In 16-bit counter disable mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. In 8-bit word counter mode, the lower 8-bits of the counter and compare register are used to configure the shifter bit count, and the upper 8-bits are used to configure the shifter word count. In 8-bit PWM low mode, the lower 8-bits of the counter and compare register are used to configure the low period of the timer shift clock and the upper 8-bits are used to configure the high period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit input capture mode, the inverted value of 16-bit counter is latched into the compare register when a timer counter disable condition is detected (as configured by TIMDIS). This also sets the timer status flag. The timer counter is immediately restarted from 0xFFFF 000b - Timer Disabled. 001b - Dual 8-bit counters baud mode. 010b - Dual 8-bit counters PWM high mode. 011b - Single 16-bit counter mode. 100b - Single 16-bit counter disable mode. 101b - Dual 8-bit counters word mode. 110b - Dual 8-bit counters PWM low mode. 111b - Single 16-bit input capture mode.

53.6.1.34 Timer Configuration N Register (TIMCFG0 - TIMCFG7)

Offset

For a = 0 to 7:

Register	Offset
TIMCFGa	480h + (a × 4h)

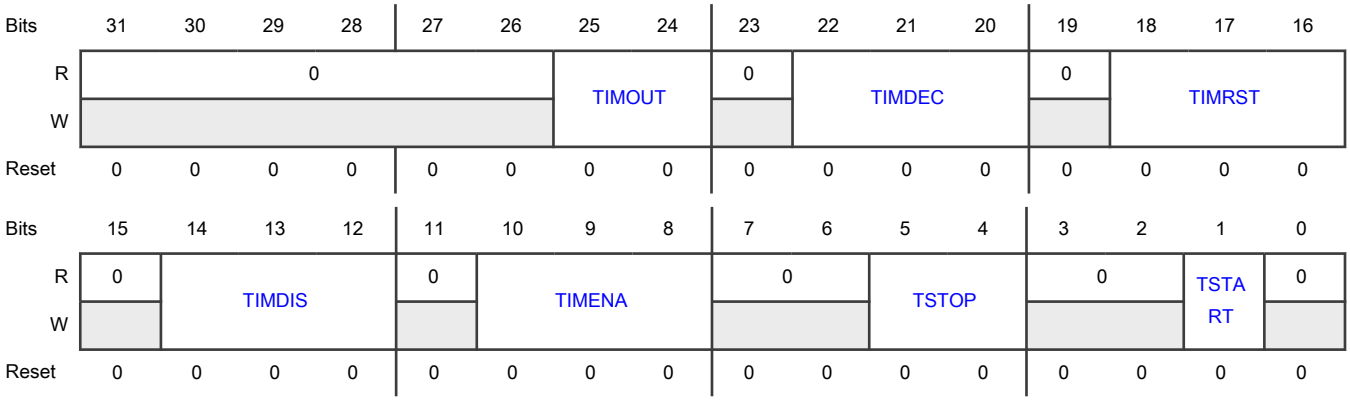
Function

The options to enable or disable the timer using the Timer N-1 enable or disable are reserved when N is evenly divisible by 4 (for example: Timer 0).

NOTE

The pin and trigger level/edges specified below refer to the signal state after being modified by the PINPOL or TRGPOL setting in the TIMCTL register. For example, "Trigger low" means that a trigger is actually at logic level 1 if the TRGPOL is set to 1 (active low).

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 TIMOUT	Timer Output Configures the initial state of the Timer Output and whether it is affected by the Timer reset. 00b - Timer output is logic one when enabled and is not affected by timer reset 01b - Timer output is logic zero when enabled and is not affected by timer reset 10b - Timer output is logic one when enabled and on timer reset 11b - Timer output is logic zero when enabled and on timer reset
23 —	Reserved
22-20 TIMDEC	Timer Decrement Configures the source of the Timer decrement and the source of the Shift clock. 000b - Decrement counter on FlexIO clock, Shift clock equals Timer output. 001b - Decrement counter on Trigger input (both edges), Shift clock equals Timer output. 010b - Decrement counter on Pin input (both edges), Shift clock equals Pin input. 011b - Decrement counter on Trigger input (both edges), Shift clock equals Trigger input. 100b - Decrement counter on FlexIO clock divided by 16, Shift clock equals Timer output. 101b - Decrement counter on FlexIO clock divided by 256, Shift clock equals Timer output.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Decrement counter on Pin input (rising edge), Shift clock equals Pin input. 111b - Decrement counter on Trigger input (rising edge), Shift clock equals Trigger input.
19 —	Reserved
18-16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset only resets the lower 8-bits that configure the baud rate. In all other modes, the timer reset resets the full 16-bits of the counter. 000b - Timer never reset 001b - Timer reset on Timer Output high. 010b - Timer reset on Timer Pin equal to Timer Output 011b - Timer reset on Timer Trigger equal to Timer Output 100b - Timer reset on Timer Pin rising edge 101b - Reserved 110b - Timer reset on Trigger rising edge 111b - Timer reset on Trigger rising or falling edge
15 —	Reserved
14-12 TIMDIS	Timer Disable Configures the condition that causes the Timer to be disabled and stop decrementing. 000b - Timer never disabled 001b - Timer disabled on Timer N-1 disable 010b - Timer disabled on Timer compare (upper 8-bits match and decrement) 011b - Timer disabled on Timer compare (upper 8-bits match and decrement) and Trigger Low 100b - Timer disabled on Pin rising or falling edge 101b - Timer disabled on Pin rising or falling edge provided Trigger is high 110b - Timer disabled on Trigger falling edge 111b - Reserved
11 —	Reserved
10-8 TIMENA	Timer Enable Configures the condition that causes the Timer to be enabled and start decrementing.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Timer always enabled 001b - Timer enabled on Timer N-1 enable 010b - Timer enabled on Trigger high 011b - Timer enabled on Trigger high and Pin high 100b - Timer enabled on Pin rising edge 101b - Timer enabled on Pin rising edge and Trigger high 110b - Timer enabled on Trigger rising edge 111b - Timer enabled on Trigger rising or falling edge
7-6 —	Reserved
5-4 TSTOP	Timer Stop Bit The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters output the contents of the stop bit when the timer is disabled. When stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00b - Stop bit disabled 01b - Stop bit is enabled on timer compare 10b - Stop bit is enabled on timer disable 11b - Stop bit is enabled on timer compare and timer disable
3-2 —	Reserved
1 TSTART	Timer Start Bit When start bit is enabled, configured shifters outputs the contents of the start bit when the timer is enabled and the timer counter reloads from the compare register on the first rising edge of the shift clock. 0b - Start bit disabled 1b - Start bit enabled
0 —	Reserved

53.6.1.35 Timer Compare N Register (TIMCMP0 - TIMCMP7)

Offset

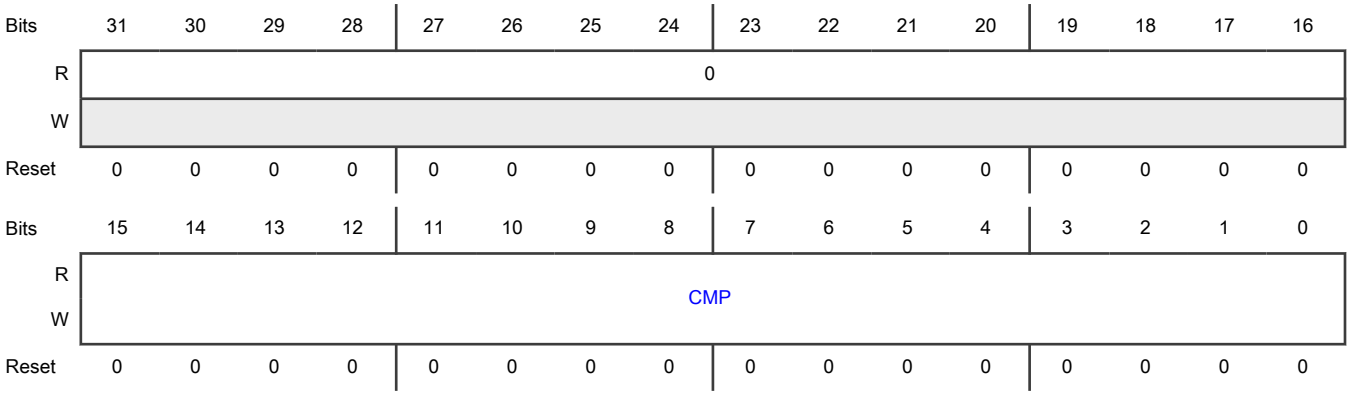
For a = 0 to 7:

Register	Offset
TIMCMPa	500h + (a × 4h)

Function

Contains the timer compare value.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 CMP	Timer Compare Value The timer compare value is loaded into the timer counter when the timer is first enabled, when the timer is reset and when the timer decrements down to zero. In 8-bit baud counter mode, the lower 8-bits configure the baud rate divider equal to (CMP[7:0] + 1) * 2. The upper 8-bits configure the number of bits in each word equal to (CMP[15:8] + 1) / 2. In 8-bit PWM high mode, the lower 8-bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2. In 16-bit counter disable mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2. In 8-bit word counter mode, the lower 8-bits configure the number of bits in each word equal to (CMP[7:0] + 1) / 2. The upper 8-bits configure the number of words to transfer equal to (CMP[15:8] + 1) / 2. In 8-bit PWM low mode, the lower 8-bits configure the low period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the high period of the output to (CMP[15:8] + 1).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In 16-bit input capture mode, the compare register is updated with the inverse of the timer counter value whenever the timer status flag is set. The timer compare register should only be read when the timer status flag is set.

53.6.1.36 Shifter Buffer N Nibble Byte Swapped Register (SHIFTBUFNBS0 - SHIFTBUFNBS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFNBSa	680h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are nibble swapped within each byte.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped within each byte. Reads return { SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4] }.

53.6.1.37 Shifter Buffer N Half Word Swapped Register (SHIFTBUFHWS0 - SHIFTBUFHWS7)

Offset

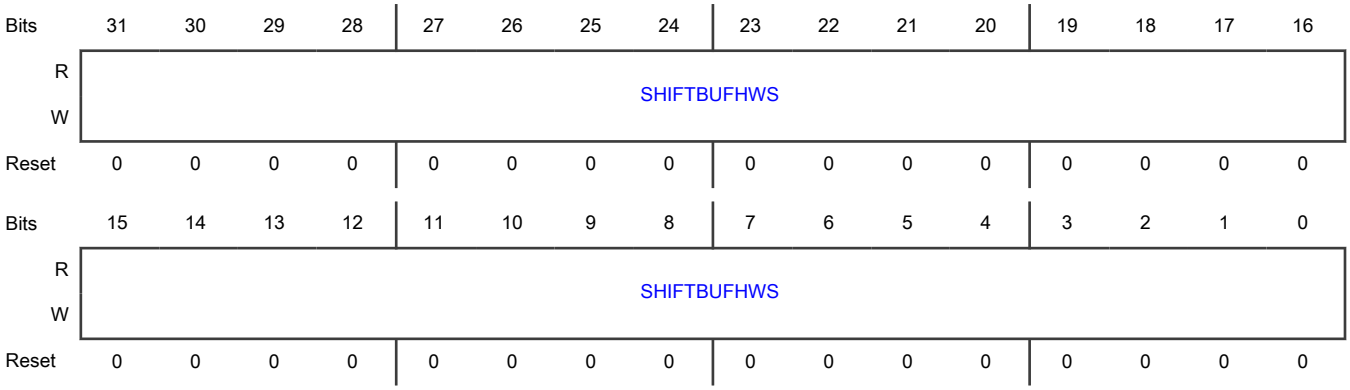
For a = 0 to 7:

Register	Offset
SHIFTBUFHWSa	700h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are halfword swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are halfword swapped. Reads return { SHIFTBUF[15:0], SHIFTBUF[31:16] }.

53.6.1.38 Shifter Buffer N Nibble Swapped Register (SHIFTBUFNIS0 - SHIFTBUFNIS7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFNISa	780h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are nibble swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFNIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped. Reads return { SHIFTBUF[3:0], SHIFTBUF[7:4], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[27:24], SHIFTBUF[31:28] }.

53.6.1.39 Shifter Buffer N Odd Even Swapped Register (SHIFTBUFOES0 - SHIFTBUFOES7)

Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFOESa	800h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register have odd and even bits partitioned separately.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFOES															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFOES															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SHIFTBUFOES	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register have the odd and even bits partitioned separately. Only 32-bit accesses are supported to this register. Reads return { SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1], SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18], SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0] }.

53.6.1.40 Shifter Buffer N Even Odd Swapped Register (SHIFTBUFEOS0 - SHIFTBUFEOS7)

Offset

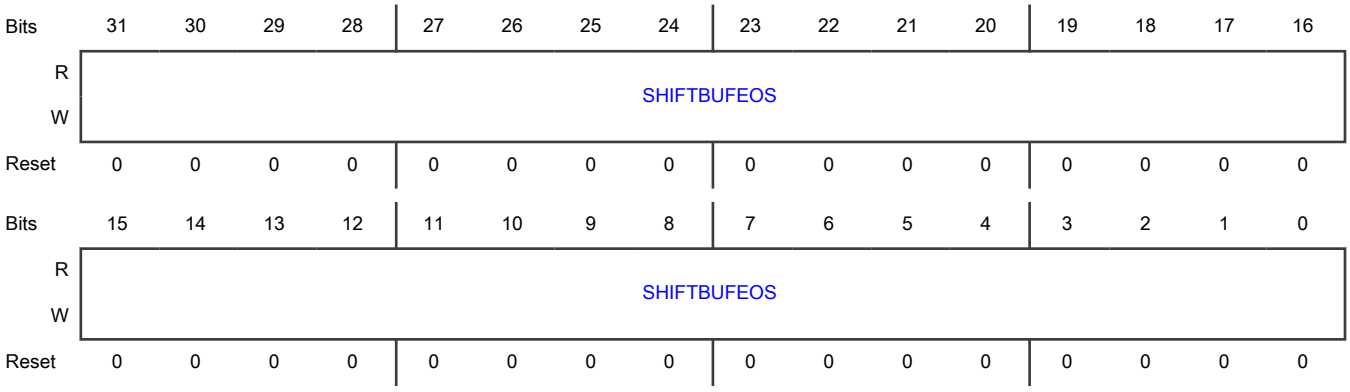
For a = 0 to 7:

Register	Offset
SHIFTBUFEOSa	880h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register have even and odd bits partitioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFEOS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register have the even and odd bits partitioned separately. Only 32-bit accesses are supported to this register. Reads return { SHIFTBUF[30], SHIFTBUF[28], SHIFTBUF[26], SHIFTBUF[24], SHIFTBUF[22], SHIFTBUF[20], SHIFTBUF[18],

Table continues on the next page...

Field	Function
	SHIFTBUF[16], SHIFTBUF[14], SHIFTBUF[12], SHIFTBUF[10], SHIFTBUF[8], SHIFTBUF[6], SHIFTBUF[4], SHIFTBUF[2], SHIFTBUF[0], SHIFTBUF[31], SHIFTBUF[29], SHIFTBUF[27],SHIFTBUF[25],SHIFTBUF[23],SHIFTBUF[21],SHIFTBUF[19],SHIFTBUF[17],SHIFTBUF[15],SHIFTBUF[13],SHIFTBUF[11],SHIFTBUF[9], SHIFTBUF[7], SHIFTBUF[5], SHIFTBUF[3], SHIFTBUF[1] }.

53.6.1.41 Shifter Buffer N Halfword Byte Swapped Register (SHIFTBUFHBS0 - SHIFTBUFHBS7)

Offset

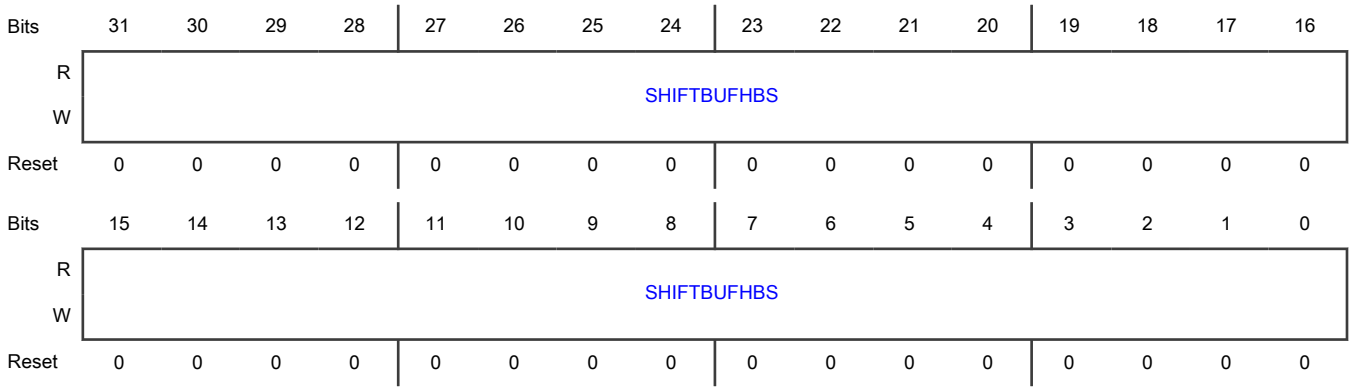
For a = 0 to 7:

Register	Offset
SHIFTBUFHBSa	900h + (a × 4h)

Function

SHIFTBUF register contents, but contents of this register are halfword byte swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFHBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are halfword byte swapped. Reads return { SHIFTBUF[23:16], SHIFTBUF[31:24], SHIFTBUF[7:0], SHIFTBUF[15:8] }.

Chapter 54
Semaphores2 (SEMA42)

54.1 Chip-specific SEMA42 information

Table 393. Reference links to related information

Topic	Related module	Reference
Full description	SEMA42	SEMA42
System memory map		System memory map
Clocking		Clock distribution
Signal multiplexing	Port control	Signal multiplexing

54.1.1 Module instances

This device has one instance of the SEMA42 module.

54.1.2 Domain ID

The following table lists modules assigned to each domain.

Table 394. SEMA42 domain ID assignment

Domain ID	Modules
0	CM33 Code and Data in normal functional mode
1	CM33 Code and Data in Debug mode
2	DMA0
3	EdgeLock Secure Enclave
4	Data Stream
5	Radio NBU

54.2 Overview

SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single - write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

54.2.1 Block diagram

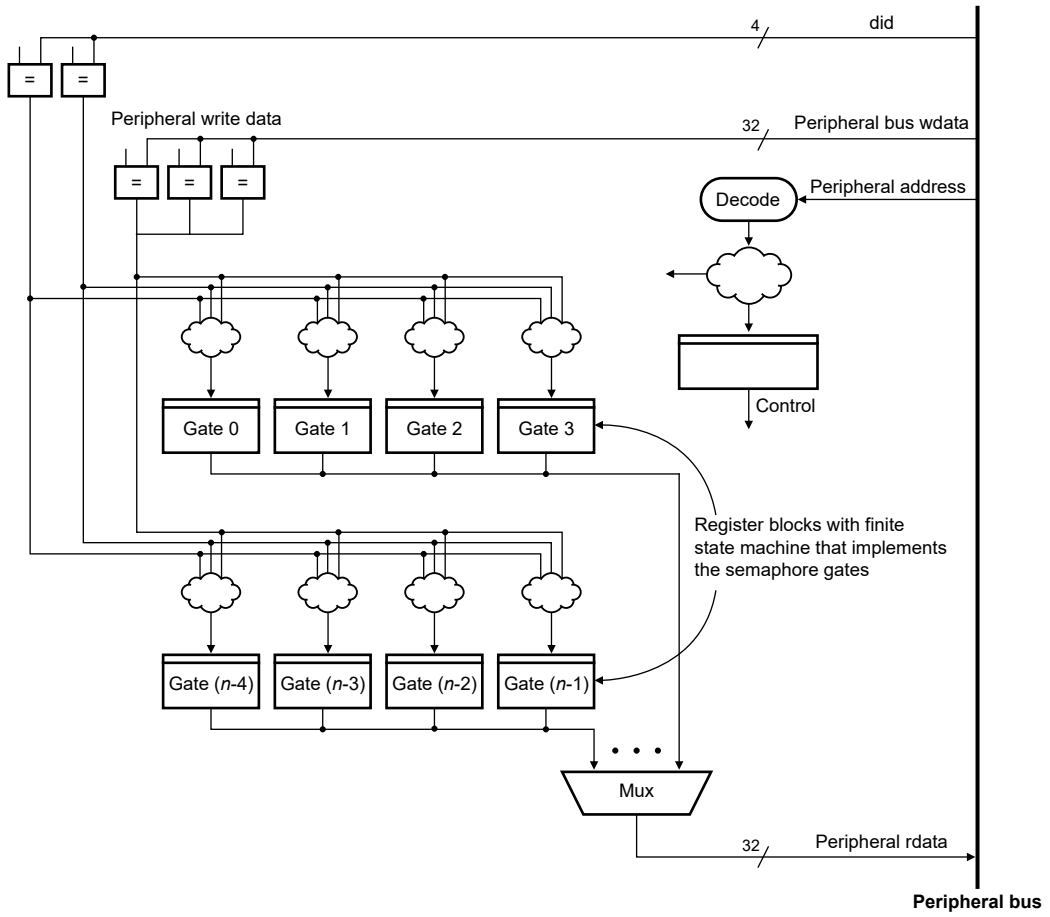


Figure 207. SEMA42 block diagram

54.2.2 Features

SEMA42 implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Support for 16 hardware-enforced gates in a multi-domain configuration that supports up to 15 domains.
- Each hardware gate appears as a 16-state, 4-bit state machine.
- Support for secure reset mechanisms to clear the contents of individual gates, as well as a clear-all capability.
- Memory-mapped slave peripheral that offers programming-model accesses.

54.3 Memory map/register definition

You can access these registers only in Supervisor mode. User accesses terminate with an error.

54.3.1 SEMA42 register descriptions

54.3.1.1 SEMA42 memory map

SEMA42 base address: 4003_F000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Fh	Gate (GATE0 - GATE15) [1]	8	RW	00h
42h	Reset Gate Read (RSTGT_R)	16	R	0000h
42h	Reset Gate Write (RSTGT_W)	16	W	See section

1. In this array, the index and offset values of the registers do not increment in direct alignment. For details, see the register description.

54.3.1.2 Gate (GATE0 - GATE15)

Offset

For n = 0 to 15:

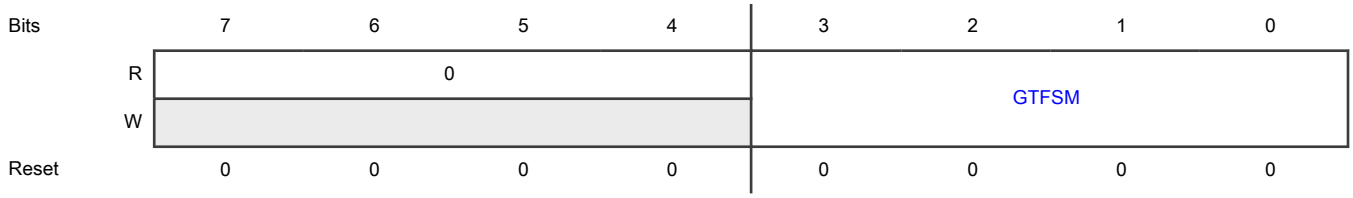
Register	Offset
GATEn	0h + (n + 3 - 2 × (n mod 4))

Function

Implements each semaphore gate in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical domain-identifier number in conjunction with the data patterns to validate all attempted write operations. Only domain masters can modify the gate registers. After a gate locks, only the locking domain must open (unlock) the gate.

You can read multiple gate values in a single access. However, you can update only a single gate at a time, via a write operation. If you attempt to write a byte-wide value that is neither the unlock value (00h) nor the appropriate lock value (domainID_number + 1), SEMA42 considers this as "no operation" and does not change any gate state. Attempts to write multiple gates in a single-aligned access with a size larger than 8 bits (byte) generate an error termination and do not allow any gate state changes.

Diagram



Fields

Field	Function
7-4 —	Reserved
3-0 GTFSM	Gate Finite State Machine Indicates the state of the gate for the last domain that locked the gate. This can be useful during system debug.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The hardware gate has a 16-state implementation, defined as: 0000b - The gate is unlocked (free). 0001b - Domain 0 locked the gate. 0010b - Domain 1 locked the gate. 0011b - Domain 2 locked the gate. 0100b - Domain 3 locked the gate. 0101b - Domain 4 locked the gate. 0110b - Domain 5 locked the gate. 0111b - Domain 6 locked the gate. 1000b - Domain 7 locked the gate. 1001b - Domain 8 locked the gate. 1010b - Domain 9 locked the gate. 1011b - Domain 10 locked the gate. 1100b - Domain 11 locked the gate. 1101b - Domain 12 locked the gate. 1110b - Domain 13 locked the gate. 1111b - Domain 14 locked the gate.

54.3.1.3 Reset Gate Read (RSTGT_R)

Offset

Register	Offset
RSTGT_R	42h

Function

Describes the specific hardware gate to be reset and records the logic number of domain. Reset Gate Write (RSTGT_W) also describe the same register showing the fields when you write it.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		RSTGSM		RSTGMS				RSTGTN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved
13-12 RSTGSM	Reset Gate Finite State Machine Indicates the encoded state machine value when you read the register. RSTGSM = 10b is valid for only a single machine cycle, so a read can never return this value. SEMA42 maintains the reset state machine in a 2-bit, 3-state implementation, defined as follows: 00b - Idle, waiting for the first data pattern write. 01b - Waiting for the second data pattern write 10b - The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. 11b - This state encoding is never used and therefore reserved.
11-8 RSTGMS	Reset Gate Domain Records the logical number of the domain performing the gate reset function. To succeed, this function requires that the same domain initiate the two consecutive writes to this register. SEMA42 updates the field each time a write to this register occurs.
7-0 RSTGTN	Reset Gate Number Specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

54.3.1.4 Reset Gate Write (RSTGT_W)

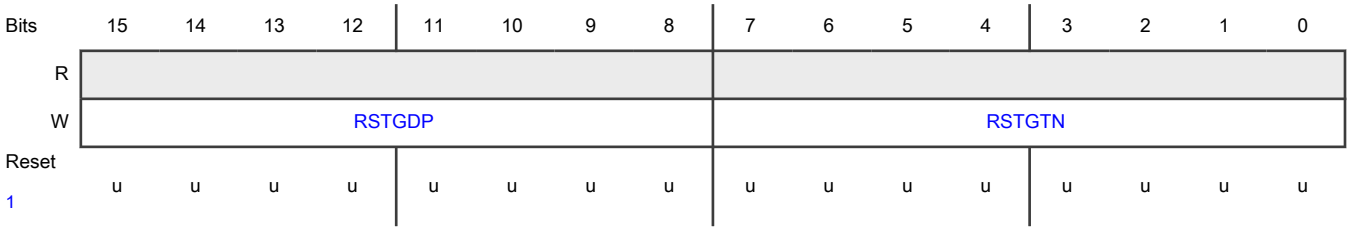
Offset

Register	Offset
RSTGT_W	42h

Function

Specifies the hardware gate to reset when data patterns are specified.

Diagram



1. Reset value is not applicable for writes.

Fields

Field	Function
15-8 RSTGDP	Reset Gate Data Pattern You access this field with the specified data patterns on the two consecutive writes to enable the gate-reset mechanism. For the first write, RSTGDP must be E2h. For the second write, RSTGDP must be 1Dh.
7-0 RSTGTN	Reset Gate Number Specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

54.4 Functional description

The intent of the hardware gate implementation is to specify a protocol where the locking domain must unlock the gate. However, some systems may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset. To support this special gate reset requirement, SEMA42 implements a secure reset mechanism that allows you to initialize a hardware gate (or all the gates) by following a specific dual-write access pattern. The secure-gate reset:

- Uses a technique similar to that required for the servicing of a software watchdog timer
- Requires two consecutive writes with pre-defined data patterns from the same domain

You must do this to force the clearing of the specified gate(s). The required access pattern as follows:

1. A domain performs a 16-bit write to the RSTGT memory location. The most significant byte (RSTGT_W[RSTGDP]) must be E2h. The value of least significant byte is irrelevant for this reference and can be anything.
2. The same domain then performs a second 16-bit write to the RSTGT location. For this write, the upper byte (RSTGT_W[RSTGDP]) is the logical complement of the first data pattern (1Dh) and the lower byte (RSTGT_W[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate or all gates to be cleared. If the same domain writes incorrect data on the second access or another domain performs the second write access, SEMA42 aborts the special gate reset sequence and does not assert an error signal.
3. Reads of the RSTGT location return information on the 2-bit reset state machine (RSTGT_R[RSTGSM]) that implements these functions:
   - The domain performing the reset (RSTGT_R[RSTGMS])
   - The last-cleared gate number(s) (RSTGT_R[RSTGTN])

   Reads of the RSTGT register do not affect the secure-reset finite state machine in any manner.

54.4.1 Multi-core programming: software gates

Multi-processor systems require a function that can be used to safely and easily provide a locking mechanism for system software to control access to shared data structures, shared hardware resources, and so on. The software uses the gating mechanisms to serialize (and synchronize) accesses to shared data and/or resources to prevent race conditions and preserve memory coherency between different processes and domains.

Consider the following description of a typical use-case: domain*X* enters a section of code. where shared data values are to be updated. The domain must first acquire a semaphore. Think of this as the locking (or closing) of a software gate. After the gate locks, a properly-architected software system does not allow other processes (or domains) to execute the same code segment or modify the shared data structure protected by the gate. In other words, the system locks out other processes/domains. Many software implementations include a spin-wait loop within the lock function until the gate locks. After domain X obtains the lock, domain X continues execution and updates the data values protected by the particular lock. After domain X completes the updates, it unlocks (or opens) the software gate, allowing other processes/domains access to the updated data values.

A correctly-implemented system solution must follow these important rules:

- A gate variable must protect all writes to shared data values or shared hardware resources.
- After a domain locks a gate, the system must block other processes/domains from accessing the shared data or resources. Software conventions enforce this.
- The domain that locks a particular gate is the only domain that can open (unlock) that gate.

Information in the hardware gate identifying the locking domain can be extremely useful for system-level debugging.

54.4.2 16 Hardware-enforced gates

Gates appear as a 16-entry byte-size array with read and write accesses.

Domains lock gates by writing "domainID_number+1" to the appropriate gate and must read back the gate value to verify that the lock operation succeeded.

After the gate locks, the locking domain unlocks the gate by writing zeroes.

- 16-state implementation

  If gate = 0h, then state = unlocked

  if gate = 1h, then state = locked by domain (master) 0

  if gate = 2h, then state = locked by domain (master) 1

  …

  if gate = Fh, then state = locked by domain (master) 14

SEMA42 uses the logical domain number and the specified data patterns as reference attributes to validate all write operation.

After a gate locks, the locking domain must unlock the gate by writing zeroes.

54.4.3 State machine of the GATE*n* registers

This section describes more about the SEMA42 functional operation and specific details of the state machines of the GATE*n* registers.

As described previously, each of the GATE*n* registers implements a 4-bit, 16-state machine. The following figure shows a simplified diagram of the state transitions for each gate.

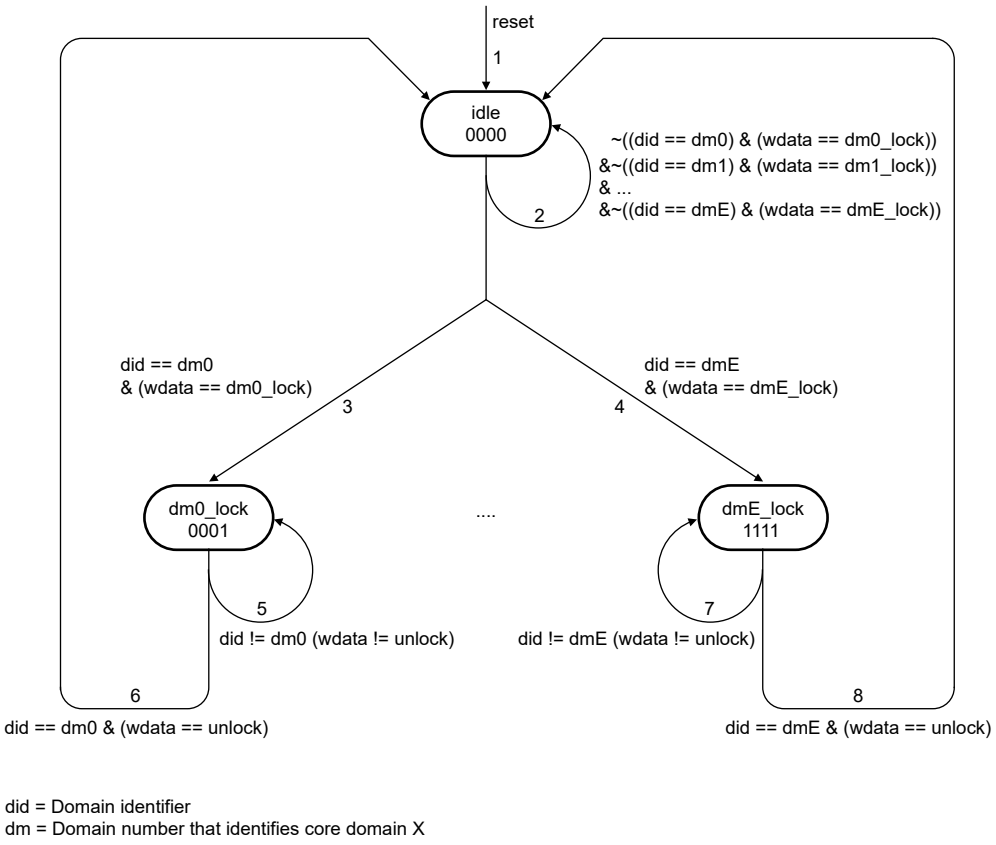


Figure 208. GATE*n* state machine

In the figure above, "dmE" represents domain 14 (E in hexadecimal). The platform passes the domain number to SEMA42.

The following table defines the GATE*n* state transitions.

Table 395. GATE*n* state transitions

Current state	Next state	Transition	Description
–	idle	1	Any reset, whether a system reset or a software-initiated gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding domain occurs, the gate remains in the idle state.
idle	dm0_lock	3	When domain 0h initiates a write of the dm0_lock data value, the gate transitions into the dm0_lock state.
idle	dmE_lock	4	When domain Eh initiates a write of the dmE_lock value, the gate transitions into the dmE_lock state.
dm0_lock	dm0_lock	5	When in this state, the gate remains here if any attempted write is not from domain 0h with the unlock data value.
dm0_lock	idle	6	The gate returns to the idle (unlocked) state after a write from domain 0h with the unlock data value occurs.

Table continues on the next page...

Table 395. GATE*n* state transitions (continued)

Current state	Next state	Transition	Description
dmE_lock	dmE_lock	7	When in this state, the gate remains here if any attempted write is not from domain Eh with the unlock data value.
dmE_lock	idle	8	The gate returns to the idle (unlocked) state after a write from domain Eh with the unlock data value occurs.

SEMA42 uses these gate data values:

- The lock data value is (domain number) + 1.
- The unlock data value is 00h.

54.4.4 Clocking

This module has no clocking considerations.

54.4.5 Interrupts

This module has no interrupts.

54.5 External signals

This module has no external signals.

54.6 Initialization

This module does not require initialization.

Chapter 55
Radio Platform

55.1 Chip-specific Radio Platform information

55.1.1 XTAL_OUT_EN

The XTAL_OUT_EN in this chapter is the RF_XTAL_OUT_ENABLE in the #unique_1319.

55.2 Introduction

55.2.1 Introduction

The 2.4GHz C40 Multi-Protocol Radio supports concurrent operation in the 2.4GHz bands.

The 2.4GHz band radio covers 2.36-2.4835GHz and is built to support multiple protocols.

- Bluetooth Low Energy v5.0 with support for all optional features
  - 1Mbps (legacy mode)
  - 2Mbps (new for v5.0)
  - Long Range 125kbps & 500kbps (new for v5.0)
- IEEE 802.15.4-2006 compliant PHY & MAC
  - Zigbee
  - Thread
- IEEE 802.15.4j-2012 Medical Body Area Network (MBAN) frequency bands spanning from 2360-2400MHz.
- Generic 2-level FSK/GFSK/MSK/GMSK
  - 250kbps
  - 500kbps
  - 1000kbps
  - 2000kbps
- Concurrent 802.15.4 and BluetoothLE operation

The Radio block diagram is provided for illustration of the internal organization of the radio.

The Narrowband Unit (NBU) is a dedicated compute subsystem for the Narrowband Radio. The NBU is comprised of an Arm Cortex-M3 and associated peripherals that, in conjunction with the Bluetooth Unit, support the BluetoothLE protocol. Among the supporting peripherals are timers, a messaging unit, and dedicated flash (distinct from the SoC flash). While the NBU allows for flexibility for evolving requirements, firmware to implement radio protocols is intended to be developed and delivered by NXP.

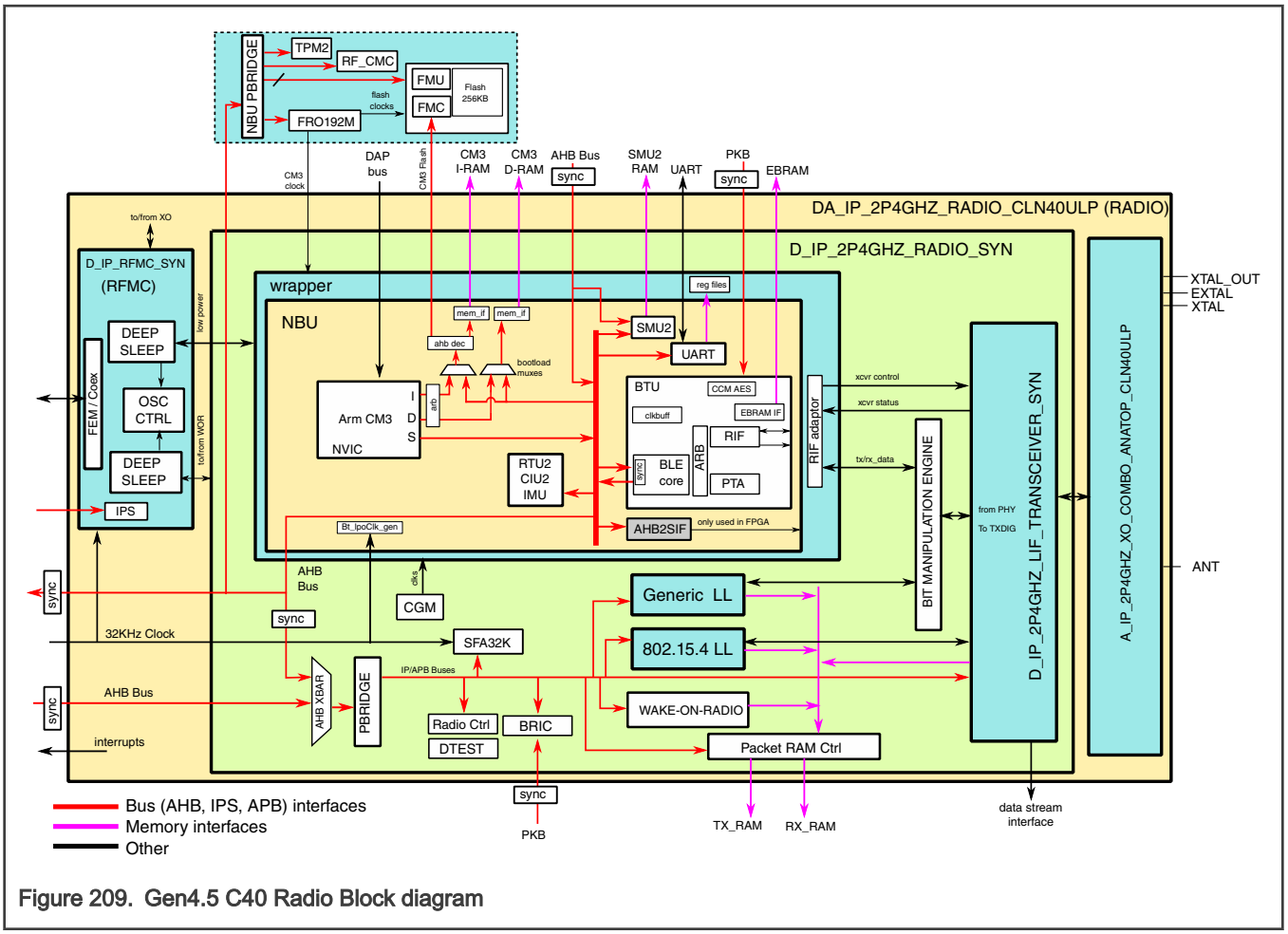


Figure 209. Gen4.5 C40 Radio Block diagram

55.3 RFMC

55.3.1 Introduction

The Radio Mode Controller (RFMC) is responsible for sequencing the power mode of the 2.4GHz radio domain and controlling the radio crystal oscillator (XO). Specifically, it supports the following features:

- Support for Deep Sleep and Power Down modes.
- Low power mode entry/exit for the 2.4GHz radio power domain.
- Enable of XO supported from both internal and external sources.
- Support for external coexistence interface for 2.4GHz frequency band
- 32kHz timer to control low power entry/exit times via Wake on Radio (WOR) or Manual (MAN) counters.
- Timer offsets to ensure XO and radio power restored prior to WOR/MAN exit event.
- Interrupts supported for XO and low power radio wakeup events.

55.3.2 Block Diagram

The following diagram gives a high-level overview of the RFMC.

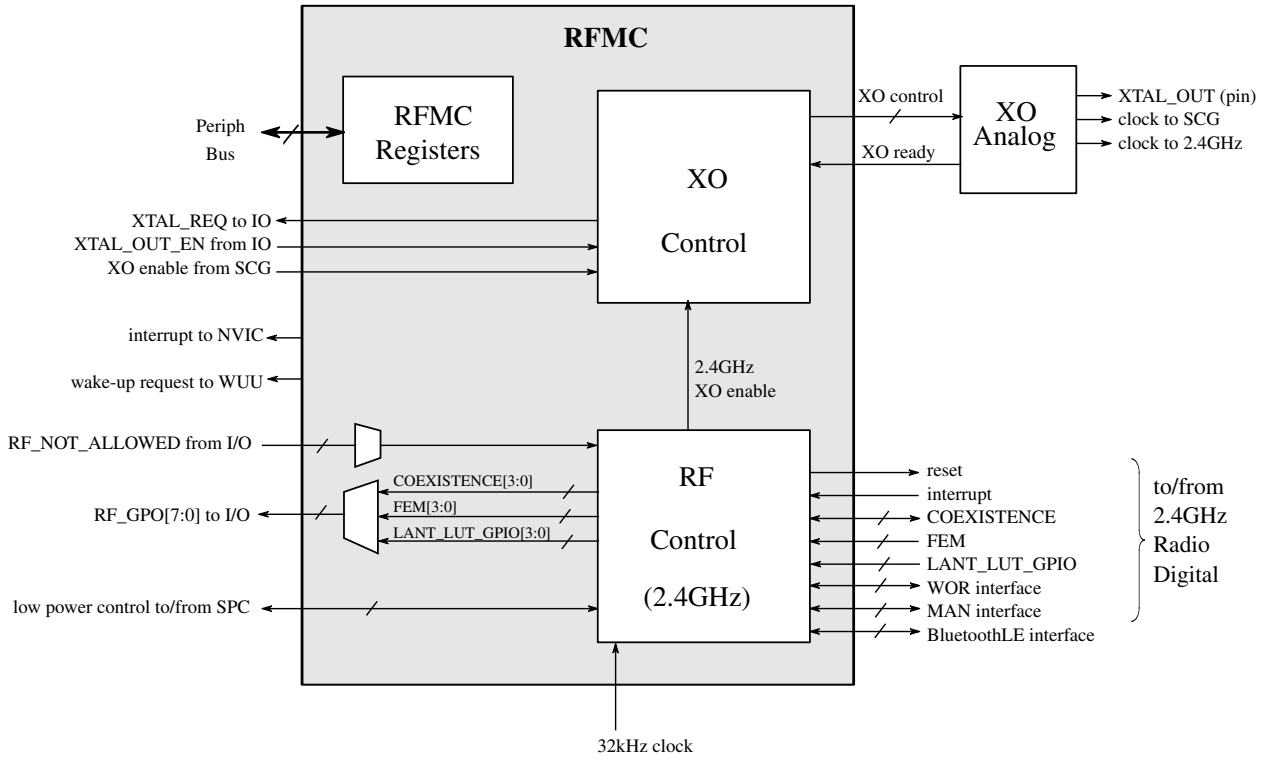


Figure 210. RFMC Block Diagram

55.3.3 Signals

Table 396. RFMC external I/O

Signal	Description	I/O
XO Interface		
XTAL_REQ	Request to enable an external crystal when internal XO is bypassed.	O
XTAL_OUT_EN	Enable from an external source to activate internal XO.	I
Radio Coexistence, FEM Control, and Localization Antenna Select Interfaces		
RF_GPO[7:0]	Software can select any two of {4 Coexistence signals, 4 FEM Control signals, 4 Localization Antenna Select signals} on this 8bit output.	O
RF_NOT_ALLOWED[4:0]	Software can select 1 of 5 input pins to use for the "RF not allowed" Coexistence signal, which blocks radio activity when asserted.	I

55.3.4 Memory Map and Registers

55.3.4.1 RFMC register descriptions

55.3.4.1.1 RFMC memory map

RFMC base address: 4004_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RFMC Version ID Register (VERID)	32	R	0200_4500h
4h	RFMC Parameter Register (PARAM)	32	R	0000_0001h
8h	RFMC Control Register (CTRL)	32	RW	0000_0000h
Ch	XO Control Register (XO_CTRL)	32	RW	0022_0000h
10h	XO Status Register (XO_STAT)	32	RW	0000_0000h
14h	XO Test Register (XO_TEST)	32	RW	0000_61E5h
18h	2.4GHz Radio Control Register (RF2p4GHz_CTRL)	32	RW	2000_0F00h
1Ch	2.4GHz Radio Status Register (RF2p4GHz_STAT)	32	RW	0000_0400h
20h	2.4GHz Radio Coexistence Register (RF2p4GHz_COEXT)	32	RW	0000_0000h
24h	2.4GHz TIMER Register (RF2p4GHz_TIMER)	32	RW	0000_0000h
28h	2.4GHz WOR Register 1 (RF2p4GHz_WOR1)	32	R	0000_0000h
2Ch	2.4GHz WOR Register 2 (RF2p4GHz_WOR2)	32	RW	0000_0000h
30h	2.4GHz MAN Register 1 (RF2p4GHz_MAN1)	32	R	0000_0000h
34h	2.4GHz MAN Register 2 (RF2p4GHz_MAN2)	32	RW	0000_0000h
38h	2.4GHz MAN Register 3 (RF2p4GHz_MAN3)	32	R	0000_0000h
3Ch	2.4GHz MAN Register 4 (RF2p4GHz_MAN4)	32	R	0000_0000h

55.3.4.1.2 RFMC Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RADIO_ID															
W																
Reset	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 MAJOR	Major RFMC Version Number This read only field returns the major version number of the RFMC.
23-16 MINOR	Minor RFMC Version Number This read only field returns the minor version number of the RFMC.
15-0 RADIO_ID	Radio Identification Number This 16bit read only field returns the Radio ID. [15:12] : Radio major family (e.g. 4'd4 = Gen4) [11: 8] : Radio minor family (e.g. 4'd0 = GenX.0) [7: 4] : Reserved [3: 0] : Radio ECO/TO number, starting from 4'd0

55.3.4.1.3 RFMC Parameter Register (PARAM)

Offset

Register	Offset
PARAM	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															RF2p4 GH...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

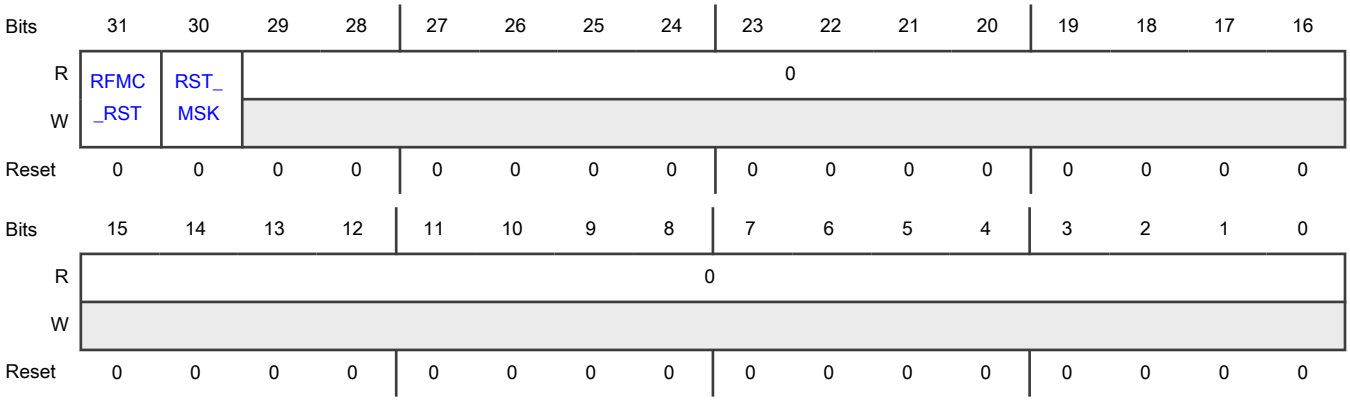
Field	Function
0 RF2p4GHz_EN	Indicates whether 2.4GHz radio is enabled (as determined by fuse bit). When disabled, the 2.4GHz radio will remain in Power Down. 0b - 2.4GHz radio disabled 1b - 2.4GHz radio enabled

55.3.4.1.4 RFMC Control Register (CTRL)

Offset

Register	Offset
CTRL	8h

Diagram



Fields

Field	Function
31 RFMC_RST	S/W System Reset for RFMC This causes a system reset for the RFMC. It resets all registers in the RFMC except for the RST_MSK and RFMC_RST register bits. This also resets the 2.4GHz radio. Note, the RFMC_RST bit is only reset on a POR/LVD event. 0b - Release the RFMC from reset 1b - Hold the RFMC in reset
30 RST_MSK	Reset Mask When set, masks system reset events (does not include POR/LVD) from resetting the RFMC, and 2.4GHz radio. Note the RFMC_RST bit is still functional, but will not reset the RST_MSK bit. The RST_MSK bit is only reset on a POR/LVD event.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-0 —	Reserved

55.3.4.1.5 XO Control Register (XO_CTRL)

Offset

Register	Offset
XO_CTRL	Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	XO_ANA_...	XO_ANA_...	XO_LDO_...	XO_LDO_...	SPARE				XTAL_RD...	XTAL_RD...	EXT_MODE	LDO_BYP...	XTAL_OU...	RDY_CNT...	RDY_CNT	
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WKUP_OFFSET						0	XTAL_EN...	XTAL_RE...	XTAL_OU...	0	EXT_IE	INT_IE	RDY_IE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 XO_ANA_EN	XO Analog Enable When the XO_ANA_OVR bit is set, this bit will control the XO analog enable. Note this bit only enables the XO analog, but will not enable its output to the SCG, XTAL_OUT pin, or 2.4GHz radio domain. 0b - XO analog disabled 1b - XO analog enabled
30 XO_ANA_OVR	XO Analog Enable Override When set, allows the XO_ANA_EN bit to override any internal or external requests for the XO. 0b - XO analog enable not overridden 1b - XO analog enable overridden by XO_ANA_EN bit
29 XO_LDO_EN	XO LDO Enable When the XO_LDO_OVR bit is set, this bit will control the XO LDO enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - XO LDO disabled 1b - XO LDO enabled
28 XO_LDO_OVR	XO LDO Enable Override When set, allows the XO_LDO_EN bit to override any internal or external requests for the XO LDO. 0b - XO LDO enable not overridden 1b - XO LDO enable overridden by XO_LDO_EN bit
27-24 SPARE	XO Spare Registers Bit 3:1 - Reserved Bit 0 - Bypass CLK_AUX_DRV_LDO
23 XTAL_RDY_OVR	XTAL Ready Override When the XTAL_RDY_OVR_EN bit is set, this bit will override the XTAL ready signal from the XO analog.
22 XTAL_RDY_OVR_EN	XTAL Ready Override Enable When set, allows the XTAL_RDY_OVR to override the XTAL ready signal from the XO analog.
21 EXT_MODE	External Clock Mode 0b - DC coupled external clock mode (amplifier powered down). 1b - AC coupled external clock mode or crystal mode (amplifier powered up).
20 LDO_BYPASS	XO LDO Bypass
19 XTAL_OUT_INV	XO Clock Output Invert Inverts the polarity of the XTAL_OUT clock relative to the internal XO clock. Also referred to as AUX_DRV_FLIP. 0b - XTAL_OUT not inverted 1b - XTAL_OUT inverted
18 RDY_CNT_OFF	XTAL Ready Count Disable 0b - XTAL Ready Count Enabled 1b - XTAL Ready Count Disabled
17-16 RDY_CNT	XTAL Ready Count Configures the ready counter value (in number of XO clock cycles) for XO startup time. 00b - 1024

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - 2048 10b - 4096 11b - 8192
15-14 —	Reserved
13-8 WKUP_OFFSET	XO Wakeup Offset Configures number of 32kHz reference clocks before a MAN/WOR wakeup event to enable the XO.
7 —	Reserved
6 XTAL_EN_IBE	XTAL_OUT_EN Input Pin Enable Enables an external source to request the XO to be enabled (see chip configuration chapter for pin mapping). 0b - XTAL_OUT_EN input pin disabled 1b - XTAL_OUT_EN input pin enabled
5 XTAL_REQ_OBE	XTAL_REQ Output Pin Enable Enables output pin for XO request (see chip configuration chapter for pin mapping). 0b - XTAL_REQ output pin disabled 1b - XTAL_REQ output pin enabled
4 XTAL_OUT_EN	XTAL_OUT Output Pin Enable Enables the XO clock and drives it to the XTAL_OUT output pin (see chip configuration chapter for pin mapping). 0b - XTAL_OUT output disabled 1b - XTAL_OUT output enabled
3 —	Reserved
2 EXT_IE	XO External Request Interrupt Enable Configures the XO to generate an interrupt when the EXT_FLAG is set. 0b - XO external request interrupt disabled 1b - XO external request interrupt enabled
1	XO Internal Request Interrupt Enable Configures the XO to generate an interrupt when the INT_FLAG is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
INT_IE	0b - XO internal request interrupt disabled 1b - XO internal request interrupt enabled
0 RDY_IE	XTAL Ready Interrupt Enable Configures the XO to generate an interrupt when the RDY_FLAG is set. 0b - XTAL ready interrupt disabled 1b - XTAL ready interrupt enabled

55.3.4.1.6 XO Status Register (XO_STAT)

Offset

Register	Offset
XO_STAT	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										XO_EN	XTAL_RDY	0	EXT_FLAG	INT_FLAG	RDY_FLAG
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 XO_EN	XO_EN Reflects the instantaneous value of the XO enable signal from RFMC to the XO. This signal combines all of the sources which can enable the XO.
4	XTAL Ready

Table continues on the next page...

Table continued from the previous page...

Field	Function
XTAL_RDY	Reflects the instantaneous value of the XTAL ready signal from the XO analog.
3 —	Reserved
2 EXT_FLAG	XO External Request Flag Flag which sets after an external source (XTAL_OUT_EN bit) has requested the XO to be enabled (write 1 to clear).
1 INT_FLAG	XO Internal Request Flag Flag which sets after an internal source (e.g. SCG or 2.4GHz radios) has requested the XO to be enabled (write 1 to clear).
0 RDY_FLAG	XTAL Ready Flag Flag which sets after the XTAL ready output from XO analog asserts (write 1 to clear).

55.3.4.1.7 XO Test Register (XO_TEST)

Offset

Register	Offset
XO_TEST	14h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											LDO_FOR...	LDO_BUMP		LDO_TRIM	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DYN_CAP	DYN_ISEL	AMP_FOR...	AUX_PD	CAP_OFF	CDAC						ISEL			
W																
Reset	0	1	1	0	0	0	0	1	1	1	1	0	0	1	0	1

Fields

Field	Function
31-21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 LDO_FORCE	XO LDO Force PTAT Startup
19-18 LDO_BUMP	XO LDO PTAT Current Bump 00b - PTAT current bump default 01b - PTAT current boost: +30%
17-16 LDO_TRIM	XO LDO Output Voltage Trim 00b - 0.92V 01b - 0.885V 10b - 0.955V 11b - 1.011V
15 —	Reserved
14 DYN_CAP	XO On-chip Load Capacitor: enable switching during startup
13 DYN_ISEL	XO Amplifier: enable current switching during startup
12 AMP_FORCE	XO Amplifier Force PTAT Startup
11 AUX_PD	XO CLK_AUX_DRV Powerdown Also referred to as CLK_AUX_DRV_PDN_FORCE.
10 CAP_OFF	XO Load Capacitor Disable
9-4 CDAC	XO On-chip Load Capacitor Trim These values are per simulation and they are not guaranteed. The parasitic capacitance vary from silicon process and temperature. It only takes into account the internal capacitances including pin, packaging/ wire bond, pads and parasitic component in the chip. It does not consider parasitic capacitance added in the PCB 00_0000b - 5.2pF 11_1111b - 14.3pF
3-0 ISEL	XO Amplifier Current Select Selectable from 40uA to 640uA, with 40uA step size. Example programming shown below

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000b - 40uA (min) 0001b - 80uA 0101b - 240uA (default) 1111b - 640uA (max)

55.3.4.1.8 2.4GHz Radio Control Register (RF2p4GHz_CTRL)

Offset

Register	Offset
RF2p4GHz_CTRL	18h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RST	RF_POR	CPU_RST	CPU_RST...	CLK_OVR				XO_EN	XO_EN_G...	LP_STOP...	SFA_TRIG_EN			LP_WKUP_DLY	
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LP_WKUP_DLY				LP_MODE				LP_ENTER	BLE_LP_...	BLE_WKUP	LP_WKUP...	RFACT_IE	BLE_WKU...	MAN_WKU...	WOR_WKU...
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31 RST	S/W Reset for 2.4GHz Radio When set, the RST bit will cause a reset of the 2.4GHz radio domain. Note, setting this bit has no effect on the RFMC's registers and this bit remains functional even when the RFMC_CTRL[RST_MSK] bits is set. 0b - Release the 2.4GHz radio from reset 1b - Hold the 2.4GHz radio in reset
30 RF_POR	S/W Power-on-Reset for 2.4GHz Radio When set, the RF_POR bit will cause a power-on-reset of the 2.4GHz radio domain. Note, setting this bit has no effect on the RFMC's registers and this bit remains functional even when the RFMC_CTRL[RST_MSK] bits is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Release the 2.4GHz radio from power-on-reset 1b - Hold the 2.4GHz radio in power-on-reset
29 CPU_RST	S/W Reset for 2.4GHz Radio CPU When set, the CPU_RST bit will cause a reset of the 2.4GHz radio CPU. Note that this bit is set after a system reset, so SOC Core software will have to clear it to bring the 2.4GHz radio CPU out of reset. Setting this bit has no effect on the RFMC's registers and this bit remains functional even when the RFMC_CTRL[RST_MSK] bits is set. 0b - Release the 2.4GHz radio CPU from reset 1b - Hold the 2.4GHz radio CPU in reset
28 CPU_RST_LOCK	LOCK for CPU_RST When set, the CPU_RST bit will be locked and unable to be reprogrammed until the next system reset. Note that this bit is not affected by the RST bit in this register, or by the RFMC CTRL[RFMC_RST] bit. This bit remains functional even when the RFMC_CTRL[RST_MSK] bits is set. 0b - CPU_RST bit is not locked 1b - CPU_RST bit is locked
27-24 CLK_OVR	Clock Gating Override Controls whether low power mode controller clocks are gated off when the logic is not enabled (default), or whether the clocks are on regardless of whether the logic is enabled. 0xxxb - Bluetooth LE power controller clock (and 32kHz clock used by Bluetooth LE link layer) only enabled when BLE_LP_EN=1 (default) 1xxxb - Bluetooth LE power controller clock (and 32kHz clock used by Bluetooth LE link layer) always enabled x0xxb - WOR power controller clock only enabled when WOR_EN=1 (default) x1xxb - WOR power controller clock always enabled xx0xb - MAN power controller clock only enabled when MAN_EN=1 (default) xx1xb - MAN power controller clock always enabled xxx0b - TIMER clock only enabled when TIM_EN=1 xxx1b - TIMER clock always enabled
23 XO_EN	XO Enable for 2.4GHz Radio Software enable of the XO analog and output to the 2.4GHz radio domain. Note that this software enable is logically or'd with enables from the 2.4GHz MAN/WOR/Bluetooth LE power controllers. 0b - XO software enable deasserted 1b - XO software enable asserted
22	XO_EN Glitch Disable for 2.4GHz Radio

Table continues on the next page...

Table continued from the previous page...

Field	Function
XO_EN_GLITCH_DIS	If set, this disables logic added to suppress potential glitches on the xo_en signal to the analog when both MAN and WOR power controllers are enabled.
21 LP_STOP_REQ_GLITCH_DIS	LP_STOP_REQ Glitch Disable for 2.4GHz Radio If set, this disables logic added to suppress potential glitches on the lp_stop_req signal to the SPC when both MAN and WOR power controllers are enabled.
20-18 SFA_TRIG_EN	SFA Trigger Enable Selects which low power controllers can cause an SFA trigger on transition from WKUP to ACTIVE state 0xxb - Bluetooth LE Low Power Controller is not allowed to cause an SFA trigger. 1xxb - Bluetooth LE Low Power Controller is allowed to cause an SFA trigger. x0xb - WOR Low Power Controller is not allowed to cause an SFA trigger. x1xb - WOR Low Power Controller is allowed to cause an SFA trigger. xx0b - MAN Low Power Controller is not allowed to cause an SFA trigger. xx1b - MAN Low Power Controller is allowed to cause an SFA trigger.
17-12 LP_WKUP_DLY	LP Wakeup Delay Configures number of 32kHz reference clocks following enable of the XO to initiate SPC recovery for a MAN/WOR/Bluetooth LE wakeup event. Note, LP_WKUP_DLY should always be set to a value less than or equal to XO_CTRL[WKUP_OFFSET].
11-8 LP_MODE	Radio Low Power Mode Configures the radio low power mode entered after any request is asserted e.g. from S/W, MAN or WOR. The mode bits should not be changed while the radio is in or transitioning through power modes (LP_REQ_STAT and LP_ACK_STAT status bits should be clear). 0000b - Active: clock gating only (only intended for debug) 0001b - Sleep: clock gating, PMC in low power mode(only intended for debug) 0011b - Deep Sleep: low power static mode with retention of digital logic and SRAMs. 0111b - Power Down: power down of radio digital logic, optional SRAM retention. 1111b - Deep Power Down: power down of radio digital logic and SRAMs.
7 LP_ENTER	S/W Low Power Entry Request Software initiated request to enter low power mode. Note that the LP_ENTER bit is clear after a POR/LVD event, resulting in the radio defaulting to a powered up state. 0b - Deassert S/W request for low power mode entry 1b - Assert S/W request for low power mode entry
6 BLE_LP_EN	Bluetooth LE Low Power Enable Enable the Bluetooth LE power controller's wakeup output to initiate Radio low power mode entry/ wakeup.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Bluetooth LE wakeup request disabled 1b - Bluetooth LE wakeup request enabled
5 BLE_WKUP	Bluetooth LE Wakeup Force early Bluetooth LE wakeup from low power mode. 0b - Bluetooth LE low power mode wakeup deasserted 1b - Bluetooth LE low power mode wakeup asserted
4 LP_WKUP_IE	Low Power Wakeup Interrupt Enable Configures the radio to generate an interrupt when the LP_WKUP_FLAG is set. 0b - Low Power wakeup interrupt disabled 1b - Low Power wakeup interrupt enabled
3 RFACT_IE	RF_ACTIVE Interrupt Enable Configures the radio to generate an interrupt when the RFACT_FLAG is set. (It is not recommended to use RFACT_IE or RFACT_FLAG except when RFACT_SRC=2'b00.) 0b - RF_ACTIVE interrupt disabled 1b - RF_ACTIVE interrupt enabled
2 BLE_WKUP_IE	Bluetooth LE Wakeup Interrupt Enable Configures the radio to generate an interrupt when the BLE_WKUP_FLAG is set. 0b - Bluetooth LE wakeup interrupt disabled 1b - Bluetooth LE wakeup interrupt enabled
1 MAN_WKUP_IE	MAN Wakeup Interrupt Enable Configures the radio to generate an interrupt when the MAN_WKUP_FLAG is set. 0b - MAN wakeup interrupt disabled 1b - MAN wakeup interrupt enabled
0 WOR_WKUP_IE	WOR Wakeup Interrupt Enable Configures the radio to generate an interrupt when the WOR_WKUP_FLAG is set. 0b - WOR wakeup interrupt disabled 1b - WOR wakeup interrupt enabled

55.3.4.1.9 2.4GHz Radio Status Register (RF2p4GHz_STAT)

Offset

Register	Offset
RF2p4GHz_STAT	1Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									0		BLE_STATE			MAN_STATE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAN_STA...	WOR_STATE			0	BLE_WKU...	LP_ACK_...	LP_REQ_...	FRO_CLK...	RST_STAT	SLP_RDY...	LP_WKUP...	RFACT_F...	BLE_WKU...	MAN_WKU...	WOR_WKU...
W												W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22-21 —	Reserved
20-18 BLE_STATE	Bluetooth LE Low Power State Returns the current state of the RFMC's Bluetooth LE low power controller. 000b - RESET state (BLE_LP_EN=0). 001b - ACTIVE state (XO enabled, RF_ACTIVE asserted, LP request deasserted). 010b - SLEEP state (XO disabled, RF_ACTIVE deasserted, LP request asserted). 011b - WAKEUP state (XO enabled, RF_ACTIVE asserted after RFACT_WKUP_DLY, LP request deasserted after LP_WKUP_DLY).
17-15 MAN_STATE	MAN Low Power State Returns the current state of the MAN low power controller. 000b - RESET state (MAN_EN=0). 001b - ACTIVE state (XO enabled, RF_ACTIVE asserted, LP request deasserted).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - SLEEP state (XO disabled, RF_ACTIVE deasserted, LP request asserted). 011b - WAKEUP state (XO enabled, RF_ACTIVE asserted after RFACT_WKUP_DLY, LP request deasserted after LP_WKUP_DLY).
14-12 WOR_STATE	WOR Low Power State Returns the current state of the WOR low power controller. 000b - RESET state (WOR_EN=0). 001b - ACTIVE state (XO enabled, RF_ACTIVE asserted, LP request deasserted). 010b - SLEEP state (XO disabled, RF_ACTIVE deasserted, LP request asserted). 011b - WAKEUP state (XO enabled, RF_ACTIVE asserted after RFACT_WKUP_DLY, LP request deasserted after LP_WKUP_DLY).
11 —	Reserved
10 BLE_WKUP_STAT	Bluetooth LE Wakeup Status Returns the current status of the Bluetooth LE wakeup request (bt_clk_req) to the RFMC.
9 LP_ACK_STAT	Low Power Acknowledge Status Returns the current status of the Radio low power acknowledge from the SPC. Note that the LP_ACK_STAT bit will be set after a POR/LVD event
8 LP_REQ_STAT	Low Power Request Status Returns the current status of the Radio low power request to the SPC. Note that the LP_REQ_STAT bit will be set after a POR/LVD event
7 FRO_CLK_VLD_STAT	FRO Clock Valid Status Returns the current status of the FRO Clock Valid signal.
6 RST_STAT	Reset Status Returns the current status of the RF_CMC's reset output. 0b - Reset is not asserted. 1b - Reset is asserted.
5 SLP_RDY_STAT	RF_CMC Sleep Ready Status Returns the current status of the Sleep Ready signal from RF_CMC.
4	Low Power Wakeup Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
LP_WKUP_FLAG	Flag which sets after deassertion of the Radio low power acknowledge from the SPC, indicating radio power domain wakeup is complete (write 1 to clear). Note, this flag will assert for all sources of radio wakeup e.g. MAN/WOR/Bluetooth LE controller or software clear of LP_ENTER bit.
3 RFACT_FLAG	RF_ACTIVE Flag This flag is associated with the RF_ACTVE pin, but only when RFACT_SRC=2'b00. (It is not recommended to use this flag for other settings of RFACT_SRC.) Write 1 to clear.
2 BLE_WKUP_FLAG	Bluetooth LE Wakeup Flag Flag which sets when Bluetooth LE low power controller transitions to WAKEUP state during a Bluetooth LE wakeup event (write 1 to clear).
1 MAN_WKUP_FLAG	MAN Wakeup Flag Flag which sets when MAN low power controller transitions to ACTIVE state after a MAN wakeup event (write 1 to clear).
0 WOR_WKUP_FLAG	WOR Wakeup Flag Flag which sets when WOR low power controller transitions to ACTIVE state after a WOR wakeup event (write 1 to clear).

55.3.4.1.10 2.4GHz Radio Coexistence Register (RF2p4GHz_COEXT)

Offset

Register	Offset
RF2p4GHz_COEXT	20h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RFNA_IBE			0	QREQ_RF...	QREQ_SO...	QREQ_SRC	0	RFACT_WKUP_DLY						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RFACT_EN	RFACT_I...	RFACT_SRC		PORTA_P...	RFGPO_SRC			RFGPO_OBE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 RFNA_IBE	RF_NOT_ALLOWED Input Buffer Enables Enables input pin for RF_NOT_ALLOWED. 000b - RF_NOT_ALLOWED input pin disabled 001b - RF_NOT_ALLOWED input pin uses PTA16 010b - RF_NOT_ALLOWED input pin uses PTA17 011b - RF_NOT_ALLOWED input pin uses PTA22 100b - RF_NOT_ALLOWED input pin uses PTC7 101b - RF_NOT_ALLOWED input pin uses PTD6
27 —	Reserved
26 QREQ_RF_EN	QUIET_REQ Enable for Radio CPU Flash Enables QUIET_REQ to suppress Radio CPU Flash activity. 0b - QUIET_REQ is not enabled for Radio CPU Flash 1b - QUIET_REQ is enabled for Radio CPU Flash
25 QREQ_SOC_EN	QUIET_REQ Enable for SOC Core Flash Enables QUIET_REQ to suppress SOC Core Flash activity. 0b - QUIET_REQ is not enabled for SOC Core Flash 1b - QUIET_REQ is enabled for SOC Core Flash
24 QREQ_SRC	QUIET_REQ Source Selects the QUIET_REQ source from the RFMC or TSM/LL outputs. 0b - QUIET_REQ is driven by the RFMC 1b - QUIET_REQ is driven by the TSM/LL
23-22 —	Reserved
21-16 RFACT_WKUP_DLY	RF_ACTIVE Wakeup Delay If RFACT_SRC=2'b00, this configures number of 32kHz reference clocks following enable of the XO to assert the RF_ACTIVE pin for a MAN/WOR/Bluetooth LE wakeup event. Note, RFACT_WKUP_DLY should always be set to a value less than or equal to XO_CTRL[WKUP_OFFSET].
15 RFACT_EN	S/W Enable of RF_ACTIVE pin If RFACT_SRC=2'b00 and RFACT_IDIS=0, this can be used by software to assert the RF_ACTIVE pin.

Table continues on the next page...

Table continued from the previous page...

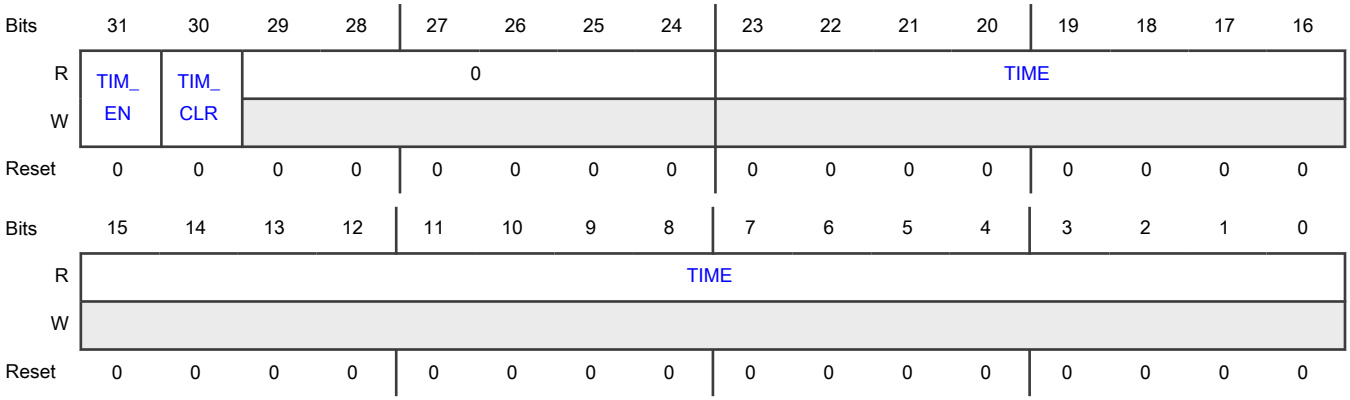
Field	Function
	0b - Take no action 1b - Assert RF_ACTIVE pin
14 RFACT_IDIS	RF_ACTIVE Idle Disable if RFACT_SRC=2'b00, this configures the RF_ACTIVE pin to deassert when TSM is idle. 0b - RF_ACTIVE does not deassert when TSM is idle (will deassert on next low power mode entry) 1b - RF_ACTIVE will deassert when TSM is idle
13-12 RFACT_SRC	RF_ACTIVE Source Selects the RF_ACTIVE source from the RFMC, TSM/LL or Bluetooth LE wakeup request. 00b - RF_ACTIVE is driven by the RFMC 01b - RF_ACTIVE is driven by the TSM/LL 10b - RF_ACTIVE is driven by the Bluetooth LE wakeup request (bt_clk_req) 11b - Reserved
11 PORTA_PWR	PORTA Power Some of the coexistence or FEM control pins are mapped to PORTA pin in the SOC. This bit controls whether PORTA pins (in the CORE_WAKE domain) remain powered when the MCU is in Deep Power Down 0b - PORTA pins do not remain powered (default behavior) 1b - PORTA pins remain powered
10-8 RFGPO_SRC	RF_GPO Source Selects the Source for RF_GPO[7:0]. In description below, coext[3:0] = {rf_priority[1:0], rf_status, rf_active}, and fem_ctrl[3:0] = {ant_b, ant_a, rx_switch, tx_switch} 000b - RF_GPO[7:0] = {coext[3:0], fem_ctrl[3:0]} 001b - RF_GPO[7:0] = {fem_ctrl[3:0], coext[3:0]} 010b - RF_GPO[7:0] = {lant_lut_gpio[3:0], fem_ctrl[3:0]} 011b - RF_GPO[7:0] = {fem_ctrl[3:0], lant_lut_gpio[3:0]} 100b - RF_GPO[7:0] = {lant_lut_gpio[3:0], coext[3:0]} 101b - RF_GPO[7:0] = {coext[3:0], lant_lut_gpio[3:0]}
7-0 RFGPO_OBE	RF_GPO Output Buffer Enable Enables output pins for RF_GPO[7:0] (see chip configuration chapter for pin mapping). 0 - output pin disabled 1 - output pin enabled

55.3.4.1.11 2.4GHz TIMER Register (RF2p4GHz_TIMER)

Offset

Register	Offset
RF2p4GHz_TIMER	24h

Diagram



Fields

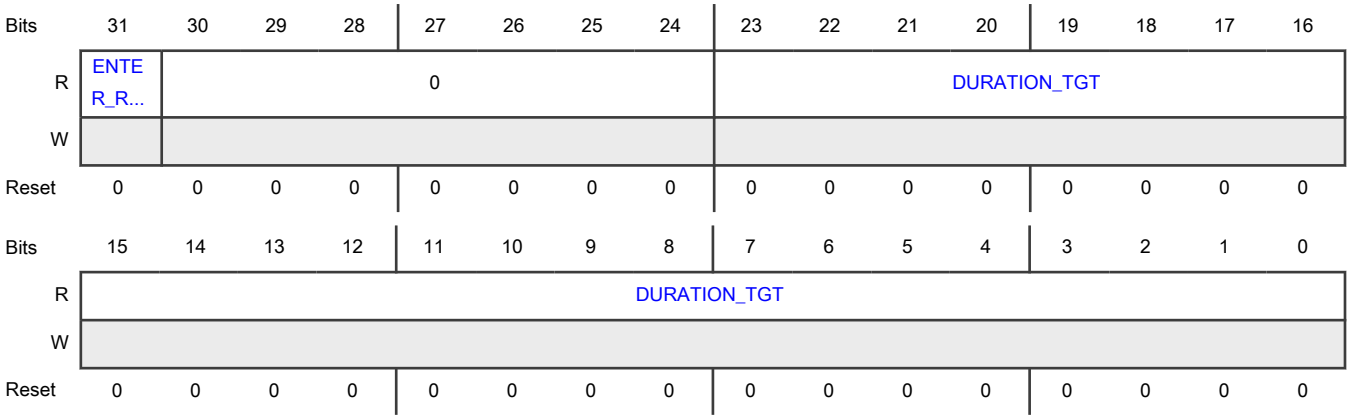
Field	Function
31 TIM_EN	Timer Enable Enables 32kHz reference clock timer (used by the MAN low power controller) to begin incrementing. 0b - Timer disabled 1b - Timer enabled
30 TIM_CLR	Timer Clear Causes 32kHz reference clock timer (used by the MAN low power controller) to clear to 0. Note after writing a 1 to TIM_CLR, user should poll TIME count to ensure it has cleared before writing 0 to TIM_CLR 0b - Timer not cleared 1b - Timer cleared
29-24 —	Reserved
23-0 TIME	Timer Count Current 32kHz reference clock time (used by the MAN low power controller).

55.3.4.1.12 2.4GHz WOR Register 1 (RF2p4GHz_WOR1)

Offset

Register	Offset
RF2p4GHz_WOR1	28h

Diagram



Fields

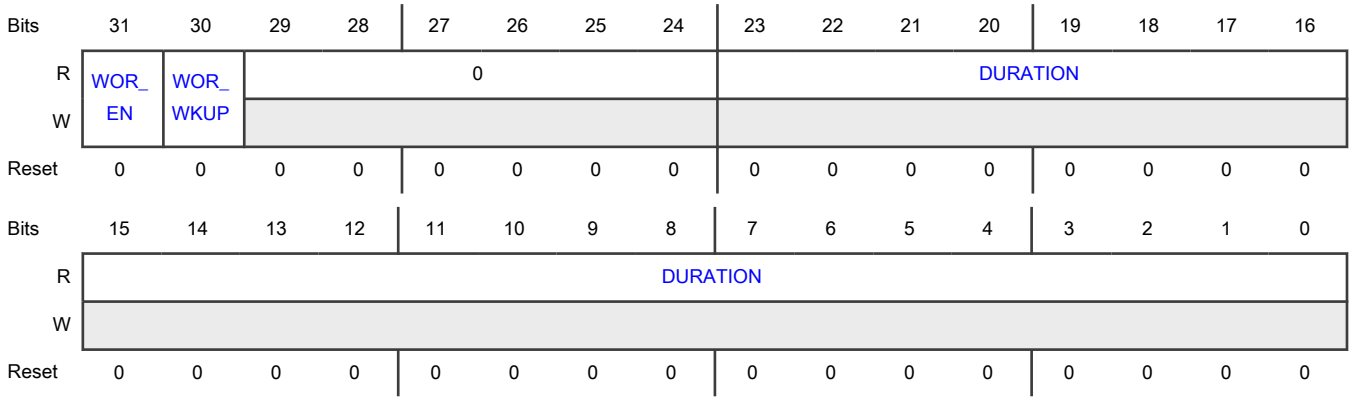
Field	Function
31 ENTER_REQ	WOR Low Power Entry Request WOR request asserted for low power mode entry. 0b - WOR low power mode request deasserted 1b - WOR low power mode request asserted
30-24 —	Reserved
23-0 DURATION_TGT	WOR Low Power Duration Target Target duration for radio to remain in low power mode. Note, value is only valid when ENTER_REQ is asserted.

55.3.4.1.13 2.4GHz WOR Register 2 (RF2p4GHz_WOR2)

Offset

Register	Offset
RF2p4GHz_WOR2	2Ch

Diagram



Fields

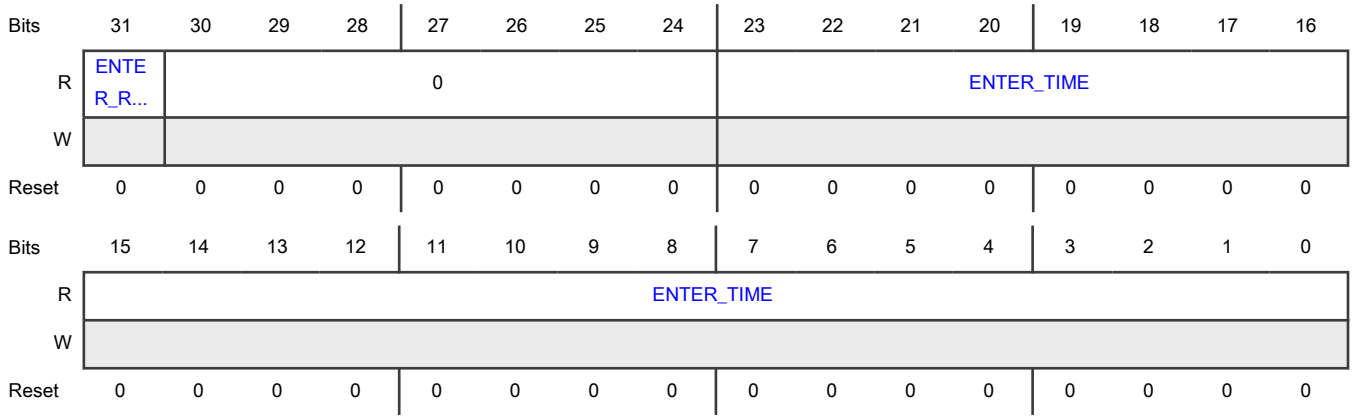
Field	Function
31 WOR_EN	WOR Enable Enable WOR to initiate low power mode entry/wakeup. 0b - WOR low power mode entry/wakeup disabled 1b - WOR low power mode entry/wakeup enabled
30 WOR_WKUP	WOR Wakeup Force early WOR wakeup from low power mode. 0b - WOR low power mode wakeup deasserted 1b - WOR low power mode wakeup asserted
29-24 —	Reserved
23-0 DURATION	WOR Low Power Duration Duration radio remained in low power mode due to WOR low power mode request. Note, value is only valid when ENTER_REQ is deasserted.

55.3.4.1.14 2.4GHz MAN Register 1 (RF2p4GHz_MAN1)

Offset

Register	Offset
RF2p4GHz_MAN1	30h

Diagram



Fields

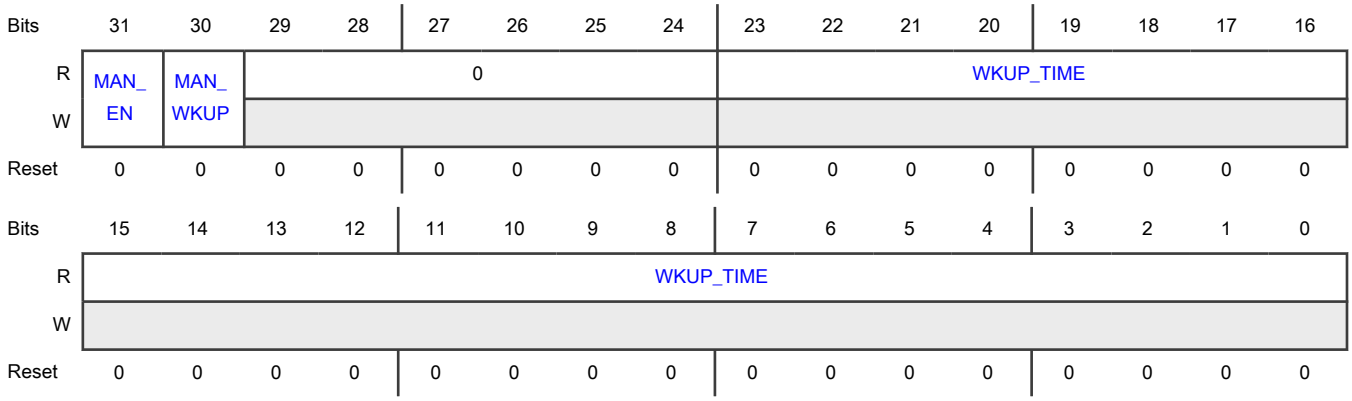
Field	Function
31 ENTER_REQ	MAN Low Power Entry Request MAN request asserted for low power mode entry. 0b - MAN low power mode request deasserted 1b - MAN low power mode request asserted
30-24 —	Reserved
23-0 ENTER_TIME	MAN Low Power Entry Time Stamp Reflects the instantaneous value of the Time stamp at which the MAN should initiate low power entry.

55.3.4.1.15 2.4GHz MAN Register 2 (RF2p4GHz_MAN2)

Offset

Register	Offset
RF2p4GHz_MAN2	34h

Diagram



Fields

Field	Function
31 MAN_EN	MAN Enable Enable MAN to initiate low power mode entry/wakeup. 0b - MAN low power mode entry/wakeup disabled 1b - MAN low power mode entry/wakeup enabled
30 MAN_WKUP	MAN Wakeup Force early MAN wakeup from low power mode. 0b - MAN low power mode wakeup deasserted 1b - MAN low power mode wakeup asserted
29-24 —	Reserved
23-0 WKUP_TIME	MAN Low Power Wakeup Time Stamp Reflects the instantaneous value of the Time stamp at which the MAN should complete low power wakeup.

55.3.4.1.16 2.4GHz MAN Register 3 (RF2p4GHz_MAN3)

Offset

Register	Offset
RF2p4GHz_MAN3	38h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								ENTER_TIME_CAPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENTER_TIME_CAPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 ENTER_TIME_CAPT	MAN Low Power Entry Time Captured Captured Time stamp for MAN entry to low power. The value of ENTER_TIME is captured when the man_sleep_req asserts.

55.3.4.1.17 2.4GHz MAN Register 4 (RF2p4GHz_MAN4)

Offset

Register	Offset
RF2p4GHz_MAN4	3Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								WKUP_TIME_CAPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WKUP_TIME_CAPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 WKUP_TIME_CAPT	MAN Low Power Wakeup Time Captured Captured Time stamp for MAN exit from low power. The value of WKUP_TIME is captured when the man_sleep_req asserts. However, if MAN_WKUP bit is set, the value is updated when that occurs to reflect the early exit time.

55.3.5 Functional description

55.3.5.1 XO Control

The XO Control sub-module is responsible for enabling the XO analog oscillator in response to a request from either the 2.4GHz radio, System Clock Generator (SCG) or an external request (XTAL_OUT_EN pin asserted). The clock output from the XO analog oscillator can be independantly driven to 3 different clocking domains under RFMC control: 2.4GHz radio digital, SCG and XTAL_OUT pin. (Additional XO clock output(s) to the 2.4GHz radio analog blocks are not shown in the figures; this is enabled by TSM.) The figure below illustrates the logic implemented by the XO control sub-module to drive each of these domains.

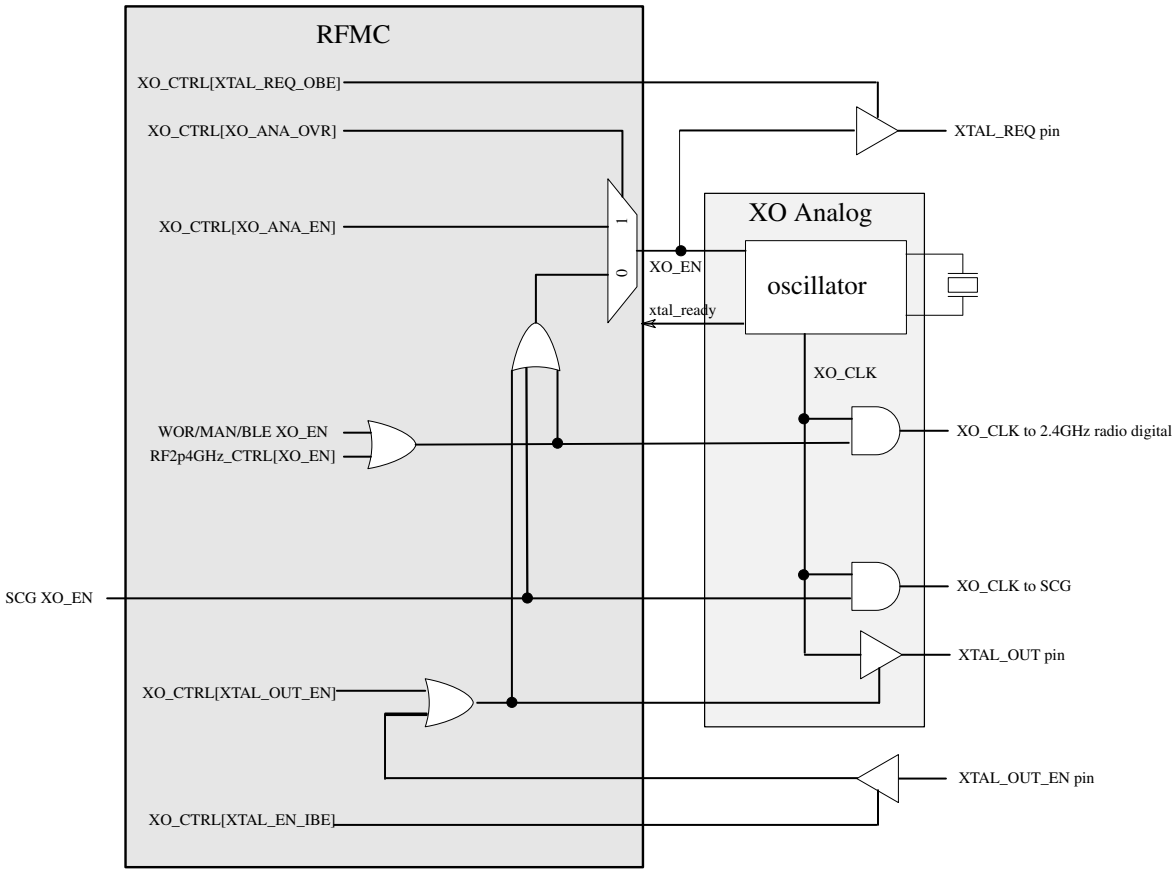


Figure 211. XO Control logic

The XO_CTRL[WKUP_OFFSET] can be used to pre-emptively enable the XO during a MAN/WOR wakeup sequence. This ensures the XO is ready and stable before the link layer begins operation at the designated MAN/WOR wakeup time.

Three flags with optional interrupts can be generated by the XO Control sub-module:

- RDY_FLAG: sets after the XO output is ready.
- INT_FLAG: sets when an internal source (SCG, 2.4GHz radio) has requested the XO.
- EXT_FLAG: sets when an external source (XTAL_OUT_EN pin) has requested the XO.

55.3.5.2 Low Power Mode Controllers

A major function of the RFMC is to control the power mode of the 2.4GHz radio . The RFMC itself resides in the system power domain and remains fully functional in all power modes except for Deep Power Down.

The 2.4GHz sub-module within the RFMC contains the following power controllers, any of which can be used to control low power entry/exit for the 2.4GHz radio:

- The MAN power controller can be used in conjunction with the Generic Link Layer or the 802.15.4 Link Layer. This controller is used to control the low power mode entry/exit time by programming the entry and wakeup time in the RFMC.
- The WOR power controller can be used in conjunction with the Generic Link Layer or the 802.15.4 Link Layer. This controller enters low power mode when directed -- and for the duration specified -- by the radio's WOR scheduler. The WOR scheduler is capable of scheduling multiple low power entry/exit and radio events.
- The Bluetooth LE power controller can only be used in conjunction with the Bluetooth LE Link Layer in the 2.4GHz radio. This power controller enters and exits low power as directed by the Bluetooth LE link layer in the radio. (Note that the Bluetooth LE link layer controls a small amount of logic residing in the system power domain and which uses the 32kHz clock to keep track of time during low power mode.)

In addition to the power controllers, software can also set/clear the RF*_CTRL[LP_ENTER] bit to enter/exit low power mode.

When software or the power controller directs the radio to enter low power, the low power mode will be as selected as programmed by the RF*_CTRL[LP_MODE] bit field, which selects between the following:

- Deep Sleep: low power static mode with clocks gated, but all radio digital logic and SRAM state is retained.
- Power Down: full power down of all radio digital logic and SRAM.

Following a POR/LVD event, RF*_CTRL[LP_ENTER] is set and RF*_CTRL[LP_MODE] is configured such that both radios will initially be in Power Down mode until the RF*_CTRL[LP_ENTER] field is cleared.

The MAN, WOR and Bluetooth LE power controllers in the RFMC each implement its own state machine to sequence the radio in and out of low power modes. Each state machine implements the following states, which are also observable by reading the RF*_STAT[MAN/WOR/BLE_STATE] register fields:

- RESET: controller reset, waiting for controller to be enabled. Outputs set to XO_EN=0, RF_ACTIVE=0, LP_REQ=0.
- ACTIVE: radio in active mode. Outputs set to XO_EN=1, RF_ACTIVE=1, LP_REQ=0.
- SLEEP: radio in low power mode. Outputs set to XO_EN=0, RF_ACTIVE=0, LP_REQ=1.
- WKUP: radio begins transition to active mode. Outputs set to XO_EN=1, RF_ACTIVE=1 (after RF*_COEXT[RFACT_WKUP_DLY]), LP_REQ=0 (after RF*_CTRL[LP_WKUP_DLY]).

The following diagram shows the transitions between states for the power controllers.

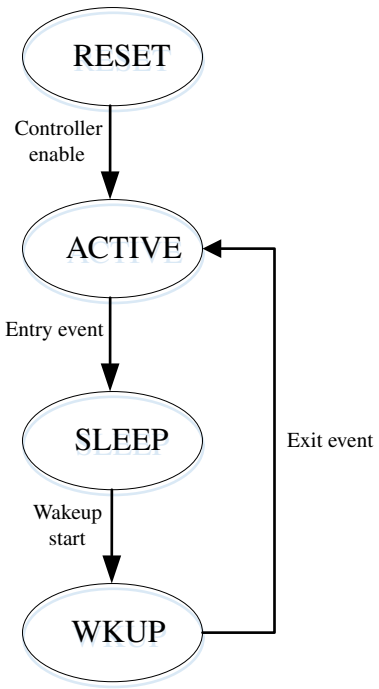


Figure 212. MAN/WOR/Bluetooth LE power control state diagram

Note that the transition from ACTIVE to SLEEP also depends on a "sleep ready" signal from the RF_CMC module, which ensures that other logic in the radio (especially the flash) is ready for the RFMC to put the radio is low power mode. This RF_CMC signal is reflected in the RF*_STAT[SLP_RDY_STAT] bit.

On transition from WKUP state to ACTIVE state, each power controller selected via RF*_CTRL[SFA_TRIG_EN] asserts a trigger to the SFA in the radio for one 32kHz clock cycle which will (if the SFA is properly configured) start a calibration cycle for the FRO32K clock.

To ensure correct sequencing of the radio power modes, it is recommended that only one of the power controllers be enabled at any given time. However, the RFMC does allow the MAN and WOR power controllers to be concurrently enabled in which case the radio will only enter low power when MAN and WOR power controllers both request low power.

The MAN/WOR/Bluetooth LE power controllers operate using the 32kHz OSC clock, and this clock must be available before enabling any of the power controllers for low power entry/exit.

55.3.5.2.1 MAN Power Controller

To enable the MAN power controller, the RF*_MAN2[MAN_EN] bit must be set. The RF*_TIMER[TIM_EN] must also be set to enable the 32kHz timer used by the MAN power control. For the 2.4GHz radio, the MAN must be assigned to a specific link layer in the 2.4GHz radio. In the WOR, the ENTER_TIME and WKUP_TIME for MAN should be programmed with the desired low power entry and exit time, respectively.

The low power entry/exit sequence is illustrated in the figure below. After the link layer has requested deep sleep entry (man_sleep_req in figure below, and reflected in the RF*_MAN1[ENTER_REQ] status bit) a MAN low power entry event is triggered when the RF*_TIMER[TIME] counter matches ENTER_TIME. Low power exit is scheduled to complete when the RF*_TIMER[TIME] counter matches WKUP_TIME. Prior to the WKUP_TIME, the MAN controller will enable the XO (xo_en) at XO_CTRL[WKUP_OFFSET] 32kHz cycles before WKUP_TIME, and request the SPC to wake the radio domain from low power mode (lp_stop_req de-assertion) at RF*_CTRL[LP_WKUP_DLY] 32kHz cycles after XO is enabled. It is expected that the WKUP_OFFSET is programmed to allow sufficient time for the XO to warmup and assert the xtal_ready signal. Likewise, the LP_WKUP_DLY is programmed to allow enough time for the SPC to bring the radio out of low power mode before

WKUP_TIME. RF_ACTIVE can also be programmed to assert at RF*_COEXT[RFACT_WKUP_DLY] after XO is enabled. Flags RF*_STAT[LP_WKUP_FLAG] and RF*_STAT[MAN_WKUP_FLAG] (with optional interrupts) are set after the lp_stop_ack (from SPC) de-asserts and at WKUP_TIME, respectively. Before the MAN power controller will enter ACTIVE state, TIME must reach WKUP_TIME and man_sleep_req must be de-asserted. If properly configured in RF*_CTRL[SFA_TRIG_EN], a trigger (sfa_meas) will be issued to the radio SFA after the MAN power controller reaches the ACTIVE state. The MAN power controller's state is reflected in RF*_STAT[MAN_STATE] for use as needed by software.

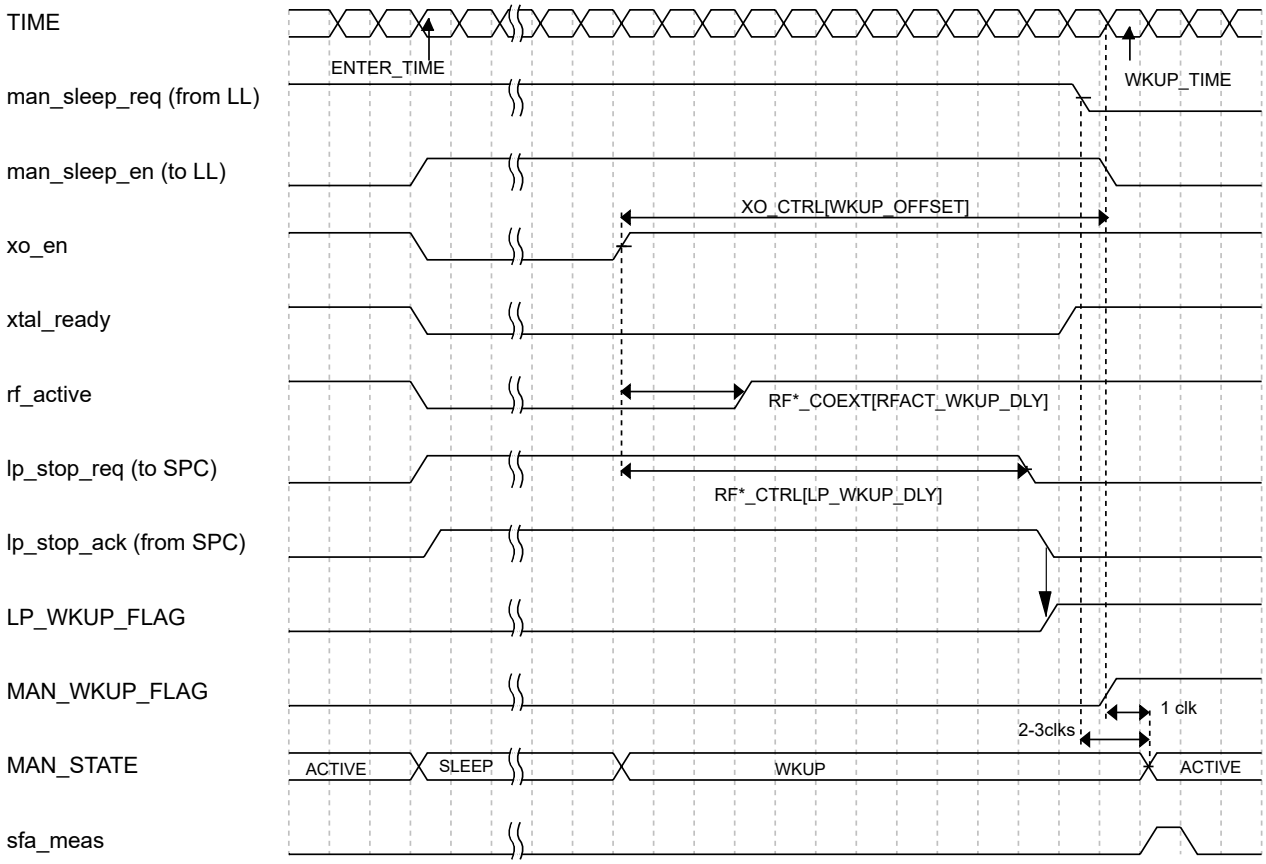


Figure 213. Low Power entry/exit for MAN Power Controller

If for some reason the radio needs to be woken before WKUP_TIME has been reached, the RF*_MAN2[MAN_WKUP] bit can be set by software to wake the radio prematurely.

55.3.5.2.2 WOR Power Controller

To enable the WOR power controller, the RF*_WOR2[WOR_EN] bit must be set. For the 2.4GHz radio, the WOR must be assigned to a specific link layer by programming the WOR_CTRL[WOR_PROTOCOL] in the 2.4GHz radio. The WOR low power entry is controlled by wor_sleep_req (reflected in RF*_WOR1[ENTER_REQ] bit) from the "WOR scheduler" (the WOR logic in the radio domain), and the desired low power duration (wor_sleep_duration -- reflected in RF*_WOR1[DURATION_TGT]) is also an input to the RFMC from the WOR scheduler.

The low power entry/exit sequence is illustrated in the figure below. Prior to low power entry, the WOR scheduler pulses the wor_sleep_duration_clr to clear the DURATION counter (RF*_WOR2[DURATION]) in the WOR power controller, then asserts wor_sleep_req. The WOR power controller needs to synchronize this request to the 32kHz clock domain, so after 2-3 cycles, low power mode is entered. Low power exit is scheduled to complete when the DURATION counter matches wor_sleep_duration. Prior to this time, the WOR controller will enable the XO (xo_en) at XO_CTRL[WKUP_OFFSET] 32kHz cycles

before wor_sleep_duration, and request the SPC to wake the radio domain from low power mode (lp_stop_req de-assertion) at RF*_CTRL[LP_WKUP_DLY] 32kHz cycles after XO is enabled. It is expected that the WKUP_OFFSET is programmed to allow sufficient time for the XO to warmup and assert the xtal_ready signal. Likewise, the LP_WKUP_DLY is programmed to allow enough time for the SPC to bring the radio out of low power mode before WKUP_TIME. RF_ACTIVE can also be programmed to assert at RF*_COEXT[RFACT_WKUP_DLY] after XO is enabled. Flags RF*_STAT[LP_WKUP_FLAG] and RF*_STAT[WOR_WKUP_FLAG] (with optional interrupts) are set after the lp_stop_ack (from SPC) de-asserts and at wor_sleep_duration, respectively. The WOR power controller does not move to ACTIVE state until after the WOR scheduler de-asserts the wor_sleep_req signal. If properly configured in RF*_CTRL[SFA_TRIG_EN], a trigger (sfa_meas) will be issued to the radio SFA after the WOR power controller reaches the ACTIVE state. The WOR power controller's state is reflected in RF*_STAT[WOR_STATE] for use as needed by software.

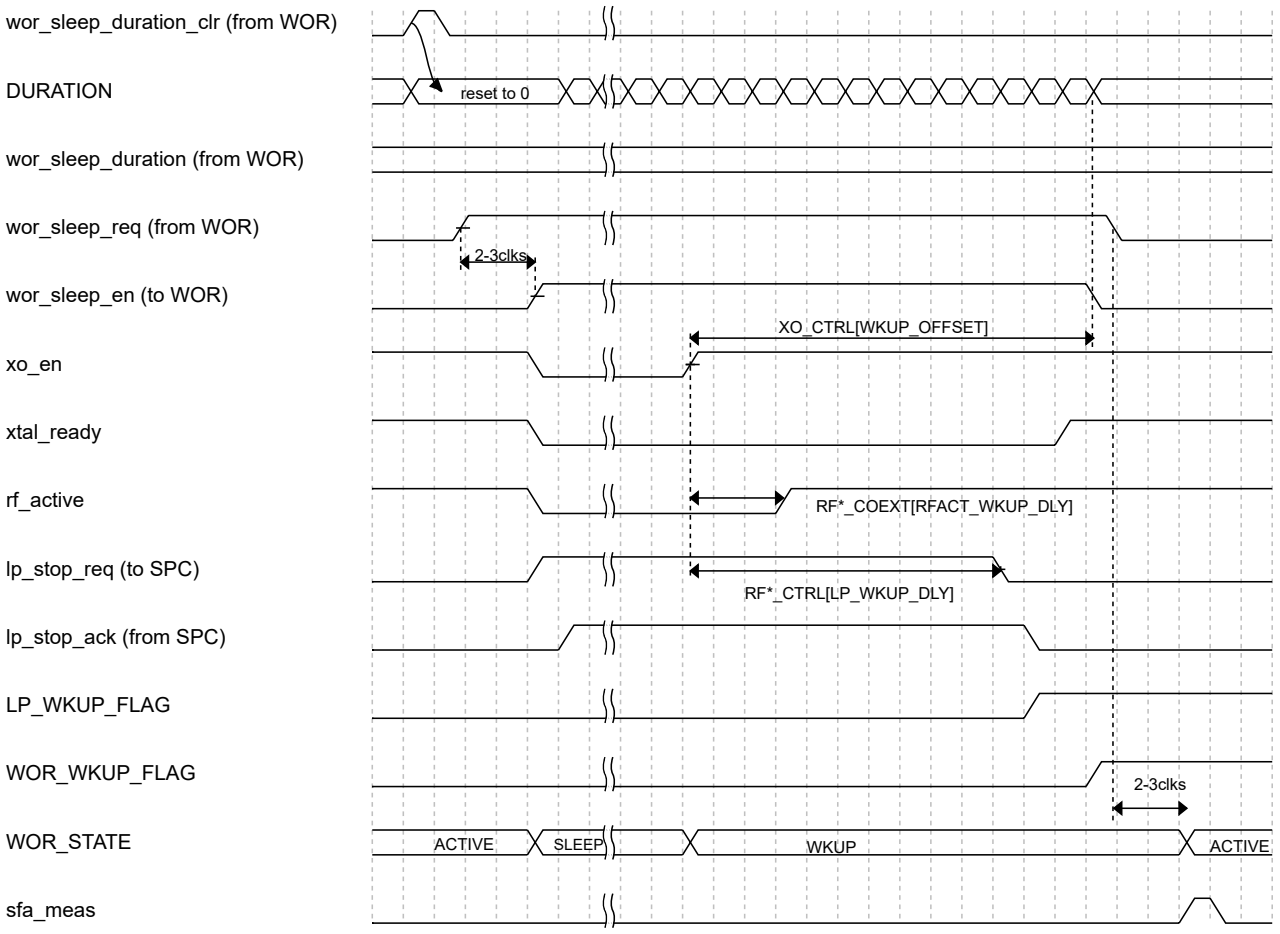


Figure 214. Low Power entry/exit for WOR Power Controller

If for some reason the radio needs to be woken before wor_sleep_duration has been reached, the RF*_WOR2[WOR_WKUP] bit can be set by software to wake the radio prematurely.

55.3.5.2.3 Bluetooth LE Power Controller

To enable the Bluetooth LE power controller, the RF2p4GHz_CTRL[BLE_LP_EN] bit must be set. A Bluetooth LE low power entry and exit event is then controlled by the Bluetooth LE link layer (see Bluetooth LE link layer programming model).

NOTE

The 32kHz clock used by both the Bluetooth LE power controller and the Bluetooth LE link layer is enabled by setting RF2p4GHz_CTRL[BLE_LP_EN]. If for some reason the Bluetooth LE link layer needs this clock to be active while the Bluetooth LE power controller is not enabled (BLE_LP_EN bit not set), software can program RF2p4GHz_CTRL[CLK_OVR] to force the clock to be enabled.

The low power entry/exit sequence is illustrated in the figure below. When the Bluetooth LE link layer wants to enter low power, it de-asserts the bt_clk_req input to RFMC (this signal is reflected in RF*_STAT[BLE_WKUP_STAT]). When the Bluetooth LE link layer wants to exit low power, it asserts bt_clk_req. At that time, the Bluetooth LE power controller will enable the XO (xo_en), and after RF*_CTRL[LP_WKUP_DLY] 32kHz cycles it will request the SPC to wake the radio domain from low power mode (lp_stop_req de-assertion). It is expected that the LP_WKUP_DLY is programmed such that the SPC will complete bringing the radio out of low power mode near the time when the XO becomes ready (xtal_ready). RF_ACTIVE can also be programmed to assert at RF*_COEXT[RFACT_WKUP_DLY] after XO is enabled. Flags RF*_STAT[LP_WKUP_FLAG] and RF*_STAT[BLE_WKUP_FLAG] (with optional interrupts) are set after the lp_stop_ack (from SPC) de-asserts and at xo_en, respectively. The Bluetooth LE power controller moves to ACTIVE state in the cycle after lp_stop_req is de-assrted. If properly configured in RF*_CTRL[SFA_TRIG_EN], a trigger (sfa_meas) will be issued to the radio SFA after the Bluetooth LE power controller reaches the ACTIVE state. The Bluetooth LE power controller's state is reflected in RF*_STAT[BLE_STATE] for use as needed by software.

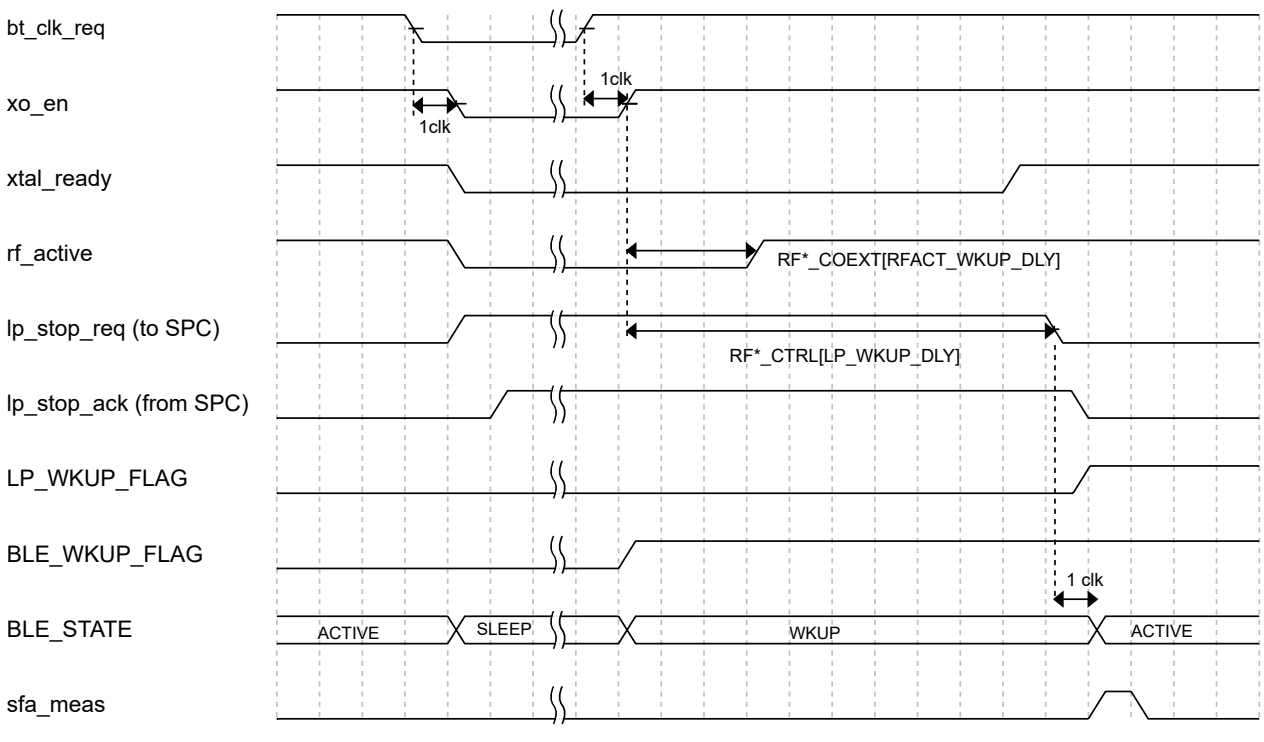


Figure 215. Low Power entr/exit for Bluetooth LE Power Controller

If for some reason the radio needs to be woken before the Bluetooth LE link layer is scheduled to assert bt_clk_req, the RF2p4GHz_CTRL[BLE_WKUP] bit can be set by software to wakeup the radio prematurely.

The Bluetooth LE power controller behavior is a bit different than the MAN and WOR power controllers. In particular

- The XO_CTRL[WKUP_OFFSET] bitfield is not used by the Bluetooth LE power controller. Instead the Bluetooth LE link layer has its own programmable bitfield "Osc_startup_delay" which is used to delay when the Bluetooth LE link layer switches timing control from the low power clock to the XO

- The Bluetooth LE power controller does not have a counter to keep track of the low power duration; that is handled in the Bluetooth LE link layer
- The BLE_WKUP_FLAG asserts at WKUP state, whereas the MAN_WKUP_FLAG and WOR_WKUP_FLAG assert at the end of the requested low power duration

55.3.5.3 Radio Coexistence

The 2.4GHz radio implements external IO signals to support coexistence with an external RF device operating within the same frequency band. The RFMC provides mux control (via RFMC's RF_GPO[7:0] output) for the coexistence outputs {RF_ACTIVE, RF_STATUS, and RF_PRIORITY[1:0]} from the 2.4GHz radio, provides additional control for the RF_ACTIVE signal, and provides control for the RF_NOT_ALLOWED coexistence input. The next few paragraphs discuss RFMC control of the RF_NOT_ALLOWED and RF_ACTIVE signals. For more information on coexistence refer to the Coexistence Interface and Coexistence/FEM/LANT connections sections of the Radio Control chapter.

For RF_NOT_ALLOWED, the RFMC provides programmable control (RF*_COEXT[RFNA_IE] bitfield) so that this coexistence signal can be sourced from any of 5 device pins, or disabled.

For RF_ACTIVE, the RFMC provides programmable control (RF*_COEXT[RFACT_SRC] bitfield) to select whether this coexistence output is sourced by the RFMC, the Bluetooth LE bt_clk_req signal, or a mux of the TSM's RF_ACTIVE signal and the active link layer's REQUEST signal. If RF_ACTIVE is sourced by the RFMC (RF*_COEXT[RFACT_SRC]=00), this signal relies on the RFMC’s low power controllers (one of the RFMC’s low power controllers should be enabled); in this configuration the RF_ACTIVE behavior is as follows:

- If RF*_COEXT[RFACT_IDIS]=0
  - Asserts during the RFMC low power controller's wakeup sequence, RF*_COEXT[RFACT_WKUP_DLY] 32kHz clock cycles after the XO is enabled
  - De-asserts when the RFMC low power controller requests that the radio enters a low power mode
  - Software can cause RF_ACTIVE to be asserted during a low power mode by setting the RF*_COEXT[RFACT_EN] bitfield
- If RF*_COEXT[RFACT_IDIS]=1
  - Asserts when the TSM is busy
  - De-asserts when the TSM is idle

Also related to RF_ACTIVE are the QUIET_REQ outputs. These also indicate when the radios are active, but are used inside the SOC to suppress SOC Core flash and/or RF Core flash activity if needed. These outputs have configurable behavior using the RF*_COEXT register's QREQ_SRC, QREQ_SOC_EN and QREQ_RF_EN bits.

55.3.5.4 Radio FEM Control

The RFMC provides mux control (via RFMC's RF_GPO[7:0] output) for the FEM (front-end module) signals {ANT_A, ANT_B, RX_SWITCH, and TX_SWITCH} from the 2.4GHz radio, but does not otherwise make use of, or affect these signals. For more information on these signals, refer to the Transceiver Localization and Antenna Control section of the Transceiver chapter and Coexistence/FEM/LANT connections sections of the Radio Control chapter.

55.3.5.5 Radio Localization Antenna Select

The RFMC provides mux control (via RFMC's RF_GPO[7:0] output) for the LANT_LUT_GPIO[3:0] (localization antenna select) signals from the 2.4GHz radio, but does not otherwise make use of, or affect these signals. For more information on these signals, refer to the Transceiver Localization and Antenna Control section of the Transceiver chapter and Coexistence/FEM/LANT connections sections of the Radio Control chapter.

55.3.5.6 Reset

The RFMC and 2.4GHz radio are reset on any system reset event or by setting the RFMC_CTRL[RFMC_RST] bit, with the exception of the following registers bits which are only reset on a POR/LVD event: RFMC_CTRL[RFMC_RST] (resets to 0) and RFMC_CTRL[RST_MSK] (resets to 0).

The RFMC can also generate software resets for the 2.4GHz radio via the RF*_CTRL[RF_POR], RF*_CTRL[RST] and RF*_CTRL[CPU_RST] bits. Setting RF*_CTRL[RF_POR], RF*_CTRL[RST], or RF*_CTRL[CPU_RST] will have no effect on the RFMC itself however. The reset associated with RF*_CTRL[CPU_RST], ipg_async_cpu_reset_lv, is used primarily to hold the CM3 core in reset for flash loading. The reset associated with RF*_CTRL[RF_POR], ipg_async_por_2p4GHz_b_lv, provides POR for the LV domain logic. On exit from Power Down or Deep Power Down low power modes, a signal from the SPC (spc_pd_pwrdwn_exit) asserts all of the LV resets.

There is also a RF*_CTRL[CPU_RST_LOCK] bit associated with the RF*_CTRL[CPU_RST]. If the CPU_RST_LOCK bit is set, the CPU_RST bit cannot be cleared until the next system reset (ipg_async_reset_1p8v). The lock bit is used by boot ROM code to force the radio CPU to remain in reset if the radio flash verification fails.

The radio also needs to be held off until the flash has finished its initialization (its IFR outputs are valid, and it is ready to accept read requests) after a reset. The radio flash output signal msf_core_hold is used by the RF_CMC module for this purpose: the RF_CMC delays de-assertion of the non-POR resets used by the radio.

For testing purposes, the RFMC and radio power domains can be isolated from system reset events by setting the RFMC_CTRL[RST_MSK] register bit. Note that the RFMC_CTRL[RFMC_RST] and RF*_CTRL[RST] register bits are still functional when the RFMC_CTRL[RST_MSK] bit is set.

The figure below illustrates reset bits and signals for the RFMC, RF_CMC and radio. The ipg_async_reset_rfmc_1p8v is an output from the RFMC to use to reset any other 1.8v logic in the radio power domain.

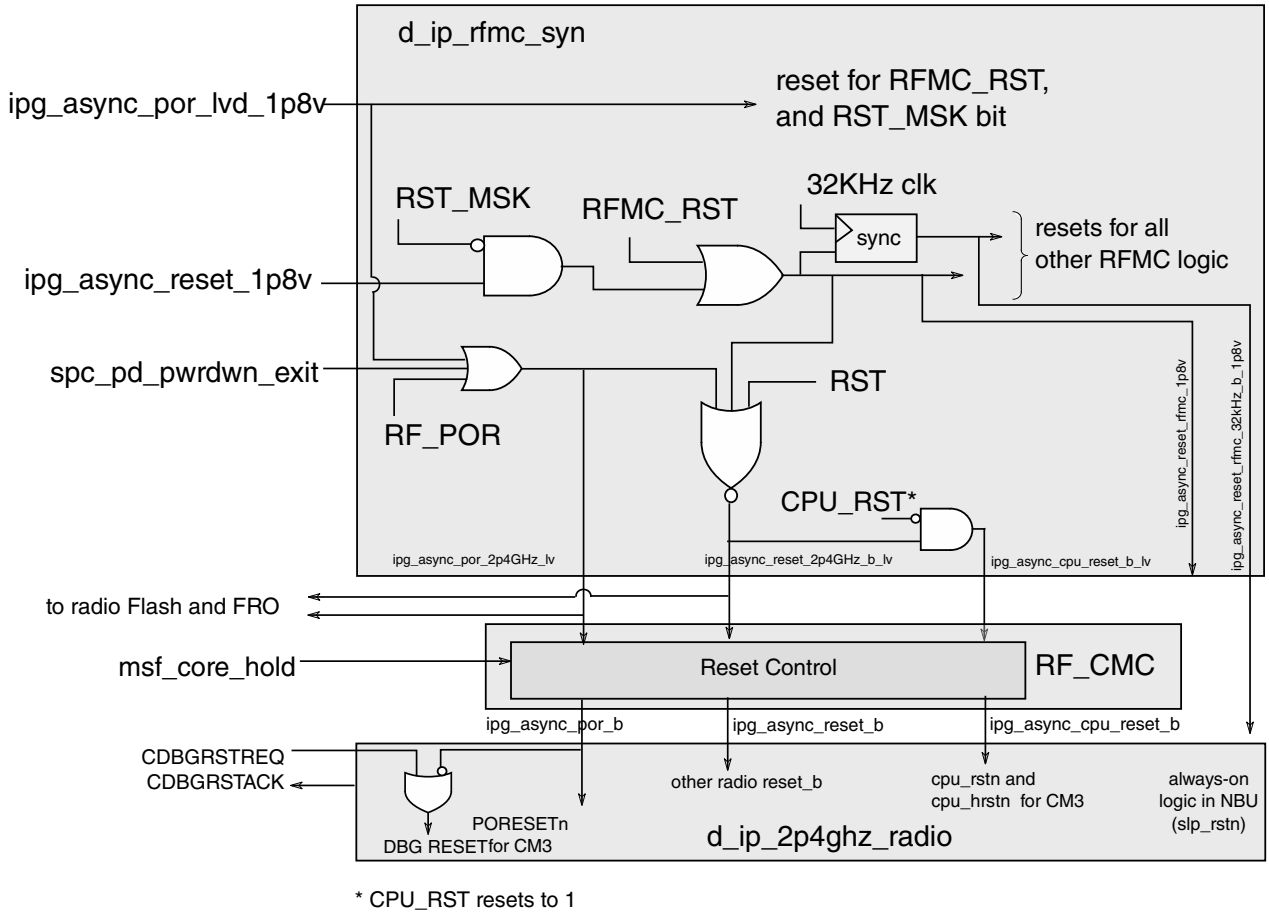


Figure 216. Reset

55.3.5.7 Interrupt and wake-up request

The RFMC generates an output for the SOC's interrupt controller (NVIC) and a wake-up request for the SOC's wake-up unit (WUU). As shown in the figure below, all enabled interrupt sources within the RFMC -- from the XO control and the 2.4GHz low power controllers -- are combined to generate the rfmc interrupt. The wake-up request for the WUU combines enabled interrupt requests from the Bluetooth LE, WOR and MAN low power controllers within the RFMC with interrupt requests from the 2.4GHz radio .

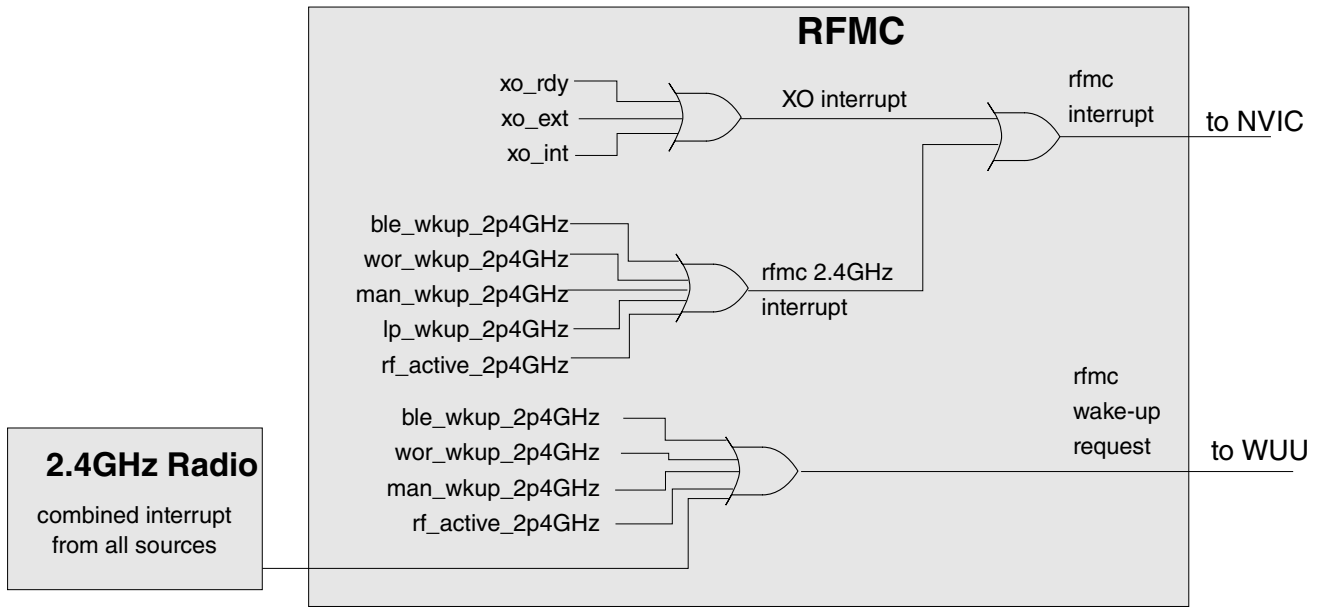


Figure 217. Interrupt and wake-up request

55.3.5.8 Power-on Sequence

After a POR/LVD event, the RFMC is released from reset, the 2.4GHz radio is powered up and released from reset, but the the XO is disabled and the NBU CPU is held in reset . The following steps are needed by software. (Note that steps 1-3 may optionally be supported by the SOC boot ROM.)

1. Configure the XO. Program the EXT_MODE, XTAL_REQ_OBE and other bits in the XO_CTRL register as needed
2. Set RF*_CTRL[XO_EN] to enable the XO, or enable of the RFMC's low power controllers to allow it to enable the XO.
3. Clear the RF*_CTRL[CPU_RST] bit to release the NBU CPU from reset.
4. Wait for XO_STAT[RDY_FLAG] to be set. An RFMC interrupt is also associated with this event. Software can access the radio modules which use the XO clock after the XO is ready. (Software can access the radio modules which use the radio FRO clock before the XO is ready.)

55.4 2.4GHz Radio

55.4.3 Bus Interfaces and Memory Map

The radio supports the following bus interfaces to allow software access to radio peirpherals and memories:

1. CM3 interface to the CM3 memories, the memory-mapped peripherals inside the NBU, access to the radio resources (Generic LL, XCVR, etc.) outside of the NBU via the radio's PBRIDGE, access to resources (radio flash, TPM2, RF_CMC, radio FRO192M) related to -- but outside of -- the radio via NBU PBRIDGE, and access to CM33 peripherals (e.g., DSB, EdgeLock Secure Enclave, SEMA42, and RFMC) via AXBS PBRIDGE0. (For a detailed list of peripherals accessible via PBRIDGE0, refer to the Peripheral Bridge section of the Core overview chapter of this Reference Manual.)
2. CM33 access some NBU resources, and NBU PBRIDGE resources. Note that this path only allows access to certain regions of the NBU/CM3 memory space, and (unlike #1 above) it is not allowed acess to the radio PBRIDGE or the AXBS's PBRIDGE0.
3. CM33 access to the radio PBRIDGE to access radio resources (Generic LL, XCVR, etc.) outside of the NBU
4. CM33 and CM3 access to RFMC registers. The CM3 access is via AXBS PBRIDGE0 (#1 above).

These buses are highlighted in the figure below.

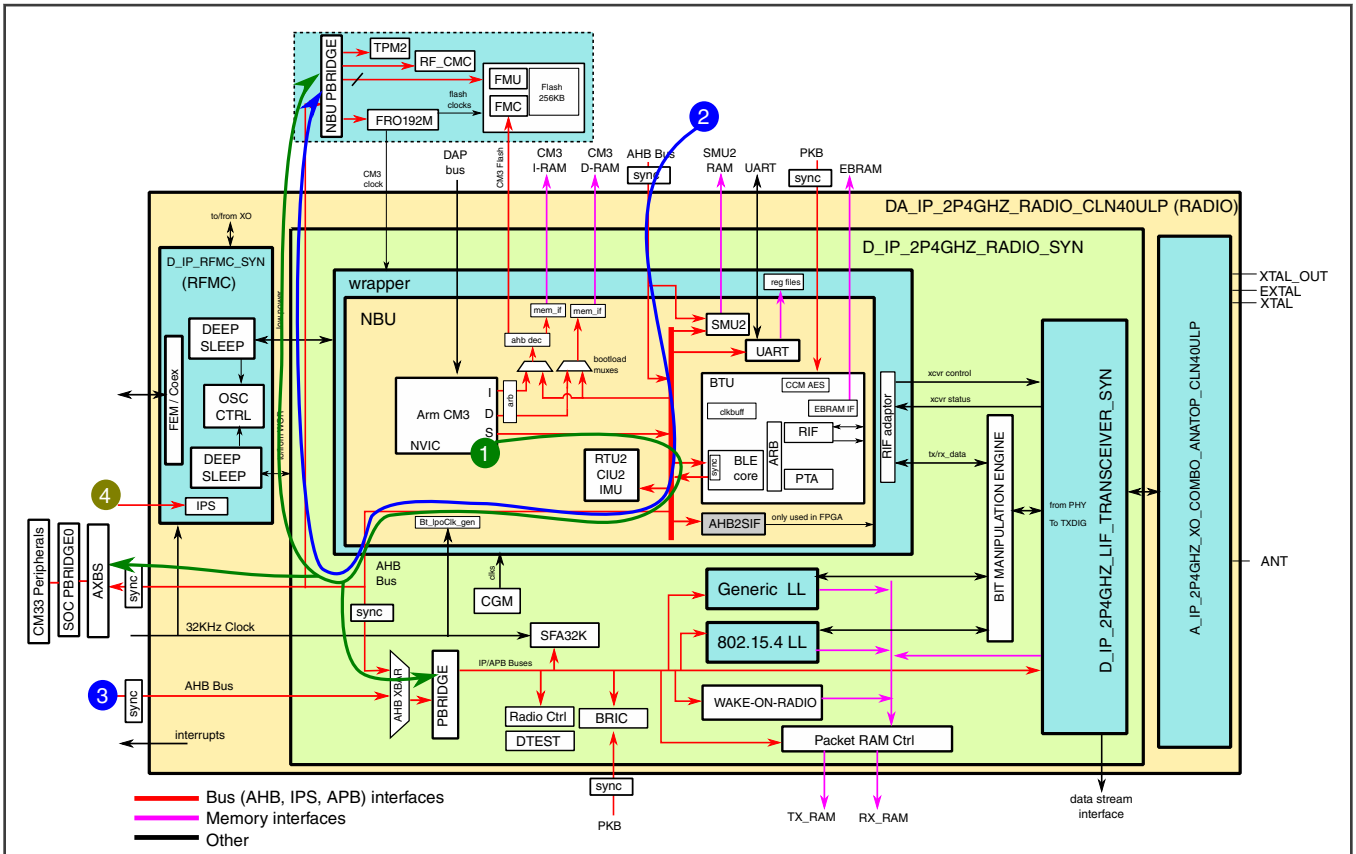


Figure 218. Radio Bus

The table below shows the radio memory space (excluding CM33 access to RFMC). The left columns show the CM3 memory space (#1 in previous figure). Columns to the right of that show the CM33 access via NBU (#2 in previous figure) and via the radio PBRIDGE (#3 in previous figure).

Table 397. Radio Memory Space (excluding CM33 access to RFMC)

CM3 Memory Space				CM33 access to radio via NBU	CM33 access to radio via PBRIDGE
Start Addr	End Addr	Size	Module/Description[1]	Start Addr	Start Addr
00000000	0003FFFF	256KB	CM3 access to flash (main array)		
00040000	00047FFF	32KB	CM3 access to flash (IFR0)		
00048000	00049FFF	8KB	CM3 access to flash (IFR1)		
0004A000	000FFFFF	728KB	CM3 access to flash (Reserved)		
00100000	001007FF	2KB	CM3 access to IMEM		
00140000	0014BFFF	48KB	CM3 access to DMEM		
00180000	9FFFFFFF	~2.5GB	Reserved		
A0000000	A003FFFF	256KB	SOC access to flash (main array)	48800000	

Table continues on the next page...

Table 397. Radio Memory Space (excluding CM33 access to RFMC) (continued)

CM3 Memory Space				CM33 access to radio via NBU	CM33 access to radio via PBRIDGE
Start Addr	End Addr	Size	Module/Description[1]	Start Addr	Start Addr
A0040000	A0047FFF	32KB	SOC access to flash (IFR0)	48840000	
A0048000	A0049FFF	8KB	SOC access to flash (IFR1)	48848000	
A004A000	A00FFFFF	728KB	SOC access to flash (Reserved)	4884A000	
A0100000	A013FFFF	256KB	Reserved (was SOC access to IMEM)		
A0140000	A017FFFF	256KB	Reserved (was SOC access to DMEM)		
A0180000	A01FFFFF	512KB	Reserved		
A0200000	A7FFFFFF	126MB	Reserved		
A8000000	A80007FF	2KB	Reserved	48940000	
A8000800	A8000FFF	2KB	AHB2 arbiter config	48940800	
A8001000	A80030FF	8.25KB	Reserved	48941000	
A8003100	A80031FF	256B	BT UART	48943100	
A8003200	A8003FFF	3.5KB	Reserved (returns OKAY response)	48943200	
A8004000	A8004FFF	4KB	BT UART PFU	48944000	
A8005000	A8007FFF	12KB	Reserved	48945000	
A8008000	A80093FF	5KB	CIU2	48948000	
A8009400	A80097FF	1KB	BT_RTU1	48949400	
A8009800	A803FFFF	218KB	Reserved (returns OKAY response)	48949800	
A9000000	A900FFFF	64KB	Reserved		
A9010000	A901FFFF	64KB	Reserved		
A9020000	A902FFFF	64KB	Bluetooth LE		
A9030000	A903FFFF	64KB	Reserved		
A9040000	A9047FFF	32KB	BRF register space (AHB2SIF)		
A9048000	A904FFFF	32KB	BRF memory		
A9050000	A90FFFFF	704KB	Reserved		
A9100000	A913FFFF	256KB	Access to Radio PBRIDGE (outside NBU), see detailed entries below		
A9100000	A9100FFF	4KB	>Reserved		48A00000
A9101000	A9101FFF	4KB	>802.15.4 LL		48A01000
A9102000	A9102FFF	4KB	>Generic LL		48A02000
A9103000	A9103FFF	4KB	>Generic LL REMAP0		48A03000

Table continues on the next page...

Table 397. Radio Memory Space (excluding CM33 access to RFMC) (continued)

CM3 Memory Space				CM33 access to radio via NBU	CM33 access to radio via PBRIDGE
Start Addr	End Addr	Size	Module/Description[1]	Start Addr	Start Addr
A9104000	A9104FFF	4KB	>Generic LL REMAP0		48A04000
A9105000	A9105FFF	4KB	>Generic LL REMAP0		48A05000
A9106000	A91060FF	256B	>Radio Control MISC		48A06000
A9106100	A91061FF	256B	>WOR		48A06100
A9106200	A91062FF	256B	>RBME		48A06200
A9106300	A91066FF	256B	>SFA		48A06300
A9106700	A91067FF	256B	>BRIC (KEY)		48A06700
A9106800	A9106FFF	2KB	>BRIC (LTC)		48A06800
A9107000	A91071FF	512B	>XCVR/RX_DIG		48A07000
A9107200	A91072FF	256B	>XCVR/TX_DIG		48A07200
A9107300	A91073FF	256B	>XCVR/PLL		48A07300
A9107400	A91074FF	256B	>XCVR/ Reserved		48A07400
A9107500	A91075FF	256B	>XCVR/802.15.4 PHY		48A07500
A9107600	A91077FF	512B	>XCVR/GEN4 PHY		48A07600
A9107800	A9107BFF	1KB	>XCVR/TSM		48A07800
A9107C00	A9107CFF	256B	>XCVR/XCVR_ANA		48A07C00
A9107D00	A9107DFF	256B	>XCVR/XCVR_MISC		48A07D00
A9107E00	A9107FFF	512B	>XCVR/ Reserved		48A07E00
A9108000	A9108FFF	4KB	>TX Packet RAM		48A08000
A9109000	A91097FF	2KB	>RX Packet RAM		48A09000
A9109800	A910FFFF	26KB	>Reserved		48A09800
A9110000	A913FFFF	192KB	>Reserved		
A9140000	A917FFFF	256KB	Access to NBU PBRIDGE (outside radio), see detailed entries below	48980000	
A9140000	A9100FFF	4KB	>Radio FRO192M	48980000	
A9141000	A9101FFF	4KB	>Radio Flash FMU	48981000	
A9142000	A9102FFF	4KB	>Radio Flash FMC	48982000	
A9143000	A9103FFF	4KB	>RF_CMC	48983000	
A9144000	A9104FFF	4KB	>TPM2	48984000	
A9145000	A9105FFF	4KB	>Radio Flash test	48985000	

Table continues on the next page...

Table 397. Radio Memory Space (excluding CM33 access to RFMC) (continued)

CM3 Memory Space				CM33 access to radio via NBU	CM33 access to radio via PBRIDGE
Start Addr	End Addr	Size	Module/Description[1]	Start Addr	Start Addr
A9146000	A917FFFF	236KB	>Reserved	48986000	
A9180000	A91FFFFF	512KB	Access to CM33 peripherals via AXBS PRBIDGE0 (outside radio)[2]		
A9200000	AFFFFFFF	110MB	Reserved		
A8040000	A8FFFFFF	15.75MB	Reserved		
B0000000	B0009FFF	40KB	SMU2	489C0000	
B000A000	B003FFFF	216KB	Reserved (returns OKAY response)	489CA000	
B0040000	FFFFFFFF	~1.31GB	Reserved		

1. Access to "Reserved" regions return ERROR response except as noted.
2. The CM3 Start Addr for the peripheral assigned to PBRIDGE0 slot "n" is 0xA9180000 + n * 0x1000. For a detailed list of peripherals accessible via PBRIDGE0, refer to the Peripheral Bridge section of the Core overview chapter of this Reference Manual.

55.4.4 Clocking

The radio uses the following clock sources:

- Radio XO: A 32MHz or 26MHz crystal is normally connected to the radio's XTAL and EXTAL pins, but a 32MHz or 26MHz clock input to EXTAL is also supported.
- Radio FRO192M: This 192MHz clock source is used to derive the clocks for the radio CM3, the radio flash and buses related to the NBU and radio flash.
- SOC 32.768kHz. This clock, from the SOC's CCM32K module, is used by the radio's "always on" logic and by the radio's SFA32K module.

The figure below shows the clock distribution, with focus on the NBU clocks. For the FRO clock outputs, the default output frequency to the NBU and PD_RF AHB, etc. is 32MHz, and the default output frequency for the flash APB and RF_CMC clock is 16MHz (with a maximum supported frequency of 24MHz).

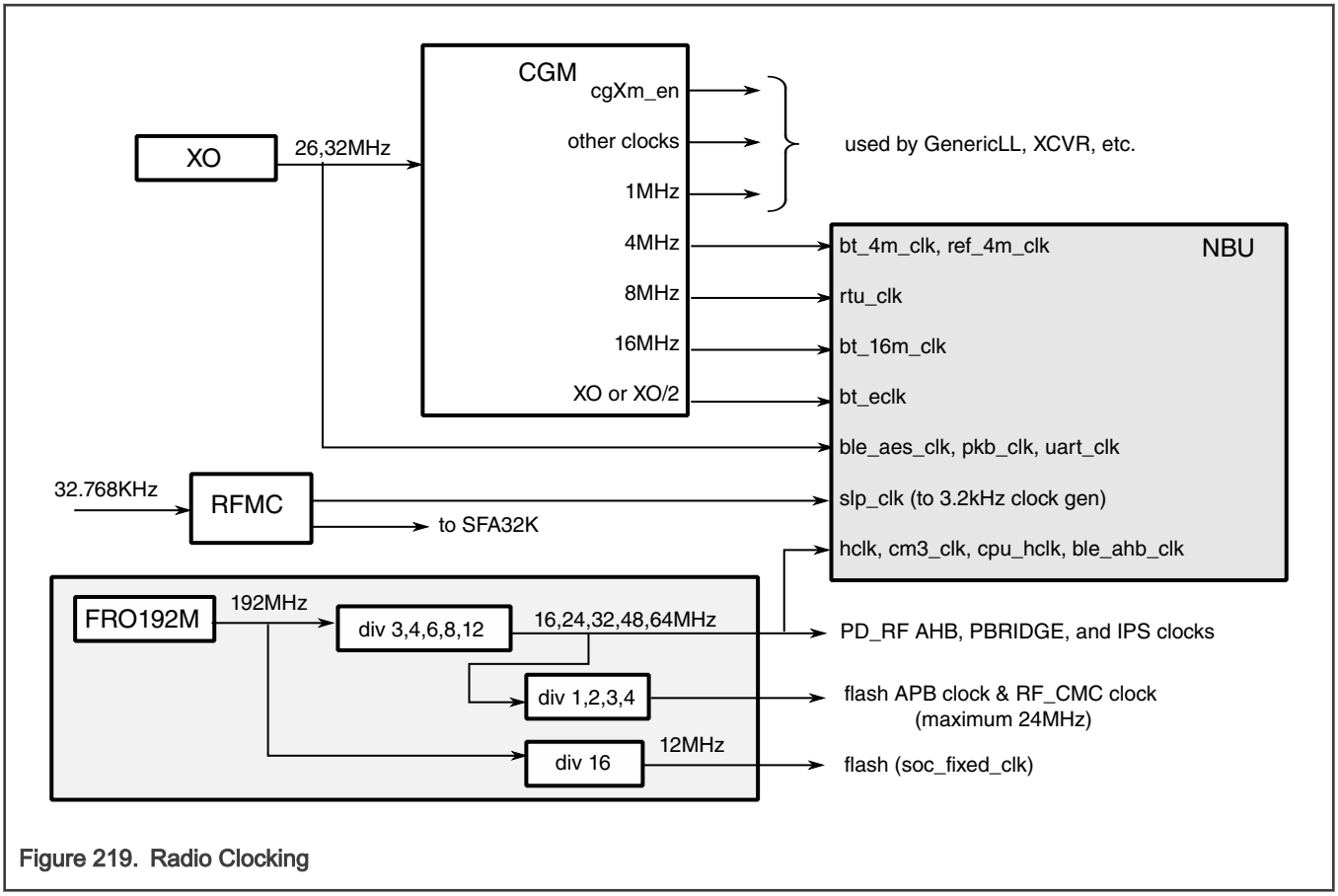


Figure 219. Radio Clocking

Inside the NBU, there is a module (bt_lpoClk_gen) which generates a 3.2KHz clock: in low power modes, this clock is derived from slp_clk (32.768KHz), while when not in low power mode the clock is derived from bt_4m_clk (4MHz).

The NBU's clock inputs are further described in the table below.

Table 398. NBU Clocks

NBU clock input	Source	Used in NBU by	Clock enabled by
cm3_hclk	FRO divider	CM3	RF_CLK_CTRL[CM3_HCLK_EN] & !cpu_sleeping & nbu_cm3_clock_en_mask[1]
cpu_hclk		flash/IMEM/ DMEM/buses	RF_CLK_CTRL[NBU_AHB_CLK_EN] & ((bt_clk_req & RF_CLK_CTRL[BT_CLK_REQ_EN]) || (!man_deep_sleep_enable & RF_CLK_CTRL[MAN_DS_EN]) || (!wor_deep_sleep_enable & RF_CLK_CTRL[WOR_DS_EN]) || ! cpu_cp15_sleep) & nbu_ahb_clock_en_mask[1]
hclk		AHB subsystem clock, SMU2, CIU2	
ble_ahb_clk		BTU	RF_CLK_CTRL[BLE_AHB_CLK_EN] & bt_clk_req
bt_eclk	XO or XO/2[2]	BTU	RF_CLK_CTRL[BT_ECLK_EN] & bt_clk_req & btll[3]
bt_16m_clk	XO divider (16MHz)	BTU	RF_CLK_CTRL[BT_16M_CLK_EN] & bt_clk_req & btll[3]
rtu_clk	XO divider (8MHz)	RTU	RF_CLK_CTRL[RTU_CLK_EN] & bt_clk_req & btll[3]

Table continues on the next page...

Table 398. NBU Clocks (continued)

NBU clock input	Source	Used in NBU by	Clock enabled by
bt_4m_clk	XO divider (4MHz)	BTU	RF_CLK_CTRL[BT_4M_CLK_EN] & bt_clk_req & btll[3]
ref_4m_clk		lpoClk_gen	RF_CLK_CTRL[BT_REF_4M_CLK_EN] & btll[3]
ble_aes_clk	XO	AES	RF_CLK_CTRL[BLE_AES_CLK_REQ] & ble_aes_clk_req & btll[3]
nbu_pkb_clk		Private Key Bus	RF_CLK_CTRL[NBU_PKB_CLK_EN]
uart_clk		UART	RF_CLK_CTRL[UART_CLK_EN] & btll[3]
slp_clk	32.768kHz	lpoClk_gen	RFMC: RF2p4GHz_CTRL register: BLE_LP_EN | CLK_OVR[3]

1. from RF_CMC
2. selected by RF_CLK_CTRL[BT_ECLK_DIV]. Should be >=16MHz, so for XO=26MHz select XO while for XO=32MHz XO/2 can be selected
3. btll = btll_en & (LL_CTRL[ACTIVE_LL]==Bluetooth LE) | RF_CLK_CTRL[BTLL_CLK_EN_OVRD], where btll_en is an IFR bit

The other radio peripherals outside of the NBU use only clocks derived from the XO. The following table describes the 802.15.4 LL, Generic LL, and WOR clocks

Table 399. Link Layer and WOR Clocks

Module	Clock	Clock enabled by
802.15.4 LL	zigbee_tsm	(tsm_clk_en | zig_force_clk_on) & zbll[1]
802.15.4 LL	zigbee	!dsm_zig_deep_sleep_enable & zbll[1]
Generic LL	generic_tmr (1MHz)	generic_tmrload & !dsm_gen_deep_sleep_en & genll[2]
Generic LL	generic_500k (500kHz)	!dsm_gen_deep_sleep_en & genll[2]
Generic LL	generic_1m (1MHz or XO when requested for tmrload)	!dsm_gen_deep_sleep_en & genll[2]
WOR	clk_wor	wor_en
WOR	clk_wor_1m (1MHz)	(wor_en | wor_clk_xtend)
WOR	clk_wor_comp (1MHz or XO when requested for computing)	(wor_en | wor_clk_xtend)
WOR	clk_wor_timestamp	(wor_en | wor_clk_xtend) & wor_timstamp_clk_en
WOR	clk_wor_ww_1m (1MHz)	(wor_en | wor_clk_xtend) & ! wor_deep_sleep_enable

1. zbll = zbll_en & (LL_CTRL[ACTIVE_LL]==ZIGBEE) | RF_CLK_CTRL[ZBLL_CLK_EN_OVRD], where zbll_en is an IFR bit
2. genll = genll_en & (LL_CTRL[ACTIVE_LL]==GENERIC) | RF_CLK_CTRL[GENLL_CLK_EN_OVRD], where genll_en is an IFR bit

55.4.5 Radio interrupts and Wake-up request

The Radio can generate some asynchronous interrupts and one asynchronous MCU wakeup signal.

The interrupt configurations are illustrated in the diagram below and detailed in the following sections.

55.4.5.1 Interrupts and Wake-up request Diagram

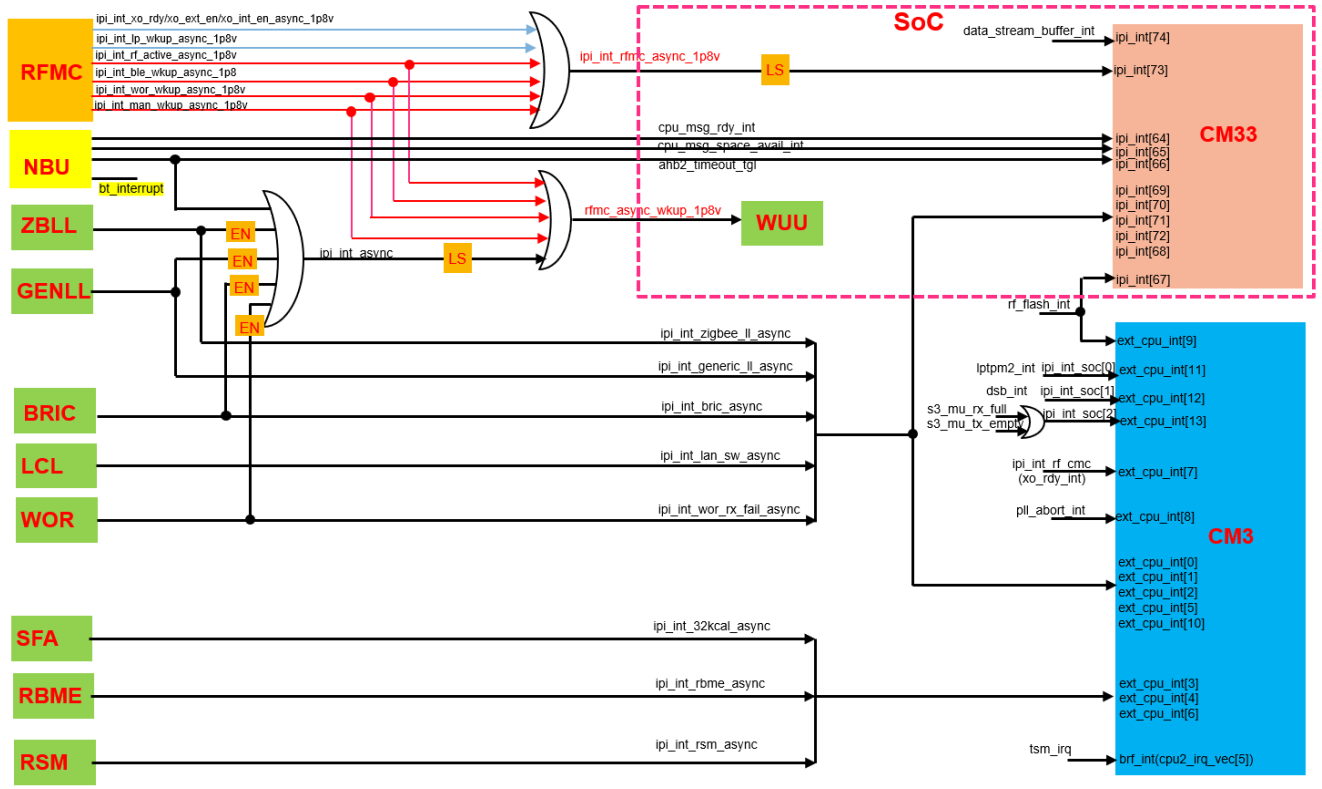


Figure 220. Radio Interrupts

55.4.5.2 Interrupt sources and connections

The radio interrupt sources and connections are summarized in the table below.

Table 400. Radio Interrupts

Interrupt Source	to NBU/CM3 NVIC	to SOC
sources inside the NBU used exclusively by CM3	connected inside the NBU	
NBU IMU interrupt to SOC Core (cpu_msg_rdy_int)		radio output to SOC Core NVIC
NBU IMU interrupt to SOC Core (cpu_msg_space_avail_int)		radio output to SOC Core NVIC
NBU timeout (ahb2_timeout_tgl)		radiout output to SOC Core NVIC *or* SOC RESET
Transceiver: TSM	connected inside radio to brf_int	
15.4 LL	connected inside radio to ext_cpu_int[0]	radio output to SOC Core NVIC
Generic LL	connected inside radio to ext_cpu_int[1]	radio output to SOC Core NVIC
BRIC	connected inside radio to ext_cpu_int[2]	radio output to SOC Core NVIC

Table continues on the next page...

Table 400. Radio Interrupts (continued)

Interrupt Source	to NBU/CM3 NVIC	to SOC
SFA32K	connected inside radio to ext_cpu_int[3]	
Transceiver: RBME	connected inside radio to ext_cpu_int[4]	
Transceiver: LCL_CTRL (lant_sw)	connected inside radio to ext_cpu_int[5]	radio output to SOC Core NVIC
Transceiver: Ranging Sequence Manager	connected inside radio to ext_cpu_int[6]	
RFMC		radio output to SOC Core NVIC
RF_CMC (XO ready)	connected inside radio to ext_cpu_int[7]	
Transceiver: PLL abort	connected inside radio to ext_cpu_int[8]	
Radio Flash FMU	radio input from PD_RF domain to ext_cpu_int[9]	PD_RF domain output to SOC Core NVIC
WOR (RX failure)	connected inside radio to ext_cpu_int[10]	radio output to SOC Core NVIC
TPM2 (in PD_RF domain)	radio input from PD_RF domain to ext_cpu_int[11]	
Data Stream Buffer (in SOC domain)	radio input from SOC domain to ext_cpu_int[12]	
EdgeLock Secure Enclave (in SOC domain) rx_full || tx_empty	radio input from SOC domain to ext_cpu_int[13]	
RFMC WUU_MxIF (OR of radio interrupts assigned to SOC Core)		radio output to SOC Core WWU

55.4.6 Radio Control

55.4.6.1 Radio Misc Register

55.4.6.1.1 RADIO_MISC register descriptions

55.4.6.1.1.1 RADIO_MISC memory map

RADIO_CTRL base address: 48A0_6000h

Offset	Register	Width (In bits)	Access	Reset value
4h	LL Control Register (LL_CTRL)	32	RW	0000_0003h
8h	Radio Control Register (RF_CTRL)	32	RW	0000_0000h
Ch	Radio Clock Control Register (RF_CLK_CTRL)	32	RW	801F_FF00h
10h	COEXISTENCE CONTROL (COEX_CTRL)	32	RW	0000_0000h
14h	Radio Control Register (UID_MSB)	32	R	0000_0000h
18h	Radio Control Register (UID_LSB)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

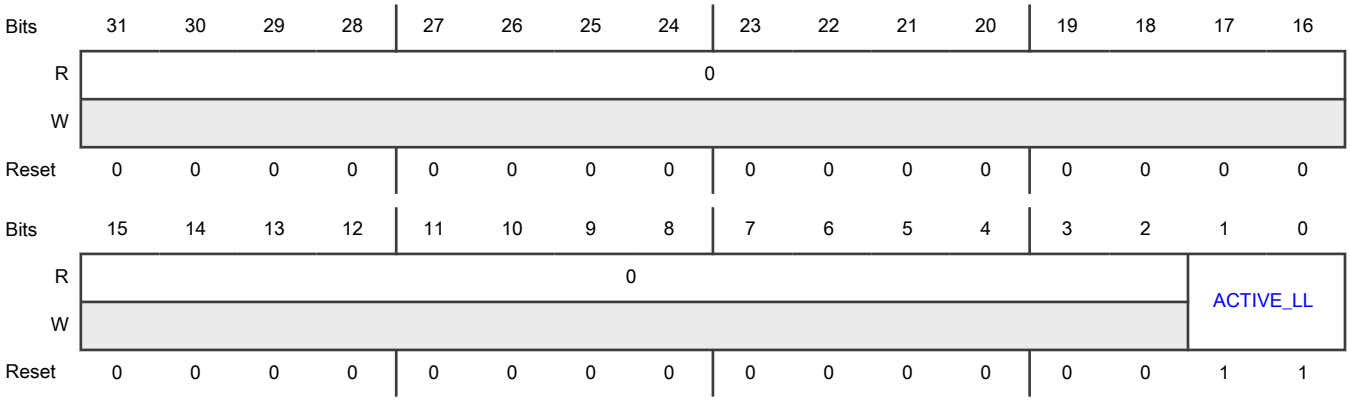
Offset	Register	Width (In bits)	Access	Reset value
1Ch	PACKET RAM Control Register (PACKET_RAM_CTRL)	32	RW	0000_0000h
20h	BLE PHY Interface Control Register (BLE_PHY_CTRL)	32	RW	0F0F_1222h
24h	DTEST Control register (DTEST_CTRL)	32	RW	0000_0000h
30h	DTEST PIN Control 2 register (DTEST_PIN_CTRL2)	32	RW	0B0A_0908h

55.4.6.1.1.2 LL Control Register (LL_CTRL)

Offset

Register	Offset
LL_CTRL	4h

Diagram



Fields

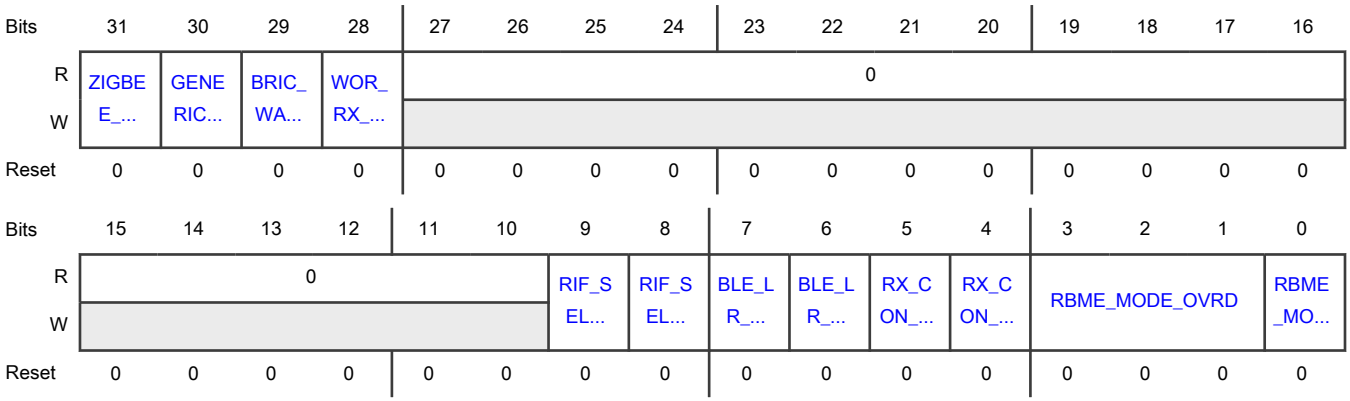
Field	Function
31-2 —	Reserved
1-0 ACTIVE_LL	link layer control register Controls access to shared resources: Packet RAM, DSM, WOR, DEMOD_SEL, LL Channel index to PLL. 00b - Bluetooth LE LL is selected 01b - ZIGBEE LL is selected 10b - GENERIC LL is selected 11b - Disabled (default)

55.4.6.1.1.3 Radio Control Register (RF_CTRL)

Offset

Register	Offset
RF_CTRL	8h

Diagram



Fields

Field	Function
31 ZIGBEE_WAKEUP_EN	Zigbee LL Wakeup Enable 0b - The Zigbee LL interrupt doesn't assert rfmc_wakeup. 1b - The Zigbee LL interrupt asserts rfmc_wakeup.
30 GENERIC_WAKEUP_EN	Generic LL Wakeup Enable 0b - The Generic LL interrupt doesn't assert rfmc_wakeup. 1b - The Genecir LL interrupt asserts rfmc_wakeup.
29 BRIC_WAKEUP_EN	BRIC Wakeup Enable 0b - The BRIC interrupt doesn't assert rfmc_wakeup. 1b - The BRIC interrupt asserts rfmc_wakeup.
28 WOR_RX_FAIL_WAKEUP_EN	WOR RX Fail Wakeup Enable 0b - The wor_rx_fail interrupt doesn't assert rfmc_wakeup. 1b - The wor_rx_fail interrupt asserts rfmc_wakeup.
27-10 —	Reserved
9	rif_sel_2mbps Override Override rif_sel_2mbps

Table continues on the next page...

Table continued from the previous page...

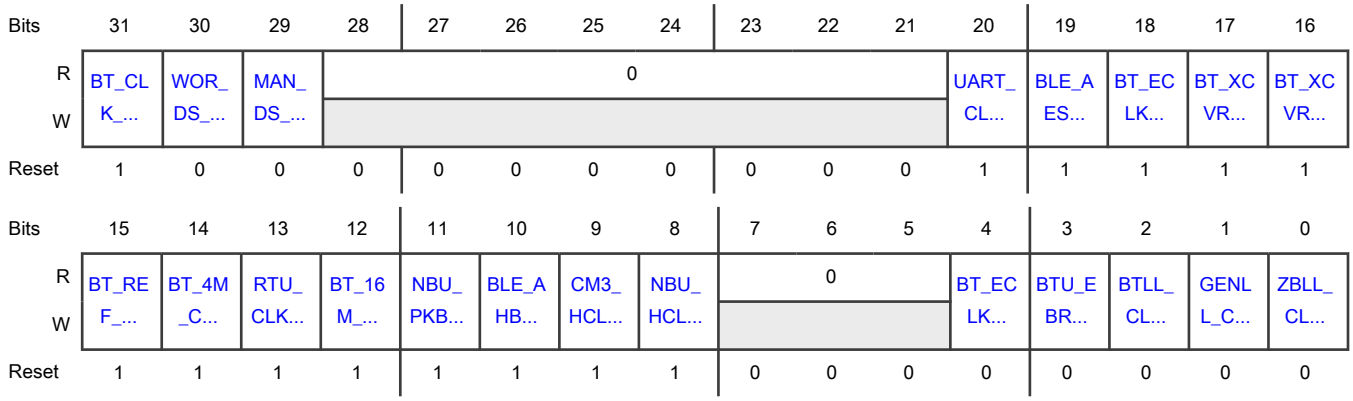
Field	Function
RIF_SEL_2MBPS_OVRD	
8 RIF_SEL_2MBPS_OVRD_EN	rif_sel_2mbps Override Enable 0b - rif_sel_2mbps Override Disable 1b - rif_sel_2mbps Override Enable
7 BLE_LR_EN_OVRD	ble_lr_en Override Typically, the value of ble_lr_en(longrange mode enable) is controlled by GEN-LL or NBU, however, which can be override by BLE_LR_EN_OVRD and BLE_LR_EN_OVRD_EN field.
6 BLE_LR_EN_OVRD_EN	ble_lr_en Override Enable 0b - ble_lr_en Override Disable 1b - ble_lr_en Override Enable
5 RX_CON_EN_OVRD	rx_con_en Override Typically, the value of rx_con_en(uncoded and coded concurrent acquisition enable) is controlled by GEN-LL or NBU, however, which can be override by RX_CON_EN_OVRD and RX_CON_EN_OVRD_EN field.
4 RX_CON_EN_OVRD_EN	rx_con_en Override Enable 0b - rx_con_en Override Disable 1b - rx_con_en Override Enable
3-1 RBME_MODE_OVRD	RBME Mode Override Typically, the value of rbme_mode is controlled by GEN-LL or NBU, however, which can be override by RBME_MODE_OVRD and RBME_MODE_OVRD_EN field.
0 RBME_MODE_OVRD_EN	RBME Mode Override Enable 0b - RBME Mode Override Disable 1b - RBME Mode Override Enable

55.4.6.1.1.4 Radio Clock Control Register (RF_CLK_CTRL)

Offset

Register	Offset
RF_CLK_CTRL	Ch

Diagram



Fields

Field	Function
31 BT_CLK_REQ_EN	BT_CLK_REQ control enable 0b - Disable the control of bt_clk_req for nbu_hclk. 1b - Enable the control of bt_clk_req for nbu_hclk.
30 WOR_DS_EN	WOR deep sleep control enable 0b - Disable the control of rfmc_wor_deep_sleep_enable for nbu_hclk. 1b - Enable the control of rfmc_wor_deep_sleep_enable for nbu_hclk.
29 MAN_DS_EN	Manual deep sleep control enable 0b - Disable the control of rfmc_man_deep_sleep_enable for nbu_hclk. 1b - Enable the control of rfmc_man_deep_sleep_enable for nbu_hclk.
28-21 —	Reserved
20 UART_CLK_EN	UART Clock Enable 0b - uart_clk is disabled. 1b - uart_clk is enabled.
19 BLE_AES_CLK_EN	BLE_AES_CLK Enable 0b - bt_aes_clk is disabled. 1b - bt_aes_clk is enabled.
18 BT_ECLK_EN	BT_ECLK Enable 0b - bt_eclk is disabled. 1b - bt_eclk is enabled.
17	BT XCVR 32M Clock Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
BT_XCVR_32M_CLK_EN	0b - bt_xcvr_32m_clk is disabled. 1b - bt_xcvr_32m_clk is enabled.
16 BT_XCVR_4M_CLK_EN	BT XCVR 4M Clock Enable 0b - bt_xcvr_4m_clk is disabled. 1b - bt_xcvr_4m_clk is enabled.
15 BT_REF_4M_CLK_EN	BT REF 4M Clock Enable 0b - bt_ref_4m_clk is disabled. 1b - bt_ref_4m_clk is enabled.
14 BT_4M_CLK_EN	BT 4M Clock Enable 0b - bt_4m_clk is disabled. 1b - bt_4m_clk is enabled.
13 RTU_CLK_EN	RTU Clock Enable 0b - rtu_clk is disabled. 1b - rtu_clk is enabled.
12 BT_16M_CLK_EN	BT 16M Clock Enable 0b - bt_16m_clk is disabled. 1b - bt_16m_clk is enabled.
11 NBU_PKB_CLK_EN	NBU PKB Clock Enable 0b - nbu_pkb_clk is disabled. 1b - nbu_pkb_clk is enabled.
10 BLE_AHB_CLK_EN	BLE_AHB CLOCK Enable 0b - ble_ahb_clk is disabled. 1b - ble_ahb_clk is enabled.
9 CM3_HCLK_EN	CM3 HCLK Enable 0b - cm3_hclk is disabled. 1b - cm3_hclk is enabled.
8 NBU_HCLK_EN	NBU HCLK Enable 0b - nbu hclk/cpu_hclk are disabled. 1b - nbu hclk/cpu_hclk are enabled.
7-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

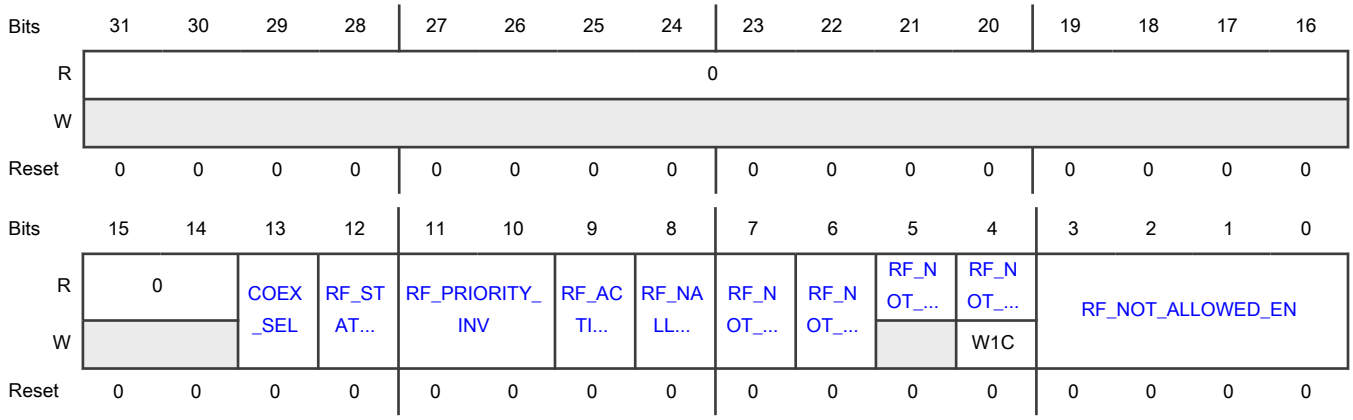
Field	Function
4 BT_ECLK_DIV	BE_ECLK Divider Enable bt_eclk divider. 0b - ref_clk is not divided as bt_eclk. 1b - ref_clk is divided by 2 as bt_eclk.
3 BTU_EBRAM_CLK_ON_OVRD	BTU EBRAM Clock Enable Override 0b - btu_ebram_clk is not forced on. 1b - btu_ebram_clk is forced on.
2 BTLL_CLK_EN_OVRD	BTLL Clock Enable Override 0b - BTLL clock force on is disabled. 1b - BTLL clock force on is enabled.
1 GENLL_CLK_EN_OVRD	GENLL Clock Enable Override 0b - GENLL clock force on is disabled. 1b - GENLL clock force on is enabled.
0 ZBLL_CLK_EN_OVRD	ZBLL Clock Enable Override 0b - ZBLL clock force on is disabled. 1b - ZBLL clock force on is enabled.

55.4.6.1.1.5 COEXISTENCE CONTROL (COEX_CTRL)

Offset

Register	Offset
COEX_CTRL	10h

Diagram



Fields

Field	Function
31-14 —	Reserved
13 COEX_SEL	COEX_SEL 0b - Select coexistence signals from LL. 1b - Select coexistence signals from TSM.
12 RF_STATUS_INV	RF_STATUS Invert The normal behavior for the RF_STATUS signal is active high. This bit can optionally invert this signal. Refer also to "Radio Coexistence/FEM/LANT Mux diagram" figure. 0b - rf_status is not inverted. 1b - rf_status is inverted.
11-10 RF_PRIORITY_INV	RF_PRIORITY Invert The normal behavior for the RF_PRIORITY signals is active high. This bitfield can optionally invert these signals. Refer also to "Radio Coexistence/FEM/LANT Mux diagram" figure. 0xb - rf_priority[1] is not inverted. 1xb - rf_priority[1] is inverted. x0b - rf_priority[0] is not inverted. x1b - rf_priority[0] is inverted.
9 RF_ACTIVE_INV	RF_ACTIVE Invert The normal behavior for the RF_ACTIVE signal is active high. This bit can optionally invert this signal. Refer also to "Radio Coexistence/FEM/LANT Mux diagram" figure. 0b - rf_active is not inverted. 1b - rf_active is inverted.
8 RF_NALLOWED_INV	RF_NALLOWED Invert The normal behavior for the RF_NALLOWED signal is active high. This bit can optionally invert this signal. Refer also to "Radio Coexistence/FEM/LANT Mux diagram" figure. 0b - rf_nallowed is not inverted. 1b - rf_nallowed is inverted.
7 RF_NOT_ALLOWED_OVRD_EN	RF_NOT_ALLOWED Override Enable 0b - RF_NALLOWED Override Disable 1b - RF_NALLOWED Override Enable
6	RF_NOT_ALLOWED Override Override rf_not_allowed

Table continues on the next page...

Table continued from the previous page...

Field	Function
RF_NOT_ALLOWED_OVRD	
5 RF_NOT_ALLOWED	RF_NOT_ALLOWED Reflects the instantaneous state of the RF_NOT_ALLOWED pin, synchronized into the RF OSC clock domain.
4 RF_NOT_ALLOWED_ASSERTED	RF_NOT_ALLOWED_ASSERTED 0b - Assertion on RF_NOT_ALLOWED has not occurred 1b - Assertion on RF_NOT_ALLOWED has occurred since the last time this bit was cleared
3-0 RF_NOT_ALLOWED_EN	RF_NOT_ALLOWED PER-LINK-LAYER ENABLE The coexistence input RF_NOT_ALLOWED can be enabled to selectively abort TX or RX sequences, with individual enables for each supported protocol, according to the following table: **xxx1** : RF_NOT_ALLOWED assertions are enabled to abort Bluetooth LE TX and RX sequences **xxx0** : RF_NOT_ALLOWED assertions are not enabled to abort Bluetooth LE TX and RX sequences **x1xx** : RF_NOT_ALLOWED assertions are enabled to abort Zigbee TX and RX sequences **x0xx** : RF_NOT_ALLOWED assertions are not enabled to abort Zigbee TX and RX sequences **1xxx** : RF_NOT_ALLOWED assertions are enabled to abort GENERIC_FSK TX and RX sequences **0xxx** : RF_NOT_ALLOWED assertions are not enabled to abort GENERIC_FSK TX and RX sequences

55.4.6.1.1.6 Radio Control Register (UID_MSB)

Offset

Register	Offset
UID_MSB	14h

Function

The UID_MSB and UID_LSB registers provide read-only access to the 40bit Radio Unique ID which is stored in Flash IFR. This can be used in part to define MAC address(es) needed by radio protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RADIO_UID_MSB							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 RADIO_UID_MSB	The most signficant 8bits of the 40bit Radio UID.

55.4.6.1.1.7 Radio Control Register (UID_LSB)

Offset

Register	Offset
UID_LSB	18h

Function

The UID_MSB and UID_LSB registers provide read-only access to the 40bit Radio Unique ID which is stored in Flash IFR. This can be used in part to define MAC address(es) needed by radio protocols.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RADIO_UID_LSB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RADIO_UID_LSB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RADIO_UID_LSB	The least signficant 32bits of the 40bit Radio UID.

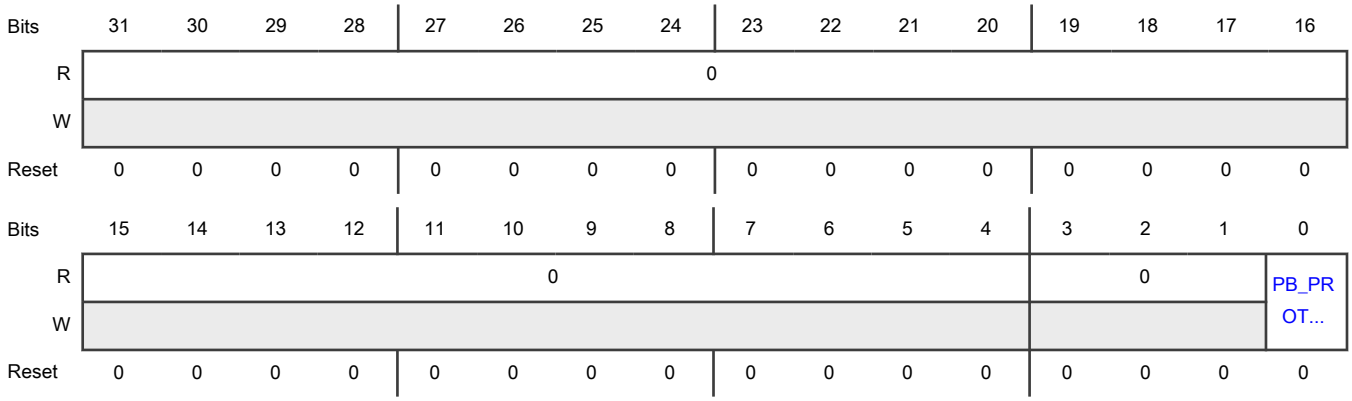
55.4.6.1.1.8 PACKET RAM Control Register (PACKET_RAM_CTRL)

Offset

Register	Offset
PACKET_RAM_CTRL	1Ch

Function
Packet RAM Control Register

Diagram



Fields

Field	Function
31-4 —	Reserved
3-1 —	Reserved
0 PB_PROTECT	PB_PROTECT The bit inhibits receive-packet overwriting of the RX section of the Packet Buffer contents, but doesn't inhibit TX content loading of the TX section of the Packet Buffer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Incoming receive data can overwrite the existing contents of the RX section of the Packet Buffer. 1b - Incoming receive data is been blocked from overwriting the existing contents of the RX section of the Packet Buffer.

55.4.6.1.1.9 BLE PHY Interface Control Register (BLE_PHY_CTRL)

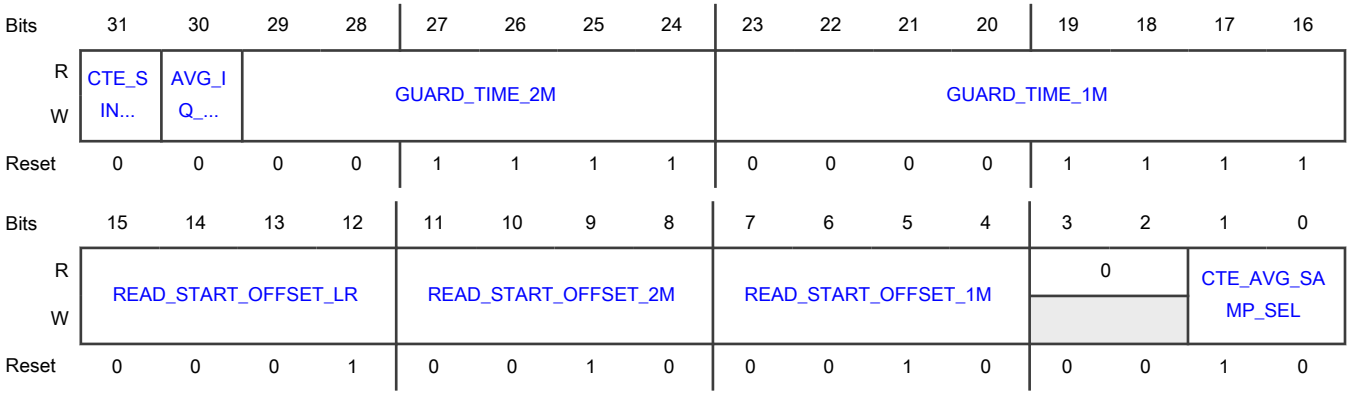
Offset

Register	Offset
BLE_PHY_CTRL	20h

Function

BLE PHY Interface Control Register

Diagram



Fields

Field	Function
31 CTE_SINGLE_BUF	Config for using single buffer for Rx data and CTE samples
30 AVG_IQ_DISABLE	Disable IQ sample averaging
29-24	Guard time offset for 2M

Table continues on the next page...

Table continued from the previous page...

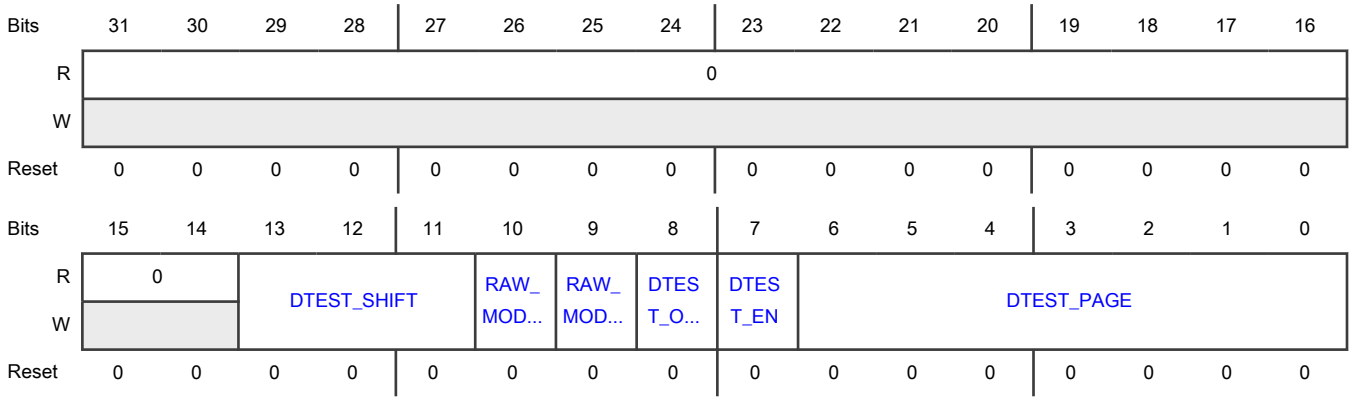
Field	Function
GUARD_TIME_2M	
23-16 GUARD_TIME_1M	Guard time offset for 1M
15-12 READ_START_OFFSET_LR	Start sending Rx data to NBU after a programmable number of symbols are received from PHY - LR
11-8 READ_START_OFFSET_2M	Start sending Rx data to NBU after a programmable number of symbols are received from PHY - 2M
7-4 READ_START_OFFSET_1M	Start sending Rx data to NBU after a programmable number of symbols are received from PHY - 1M
3-2 —	Reserved
1-0 CTE_AVG_SAMP_SEL	Sampling select

55.4.6.1.1.10 DTEST Control register (DTEST_CTRL)

Offset

Register	Offset
DTEST_CTRL	24h

Diagram



Fields

Field	Function
31-14 —	Reserved
13-11 DTEST_SHIFT	DTEST shift control 1.Shift control for rx_dig_iq 0bx00: rx_dig_iq[11:0] 0bx01: {rx_dig_iq[11],rx_dig_iq[11:1]} 0bx10: {rx_dig_iq[11],rx_dig_iq[11],rx_dig_iq[11:2]} 0bx11: Reserved - same as x00 2.Shift control for pll_ripple_counter 0b000: pll_ripple_counter[18:5] 0b001: pll_ripple_counter[17:4] 0b010: pll_ripple_counter[16:3] 0b011: pll_ripple_counter[15:2] 0b100: pll_ripple_counter[14:1] 0b101: pll_ripple_counter[13:0] 0b110: Reserved - same as 0b000 0b111: Reserved - same as 0b000
10 RAW_MODE_Q	Select rx_dig_q as DTEST RX_IQ page
9 RAW_MODE_I	Select rx_dig_i as DTEST RX_IQ page
8	Enable/Disable register dtest signal

Table continues on the next page...

Table continued from the previous page...

Field	Function
DTEST_OUT_REG_EN	0b - output dtest signal directly 1b - output dtest signal after registered
7 DTEST_EN	DTEST_EN control 0b - disable dtest feature 1b - enable dtest feature
6-0 DTEST_PAGE	DTEST PAGE Number Refer to the detailed dtest page definition table

55.4.6.1.1.11 DTEST PIN Control 2 register (DTEST_PIN_CTRL2)

Offset

Register	Offset
DTEST_PIN_CTRL2	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DTEST_PIN11_OVRD_SEL				DTEST_PIN11_MUX_SEL				DTEST_PIN10_OVRD_SEL				DTEST_PIN10_MUX_SEL			
Reset	0	0	0	0	1	0	1	1	0	0	0	0	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DTEST_PIN9_OVRD_SEL				DTEST_PIN9_MUX_SEL				DTEST_PIN8_OVRD_SEL				DTEST_PIN8_MUX_SEL			
Reset	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0

Fields

Field	Function
31-28 DTEST_PIN11_OVRD_SEL	DTEST_PIN11_OVRD_SEL Select the override signal for DTEST PIN11 0000b - override is disabled 0001b - aa_sfd_matched 0010b - rx_pd_fnd

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0011b - agc_gain_change 0100b - tsm_combined_tx_en 0101b - tsm_combined_rx_en 0110b - crc_fail 0111b - decode_data_out 1000b - tx_data_out 1001b - nbu_testbus[14] 1010b - nbu_testbus[15] 1011b-1111b - reserved
27-24 DTEST_PIN11_MUX_SEL	DTEST_PIN11_MUX_SEL Select DTEST PIN11 signal from the signal on DTEST PIN[13:0].
23-20 DTEST_PIN10_OVRD_SEL	DTEST_PIN10_OVRD_SEL Select the override signal for DTEST PIN10 0000b - override is disabled 0001b - aa_sfd_matched 0010b - rx_pd_fnd 0011b - agc_gain_change 0100b - tsm_combined_tx_en 0101b - tsm_combined_rx_en 0110b - crc_fail 0111b - decode_data_out 1000b - tx_data_out 1001b - nbu_testbus[14] 1010b - nbu_testbus[15] 1011b-1111b - reserved
19-16 DTEST_PIN10_MUX_SEL	DTEST_PIN10_MUX_SEL Select DTEST PIN10 signal from the signal on DTEST PIN[13:0].
15-12 DTEST_PIN9_OVRD_SEL	DTEST_PIN9_OVRD_SEL Select the override signal for DTEST PIN9 0000b - override is disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0001b - aa_sfd_matched 0010b - rx_pd_fnd 0011b - agc_gain_change 0100b - tsm_combined_tx_en 0101b - tsm_combined_rx_en 0110b - crc_fail 0111b - decode_data_out 1000b - tx_data_out 1001b - nbu_testbus[14] 1010b - nbu_testbus[15] 1011b-1111b - reserved
11-8 DTEST_PIN9_MUX_SEL	DTEST_PIN9_MUX_SEL Select DTEST PIN9 signal from the signal on DTEST PIN[13:0].
7-4 DTEST_PIN8_OVRD_SEL	DTEST_PIN8_OVRD_SEL Select the override signal for DTEST PIN8 0000b - override is disabled 0001b - aa_sfd_matched 0010b - rx_pd_fnd 0011b - agc_gain_change 0100b - tsm_combined_tx_en 0101b - tsm_combined_rx_en 0110b - crc_fail 0111b - decode_data_out 1000b - tx_data_out 1001b - nbu_testbus[14] 1010b - nbu_testbus[15] 1011b-1111b - reserved
3-0 DTEST_PIN8_MUX_SEL	DTEST_PIN8_MUX_SEL Select DTEST PIN8 signal from the signal on DTEST PIN[13:0].

55.4.6.2 Wake On Radio

55.4.6.2.1 Introduction

The 2.4GHz Wake-On-Radio (WOR) Scheduler, is a new hardware automation block, intended to reduce radio and SoC dynamic power consumption, and facilitate radio frequency hopping.

55.4.6.2.1.1 Overview

The 2.4GHz Radio Wake-On-Radio Scheduler provides hardware automation for the insertion of the radio digital logic into Deep Sleep Mode (DSM), and subsequent DSM exit, as well as the sequencing of RF operations, both TX and RX, between DSM cycles, without MCU intervention. Wake-On-Radio (WOR) substantially increases the autonomy of the radio within the SoC, taking advantage of the separated power domains for the MCU and the Radio in the SoC, by providing sufficient hardware automation to ensure very long MCU low-power intervals, encompassing multiple RF operations, whenever possible. Two different scheduling methods are supported: Timeslot-based scheduling, and Constant-Interval scheduling. WOR also includes hardware automation for RF Frequency Hopping. WOR supports the radio's 802.15.4, and GENERIC_FSK protocol engine(s), and interfaces to the radio's RFMC (Radio Mode Controller) for the scheduling of DSM entry/exit. WOR divides future RF operations into "timeslots", which can occur at regularly-, or non-regularly-scheduled intervals. WOR provides a descriptor-based format for software to program timeslot information. WOR interacts with the protocol engines to dynamically manage radio interrupts, allowing link layer interrupts to propagate only when it is necessary to notify the MCU of required action.

55.4.6.2.1.2 Block Diagram

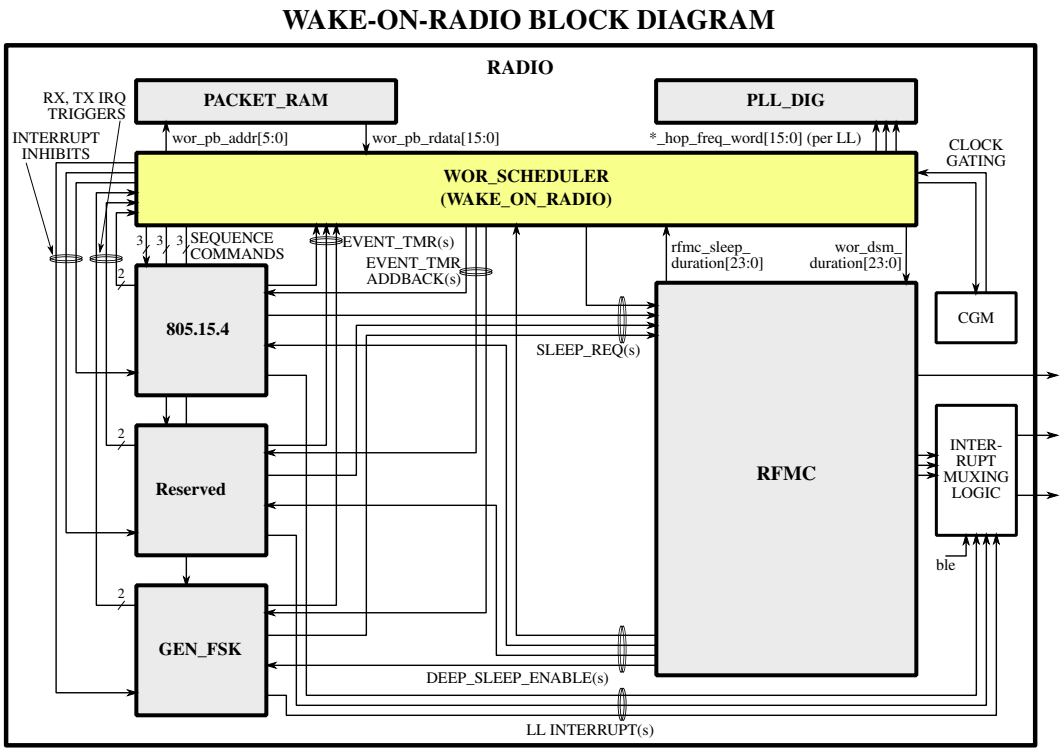


Figure 221. WAKE-ON-RADIO BLOCK DIAGRAM

55.4.6.2.2 Memory Map and Register Definition

Programmable registers to provide configuration and status for the Wake-On-Radio Scheduler are mapped into Transciever Address Space. WOR registers are described in the table below.

55.4.6.2.2.1 WOR register descriptions

55.4.6.2.2.1.1 WOR_ADDR memory map

WOR_REGS base address: 48A0_6100h

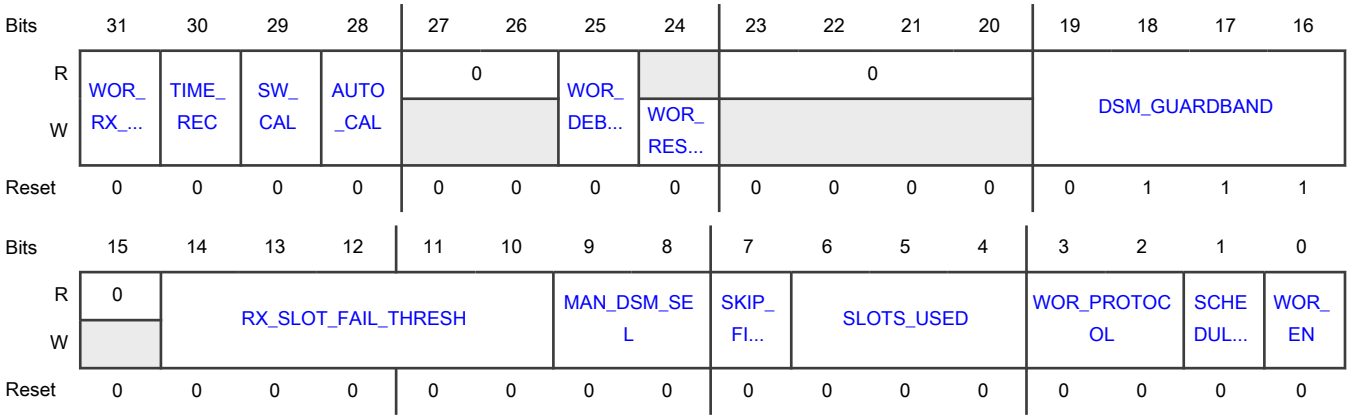
Offset	Register	Width (In bits)	Access	Reset value
0h	WAKE-ON-RADIO CONTROL REGISTER (CTRL)	32	RW	0007_0000h
4h	WAKE-ON-RADIO TIMEOUT REGISTER (TIMEOUT)	32	RW	00FF_0000h
8h	WAKE-ON-RADIO TIMESTAMP 1 (TIMESTAMP1)	32	R	See section
Ch	WAKE-ON-RADIO TIMESTAMP 2 (TIMESTAMP2)	32	R	See section
10h	WAKE-ON-RADIO TIMESTAMP 3 (TIMESTAMP3)	32	R	See section
14h	WAKE-ON-RADIO STATUS REGISTER (STATUS)	32	RW	0000_0000h
18h	WINDOW-WIDENING CONTROL REGISTER (WW_CTRL)	32	RW	0000_0002h
1Ch	FREQUENCY HOP CONTROL REGISTER (HOP_CTRL)	32	RW	0030_0000h
20h	SLOT 0 DESCRIPTOR (LSB) (SLOT0_DESC0)	32	RW	0000_0000h
24h	SLOT 0 DESCRIPTOR (MSB) (SLOT0_DESC1)	32	RW	0000_0000h
28h	SLOT 1 DESCRIPTOR (LSB) (SLOT1_DESC0)	32	RW	0000_0000h
2Ch	SLOT 1 DESCRIPTOR (MSB) (SLOT1_DESC1)	32	RW	0000_0000h
30h	SLOT 2 DESCRIPTOR (LSB) (SLOT2_DESC0)	32	RW	0000_0000h
34h	SLOT 2 DESCRIPTOR (MSB) (SLOT2_DESC1)	32	RW	0000_0000h
38h	SLOT 3 DESCRIPTOR (LSB) (SLOT3_DESC0)	32	RW	0000_0000h
3Ch	SLOT 3 DESCRIPTOR (MSB) (SLOT3_DESC1)	32	RW	0000_0000h
40h	Auto Drift Calculation Register 1 (AUTO_DRIFT1)	32	RW	0000_0000h
44h	Auto Drift Calculation Register 2 (AUTO_DRIFT2)	32	RW	0000_0000h
48h	Auto Drift Calculation Register 3 (AUTO_DRIFT3)	32	RW	0000_0000h
4Ch	Auto Drift Calculation Register 4 (AUTO_DRIFT4)	32	RW	0000_0000h
98h	Timer Count (TIME)	32	R	0000_0000h
9Ch	MAN Low Power Entry Time Captured (ENTER_TIME_CAPT)	32	R	0000_0000h
A0h	MAN Low Power Wakeup Time Captured (WKUP_TIME_CAPT)	32	R	0000_0000h
A4h	MAN Low Power Entry Time Stamp (ENTER_TIME)	32	RW	0000_0000h
A8h	MAN Low Power Wakeup Time Stamp (WKUP_TIME)	32	RW	0000_0000h

55.4.6.2.2.1.2 WAKE-ON-RADIO CONTROL REGISTER (CTRL)

Offset

Register	Offset
CTRL	0h

Diagram



Fields

Field	Function
31 WOR_RX_FAIL_IRQ_EN	WOR_RX_FAIL_IRQ Enable 1: WOR_RX_FAIL_IRQ interrupt is enabled 0: WOR_RX_FAIL_IRQ interrupt is not enabled
30 TIME_REC	Enable the WOR HW to record the timing information to the Packet RAM. 1: Record the timing information to the Packet RAM 0: Do not record the timing information to the Packet RAM.
29 SW_CAL	Enable the WOR SW to calculate the drift. Only when AUTO_CAL is set. Software can calculate the drift based on the timing information stored in the Packet RAM. When AUTO_CAL is enabled, this calculated drift can override the HW result. 1: Enable the software drift override. 0: Disable the software drift override.
28 AUTO_CAL	Auto calculate and track the drift enable Enable the WOR HW to auto calculate the drift. For Constant-Interval mode only. 1: Enable the drift auto calculation. 0: Disable the drift auto calculation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25 WOR_DEBUG_REG	WAKE-ON-RADIO Debug Register Enable WOR_DEBUG_REG alters the readback values of several read-only registers, for debug purposes. The registers which have a different readback value when WOR_DEBUG_REG=1 are: • TIMESTAMP1 • TIMESTAMP2 • TIMESTAMP3 See the description of those registers, which indicates their nominal, and debug-mode readback values.
24 WOR_RESUME	WAKE-ON-RADIO Resume This bit causes the WOR state machine to advance from its WAIT_RESUME state. The state machine will enter WAIT_RESUME on any slot in which an MCU wakeup occurs (based on descriptor WAKE_ON_COMPLETE and/or WAKE_ON_TIMEOUT fields). After waking up (interrupting) the MCU, WOR will hold in WAIT_RESUME, giving software whatever time it needs to complete all required activity. Software then sets WOR_RESUME=1, causing the state machine to break out of WAIT_RESUME, advance to the next slot, and enter DSM. This bit is write-only. Writes of '0' have no effect. Readback value is always 0. Setting WOR_RESUME has no effect if WOR state machine is not in WAIT_RESUME. 1: Causes WOR state machine to advance out of WAIT_RESUME state, and begin the next slot 0: No effect.
23-20 —	Reserved
19-16 DSM_GUARDBAND	WAKE-ON-RADIO DSM Guardband Determines the number of 32.768KHz clock cycles of guardband used by the WOR state machine in computing duration of the DSM for any slot. Increasing this setting by 1 LSB causes WOR to shorten the DSM duration by 1 32.768KHz clock cycle. This guardband takes into account synchronization, and other, delays within the RFMC, and ensures that DSM will wakeup will never occur "too late" with respect to the slot descriptor's START_TIME field. The current default setting is 7. Values below 7 should not be used.
15 —	Reserved
14-10 RX_SLOT_FAIL_THRESH	RX Slot Fail Thresh When the number of missed slot is equal to (RX_SLOT_FAIL_THRESH + 1), the WOR_RX_FAIL_IRQ is set.
9-8 MAN_DSM_SEL	Manual DSM Selector Assigning Manual DSM to a protocol is done by way of the MAN_DSM_SEL[1:0] register, as shown in the following table:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0: Manual DSM Not Assigned 1: Manual DSM Assigned to 802.15.4 2: Manual DSM Assigned to GENERIC_FSK
7 SKIP_FIRST_DSM	WAKE-ON-RADIO Skip DSM On First Slot Setting this bit causes the WOR State Machine to skip the DSM entry/exit cycle for the first SLOT (SLOT 0) encountered after setting SLOTS_USED > 0, regardless of the setting of the NO_DSM field of the slot's descriptor. Subsequent slots, including SLOT 0 when it is re-encountered, will enter/exit DSM based on the descriptor NO_DSM. The SKIP_FIRST_DSM feature is useful, especially in Constant Interval Scheduling mode, when the desire is to begin RF operations immediately (at the start of the first EVENT_INTERVAL), as programmed in the descriptor START_TIME, without waiting out a DSM cycle 1: WOR skips DSM entry/exit on the first SLOT 0 after setting SLOTS_USED > 0, regardless of descriptor NO_DSM 0: WOR always obeys descriptor NO_DSM **Note:** SKIP_FIRST_DSM should not be used if the SLOT 0 descriptor FUNCTION is set to "NO_RF".
6-4 SLOTS_USED	WAKE-ON-RADIO Number Of Slots Used Determines the number of timeslots used to schedule future DSM and RF operations, for either scheduling mode. Up to 4 timeslots can be enabled. Increasing the number of timeslots used decreases average power consumption. Valid values are 0 ≤ SLOTS_USED ≤ 4. The SLOTS_USED register is the mechanism by which the WOR state machine is activated and de-activated. Setting SLOTS_USED to 0 disables the WOR state machine and puts the WOR functionality in an IDLE state. A subsequent write of a non-zero value to SL0TS_USED activates the state machine, initializing to SLOT 0, and starts a new WOR "session".
3-2 WOR_PROTOCOL	WAKE-ON-RADIO Protocol Selector Assigns the Wake-On-Radio function to one of the radio's protocol engines. 0: Not assigned 1: Assigned to 802.15.4 2: Assigned to GENERIC_FSK 3: Reserved
1 SCHEDULING_MODE	WAKE-ON-RADIO Scheduling Mode Determines the scheduling mode used by the WOR HW to organize future DSM and RF operations. 1: Constant-Interval Scheduling mode. 0: Timeslot Scheduling mode.
0 WOR_EN	WAKE-ON-RADIO Enable Wake-On-Radio Master Enable. In addition to enabling the block function, this bit gates the clock to the WOR module. This bit should be set, and remain set, whenever WOR is in use. 1: Wake-On-Radio is enabled. 0: Wake-On-Radio is disabled, and clocks to the module are gated off.

55.4.6.2.2.1.3 WAKE-ON-RADIO TIMEOUT REGISTER (TIMEOUT)

Offset

Register	Offset
TIMEOUT	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WOR_SLOT_COUNT								WAKE_ON_NTH_SLOT							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RECEIVE_TIMEOUT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 WOR_SLOT_COUNT	WAKE-ON-RADIO Absolute Slot Count This read-only register indicates the number of slots that have elapsed since SLOTS_USED was set to a non-zero value to activate the WOR state machine. WOR_SLOT_COUNT can be considered an Absolute Slot Number. WOR_SLOT_COUNT counts slots up to a maximum of 255. If the number of elapsed slots exceeds 255, WOR_SLOT_COUNT will remain at 255. WOR_SLOT_COUNT is "sticky" in the sense that it will hold its value after the WOR session has ended by setting SLOTS_USED=0; WOR_SLOT_COUNT will clear to 0 when a new WOR session is started by taking SLOTS_USED from 0 to any non-zero value.
23-16 WAKE_ON_NTH_SLOT	WAKE-ON-RADIO Force Wake On Nth Slot If the WAKE_ON_COMPLETE bit is not set in any descriptor, WOR will not awaken the MCU even after a number of good packets are received. Additionally, if WAKE_ON_TIMEOUT is also not set in any descriptor, WOR will not wake up the MCU under any circumstances, which would not be desirable. This register provides a backup method to awaken the MCU, after *N+1* slots have elapsed, where *N* = WAKE_ON_NTH_SLOT[7:0], regardless of whether or not any good packets have been received, and regardless of the setting of the WAKE_ON_* descriptor bits. Applies to Constant-Interval Mode only. (For Timeslot mode, at least one slot *must* have a WAKE_ON_* bit set in its descriptor).
15-0 RECEIVE_TIMEOUT	WAKE-ON-RADIO Receive Timeout For any RX slot, determines how long RX stays on after the descriptor's START_TIME is reached, in microseconds. A timeout is declared if a good packet is not received before the expiration of this time period (START_TIME + RECEIVE_TIMEOUT + WINDOW_WIDENING). Applies to TIMESLOT mode only.

55.4.6.2.2.1.4 WAKE-ON-RADIO TIMESTAMP 1 (TIMESTAMP1)

Offset

Register	Offset
TIMESTAMP1	8h

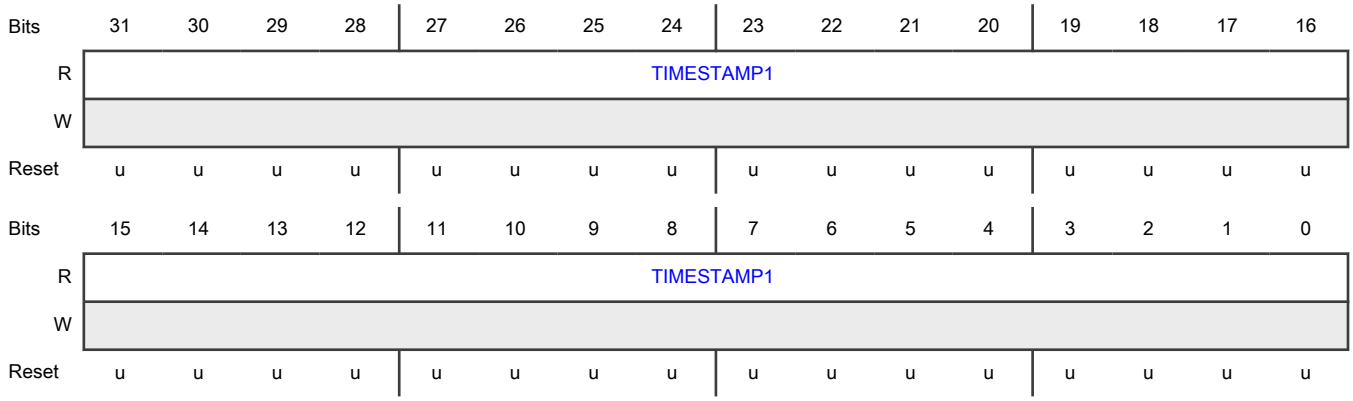
Function

Timestamp captured for a packet received on SLOT 1. During SLOT 1, any valid Access Address or SFD detection will result in the contents of the respective Link Layer's Event Timer being captured into this register at the instant of detection (there is no offset applied). TIMESTAMP1 remains "sticky" until a new AA or SFD detection occurs, during either the same, or a subsequent, SLOT 1.

Note: Timestamp for SLOT 0 is captured into the respective Link Layer's TIMESTAMP register (external to WOR), and can be read from that Link Layer's address space.

Note: If WOR_DEBUG=1, this register instead reads back WOR_ADDBACK[27:0], which is the number of microseconds that the WOR state machine "added back" to the assigned Link Layer's Event Timer, to correct for the amount of time that Link Layer was in DSM (Event Timer frozen). The WOR_ADDBACK readback value applies to the most recently-completed DSM cycle.

Diagram



Fields

Field	Function
31-0 TIMESTAMP1	WAKE-ON-RADIO TIMESTAMP1 Timestamp captured for a packet received on SLOT 1. During SLOT 1, any valid Access Address or SFD detection will result in the contents of the respective Link Layer's Event Timer being captured into this register at the instant of detection (there is no offset applied). TIMESTAMP1 remains "sticky" until a new AA or SFD detection occurs, during either the same, or a subsequent, SLOT 1.

55.4.6.2.2.1.5 WAKE-ON-RADIO TIMESTAMP 2 (TIMESTAMP2)

Offset

Register	Offset
TIMESTAMP2	Ch

Function

Timestamp captured for a packet received on SLOT 2. During SLOT 2, any valid Access Address or SFD detection will result in the contents of the respective Link Layer's Event Timer being captured into this register at the instant of detection (there is no offset applied). TIMESTAMP2 remains "sticky" until a new AA or SFD detection occurs, during either the same, or a subsequent, SLOT 2.

Note: Timestamp for SLOT 0 is captured into the respective Link Layer's TIMESTAMP register (external to WOR), and can be read from that Link Layer's address space.

Note: If WOR_DEBUG=1, this register instead reads back WW_DSM_ADJUST[23:0], which is the accumulated DSM component of the Window Widening adjustment value that is used to offset the start and end of receive operations during the timeslot.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TIMESTAMP2															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMESTAMP2															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 TIMESTAMP2	WAKE-ON-RADIO TIMESTAMP2 Timestamp captured for a packet received on SLOT 2. During SLOT 2, any valid Access Address detection or SFD detection will result in the contents of the respective Link Layer's Event Timer being captured into this register at the instant of detection (there is no offset applied). TIMESTAMP2 remains "sticky" until a new AA or SFD detection occurs, during either the same, or a subsequent, SLOT 2.

55.4.6.2.2.1.6 WAKE-ON-RADIO TIMESTAMP 3 (TIMESTAMP3)

Offset

Register	Offset
TIMESTAMP3	10h

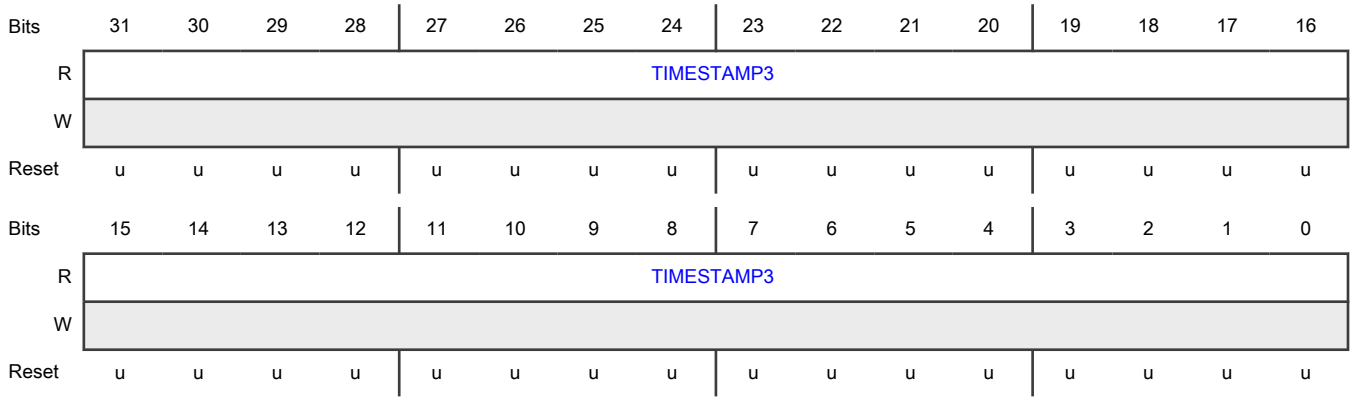
Function

Timestamp captured for a packet received on SLOT 3. During SLOT 3, any valid Access Address detection or SFD detection will result in the contents of the respective Link Layer's Event Timer being captured into this register at the instant of detection (there is no offset applied). TIMESTAMP3 remains "sticky" until a new AA or SFD detection occurs, during either the same, or a subsequent, SLOT 3.

Note: Timestamp for SLOT 0 is captured into the respective Link Layer's TIMESTAMP register (external to WOR), and can be read from that Link Layer's address space.

Note: If WOR_DEBUG=1, this register instead reads back WW_RUN_ADJUST[19:0], which is the accumulated "run time" component of the Window Widening adjustment value that is used to offset the start and end of receive operations during the timeslot.

Diagram



Fields

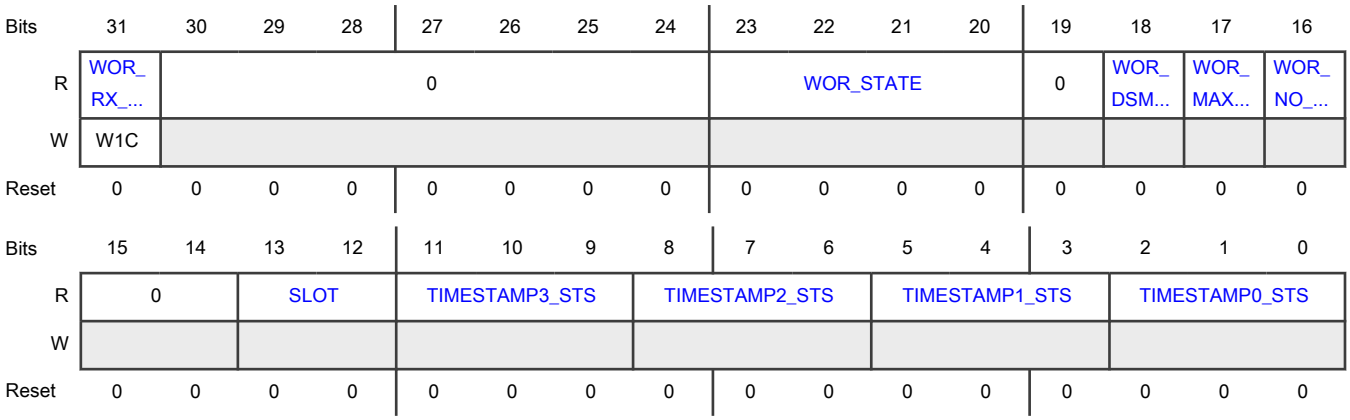
Field	Function
31-0 TIMESTAMP3	WAKE-ON-RADIO TIMESTAMP3 Timestamp captured for a packet received on SLOT 3. During SLOT 3, any valid Access Address detection or SFD detection will result in the contents of the respective Link Layer's Event Timer being captured into this register at the instant of detection (there is no offset applied). TIMESTAMP3 remains "sticky" until a new AA or SFD detection occurs, during either the same, or a subsequent, SLOT 3.

55.4.6.2.2.1.7 WAKE-ON-RADIO STATUS REGISTER (STATUS)

Offset

Register	Offset
STATUS	14h

Diagram



Fields

Field	Function
31 WOR_RX_FAIL_IRQ	WOR RX Fail Interrupt Flag Will assert when the number of missed slot is equal to RX_SLOT_FAIL_THRESH.
30-24 —	Reserved
23-20 WOR_STATE	WAKE-ON-RADIO Current State This read-only register reflects the instantaneous state of the Wake-On-Radio state vector. A listing of the Wake-On-Radio states, along with the associated state vector, is below: • DISABLED (0) • COMP_DURATION1 (1) • COMP_DURATION2 (2) • COMP_DURATION3 (10) • SLEEP_REQ (3) • IN_DSM (4) • COMP_ADDBACK (5) • COMP_WW (6)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• WAIT_SEQ_START (7) • WAIT_SEQ_END (8) • WAIT_RESUME (9)
19 —	Reserved
18 WOR_DSM_EXIT_FLAG	WAKE-ON-RADIO Early DSM Exit Flag This read-only flag indicates that, in the current slot, an event external to the radio has forced an early exit from DSM, before the WOR-computed DSM duration has expired. The WOR_IRQ (WAKE_IRQ) interrupt inside the assigned Link Layer will become set under these conditions, and this flag indicates that the reason for WOR_IRQ was due to an "DSM Early Exit wakeup", as described by the conditions above. The WOR_IRQ will become set, and the MCU awakened, even if the slot descriptor's WAKE_ON_COMPLETE and WAKE_ON_TIMEOUT bits are both set to 0. The WOR state machine is holding in the WAIT_RESUME state, and once software has completed all activities pursuant to the "DSM Early Exit wakeup", software must set WOR_RESUME to advance to the next slot.
17 WOR_MAX_SLOT_FLAG	WAKE-ON-RADIO Maximum Slot Count Reached Flag This read-only flag indicates that, in the current slot, the following conditions have all been met: 1. SCHEDULING_MODE is set to 1 (Constant Interval) 2. WOR_SLOT_COUNT[7:0] (absolute slot number) has matched WAKE_ON_NTH_SLOT[7:0] The WOR_IRQ (WAKE_IRQ) interrupt inside the assigned Link Layer will become set under these conditions, and this flag indicates that the reason for WOR_IRQ was due to a "Maximum Slot Reached wakeup", as described by the conditions above. The WOR_IRQ will become set, and the MCU awakened, even if the slot descriptor's WAKE_ON_COMPLETE and WAKE_ON_TIMEOUT bits are both set to 0. The WOR state machine is holding in the WAIT_RESUME state, and once software has completed all activities pursuant to the "Maximum Slot Reached wakeup", software must set WOR_RESUME to advance to the next slot
16 WOR_NO_RF_FLAG	WAKE-ON-RADIO NO_RF Slot Flag This read-only flag indicates that, in the current slot, the following conditions have all been met: 1. The slot's descriptor FUNCTION is set to "NO_RF" 2. A DSM cycle has just completed (exited) during the slot 3. The slot's descriptor WAKE_ON_COMPLETE bit was set (or the WOR_SLOT_COUNT matched WAKE_ON_NTH_SLOT), requiring an MCU wakeup. The WOR_IRQ (WAKE_IRQ) interrupt inside the assigned Link Layer will become set under these conditions, and this flag indicates that the reason for WOR_IRQ was due to a "NO_RF wakeup", as described by the conditions above. The WOR state machine is holding in the WAIT_RESUME state, and once software has completed all activities pursuant to the "NO_RF wakeup", software must set WOR_RESUME to advance to the next slot
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13-12 SLOT	WAKE-ON-RADIO Current Slot This read-only register reflects the instantaneous state of the WOR's SLOT counter, indicating which SLOT is currently in service by the WOR state machine. Software can determine which of the 4 descriptors is currently in effect based on this register.
11-9 TIMESTAMP3_STS	WAKE-ON-RADIO Timestamp 3 Status This read-only register comprises 3 bits which convey information about the validity of the TIMESTAMP3 register contents: **TIMESTAMP3_STS[0]** -- "AA/SFD Status". If set, this bit indicates at least 1 AA or SFD detection occured during the current, or most recent, SLOT 3 RX operation, and therefore a timestamp, valid or not, was captured into the TIMESTAMP3 register. **TIMESTAMP3_STS[1]** -- "CRC Status". If set, this bit indicates a packet was received which passed CRC verification during the current, or most recent, SLOT 3 RX operation **TIMESTAMP3_STS[2]** -- "Filter Status". If set, this bit indicates a packet was received which passed Packet Filtering rules during the current, or most recent, SLOT 3 RX operation
8-6 TIMESTAMP2_STS	WAKE-ON-RADIO Timestamp 2 Status This read-only register comprises 3 bits which convey information about the validity of the TIMESTAMP2 register contents: **TIMESTAMP2_STS[0]** -- "AA/SFD Status". If set, this bit indicates at least 1 AA or SFD detection occured during the current, or most recent, SLOT 2 RX operation, and therefore a timestamp, valid or not, was captured into the TIMESTAMP2 register. **TIMESTAMP2_STS[1]** -- "CRC Status". If set, this bit indicates a packet was received which passed CRC verification during the current, or most recent, SLOT 2 RX operation **TIMESTAMP2_STS[2]** -- "Filter Status". If set, this bit indicates a packet was received which passed Packet Filtering rules during the current, or most recent, SLOT 2 RX operation
5-3 TIMESTAMP1_STS	WAKE-ON-RADIO Timestamp 1 Status This read-only register comprises 3 bits which convey information about the validity of the TIMESTAMP1 register contents: **TIMESTAMP1_STS[0]** -- "AA/SFD Status". If set, this bit indicates at least 1 AA or SFD detection occured during the current, or most recent, SLOT 1 RX operation, and therefore a timestamp, valid or not, was captured into the TIMESTAMP1 register. **TIMESTAMP1_STS[1]** -- "CRC Status". If set, this bit indicates a packet was received which passed CRC verification during the current, or most recent, SLOT 1 RX operation **TIMESTAMP1_STS[2]** -- "Filter Status". If set, this bit indicates a packet was received which passed Packet Filtering rules during the current, or most recent, SLOT 1 RX operation
2-0	WAKE-ON-RADIO Timestamp 0 Status This read-only register comprises 3 bits which convey information about the validity of the TIMESTAMP0 register contents:

Table continues on the next page...

Table continued from the previous page...

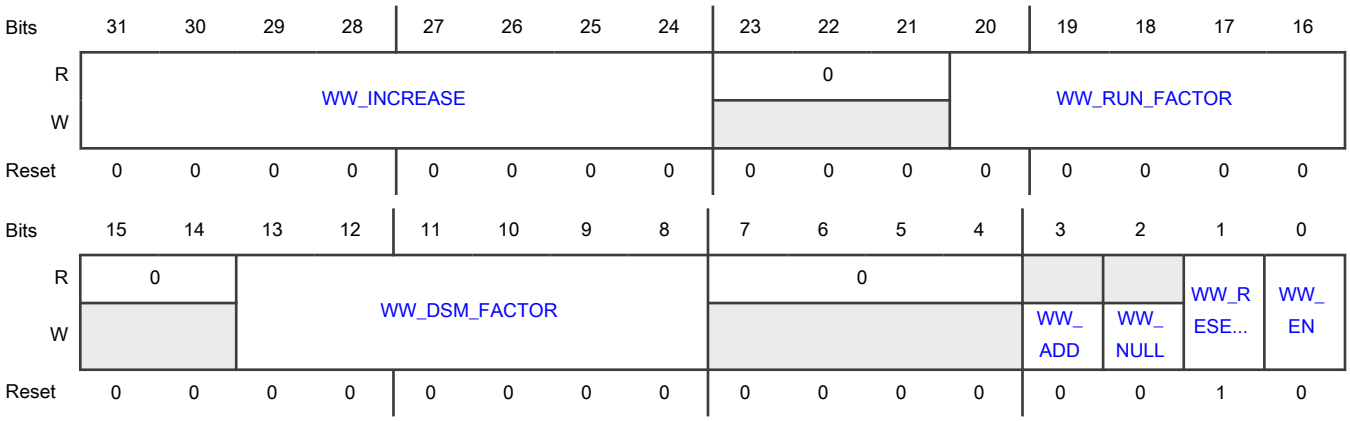
Field	Function
TIMESTAMP0_STS	**TIMESTAMP0_STS[0]** -- "AA/SFD Status". If set, this bit indicates at least 1 AA or SFD detection occured during the current, or most recent, SLOT 0 RX operation, and therefore a timestamp, valid or not, was captured into the TIMESTAMP0 register. **TIMESTAMP0_STS[1]** -- "CRC Status". If set, this bit indicates a packet was received which passed CRC verification during the current, or most recent, SLOT 0 RX operation **TIMESTAMP0_STS[2]** -- "Filter Status". If set, this bit indicates a packet was received which passed Packet Filtering rules during the current, or most recent, SLOT 0 RX operation **Note:** TIMESTAMP0 register (timestamp for SLOT 0) is captured into the assigned Link Layer's TIMESTAMP register (external to WOR), and can be read from that Link Layer's address space. However TIMESTAMP0_STS[2:0], which applies to that register, resides in WOR address space.

55.4.6.2.2.1.8 WINDOW-WIDENING CONTROL REGISTER (WW_CTRL)

Offset

Register	Offset
WW_CTRL	18h

Diagram



Fields

Field	Function
31-24 WW_INCREASE	Window-widening Manual Increase Amount This register allows software to increase (add to) the HW-computed window-widening adjustment. This incremental amount is added whenever a 1 is written to the WW_ADD register bit. The units of WW_INCREASE[7:0] are microseconds. It is an unsigned (positive only) addition. This is a one-time only addition (whever WW_ADD is set), and the window-widening adjustment value continues to accumulate

Table continues on the next page...

Table continued from the previous page...

Field	Function
	after the increment. The capability to add to the HW-computed adjustment is intended for contingency or debug purposes only, as the HW-computed value should be accurate.
23-21 —	Reserved
20-16 WW_RUN_FACTOR	Window-widening Runtime Factor Determines the rate of Window-widening that applies during RUNTIME (Reference Oscillator On). This setting should be based on the combined Sleep Clock Accuracy (SCA) of the Master + Slave. Each LSB is worth 64ppm of SCA drift. Valid values are 0 -> 16. The lower this factor, the greater the power savings.
15-14 —	Reserved
13-8 WW_DSM_FACTOR	Window-widening DSM Factor WW_DSM_FACTOR[5:0]: Determines the rate of Window-widening that applies during DSM (Reference Oscillator Off). This setting should be based on the combined Sleep Clock Accuracy (SCA) of the Master + Slave. Each LSB is worth 16ppm of combined SCA drift. Valid values are 0 -> 63. The lower this factor, the greater the power savings.
7-4 —	Reserved
3 WW_ADD	Window-widening Add Command This bit allows software to increase (add to) the HW-computed window-widening adjustment. The incremental amount is determined by the WW_INCREASE[7:0] register. The capability to add to the HW-computed adjustment is intended for contingency or debug purposes only, as the HW-computed value should be accurate. This bit is write-only. Readback value is always 0. 1: Increase the HW-computed window-widening adjustment by amount WW_INCREASE[7:0]. 0: No effect
2 WW_NULL	Window-widening Null Command This bit allows software to reset the accumulated window-widening adjustment to 0. This bit is write-only. Readback value is always 0. 1: Reset the accumulated window-widening adjustment to 0. 0: No effect.
1 WW_RESET_ON_RX	Window-widening Reset on Received Good Packet Enable a good packet reception to reset the accumulated window-widening adjustment to 0. A good packet is one that passes CRC verification, and passes packet filtering rules. Filtering rules vary by protocol. 1: Allow good packet reception to reset window-widening adjustment. This is the default. 0: Block good packet reception from resetting window-widening adjustment.

Table continues on the next page...

Table continued from the previous page...

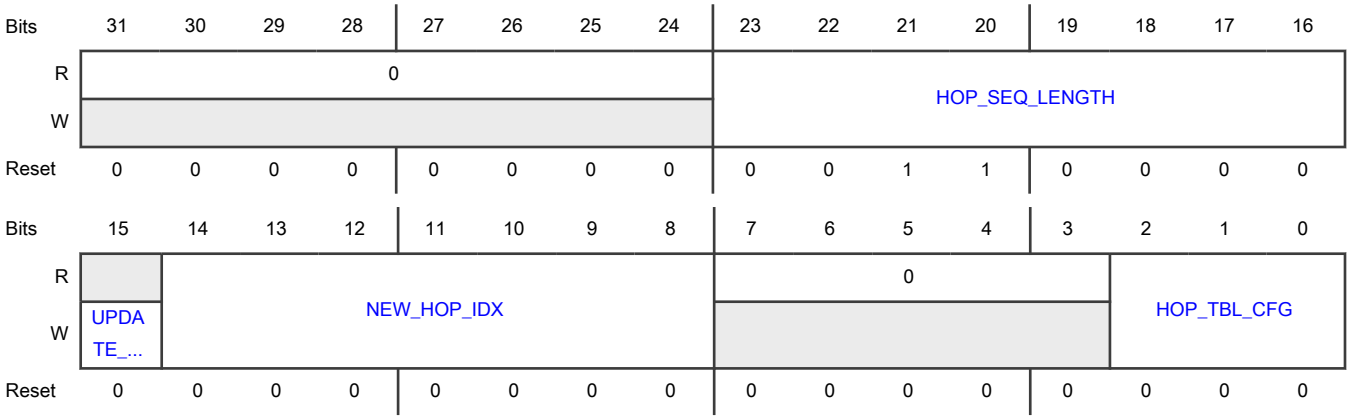
Field	Function
0 WW_EN	Window-widening Enable Master Enable for Window Widening. Typically, only slave devices employ window widening. 1: Window Widening is enabled. 0: Window Widening is disabled.

55.4.6.2.2.1.9 FREQUENCY HOP CONTROL REGISTER (HOP_CTRL)

Offset

Register	Offset
HOP_CTRL	1Ch

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 HOP_SEQ_LENGTH	New Hop Table Index The HOP_SEQ_LENGTH[7:0] register determines the number of entries in the WOR Hop Table, regardless of how the table is configured by HOP_TBL_CFG[2:0]. During WOR operation, once the Hop Table Index reaches HOP_SEQ_LENGTH - 1, the next Hop Table Index will wrap to 0. HOP_SEQ_LENGTH must not exceed the table size specified by HOP_TBL_CFG (128 entries).
15	Update Hop Table Index

Table continues on the next page...

Table continued from the previous page...

Field	Function
UPDATE_HOP_IDX	Software can jump to a new index (read pointer address) of the WOR Hop Table at any time by writing the desired index to the NEW_HOP_IDX[6:0] register, and setting the UPDATE_HOP_IDX bit to 1. This bit is write-only. Readback value is always 0. 1: Jump to new Hop Table Index pointed to by NEW_HOP_IDX[6:0] 0: No effect
14-8 NEW_HOP_IDX	New Hop Table Index Software can jump to a new index (read pointer address) of the WOR Hop Table at any time by writing the desired index to this register, and setting the UPDATE_HOP_IDX bit.
7-3 —	Reserved
2-0 HOP_TBL_CFG	Hop Table Configuration For maximum flexibility in Frequency Hop scheduling, 4 hop table configurations are defined. The configurations affect: 1. The source of the RF channel information (Link Layer register or WOR Hop Table) 2. How the RF channel information is organized in the WOR Hop Table 3. How the hop frequency word is interpreted by the PLL Digital Block Source of RF channel information: 0: Link Layer CHANNEL_NUM register (legacy configuration). The Link Layer is the one assigned by WOR_PROTOCOL register. 1: WOR Hop Table 2: WOR Hop Table 3: WOR Hop Table 4-7: Reserved Hop Table Organization: 0: N/A 1: Table size is 128 entries. Two 7-bit CHANNEL_NUM fields are packed into a 16-bit hop frequency word for RAM storage 2-3: Table size is 128 entries. Each 16-bit hop frequency word in RAM contains a specification for a single RF channel 4-7: Reserved PLL Digital Interpretation of Hop Frequency Word: 0: Lower 7 bits of hop frequency word contain a 7-bit mapped channel (legacy format). 1: Lower 7 bits of hop frequency word contain a 7-bit mapped channel (legacy format). 2-3: See Chapter *PLL Digital*

Table continues on the next page...

Table continued from the previous page...

Field	Function
	4-7: Reserved

55.4.6.2.2.1.10 SLOT 0 DESCRIPTOR (LSB) (SLOT0_DESC0)

Offset

Register	Offset
SLOT0_DESC0	20h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SLOT0_DESC0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLOT0_DESC0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

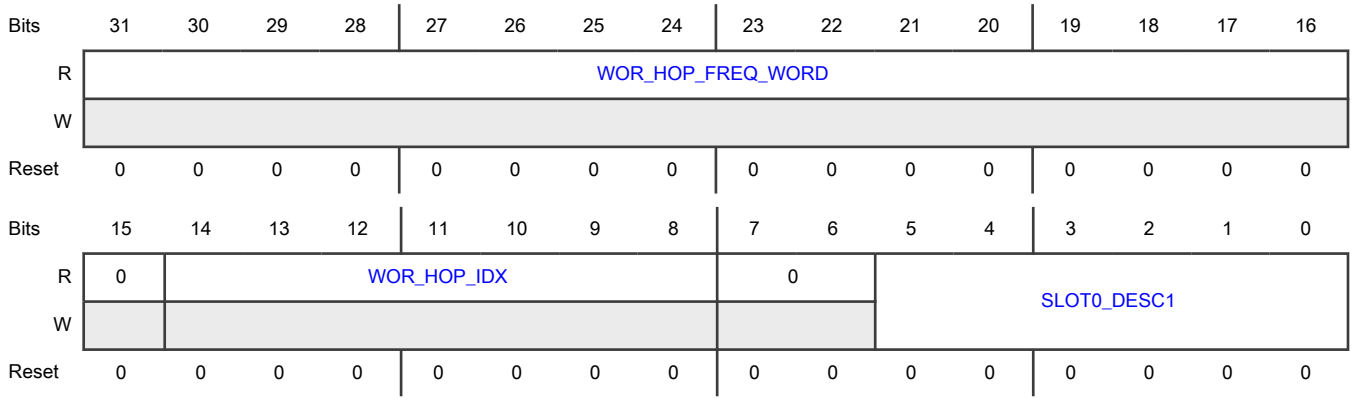
Field	Function
31-0 SLOT0_DESC0	Slot 0 Descriptor (LSB's) This register represents the 32 LSB's of the WOR descriptor for Slot 0

55.4.6.2.2.1.11 SLOT 0 DESCRIPTOR (MSB) (SLOT0_DESC1)

Offset

Register	Offset
SLOT0_DESC1	24h

Diagram



Fields

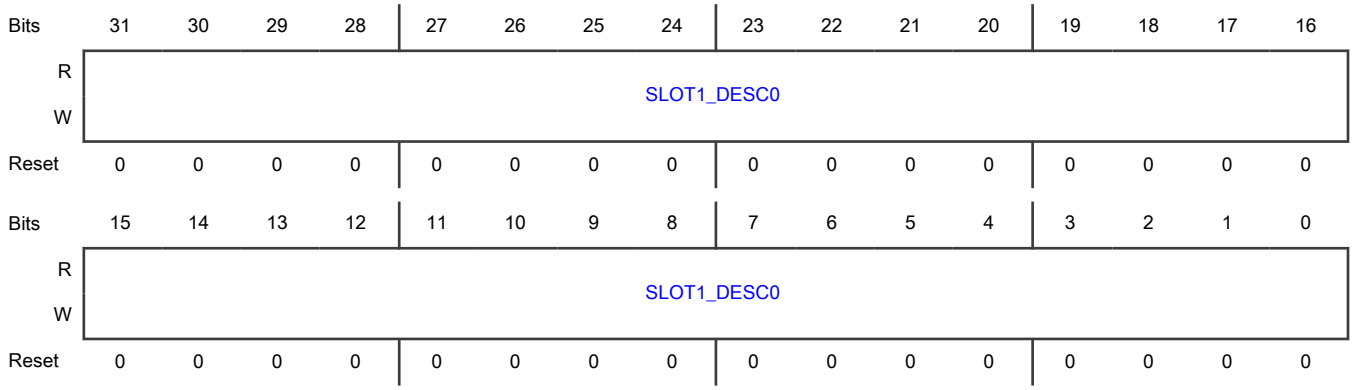
Field	Function
31-16 WOR_HOP_FREQ_WORD	Current Hop Frequency Word This read-only register reflects the current Hop Frequency Word, formulated by WOR and transmitted to the PLL Digital Block
15 —	Reserved
14-8 WOR_HOP_IDX	Current Hop Table Index This read-only register reflects the current Hop Table Index (read address pointer) into the WOR Hop Table
7-6 —	Reserved
5-0 SLOT0_DESC1	Slot 0 Descriptor (MSB's) This register represents the 6 MSB's of the WOR descriptor for Slot 0

55.4.6.2.2.1.12 SLOT 1 DESCRIPTOR (LSB) (SLOT1_DESC0)

Offset

Register	Offset
SLOT1_DESC0	28h

Diagram



Fields

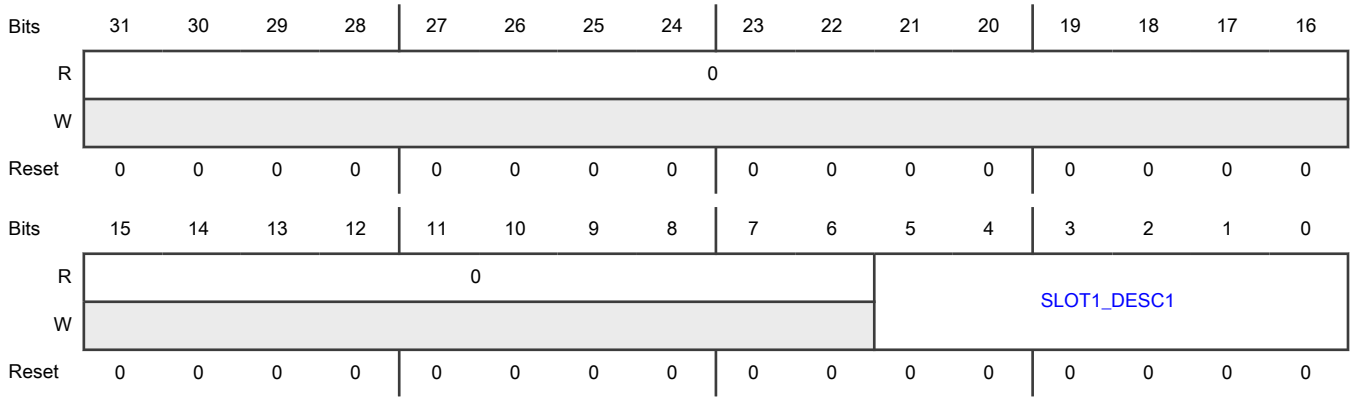
Field	Function
31-0 SLOT1_DESC0	Slot 1 Descriptor (LSB's) This register represents the 32 LSB's of the WOR descriptor for Slot 1

55.4.6.2.2.1.13 SLOT 1 DESCRIPTOR (MSB) (SLOT1_DESC1)

Offset

Register	Offset
SLOT1_DESC1	2Ch

Diagram



Fields

Field	Function
31-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 SLOT1_DESC1	Slot 1 Descriptor (MSB's) This register represents the 6 MSB's of the WOR descriptor for Slot 1

55.4.6.2.2.1.14 SLOT 2 DESCRIPTOR (LSB) (SLOT2_DESC0)

Offset

Register	Offset
SLOT2_DESC0	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SLOT2_DESC0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLOT2_DESC0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

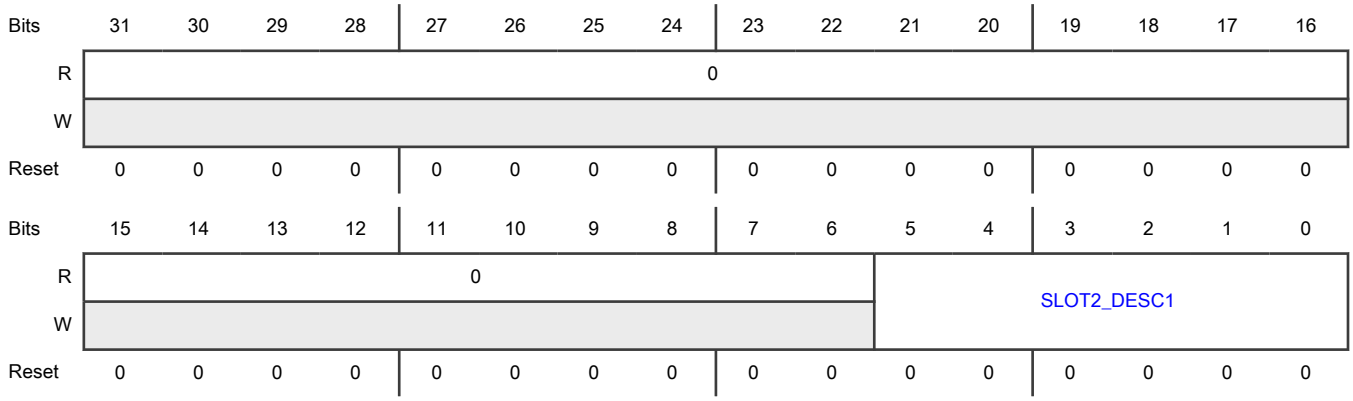
Field	Function
31-0 SLOT2_DESC0	Slot 2 Descriptor (LSB's) This register represents the 32 LSB's of the WOR descriptor for Slot 2

55.4.6.2.2.1.15 SLOT 2 DESCRIPTOR (MSB) (SLOT2_DESC1)

Offset

Register	Offset
SLOT2_DESC1	34h

Diagram



Fields

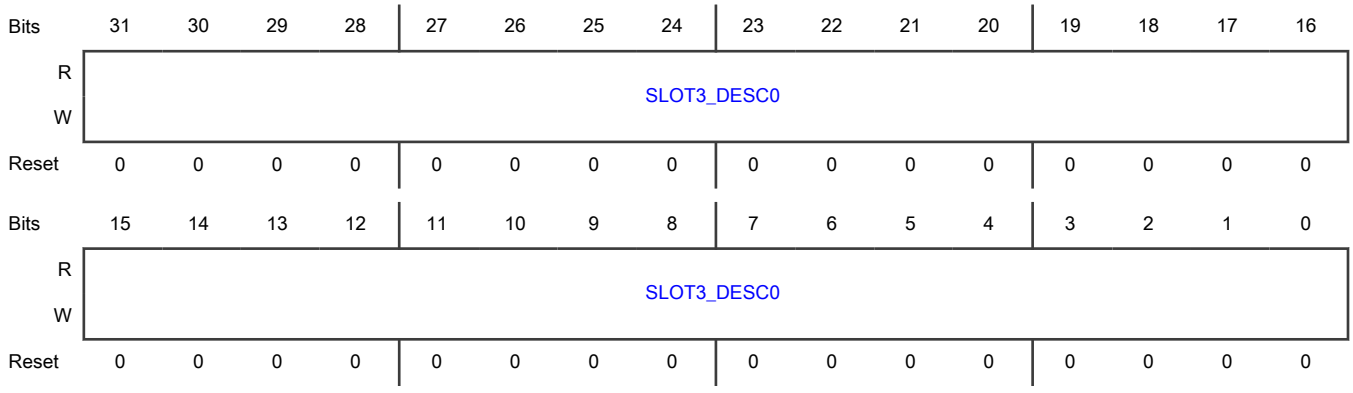
Field	Function
31-6 —	Reserved
5-0 SLOT2_DESC1	Slot 2 Descriptor (MSB's) This register represents the 6 MSB's of the WOR descriptor for Slot 2

55.4.6.2.2.1.16 SLOT 3 DESCRIPTOR (LSB) (SLOT3_DESC0)

Offset

Register	Offset
SLOT3_DESC0	38h

Diagram



Fields

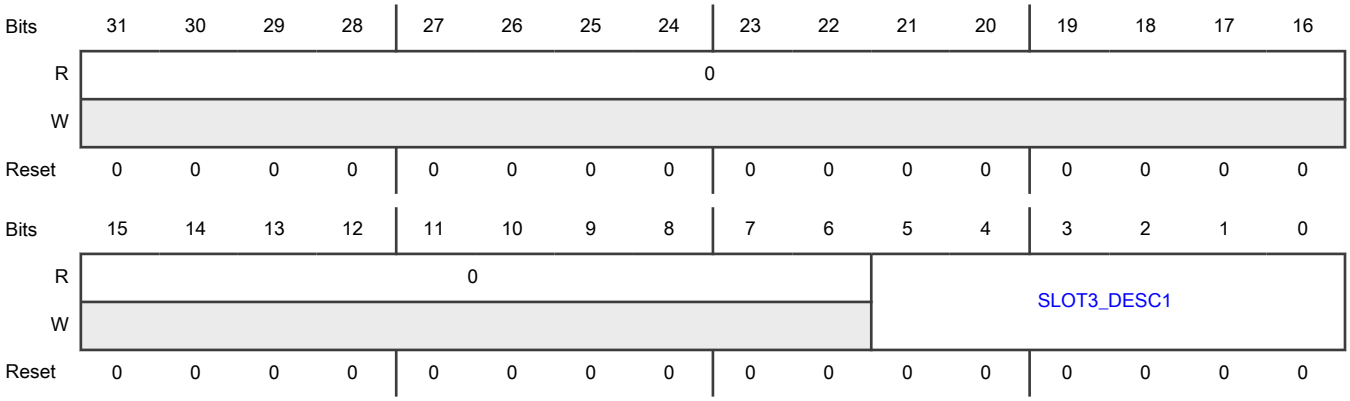
Field	Function
31-0 SLOT3_DESC0	Slot 3 Descriptor (LSB's) This register represents the 32 LSB's of the WOR descriptor for Slot 3

55.4.6.2.2.1.17 SLOT 3 DESCRIPTOR (MSB) (SLOT3_DESC1)

Offset

Register	Offset
SLOT3_DESC1	3Ch

Diagram



Fields

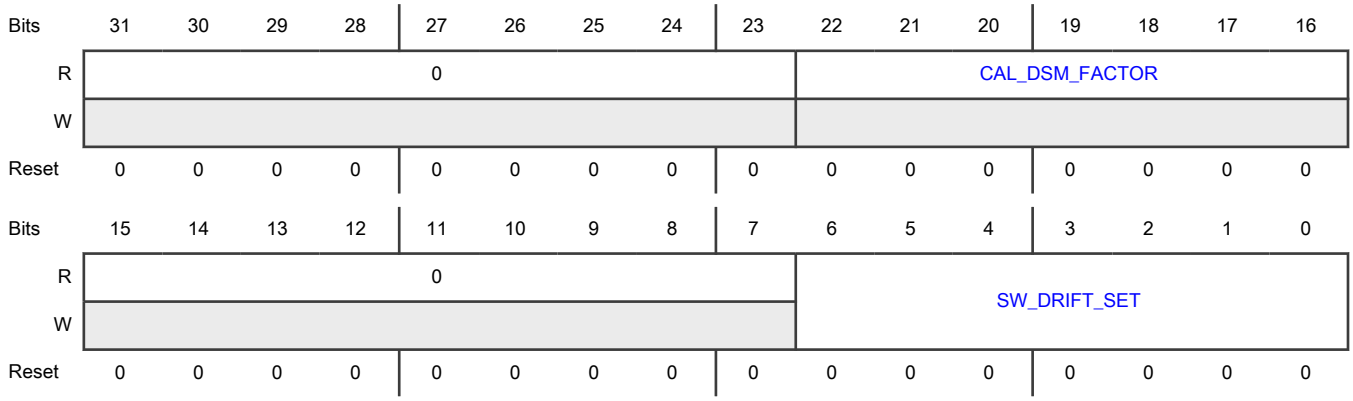
Field	Function
31-6 —	Reserved
5-0 SLOT3_DESC1	Slot 3 Descriptor (MSB's) This register represents the 6 MSB's of the WOR descriptor for Slot 3

55.4.6.2.2.1.18 Auto Drift Calculation Register 1 (AUTO_DRIFT1)

Offset

Register	Offset
AUTO_DRIFT1	40h

Diagram



Fields

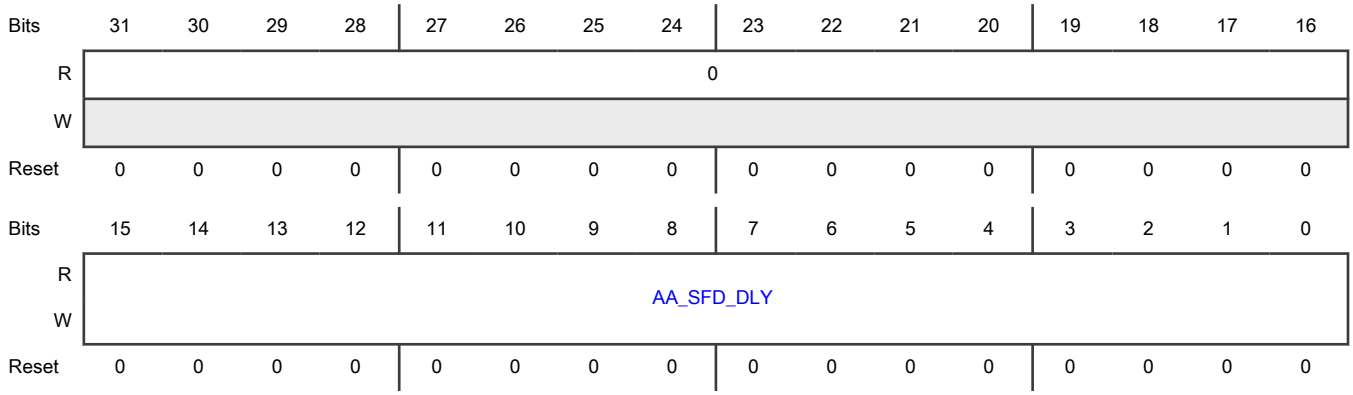
Field	Function
31-23 —	Reserved
22-16 CAL_DSM_FACTOR	Hardware calculated drift. This is a two's complement value. Each LSB is worth 16ppm. Read only.
15-7 —	Reserved
6-0 SW_DRIFT_SET	Software calculated drift. This is a two's complement value. Each LSB is worth 16ppm.

55.4.6.2.2.1.19 Auto Drift Calculation Register 2 (AUTO_DRIFT2)

Offset

Register	Offset
AUTO_DRIFT2	44h

Diagram



Fields

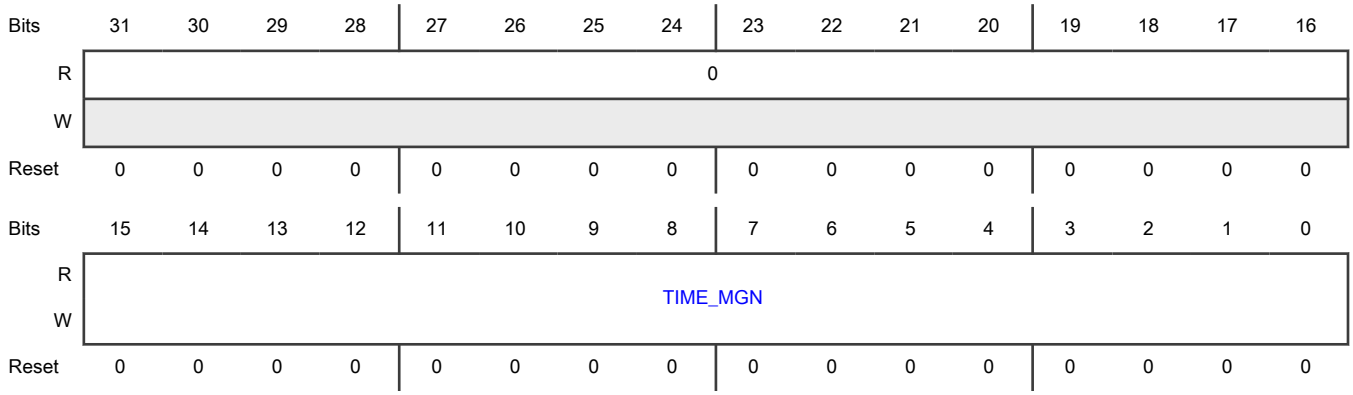
Field	Function
31-16 —	Reserved
15-0 AA_SFD_DLY	The time duration of Preamble and Sync Address plus the RX warm up duration.

55.4.6.2.2.1.20 Auto Drift Calculation Register 3 (AUTO_DRIFT3)

Offset

Register	Offset
AUTO_DRIFT3	48h

Diagram



Fields

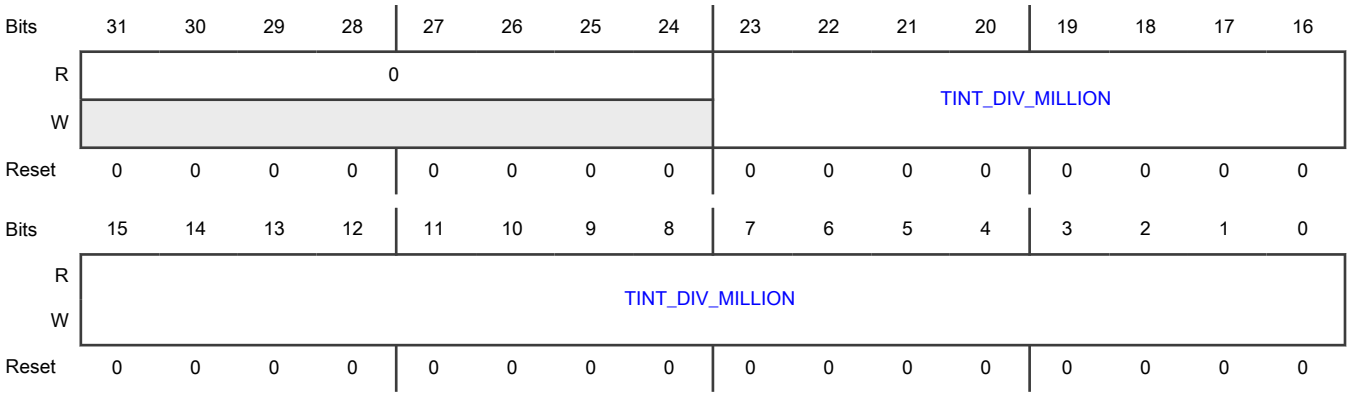
Field	Function
31-16 —	Reserved
15-0 TIME_MGN	The time margin applied to the start time and timeout.

55.4.6.2.2.1.21 Auto Drift Calculation Register 4 (AUTO_DRIFT4)

Offset

Register	Offset
AUTO_DRIFT4	4Ch

Diagram



Fields

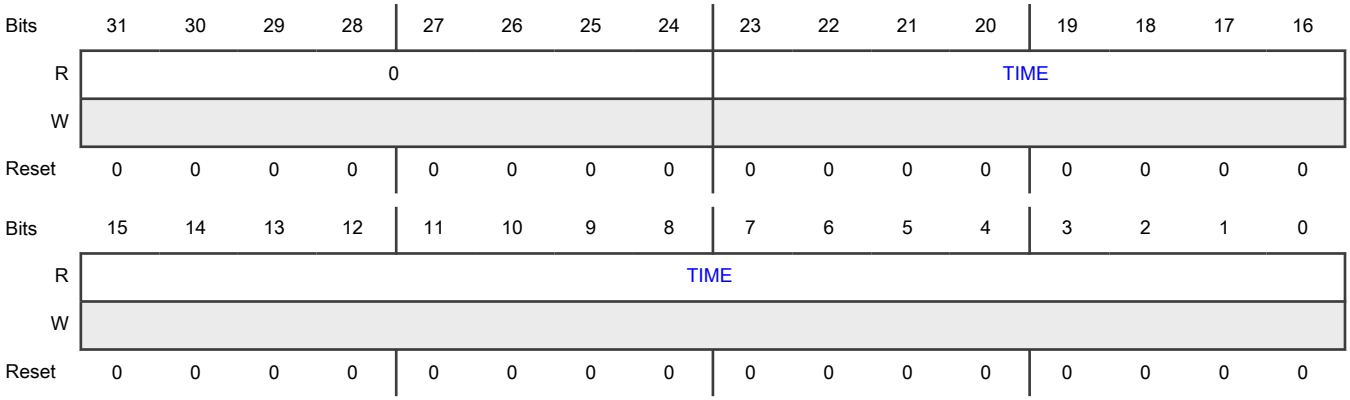
Field	Function
31-24 —	Reserved
23-0 TINT_DIV_MILLION	User programed value to help the hardware to calculate the drift.The value is (2^17 * 1,000,000 / WOR_EVENT_INTERVAL / 16)(rounded)

55.4.6.2.2.1.22 Timer Count (TIME)

Offset

Register	Offset
TIME	98h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 TIME	Current 32kHz reference clock time (used by the MAN low power controller).

55.4.6.2.2.1.23 MAN Low Power Entry Time Captured (ENTER_TIME_CAPT)

Offset

Register	Offset
ENTER_TIME_CAPT	9Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								ENTER_TIME_CAPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENTER_TIME_CAPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 ENTER_TIME_CAPT	Captured Timer count for MAN entry to low power. The value of ENTER_TIME is captured when the man_sleep_req asserts.

55.4.6.2.2.1.24 MAN Low Power Wakeup Time Captured (WKUP_TIME_CAPT)

Offset

Register	Offset
WKUP_TIME_CAPT	A0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								WKUP_TIME_CAPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WKUP_TIME_CAPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

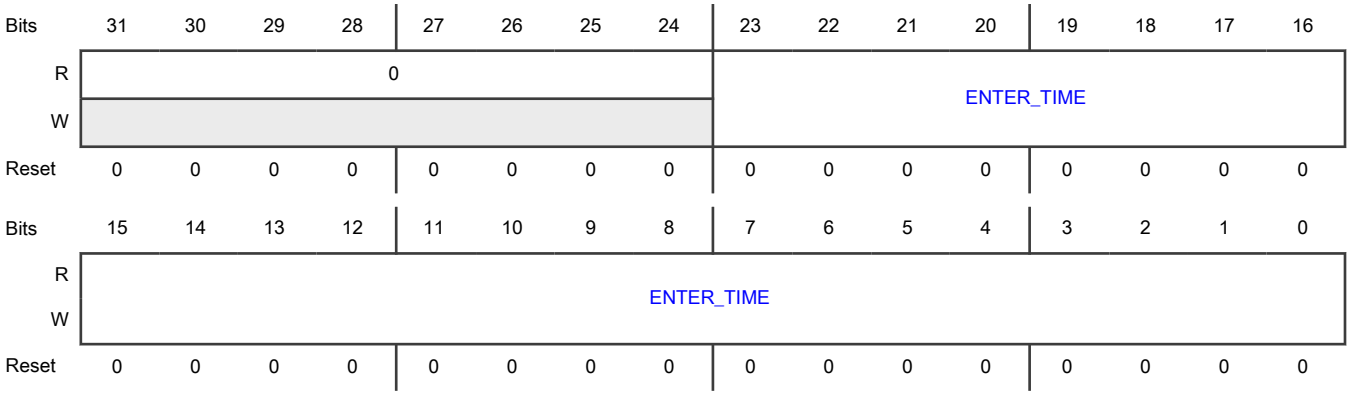
Field	Function
31-24 —	Reserved
23-0 WKUP_TIME_CAPT	Captured Timer count for MAN exit from low power. The value of WKUP_TIME is captured when the man_sleep_req asserts. However, if RFMC's MAN_WKUP bit is set, the value is updated to reflect the early exit time.

55.4.6.2.2.1.25 MAN Low Power Entry Time Stamp (ENTER_TIME)

Offset

Register	Offset
ENTER_TIME	A4h

Diagram



Fields

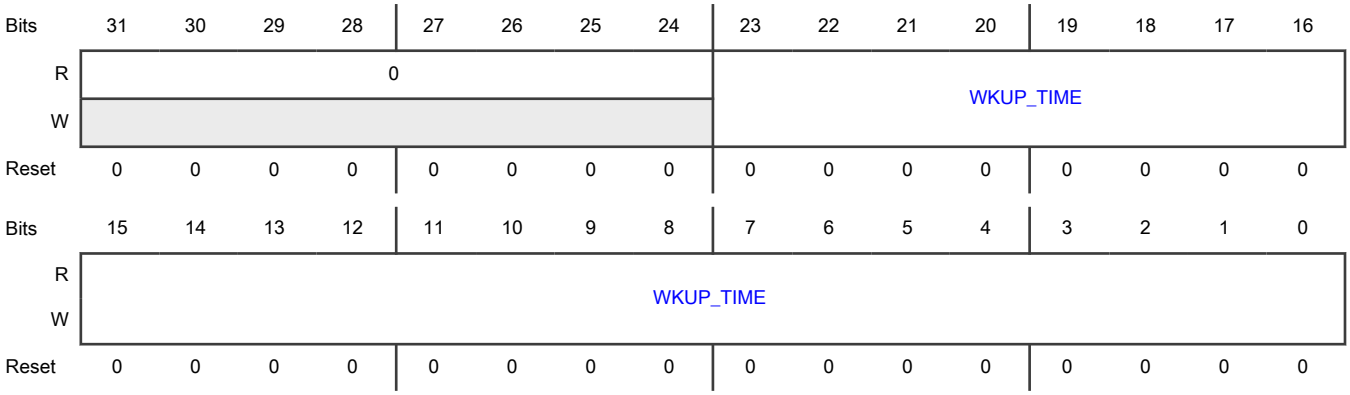
Field	Function
31-24 —	Reserved
23-0 ENTER_TIME	Timer count at which to initiate low power entry by the MAN.

55.4.6.2.2.1.26 MAN Low Power Wakeup Time Stamp (WKUP_TIME)

Offset

Register	Offset
WKUP_TIME	A8h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 WKUP_TIME	Timer count at which to complete low power wakeup by the MAN.

55.4.6.2.3 Functional Description

55.4.6.2.3.1 Assign To Protocol

The 2.4GHz radio supports multiple protocols, each with its own protocol engine. All protocols can benefit from scheduling in advance several RF operations and DSM cycles, with the radio conducting the successive operations autonomously, allowing the MCU to remain in low power state for the duration. This is the concept behind Wake-On-Radio (WOR).

The WOR function can be assigned to any one of the following protocol(s): 802.15.4, GENERIC_FSK. (Bluetooth LE has its own, internal DSM and wake-on-radio control scheme, as well as hardware-controlled frequency hopping, built into the IP, so it does not interact with the WOR Scheduler).

Assigning WOR to a protocol is done by way of the WOR_PROTOCOL register, as shown in the following table:

WOR_PROTOCOL[2:0]	ASSIGNED PROTOCOL ENGINE
0	Not Assigned

Table continues on the next page...

Table continued from the previous page...

WOR_PROTOCOL[2:0]	ASSIGNED PROTOCOL ENGINE
1	802.15.4
2	GENERIC_FSK
3	Reserved

Assigning to protocol means that the assigned Link Layer's WOR interface signals are routed, exclusively, to and from the WOR state machine. WOR interface signals include the event timers, sequence start- and stop-command signals, interrupt inhibits, and event timer addback commands. These interface signals are covered in more details in later sections.

The following diagram illustrates how 1 of the 3 WOR-capable Link Layers is assigned to the WOR function:

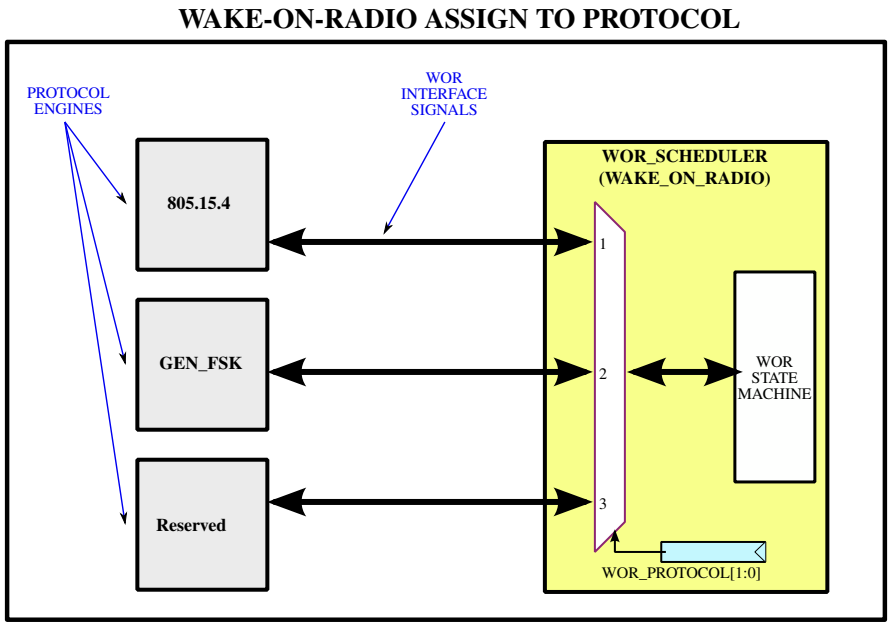


Figure 222. WAKE-ON-RADIO ASSIGN TO PROTOCOL

In a multi-protocol setting, assigning a protocol to Wake-On-Radio does not mean that unassigned protocols are incapable of DSM. There is a back-up method, Manual DSM, which uses software-driven DSM entry and exit. In a multi-protocol setting, any 2 of the radio's protocols can be assigned to different DSM control scheme, with the control schemes being Wake-On-Radio, Manual DSM, and Bluetooth LE DSM.

55.4.6.2.3.2 Scheduling Modes

For maximum flexibility in scheduling future RF operations and Deep Sleep cycles, Wake-On-Radio (WOR) defines 2 scheduling modes: Timeslot Mode, and Constant Interval Mode. These modes are described in the sections below.

Timeslot Mode

Timeslot mode is desirable when the schedule for RF transactions is sporadic, varies over time, or is otherwise non-periodic. Timeslot mode is based loosely on the TSCH (Timeslotted Channel Hopping) concept, first defined in IEEE Std 802.15.4-2015. Key principles of TSCH are adopted by WOR, but far short of full adherence.

In TSCH, time is divided into slots of fixed duration. Slots are grouped into slotframes, containing fixed number of slots. Once a TSCH PAN (network) is established, slotframes will repeat forever, back-to-back. Devices can be assigned to 1 or more slots

within a slotframe. For any given device on any given slot, the assigned slot can be TX or RX. One device can be assigned to transmit in a given slot; another device can be assigned to receive in the same slot. Assigning Device A to Transmit, and Device B to Receive, in the same slot, creates a link. Links can be dedicated (one TX and one RX device assigned). Or, links can be shared (multiple TX devices, using CSMA, and multiple RX devices). Typically, each slot uses a different hop frequency.

Slotframes are TDMA structures, with time on one axis and hop frequency on the other. The diagram below shows one "row" of a typical slotframe, with links established for Devices A, B, and C, all on the same hop frequency. This slotframe consists of 3 timeslots: TS0, TS1, and TS2.

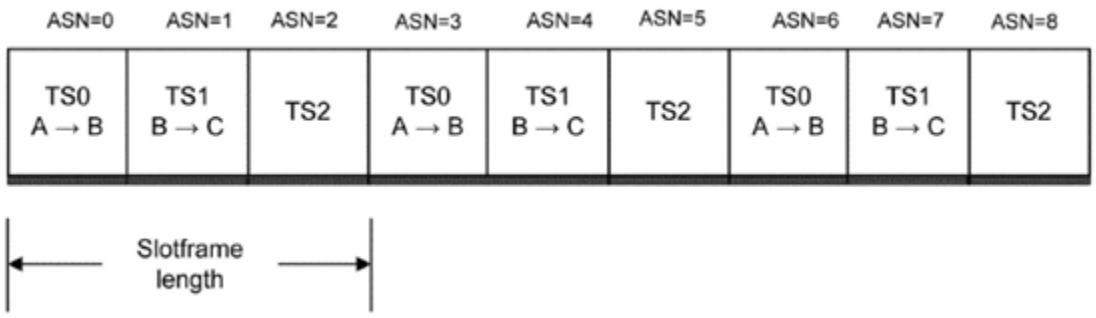


Figure 223. IEEE Std 802.15.4 TSCH 1-D SLOTFRAME

Another diagram below, shows a slotframe of 2 dimensions. The time axis has 100 slots, and the frequency axis has 4 indices into a table of available hop frequencies. Devices A, B, C, and D are assigned to various slots, and links are created amongst them.

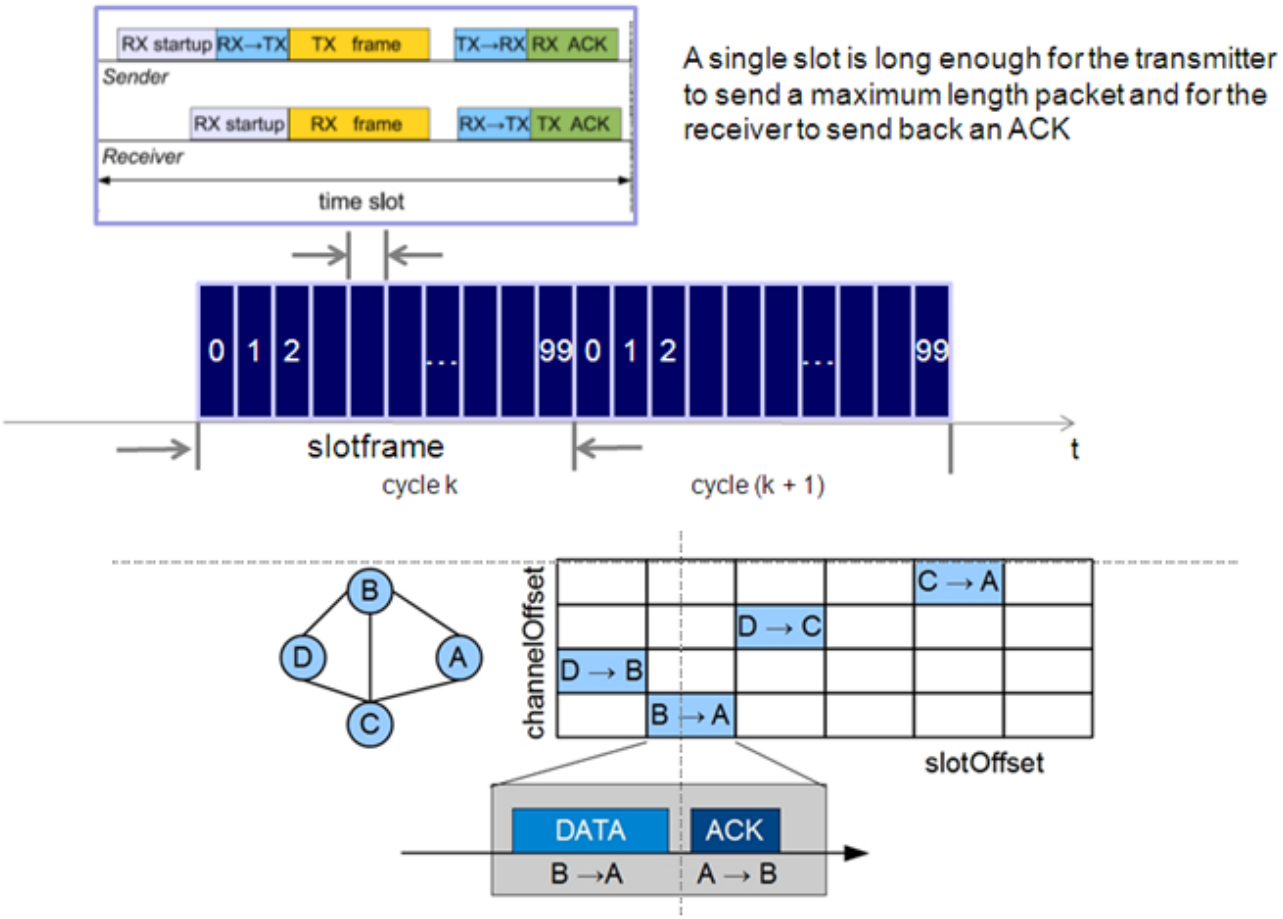


Figure 224. IEEE Std 802.15.4 TSCH 2-D SLOTFRAME

When RF operations are to be scheduled using Wake-On-Radio in a TIMESLOT context, the start times for the next *N* operations are programmed into the descriptor table to assign them to slots. One field of each slot descriptor, START_TIME, holds this information for WOR. See Section Descriptor Format for more information on the descriptor fields, and how they are packed into the descriptor. *N* is the number of slots used, and can be any integer from 2 to 4. The register SLOTS_USED is programmed with this value. The start times programmed into the descriptors, are relative to the Event Timer of the assigned Link Layer. WOR software will read the Event Timer, add to this value the desired time-to-next-event, and write the sum into the descriptor START_TIME. WOR hardware will monitor the assigned Link Layer's Event Timer; when it matches the START_TIME field of the current slot's descriptor, WOR hardware will initiate an RF sequence. The type of sequence is determined by the descriptor's FUNCTION field, and can be TX or RX or RT or TR. The RF operation will be allowed to continue, until it either self-terminates, or a timeout is reached. The timeout period for TIMESLOT scheduling mode is determined by the RECEIVE_TIMEOUT[15:0] register. The units of this register are microseconds. If a timeout is reached, i.e.,

EVENT_TMR = START_TIME + RECEIVE_TIMEOUT,

WOR hardware will terminate the sequence. Timeout applies to RX sequences only. RX sequences may result in no packet being received, a packet being received which is not addressed to the device, or a packet which is received corruptly. Any of these scenarios will result in a timeout and a WOR shutdown of the sequence. A correctly received packet addressed to the device will self-terminate the RX sequence before the timeout is reached. Timeout is not relavent to TX sequences, since these always self-terminate. The following diagram depicts how the WOR hardware processes the descriptor table, the assigned Event Timer, and the RECEIVE_TIMEOUT register, to start and terminate an RF operation.

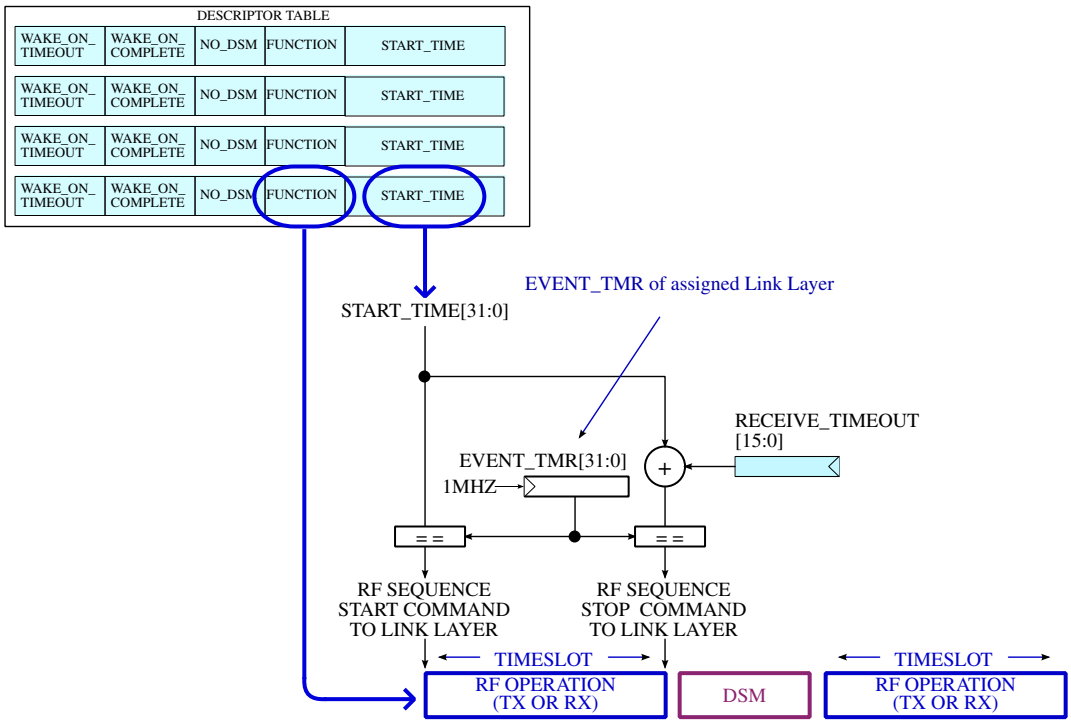


Figure 225. EVENT SCHEDULING IN TIMESTLOT SCHEDULING MODE

If the assigned protocol engine is 802.15.4 or GENERIC_FSK, an autosequence is available which is a TX sequence preceded by a CCA (Clear Channel Assessment). If the CCA results in a busy channel, the autosequence terminates without transmission; otherwise the transmission proceeds. This autosequence can be used in WOR Timeslot mode by appropriately configuring the 802.15.4 or GENERIC_FSK Link Layer.

In any timeslot, if the assigned FUNCTION is RX or RT, a TIMESTAMP can be captured to record the instant the packet's Access Address or SFD arrives at the Link Layer packet processor. There are 4 TIMESTAMP registers, once for each slot; hence, timestamps for the last 4 slots during which packets are received, can be recorded. See Section Timestamps for more details.

To execute the function RT, Link Layer should be configured with **AUTOACK** set. When Event Timer match, WOR will assert **wor_rx_start_ext signal** to Link Layer to start the RX sequence. After Link Layer completes the Auto-ACK sequence, it will send the signal **tx_irq_trig** to WOR to advance to the next state. If the RX operation timeout occurs, either nominal or delayed, WOR will assert **wor_seq_stop_ext** to the assigned Link Layer, causing an immediate abort of the RX sequence. RT function is effected by the window widening feature. A TIMESTAMP is captured in the RT sequence.

To execute the function TR, when Event Timer match, WOR will assert **wor_tx_start_ext** signal as well as **wor_function_is_tr** signal to Link Layer to start the TR sequence. After Link Layer completes the TR sequence, it will send signal **rx_irq_trig** to WOR to advance to the next state. If the RX operation timeout occurs, either nominal or delayed, WOR will **assert wor_seq_stop_ext** to the assigned Link Layer, causing an immediate abort of the RX sequence. TR function is not effected by the window widening feature. A TIMESTAMP is captured in the TR sequence.

Typically, for RX timeslots, it is desirable to wake the MCU on good packets only. Thus, the slot descriptor's WAKE_ON_COMPLETE field should be set to 1. If it is desired to wake the MCU on no-packet, or on incorrecty-received packets, the WAKE_ON_TIMEOUT descriptor field can be set to 1. If WAKE_ON_TIMEOUT is set to 1 on any slot's descriptor, the WAKE_ON_COMPLETE field is *implied* to be set to 1 as well, regardless of its actual programmed value.

Typically, for TX timeslots, it is desirable to wake the MCU after each transmission, to service the Packet Buffer. In such scenarios, WAKE_ON_COMPLETE should be set to 1. If it is required to transmit the same packet repeatedly, across multiple slots, such that the Packet Buffer does not require regular service, WAKE_ON_COMPLETE can be set to 0, to skip the MCU wakeup, on any slot.

In Timeslot Scheduling mode, the MCU *must* be awakened at least once every *N* slots, to refresh the descriptor table (or at least, to refresh the START_TIME's), with *N* being the value of SLOTS_USED. This periodic wakeup requirement must be enforced by

appropriate setting of the WAKE_ON_COMPLETE and WAKE_ON_TIMEOUT fields of the 4 slot descriptors. This is a requirement imposed on WOR software. (The same period wakeup requirement does not necessarily apply to Constant Interval scheduling mode. See next section.)

Constant-Interval Mode

Constant-Interval mode is appropriate when the schedule for RF transactions is perfectly periodic. A good example of such a scheduling regime, is the Bluetooth Low Energy Scanning State. In this state, a SCAN_INTERVAL is defined, which sets the interval between receive operations, and a SCAN_WINDOW is defined, which sets the duration of receiver "ON" time within the SCAN_INTERVAL. Wake-On-Radio adopts the principles of Bluetooth LE SCAN_INTERVAL and SCAN_WINDOW, broadening their scope to apply to any type of RF operation (TX or RX), and modifying the nomenclature to better represent the nature of the operations, to WOR_EVENT_INTERVAL and WOR_EVENT_WINDOW.

Instead of programming timeslot START_TIME's into the slot descriptors, Contant-Interval requires that WOR_EVENT_INTERVAL and WOR_EVENT_WINDOW be programmed instead. An additional parameter, WOR_EVENT_START, determines the instant of the first RF operation of the Constant-Interval WOR session, in case the desired start time is not immediate. WOR_EVENT_START is also programmed into the descriptor table. In Constant-Interval scheduling mode, WOR hardware processes the descriptors, and uses WOR_EVENT_INTERVAL, WOR_EVENT_WINDOW, and WOR_EVENT_START, to calculate the start time (Event Timer match), for the first RF operation, as well as all subsequent RF operations, on a rolling basis. WOR software is not required to refresh the descriptor table on a periodic basis, and would only intervene in the descriptors if a change was required to any parameter (e.g., a Bluetooth LE "Anchor Point Move" during a connection would require descriptor updates). In Constant-Interval mode, the start of WOR_EVENT_WINDOW always aligns to the start of WOR_EVENT_INTERVAL.

Aside from replacing the Timeslot descriptor START_TIME parameters with WOR_EVENT_INTERVAL, WOR_EVENT_WINDOW, and WOR_EVENT_START, the descriptor format is identical across the 2 scheduling modes, and the descriptor storage space (registers) is completely shared, since only 1 scheduling mode can be in effect at any given time. See Section Descriptor Format for more information on the descriptor fields, and how they are packed into the descriptor, and how the descriptor table is shared between the 2 scheduling modes.

A diagram depicting an example of Constant-Interval scheduling mode is shown below. In this diagram, the scanning device is in Constant-Interval mode, with the WOR_EVENT_INTERVAL and WOR_EVENT_WINDOW indicated.

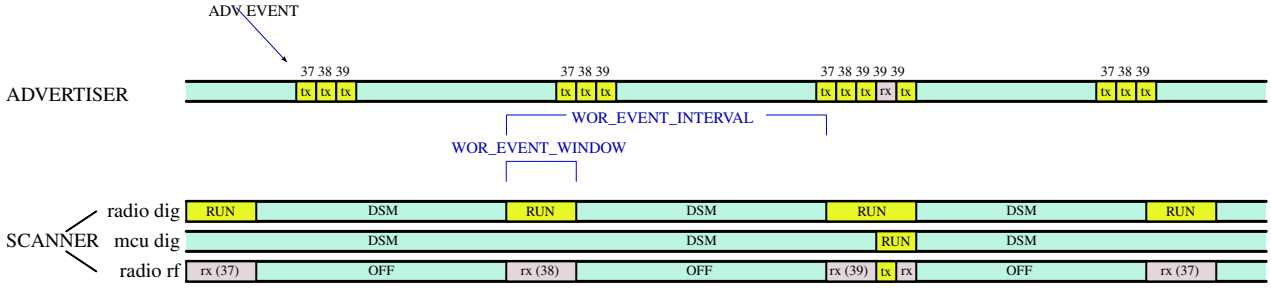


Figure 226. CONSTANT-INTERVAL SCHEDULING MODE EXAMPLE

When RF operations are to be scheduled using Wake-On-Radio in a Constant-Interval context, the WOR_EVENT_START, WOR_EVENT_INTERVAL, and WOR_EVENT_WINDOW parameters for the Constant-Interval operation are programmed into the descriptor table. These need be programmed only once, unless a change in network timing require an on-the-fly change to one of these parameters. The units for all 3 WOR_EVENT parameters are microseconds. See Section Descriptor Format for more information on the descriptor fields, and how they are packed into the descriptor. WOR hardware will process these 3 parameters to calculate the initial start time, and all subsequent start times, on a rolling basis. The start times computed by WOR hardware, are relative to the Event Timer of the assigned Link Layer. WOR hardware will monitor the assigned Link Layer's Event Timer; when it matches the next computed start time, WOR hardware will initiate an RF sequence. The type of sequence is determined by the current slot descriptor's FUNCTION field, and can be TX or RX. The RF operation will be allowed to continue, until it either self-terminates, or a timeout is reached. The timeout period for Constant-Interval scheduling mode is determined by the WOR_EVENT_WINDOW descriptor field. If a timeout is reached, i.e.,

EVENT_TMR = START_TIME (computed) + WOR_EVENT_WINDOW (descriptor),

WOR hardware will terminate the sequence. Timeout applies to RX sequences only. RX sequences may result in no packet being received, a packet being received which is not addressed to the device, or a packet which is received corruptly. Any of these scenarios will result in a timeout and a WOR shutdown of the sequence. A correctly received packet addressed to the device will self-terminate the RX sequence before the timeout is reached. Timeout is not relavent to TX sequences, since these always self-terminate. The concept of slots still applies to Constant-Interval Mode. The number of slots used can be any integer from 3 to 4 for Constant-Interval mode. The register SLOTS_USED is programmed with this value. Consecutive RF operations are assigned to consecutive slots, with the slot descriptor's parameters (e.g. FUNCTION, or WAKE_ON_COMPLETE) applying to the current slot. As in Timeslot mode, in Constant-Interval mode, WOR hardware will increment the slot number for each new RF operation, modulo SLOTS_USED.

In any timeslot, the assigned FUNCTION is a simple TX or RX operation. There is no automatic acknowledgement of packets. If acknowledgement is required during a timeslot, the MCU must be awakened (by setting the WAKE_ON_COMPLETE bit of the slot's descriptor), and the software must handle the acknowledgement, using standard Link Layer (not WOR) resources, within the turnaround time requirement (if any). WOR hardware will hold its state and wait for software to complete the acknowledgement, and other required tasks, before advancing to the next slot on software command.

If the assigned protocol engine is 802.15.4 or GENERIC_FSK, an autosequence is available which is a TX sequence preceded by a CCA (Clear Channel Assessment). If the CCA results in a busy channel, the autosequence terminates without transmission; otherwise the transmission proceeds. This autosequence can be used in WOR Timeslot mode by appropriately configuring the 802.15.4 or GENERIC_FSK Link Layer.

In any timeslot, if the assigned FUNCTION is RX, a TIMESTAMP can be captured to record the instant the packet's Access Address or SFD arrives at the Link Layer packet processor. There are 4 TIMESTAMP registers, once for each slot; hence, timestamps for the last 4 slots during which packets are received, can be recorded. See Section Timestamps for more details.

Typically, for RX timeslots, it is desirable to wake the MCU on good packets only. Thus, the slot descriptor's WAKE_ON_COMPLETE field should be set to 1. If it is desired to wake the MCU on no-packet, or on incorrecty-received packets, the WAKE_ON_TIMEOUT descriptor field can be set to 1. If WAKE_ON_TIMEOUT is set to 1 on any slot's descriptor, the WAKE_ON_COMPLETE field is *implied* to be set to 1 as well, regardless of its actual programmed value.

Typically, for TX timeslots, it is desirable to wake the MCU after each transmission, to service the Packet Buffer. In such scenarios, WAKE_ON_COMPLETE should be set to 1. If it is required to transmit the same packet repeatedly, across multiple slots, such that the Packet Buffer does not require regular service, WAKE_ON_COMPLETE can be set to 0, to skip the MCU wakeup, on any slot.

Unlike Timeslot mode, in Constant-Interval mode, it is not required to wake up the MCU at least every *Nth* slot to refresh the descriptor start times, because WOR hardware is able to compute all future start times on a rolling basis. If all slots are of the same type (same FUNCTION, TX or RX, and same descriptor information), there is no hard limit on the number of consecutive slots that can be executed between MCU wakeups. To set such a upper limit, the WAKE_ON_NTH_SLOT[7:0] register can be programmed with a value. When the number of slots elapsed after setting SLOTS_USED to a non-zero value to begin a WOR session, reaches the value of WAKE_ON_NTH_SLOT, the MCU will automatically be awakened at the end of the current slot, regardless of the setting of the WAKE_ON_COMPLETE and WAKE_ON_TIMEOUT bits in the descriptor. The WAKE_ON_NTH_SLOT register is only relavent to Constant-Interval scheduling mode, and has no effect in Timeslot mode. In Constant-Interval mode, if automatic wake on *Nth* slot is not desired, set WAKE_ON_NTH_SLOT=0 to disable this wakeup upper-limit. However, if a zero setting is used, software *must* program at least one of the slot descriptors to schedule an MCU wakeup by setting the either WAKE_ON_COMPLETE or WAKE_ON_TIMEOUT fields of the descriptor to 1; otherwise a situation could arise where the MCU is never awakened by WOR.

55.4.6.2.3.3 Descriptor Format

As mentioned in previous sections, Wake-On-Radio allows the next several RF operations to be scheduled in advance. For scheduling purposes, each of the next, several RF operations is assigned to a slot. Each slot is associated with a descriptor, which is a hardware-based data structure containing information about the characteristics of the slot, and instructions for how WOR hardware should execute it. Characterisics include start time of the RF operation, and type (function) or operation to be conducted by WOR. Software schedules the next *N* RF operations (slots) by programming the next *N* descriptors with the slots' characteristics. Software then sets the SLOTS_USED register to *N* to begin a WOR session. WOR hardware includes a 2-bit slot counter that cycles through the slots, in order, starting with SLOT 0; the slot counter is used to select the "current descriptor". The current descriptor's slot characteristics are forwarded to the WOR state machine, which executes the RF operation per the

instructions in the descriptor. After the RF operation is complete, the slot counter is incremented by 1 (modulo SLOTS_USED), and the next slot's descriptor becomes the "current" one. The process repeats indefinitely until the WOR session is ended by setting SLOTS_USED=0. WOR hardware automaticlly inserts (unless instructed otherwise in the descriptor) a DSM cycle, after the current slot's RF operation completes, with a DSM duration that lasts until the start time of the *next* slot, so that time between RF operations is always filled with DSM, for maximum power savings. After setting SLOTS_USED to a non-zero value to begin a WOR session, the MCU can be put into a low power state, and remain there, until the last (*Nth*) slot completes, assuming Timeslot scheduling mode. (For Constant Interval mode, MCU wake up on the *Nth* slot is not required. See Section Scheduling Modes for information on when and how the MCU is to be awakened for Constant-Interval mode.)

The descriptor format for Timeslot scheduling mode is shown in the diagram below, indicating the various fields in the descriptor, and the bits assigned to each field:

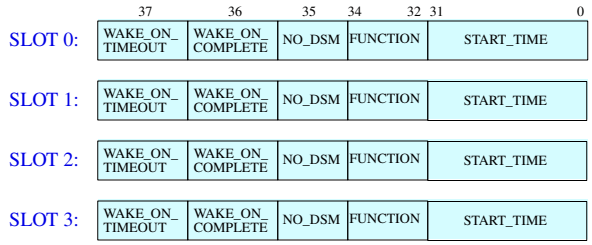


Figure 227. DESCRIPTOR FORMAT (TIMESLOT MODE)

The descriptor format for Constant-Interval scheduling mode is shown in the diagram below, indicating the various fields in the descriptor, and the bits assigned to each field:

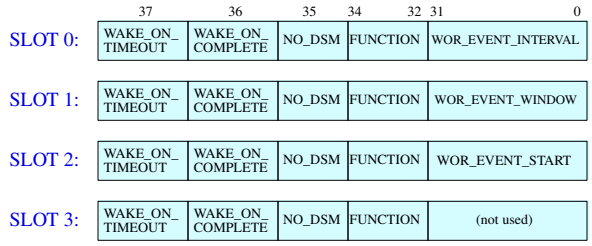


Figure 228. DESCRIPTOR FORMAT (CONSTANT-INTERVAL MODE)

The descriptor formats for Timeslot and Constant-Interval scheduling modes are almost identical, and so the desciptor hardware is almost completely shared by the 2 scheduling modes, as shown in the diagram below:

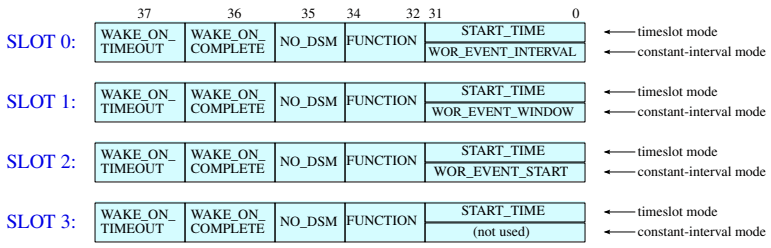


Figure 229. DESCRIPTOR FORMAT (SHARED)

Descriptor fields are described in the table below:

FIELD	DESCRIPTION
START_TIME	The instant at which to begin the current slot's RF operation. START_TIME is relative to the assigned Link Layer's Event Timer. Units are microseconds. Because the Event Timers' bit widths vary with protocol, see START_TIME programming information later in this section. Applies to Timeslot mode only. WOR hardware will automatically adjust this value with window-widening, if enabled.
WOR_EVENT_INTERVAL	Sets the interval between the start of RF operations for consecutive slots. Units are microseconds. Applies to Constant-Interval mode only. WOR hardware will automatically adjust this value with window-widening, if enabled.
WOR_EVENT_WINDOW	Maximum duration of the transceiver "ON" time for the current slot. "ON" time will be less than this if the RF operation self-terminates. Units are microseconds. Applies to Constant-Interval mode only. The start of the WOR_EVENT_WINDOW aligns to the start of the WOR_EVENT_INTERVAL. WOR hardware will automatically adjust this value with window-widening, if enabled.
WOR_EVENT_START	The instant at which to begin the RF operation for the first SLOT 0 of the WOR session. WOR_EVENT_START is relative to the assigned Link Layer's Event Timer. Units are microseconds. Because the Event Timers' bit widths vary with protocol, see WOR_EVENT_START programming information later in this section. Applies to Constant-Interval mode only. WOR hardware will automatically adjust this value with window-widening, if enabled.
FUNCTION	This 3 -bit field defines the RF function of the slot: 0: No RF 1: TX 2: RX 3: RT 4: TR 5~7: Reserved
NO_DSM	By default, a DSM cycle is automatically entered at the end of the current slot's RF operation, and the DSM cycle is timed to expire just before the start time of the next consecutive slot. If this field is set to 1, DSM is not entered, and the WOR state machine holds the radio in RUN state until the start time of the next slot is reached. NO_DSM can be used to execute back-to-back, or closely spaced RF operations, such as a Bluetooth LE 3-channel advertising event.
WAKE_ON_COMPLETE	Instructs the WOR state machine to permit an interrupt to wake up the MCU at the end of a succesfully completed

Table continues on the next page...

Table continued from the previous page...

FIELD	DESCRIPTION
	RF operation. RX operations are successfully completed if a packet is received before the timeout, which passes CRC and packet filtering rules. TX operations are always successfully completed. A successfully completed operation with WAKE_ON_COMPLETE=1 will cause the WOR state machine to temporarily remove its Link Layer interrupt inhibits, allowing the assigned Link Layer's RF interrupts (RXIRQ, TXIRQ, and/or SEQ_END_IRQ) to propagate, resulting in an MCU wakeup. If WAKE_ON_TIMEOUT=1, WAKE_ON_COMPLETE is *implied* to be 1 as well, regardless of the setting of this field.
WAKE_ON_TIMEOUT	Instructs the WOR state machine to permit an interrupt to wake up the MCU at the point of timeout of an RX operation. For Timeslot mode, timeout is determined by the RECEIVE_TIMEOUT[15:0] register; for Constant-Interval mode, timeout is determined by WOR_EVENT_WINDOW descriptor field. In either case, window-widening will automatically adjust the timeout instant, if enabled. A timeout which occurs with WAKE_ON_TIMEOUT=1 will cause the WOR state machine to temporarily remove its Link Layer interrupt inhibits, allowing the assigned Link Layer's SEQ_END_IRQ to propagate, resulting in an MCU wakeup, with a notification of timeout.

Event Timer bit width various across protocols. For 802.15.4, Event Timer is 28 bits; for GENERIC_FSK, Event Timer is 32 bits. In all cases, the Event Timer clock rate is 1MHz.

When FUNCTION is set to "NO_RF", no RF operation is performed during the slot. In this scenario, the WOR state machine will wait until the slot's programmed (or computed) START_TIME, and then automatically re-enter DSM until the next consecutive slot's START_TIME is reached. This allows multiple (up to 4) DSM-only slots which effectively extends the maximum DSM duration fourfold, to 1073 seconds (about 18 minutes), for the 802.15.4 and GENERIC_FSK protocol. However the RF Oscillator is activated breifly at each slot's START_TIME so that the WOR can process the current slot descriptor, and update the assigned Link Layer's Event Timer. Extending the DSM duration using the "NO_RF" method is referred to as "DSM cascading".

During an RX operation on any slot, if an AA or SFD has been detected at the point of timeout, the timeout (and the automatic shudown of the sequence by the WOR state machine) is *deferred* until either:

1: The packet is fully received and determined to be good. In this case the WAKE_ON_COMPLETE bit determines if the MCU is to be awakened.

2: The packet is fully received and determined not to be good. In this case the WAKE_ON_TIMEOUT bit determines if the MCU is to be awakened.

There are no time restrictions on programming the descriptor table. All descriptor fields can be modified at any time. Needless to say, caution should be exercised with modifying descriptor START_TIME, and other parameters, for the *current* slot; START_TIME (or WOR_EVENT_START) must be in the future, and FUNCTION should not be modified once the START_TIME has been reached.

55.4.6.2.3.4 State Machine

In Wake-On-Radio, future RF operations are scheduled by programming instructions into the WOR Descriptor table. Based on descriptor specifications, WOR hardware organizes the scheduled RF operations into slots, with one RF operation per slot. The slot includes more than just the RF operation. Idle time between the current time, and the start of the RF operation, is filled

with DSM. WOR hardware must compute the required DSM duration, and request DSM from RFMC (Radio Mode Controller). A diagram showing how WOR hardware processes the descriptor table, and organizes the required operations into consecutive slots, encompassing both DSM and RF, is shown below.

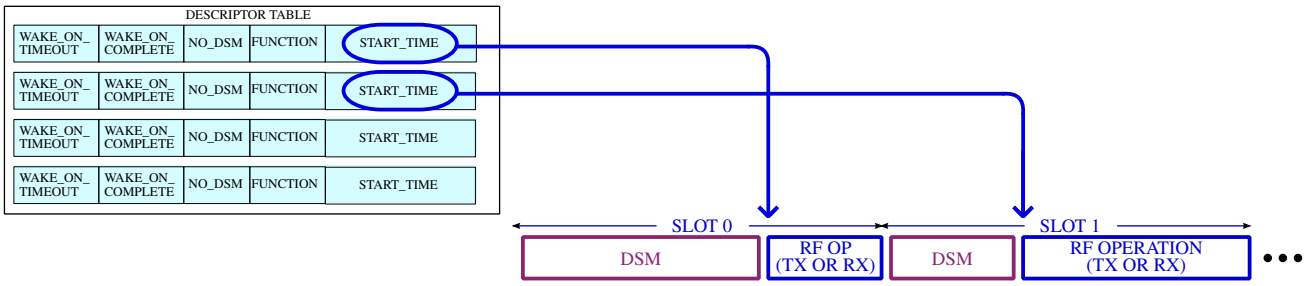


Figure 230. TIMELINE OF CONSECUTIVE SLOTS

Once a WOR session has been initiated by setting SLOTS_USED to a non-zero value, the timeline of consecutive slots shown, will repeat indefinitely, with the slot number increasing by 1 after each slot, modulo SLOTS_USED.

During each slot, WOR hardware must perform a series of tasks, e.g., computing the DSM duration, commanding the the assigned Link Layer to start and stop the RF sequence, etc. Some of the required WOR hardware tasks are superimposed on the slot timeline, in the diagram below.

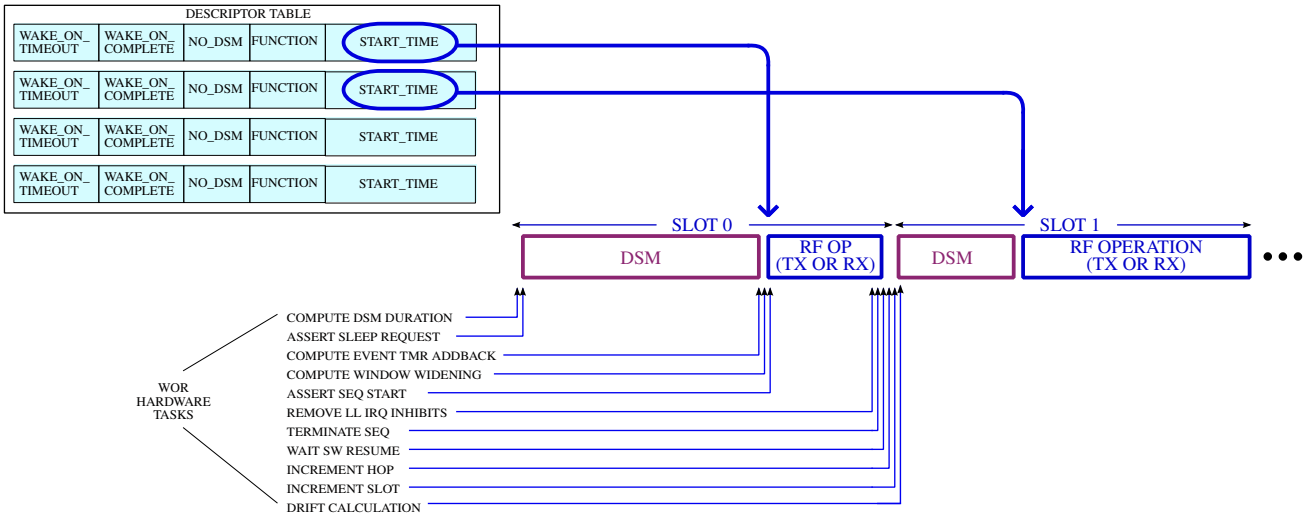


Figure 231. SLOT TIMELINE WITH HARDWARE TASKS

The WOR state machine is responsible for processing the descriptor table, and performing all the required tasks during each slot. The 11 -state machine interfaces with the assigned Link Layer for sequence operations, and with RFMC for DSM operations. The state machine also controls the incrementing of the the slot counter, and the frequency hop table index. A state diagram of the WOR state machine is shown below.

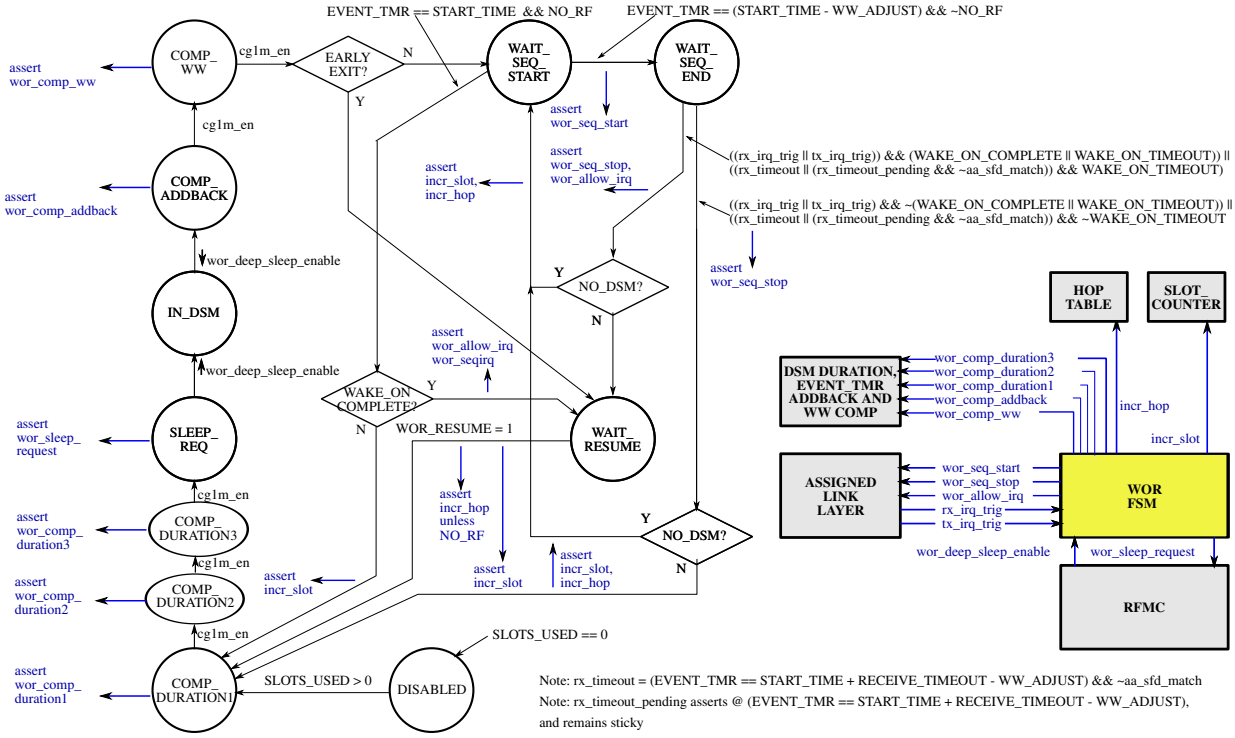


Figure 232. WAKE-ON-RADIO STATE DIAGRAM

A description of the WOR states follows.

DISABLED

WOR is in its disabled state whenever SLOTS_USED=0. WOR can be taken to "DISABLED" from any state by setting SLOTS_USED=0. WOR can be enabled by setting SLOTS_USED to a non-zero value, which initiates a new WOR "session".

COMP_DURATION1

WOR must compute the duration of the upcoming DSM cycle. This is done in 3 stages (states). COMP_DURATION1 is the first stage. Whenever the slot counter is incremented to begin a new slot, the slot's (descriptor's) START_TIME is in the future. In most cases, the START_TIME is distant enough that DSM is justified. During COMP_DURATION1, WOR computes the time to the next START_TIME, by subtracting both the current time (the assigned Link Layer's Event Timer) and the time-accumulated window-widening (if enabled) from the slot descriptor's START_TIME. (See Section Window Widening for more details on this feature.) The result of this subtraction is MICROSECONDS_TO_START_TIME[27:0]. This value must be converted to 32.768KHz cycles to be of use to RFMC. The multiplier is (549755 / 2^24), which is an approximation to the ideal conversion factor. The product is truncated to integer, the result represents the number of 32.768KHz cycles to remain in DSM prior to the next START_TIME. The hardware computations which take place during COMP_DURATION1 are shown in the following diagram.

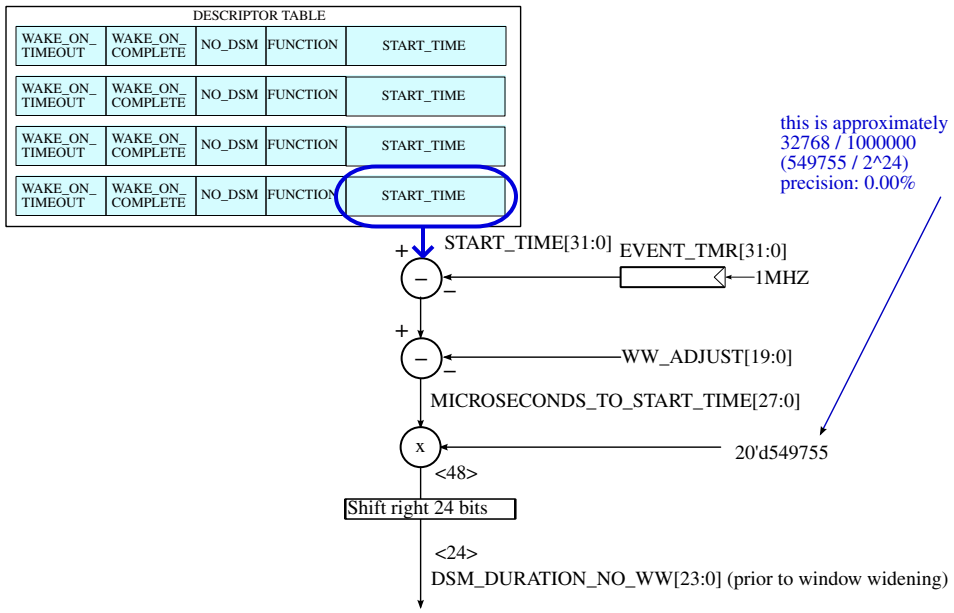


Figure 233. COMP_DURATION1

However this intermediate result does not account for the fact that window-widening for the upcoming DSM cycle itself, must be subtracted from this product. This is separate from the "accumulated" window-widening which has built up during the WOR session so far, which has already been accounted for. The estimate for the next DSM cycle's window widening is handled in the next state, COMP_DURATION2

The WOR output wor_clr_dsm_duration is asserted in this state, which is sent to the RFMC to proactively clear the DSM Duration Counter, which becomes active in the IN_DSM state (see below). The wor_clr_dsm_duration is an active-high, registered (glitch-proof) output signal.

COMP_DURATION2 and COMP_DURATION3

If window-widening is enabled, the DSM duration computed during COMP_DURATION1 may be long enough that it will need to be shortened slightly to account for window-widening which will accumulate during the DSM cycle itself. The computation for this window-widening is performed during this state. (See Section Window Widening for more details on this computation, and the usage of WW_DSM_FACTOR.) The (intermediate) duration from COMP_DURATION1 is multiplied by the DSM window-widening factor, and the product is subtracted from the duration estimate. The difference is a more accurate duration that accounts for window-widening during the DSM cycle. In a final step, a guardband is subtracted from the DSM duration. The guardband, with units of 32.768KHz clock cycles, guarantees that the DSM will complete on time before the next START_TIME is reached, accounting for RFMC synchronization and other delays in processing the WOR's sleep request. The guardband, which applies to all DSM cycles initiated by WOR, is determined by the register DSM_GUARDBAND[3:0]. The result of this step, is the final DSM duration (wor_dsm_duration[23:0]) which is then sent to the RFMC, which is responsible for radio DSM entry and exit. The hardware computations which take place during COMP_DURATION2 and COMP_DURATION3(alongside those covered by COMP_DURATION1) are shown in the following diagram.

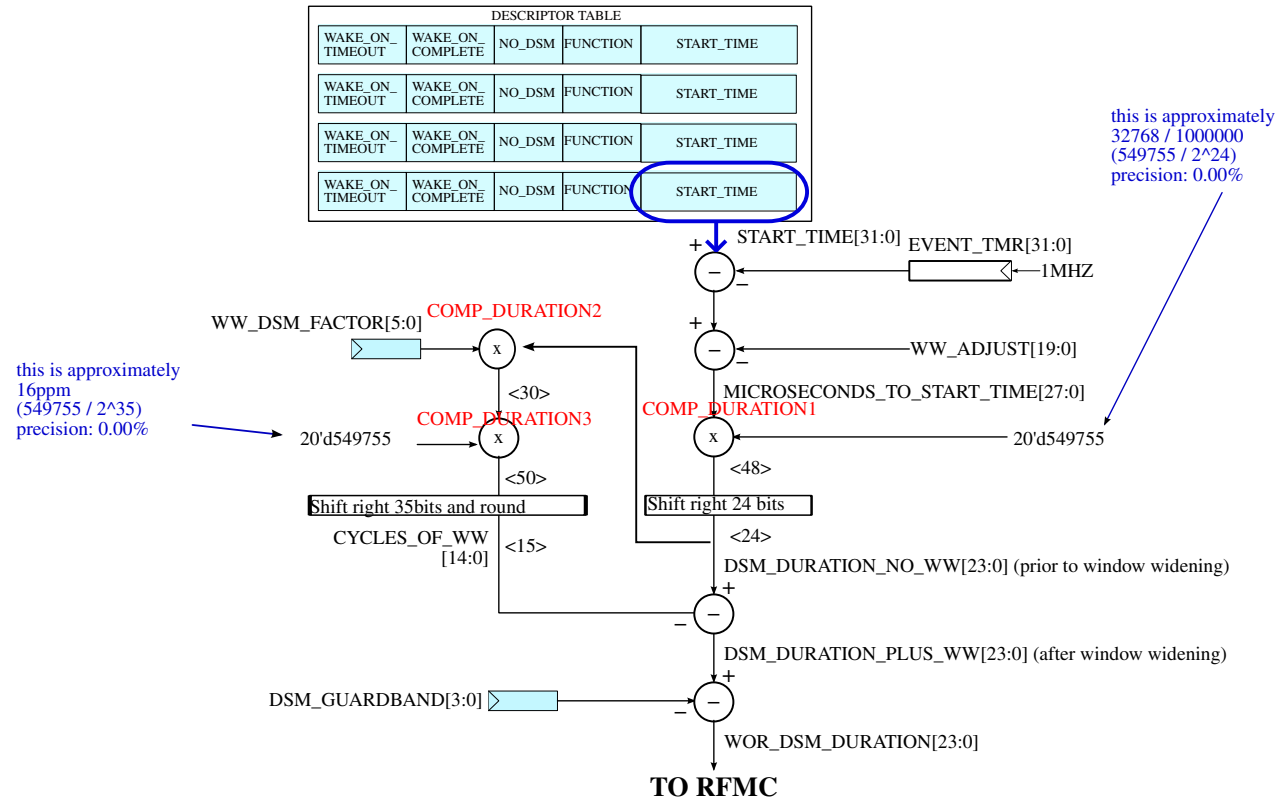


Figure 234. COMP_DURATION2 and COMP_DURATION3

SLEEP_REQ

In this state, WOR asserts its sleep request (wor_sleep_request) to the RFMC. This is a level, active-high, and registered (glitch-proof) signal to RFMC to indicate a DSM cycle is being requested of precise duration wor_dsm_duration[23:0], which was computed during the prior 2 WOR states. The units of wor_dsm_duration[23:0] are 32.768KHz clock cycles. RFMC shall synchronize wor_sleep_request into the 32.768KHz clock domain, and use it to activate the RFMC's DSM state machine. The RFMC DSM state machine shall place the radio in DSM, and simultaneously assert the output: wor_deep_sleep_enable. This signal is the precise indicator of DSM being "in effect", and is used to gate the assigned Link Layer's clocks and freeze its Event Timer.

IN_DSM

DSM is in effect during this state. Because the RF Oscillator has been disabled, there are no clocks to WOR, including its state machine, or the Link Layer, in this state. This is merely a holding state that prevails until DSM exits and the clocks to WOR and Link Layer are restarted. During this state, RFMC will use wor_deep_sleep_enable to enable a 24-bit counter which counts out the precise number of cycles in wor_dsm_duration[23:0]. When this counter reaches wor_dsm_duration, RFMC shall exit DSM and de-assert wor_deep_sleep_enable. RFMC shall provide the final count of the 24-bit counter which was used to time the DSM, using output signal rfmc_sleep_duration[23:0]. In most cases, rfmc_sleep_duration will be an exact match to wor_dsm_duration; the exceptional case is an unscheduled, early exit of DSM (see Section DSM Early Exit.) The interface signals between WOR and RFMC, and a high-level description of the hardware tasks performed by RFMC during IN_DSM state, is shown in the following diagram.

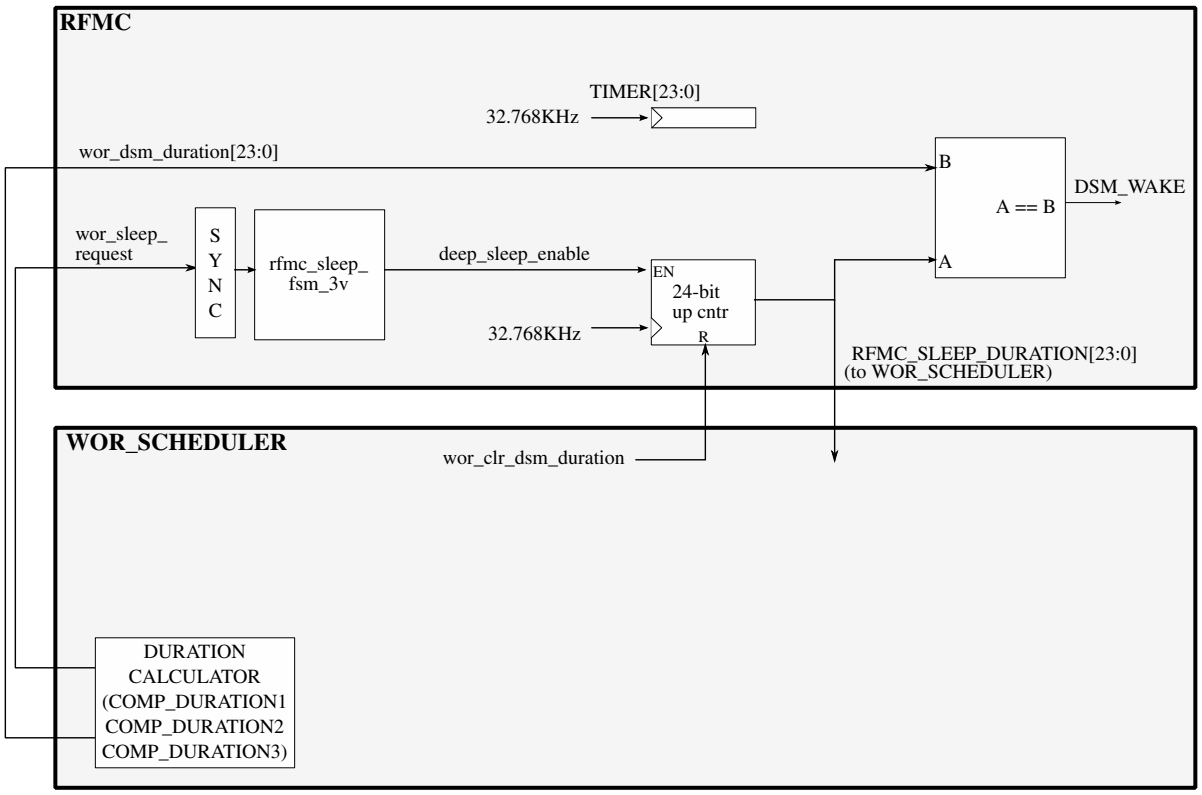


Figure 235. IN_DSM

COMP_ADDBACK

During the previous state, IN_DSM, the Reference Oscillator was disabled, and so the Event Timer of the assigned Link Layer was frozen. After DSM completes, clocks are restored and Event Timer resumes counting, but it is "behind" by the amount of time that the radio was in DSM. WOR hardware must compute the amount of DSM time, convert it to microseconds, and "add it back" to the Event Timer, so that after the addback, the Event Timer timebase is restored to where it would have been, had there been no DSM. (Note: there is a maximum of 1us of "loss of precision" upon every DSM exit; this timebase error is random and will not accumulate, it will average out over many DSM cycles). In this state, WOR computes the addback value by multiplying the rfmc_sleep_duration[23:0] (the precise number of 32.768KHz clock cycles during which RFMC held the radio in DSM) by 30.517578125 (the number of microseconds per 32.768KHz clock cycle). The 30.517578125 is an unsigned 14-bit value with 5 integer bits. The other multipicand, rfmc_sleep_duration is an unsigned integer. The product is the DSM elapsed time in microseconds, a 38 unsigned bit value with 29 integer bits. The product is rounded to the nearest integer and the 1 MSB's are discarded, resulting in the final wor_addback[27:0] value. At the end of this state, this result is sent to the assigned Link Layer's Event Timer, along with the wor_addback command (pulse), indicating to the Link Layer that it must add this amount to the current state of its Event Timer. After this point the protocol's timebase has been restored and operations can now be performed referencing the Event Timer. The following diagram depicts the hardware tasks performed in the COMP_ADDBACK state.

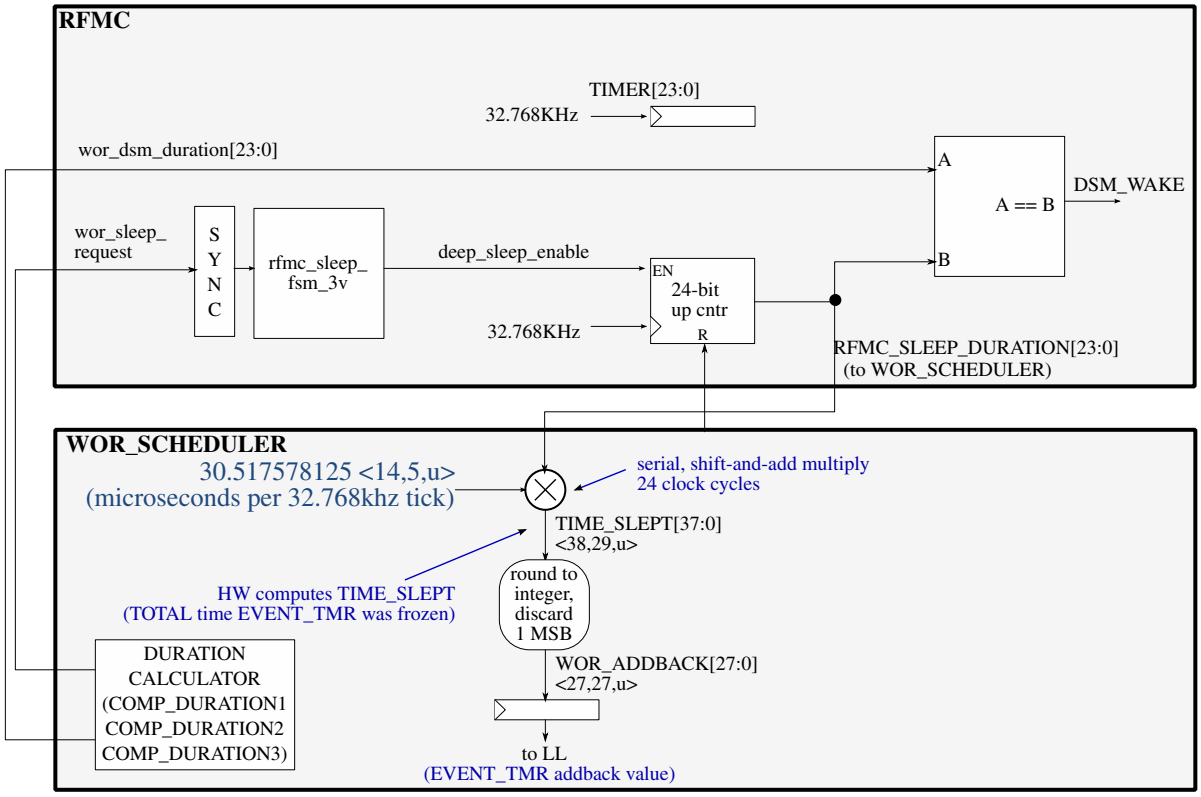


Figure 236. COMP_ADDBACK

COMP_WW

If window-widening is enabled, WOR must compute how much to increase the receive window by, based on the duration of the just-completed DSM cycle, so that this early-start offset to the receive window is ready in advance of the slot descriptor's START_TIME. The window-widening increment is computed during this state. Details of this hardware task are described in Section Window Widening.

WAIT_SEQ_START

After COMP_WW state, the WOR is ready to command an RF operation, and is only waiting for the slot descriptor's START_TIME to arrive. This start time will be in the near future since the DSM duration was timed to wake up "just in time" to start the RF operation, with sufficient guardband to guarantee that a DSM wake up can never occur too late. The WAIT_SEQ_START state merely waits for the assigned Event Timer to match the current slot descriptor's START_TIME. If the slot's FUNCTION is RX or RT, and window widening is enabled, START_TIME is modified by the hardware to be:

START_TIME - ww_adjust[19:0],

where ww_adjust[19:0] is the time-accumulated window-widening "early start" amount for the RX operation (see Section Window Widening.) If the FUNCTION is not RX or if window-widening is disabled, START_TIME is unmodified. Once a START_TIME match to the Event Timer occurs, WOR asserts the Link Layer interface signal wor_tx_start_ext (if FUNCTION is TX or TR), or wor_rx_start_ext (if FUNCTION is RX or RT). Neither interface signal is asserted if FUNCTION is NO_RF. The assigned Link Layer will commence an RF operation, TX or RX, based on this WOR command. At this point, the WOR state machine arrives at a fork:

- If FUNCTION = NO_RF and the descriptor WAKE_ON_COMPLETE=1, the next state is WAIT_RESUME
- Else if FUNCTION = NO_RF and WAKE_ON_COMPLETE=0, the next state returns to COMP_DURATION1 to begin the next slot. The slot counter and hop table index are incremented.
- Else, this is an RF operation that must be waited for. The state machine advances to the WAIT_SEQ_END state.

WAIT_SEQ_END

This state has been reached because an RF operation was launched during the previous state, and WOR is waiting for that operation to end. There are 3 ways for an RF operation to end:

- A TX operation self-terminates, indicated when the Link Layer asserts tx_irq_trig to WOR.
- A RX operation self-terminates due to a good packet reception, indicated when the Link Layer asserts rx_irq_trig to WOR.
- A RXor RT or TR operation times out. The rx_irq_trig is not asserted before the timeout. See Section Scheduling Modes for a description on the 2 methods WOR uses to compute the point-of-timeout, based on Scheduling Mode.

If either tx_irq_trig or rx_irq_trig is asserted, WOR advances to the next state (see below). If a point-of-timeout is reached without an rx_irq_trig, WOR hardware checks to see if Access Address Match has occurred or if SFD has been detected. If either is the case, timeout is deferred until either:

- The rx_irq_trig is ultimately asserted to indicate a good packet was received. This is a delayed good packet reception.
- The packet reception underway at point-of-timeout ultimately results in a CRC- or filtering-violation. This will automatically trigger a Link Layer RX recycle which will negate the AA-match or SFD-match indicator. WOR will detect this negation and declare a delayed timeout at this point.

If a timeout occurs, either nominal or delayed, WOR will assert wor_seq_stop_ext to the assigned Link Layer, causing an immediate abort of the RX sequence. This abort shall be handled by the protocol engine like a software abort. The wor_seq_stop_ext will not assert for a deferred timeout which results in a delayed good packet reception.

Nominally, WOR blocks the assigned Link Layer's RF-related interrupts from propagating to the MCU, to prevent unintended MCU wakeups during a WOR session. The RF-related interrupts for the supported protocol engines, are nearly identical in function across a 3 protocols, and are called RXIRQ, TXIRQ, and SEQ_END_IRQ. At the conclusion of an RF operation on a slot, WOR must decide whether to temporarily remove the Link Layer interrupt inhibits, to allow the Link Layer interrupt to be asserted to wake the MCU, based on which of the 5 outcomes to the RF operation occurred, and the state of the WAKE_ON_COMPLETE and WAKE_ON_TIMEOUT descriptor fields. The following table specifies whether WOR blocks, or allows, the RF interrupts to propagate, at the conclusion of the RF operation.

RF OPERATION OUTCOME	WAKE_ON_TIMEOUT=0, WAKE_ON_COMPLETE=0	WAKE_ON_TIMEOUT=0, WAKE_ON_COMPLETE=1	WAKE_ON_TIMEOUT=1, WAKE_ON_COMPLETE=x
tx_irq_trig	BLOCK	ALLOW	ALLOW
rx_irq_trig (nominal)	BLOCK	ALLOW	ALLOW
rx_irq_trig (delayed)	BLOCK	ALLOW	ALLOW
no rx_irq_trig @ nominal timeout	BLOCK	BLOCK	ALLOW
no rx_irq_trig @ delayed timeout	BLOCK	BLOCK	ALLOW

At the conclusion of the RF operation, the WOR next state is a function of whether the Link Layer RF interrupts were BLOCKED or ALLOWED (see previous table), as well as the NO_DSM descriptor field as illustrated in the following table.

RF OPERATION OUTCOME	INTERRUPTS BLOCKED, NO_DSM=0	INTERRUPTS ALLOWED, NO_DSM=0	NO_DSM=1
tx_irq_trig	COMP_DURATION1 (increment slot, hop idx)	WAIT_RESUME	WAIT_SEQ_START (increment slot, hop idx)
rx_irq_trig (nominal)	COMP_DURATION1 (increment slot, hop idx)	WAIT_RESUME	WAIT_SEQ_START (increment slot, hop idx)

Table continues on the next page...

Table continued from the previous page...

RF OPERATION OUTCOME	INTERRUPTS BLOCKED, NO_DSM=0	INTERRUPTS ALLOWED, NO_DSM=0	NO_DSM=1
rx_irq_trig (delayed)	COMP_DURATION1 (increment slot, hop idx)	WAIT_RESUME	WAIT_SEQ_START (increment slot, hop idx)
no rx_irq_trig @ nominal timeout	COMP_DURATION1 (increment slot, hop idx)	WAIT_RESUME	WAIT_SEQ_START (increment slot, hop idx)
no rx_irq_trig @ delayed timeout	COMP_DURATION1 (increment slot, hop idx)	WAIT_RESUME	WAIT_SEQ_START (increment slot, hop idx)

WAIT_RESUME

If WOR has allowed the MCU to be awakened (interrupted) by temporarily negating the RF interrupt inhibits, WOR must enter a state which permits the MCU executing WOR or Link Layer software, to complete all its assigned tasks on the slot. These tasks may include processing and decrypting a received packet, servicing the packet buffer, and formulating and transmitting an acknowledge packet. The WAIT_RESUME state serves that purpose. The WAIT_RESUME state is reached whenever all of the following conditions are met:

- Current state is WAIT_SEQ_END
- FUNCTION is TX or RXor RT or TR
- RF operation completes with any outcome (rx_irq_trig, tx_irq_trig, nominal, or delayed timeout)
- NO_DSM=0
- RF Interrupts are ALLOWED (see table, WAIT_SEQ_END state)

WAIT_RESUME is also reached whenever all of these following conditions are met:

- Current state is WAIT_SEQ_START
- FUNCTION is NO_RF
- START_TIME match to the assigned Link Layer's Event Timer
- WAKE_ON_COMPLETE=1

Once WAIT_RESUME is reached, WOR software must authorize WOR hardware to resume the WOR session and advance to the next slot, by setting the WOR_RESUME register bit to 1. When WOR_RESUME is set to 1 in WAIT_RESUME state, WOR hardware takes the following steps:

1. Advances to the next slot (slot counter++ % SLOTS_USED)
2. Increments the frequency hop table index *unless* FUNCTION = NO_RF
3. Enters COMP_DURATION1 state on the new slot
4. Reimposes Link Layer interrupt inhibits

55.4.6.2.3.5 Window Widening

OVERVIEW

Because a device in a connected state is not receiving all the time, its timebase will invariably drift with respect to the timebase of other device(s) in the network, due to finite crystal oscillator tolerances amongst the networked devices. Reducing the duty cycle of the receiver, albeit necessary to achieve power savings, exacerbates the drift problem.

To compensate for drift, Receive Window Widening can be employed. As time elapses in a network connection between successful packet receptions, the window-widening algorithm gradually moves future receiver turn-on times to the left (advance) and the turn-off time to the right (retard) to allow the transmitted packet to fall within the receiver's window in the presence of a

drifting timebase. The effect of window-widening on successive receive cycles, none of which result in a good packet reception, is illustrated in the following diagram.

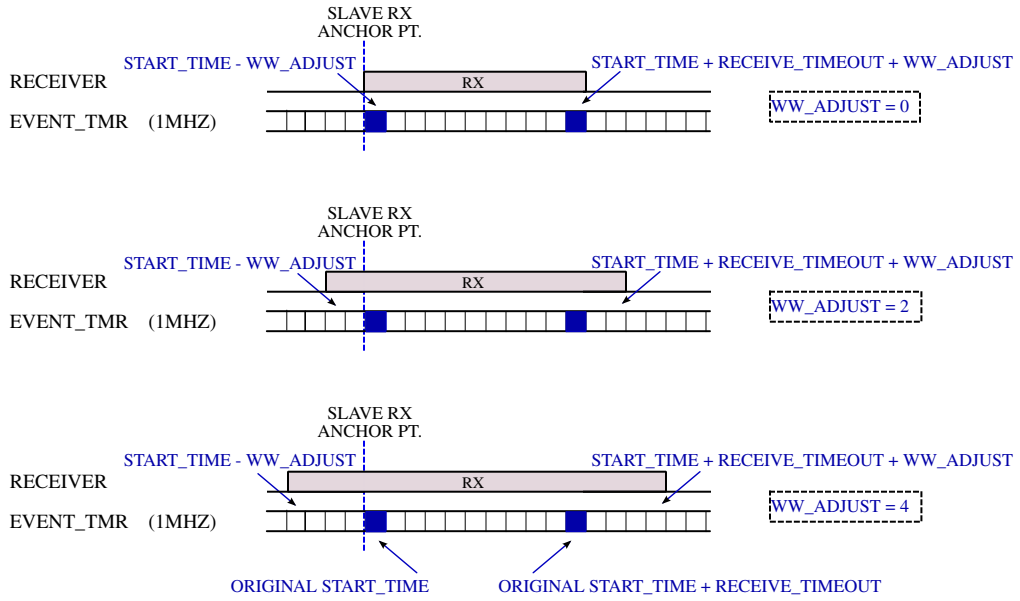


Figure 237. EFFECT OF WINDOW WIDENING ON RECEIVE WINDOW

Window widening is typically applicable only to the slave device(s) in a network, as the master (coordinator) of the network is normally the master of network timing. Thus, in Wake-On-Radio, window widening is optional, enabled by the by setting the WW_EN register bit to 1.

The Window widening algorithm needs to take into account the "Sleep Clock Accuracy" of the devices in the network. Sleep Clock Accuracy (SCA) is a *Bluetooth Low Energy* concept, and the Bluetooth LE nomenclature is adopted by Wake-On-Radio, and extrapolated to all supported protocols.

In Bluetooth LE, drift encountered by a slave device must be compensated by accounting for the SCA of both the master and the slave (self). The master transmits its SCA to the slave in the connection request (CONNECT_REQ), an advertising channel packet which informs an advertising slave of the master's desire to form a connection with it; the parameters on the CONNECT_REQ include the master's SCA. The slave device knows its own SCA, so the composite SCA is known at the point of connection establishment. Wake-On-Radio adopts this concept, although the precise method of informing the slave of Master SCA is protocol-dependent and takes place at a higher level, beyond the scope of this document.

Bluetooth LE subdivides the allowed Master SCA into coarse ranges, to reduce the number of over-the-air bits required to inform the slave. The ranges are shown in the following diagram.

SCA	masterSCA
0	251 ppm to 500 ppm
1	151 ppm to 250 ppm
2	101 ppm to 150 ppm
3	76 ppm to 100 ppm
4	51 ppm to 75 ppm
5	31 ppm to 50 ppm
6	21 ppm to 30 ppm
7	0 ppm to 20 ppm

Figure 238. SCA RANGES

The slave SCA must be accounted for as well, so the maximum composite SCA is 1000ppm. For Wake-On-Radio, this is the maximum allowed drift as well, and window-widening programmability will allow for drift amounts within these ranges to be compensated.

FACTS

In the WOR hardware, the window-widening adjustment, is a value, ww_adjust[19:0], that is subtracted from the receiver START_TIME, and added to the receiver TIMEOUT, when the slot's FUNCTION = RX, as shown in the diagram below.

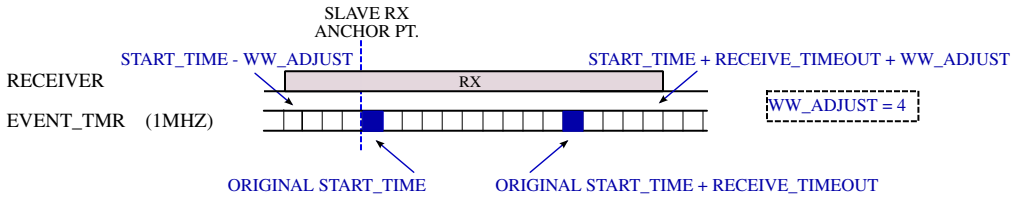


Figure 239. WINDOW WIDENING EFFECT ON RX SEQUENCE

The ww_adjust[19:0] adjustment, has no effect on START_TIME for slots using FUNCTION TX or NO_RF.

After system reset, if window-widening is enabled using WW_EN, ww_adjust[19:0] will increase linearly over time.

The adjustment, ww_adjust[19:0], will be reset to 0 under any of the following conditions:

1. system reset
2. good packet received (CRC verified and packet filtering passes)
3. Software writes a 1 to WW_NULL register bit

The reset-on-good-packet (item #2 above), can be inhibited by setting register bit WW_RESET_ON_RX=0. By default this bit is set.

The adjustment, ww_adjust[19:0], consists of 2 components, which are summed together:

1. **DSM adjustment:** computed at the conclusion of every DSM cycle based on rfmc_sleep_duration[23:0]
2. **RUNTIME adjustment:** computed by periodically incrementing a binary counter by 1 every *N* EVENT_TMR ticks whenever NOT in DSM

The rates at which window-widening accumulates during DSM and RUNTIME, are controlled by 2 programmable registers. The programmed values of these registers should be informed by the combined (master + slave) SCA.

The 2 components of the window-widening adjustment, and the reset conditionality, are illustrated in the following diagram:

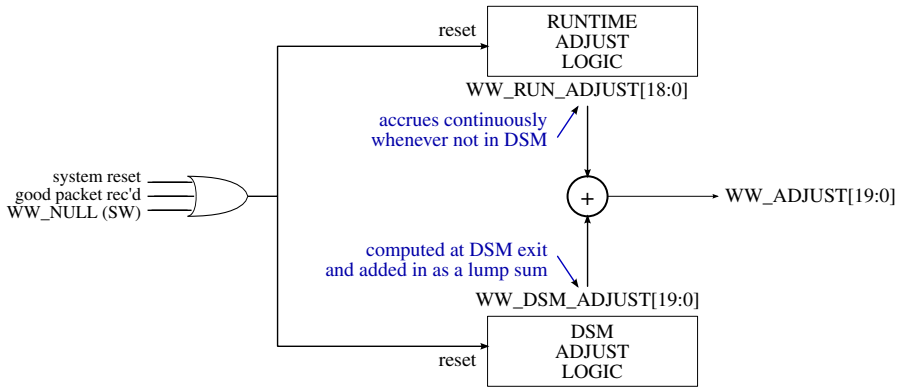


Figure 240. WINDOW WIDENING COMPONENTS AND RESET CONDITIONALITY

WINDOW WIDENING DSM ADJUSTMENT

In Wake-On-Radio, every slot includes a DSM cycle (unless NO_DSM=1 in the descriptor). WOR hardware computes the window-widening DSM adjustment after the DSM cycle completes, during the WOR COMP_WW state. This state immediately follows the COMP_ADDBACK state which updates the Event Timer for time-slept. The WW DSM adjustment, ww_dsm_adjust[19:0], is computed, and then added to the composite, accumulated adjustment, ww_adjust[19:0], as a lump sum. During COMP_WW state, WOR hardware multiplies rfmc_sleep_duration[23:0], which is the precise number of 32.768KHz cycles for which the RFMC held the radio in DSM, by a programmable factor which represents the number of microseconds per 32.768KHz clock cycle, scaled by the combined SCA of the master and slave (drift), i.e.,

DSM_DURATION (from RFMC) * (masterSCA + slaveSCA), converted to µs

Programmable register WW_DSM_FACTOR maps the composite SCA (masterSCA + slaveSCA), in parts per million, to a multiplicand which represents

N *** 16ppm * 30.5178125µs,**

where *N* = WW_DSM_FACTOR, and

30.5178125 is the *precise* number of microseconds per 32.768KHz tick, not an approximation.

Register WW_DSM_ADJUST allows SCA to be applied in coarse ranges, as described earlier in this section. Each LSB of WW_DSM_ADJUST is worth 16ppm of 30.5178125µs. Valid values are 0 ≤ WW_DSM_FACTOR ≤ 63. Consider the table below as a guide for selecting WW_DSM_FACTOR based on composite SCA:

masterSCA + slaveSCA	Use this WW_DSM_FACTOR
251 ppm to 500 ppm	16 (256ppm) - 32(512ppm)
151 ppm to 250 ppm	10 (160ppm) - 16(256ppm)
101 ppm to 150 ppm	7 (112ppm) - 10(160ppm)
76 ppm to 100 ppm	5 (80ppm) - 6 (96ppm)
51 ppm to 75 ppm	4 (64ppm)
31 ppm to 50 ppm	3 (48ppm)
21 ppm to 30 ppm	2 (32ppm)
0 ppm to 20 ppm	1 (16ppm)

Figure 241. WW_DSM_FACTOR BASED ON COMPOSITE SCA

A summary of all the hardware tasks performed to compute ww_dsm_adjust[19:0] during the COMP_WW state, is shown in the diagram below.

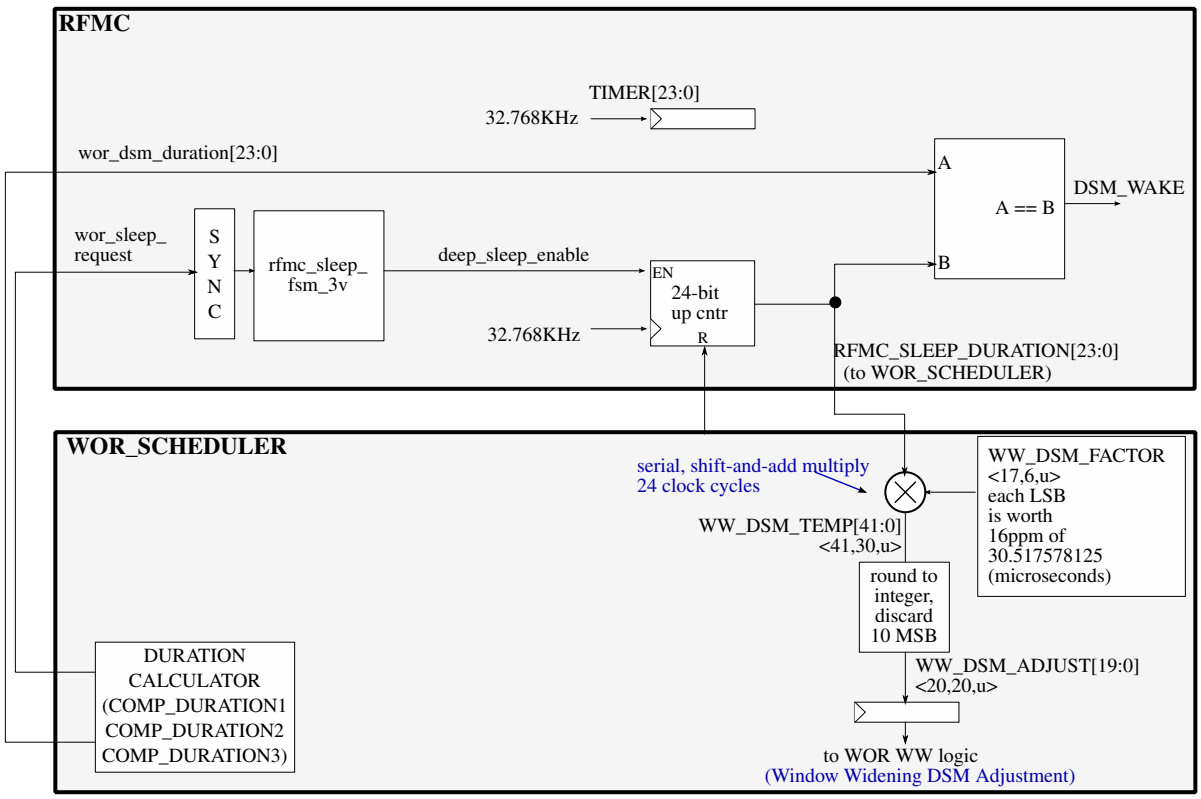


Figure 242. HARDWARE TASKS DURING COMP_WW STATE

WINDOW WIDENING RUNTIME ADJUSTMENT

The window-widening RUNTIME adjustment is computed by incrementing a binary counter every *N* EVENT_TMR ticks whenever NOT in DSM (i.e., EVENT_TMR is running). *N* is a variable based on programmable register WW_RUN_ADJUST[4:0]. The programmable resolution, or granularity, of the DSM adjustment (16ppm), will be matched by the RUNTIME adjustment. Unlike the DSM adjustment, the RUNTIME adjustment is an approximation to the ideal value of *N*, since *N* is the result of a division operation whose quotient is not always an integer number of microseconds. However, the objective in Wake-On-Radio is to spend as little time in RUN between events as possible, so the DSM adjustment will dominate the sum. Therefore, additional resolution in the RUNTIME adjustment, although acheivable, would have little benefit in overall power savings.

N is a modulo for the binary counter which tracks the runtime adjustment, and represents the ideal number of microseconds between each increment to the adjustment, which is simply the reciprocal of the composite SCA, i.e.,

***N* = 1 / (master SCA + slave SCA)**

Using an example of composite SCA = 192ppm,

***N* = 1 / 192e-6 = 5208.333**

For implementation, each SCA range (the same table of ranges defined for the DSM adjustment) becomes an input to a look-up table (LUT) which approximates this division, rounding the quotient to the nearest integer. SCA range is represented by the WW_RUN_FACTOR. If WW_RUN_FACTOR=0, the RUNTIME adjustment never increments (special case). Valid values are 0 ≤ WW_RUN_FACTOR ≤ 16.

Every time the modulo *N* is reached, the binary counter resets to 0, and the RUNTIME adjustment is incremented by 1, meaning that the start of the next receive operation, has been advanced by 1µs. The hardware which implements the window widening RUNTIME adjustment is shown in the following diagram.

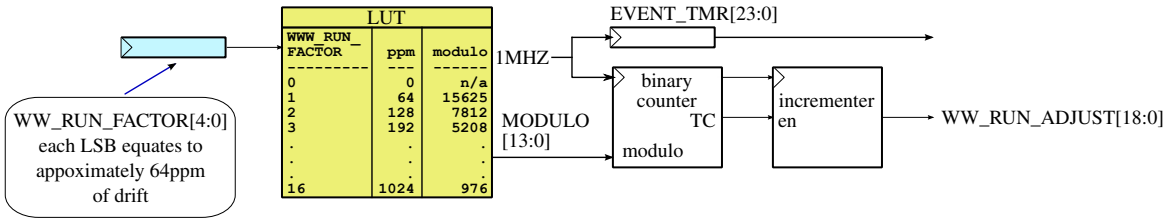


Figure 243. WINDOW WIDENING RUNTIME ADJUSTMENT

COMPENSATING DSM DURATION FOR WINDOW WIDENING

In executing each slot as specified by the WOR descriptor, there is one other instance where window widening must be accounted for. During the COMP_DURATION1 state, WOR computes the DSM duration based on descriptor START_TIME and current (Event Timer) time. This duration is timed to complete the DSM cycle as close as possible to the desired START_TIME, but with sufficient guardband to ensure wakeup can never occur too late. But the duration computed during this state may be sufficiently long that the desired START_TIME needs to be advanced, due to drift within the DSM cycle itself. Thus, window widening needs to be estimated and subtracted from this computation. This window widening estimation is used only for the purpose of the duration estimation, and does not accumulate from one DSM cycle to the next (i.e., from one slot to the next). This window widening compensation to DSM duration occurs during the COMP_DURATION2 and COMP_DURATION3 state. See Section State Machine for a description of WOR states.

The difference between this window widening computation, and the computation during COMP_WW, is that the result must be in units 32.768KHz clock cycles, because it is targetted for RFMC, instead of microseconds for an Event Timer offset. The COMP_DURATION2 and COMP_DURATION3 computation multiplies the DSM duration (in 32K cycles) computed in COMP_DURATION1 by an estimate of the SCA (drift), to arrive at a product which represents the number of 32.768KHz cycles by which to foreshorten the upcoming DSM cycle due to the widening requirement. So there is no conversion of units.

CYCLES_OF_WW[14:0] = DSM_DURATION_NO_WW[23:0] (result of COMP_DURATION1, neglecting widening) * *WW_DSM_FACTOR* * 549755 / 2^35

This product is subtracted from the DSM duration estimate, along with the DSM_GUARDBAND[3:0], to arrive at the final wor_dsm_duration[23:0] sent to RFMC for execution:

WOR_DSM_DURATION[23:0] = DSM_DURATION_NO_WW[23:0] - CYCLES_OF_WW[14:0] - DSM_GUARDBAND[3:0].

The following diagram illustrates all of the hardware tasks performed during COMP_DURATION2 and COMP_DURATION3 state:

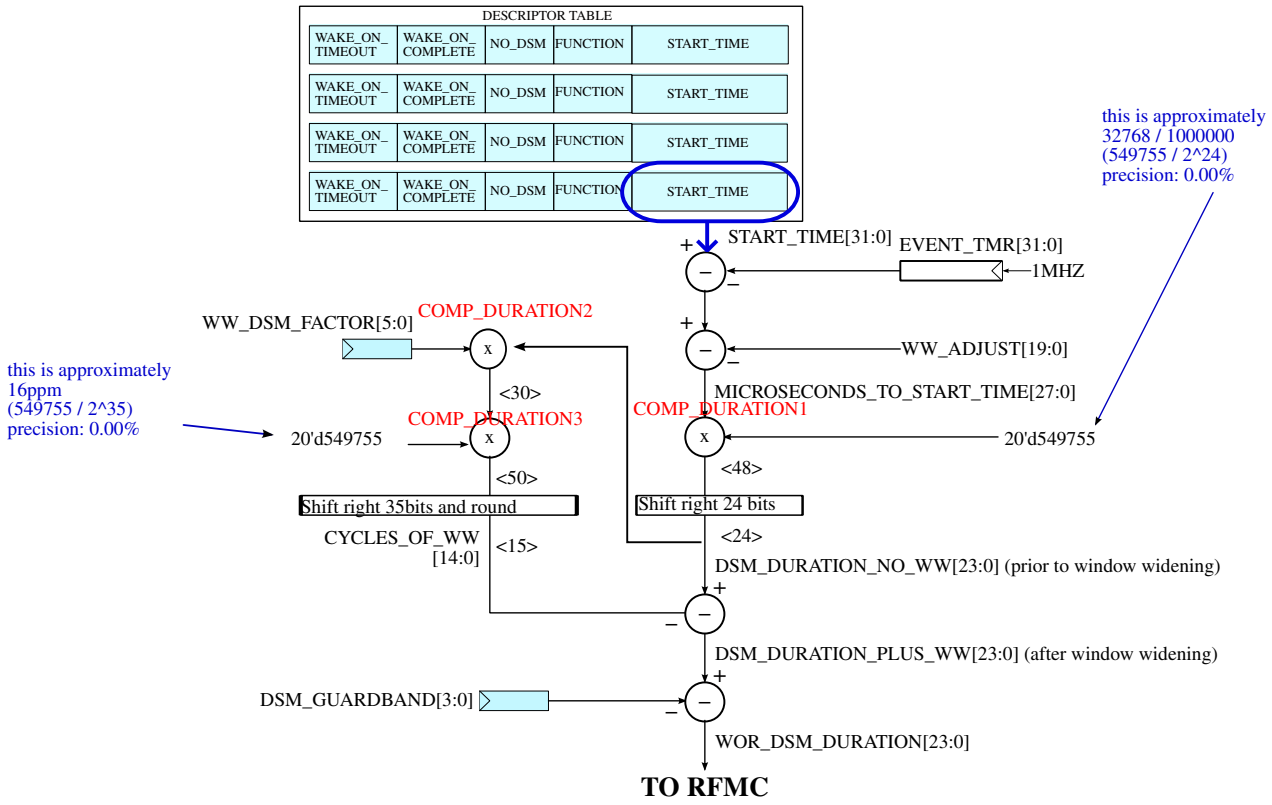


Figure 244. COMP_DURATION2

APPLYING THE WINDOW WIDENING

Now that the window-widening adjustment has been computed, it must be put into effect, by modifying the START_TIME of the RF operation to advance it by the number of microseconds in ww_adjust[19:0], which is the combined DSM and RUNTIME window widening adjustment. This occurs during the WAIT_SEQ_START state. The same adjustment is applied again, by modifying the TIMEOUT of the RF operation (assuming timeout is reached), by delaying it by the same number of microseconds that START_TIME was advanced by. This occurs during the WAIT_SEQ_END state. The following diagram illustrates how window-widening is applied in Timeslot Scheduling mode.

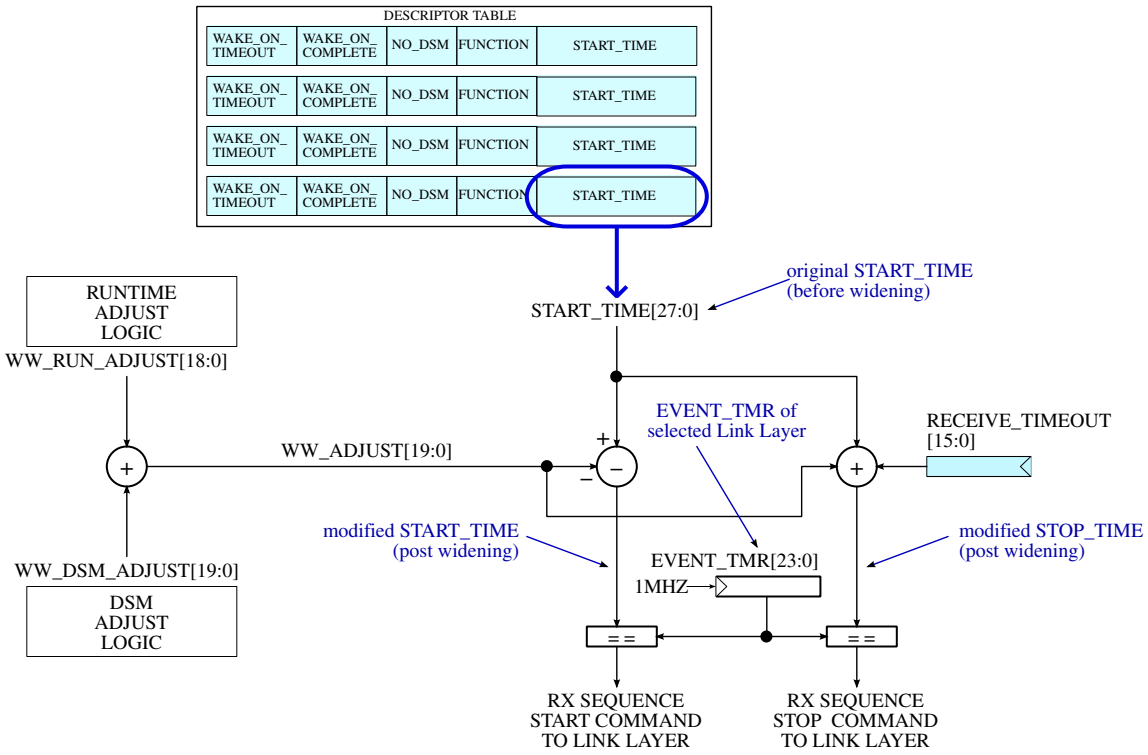


Figure 245. APPLYING WINDOW WIDENING (TIMESLOT MODE)

The application of window-widening in Constant-Interval scheduling mode, is largely the same as Timeslot mode; however the way the WOR hardware determines START_TIME and TIMEOUT is different, these need to be computed by WOR on a slot-by-slot basis. The following diagram illustrates how window-widening is applied in Constant-Interval Scheduling mode.

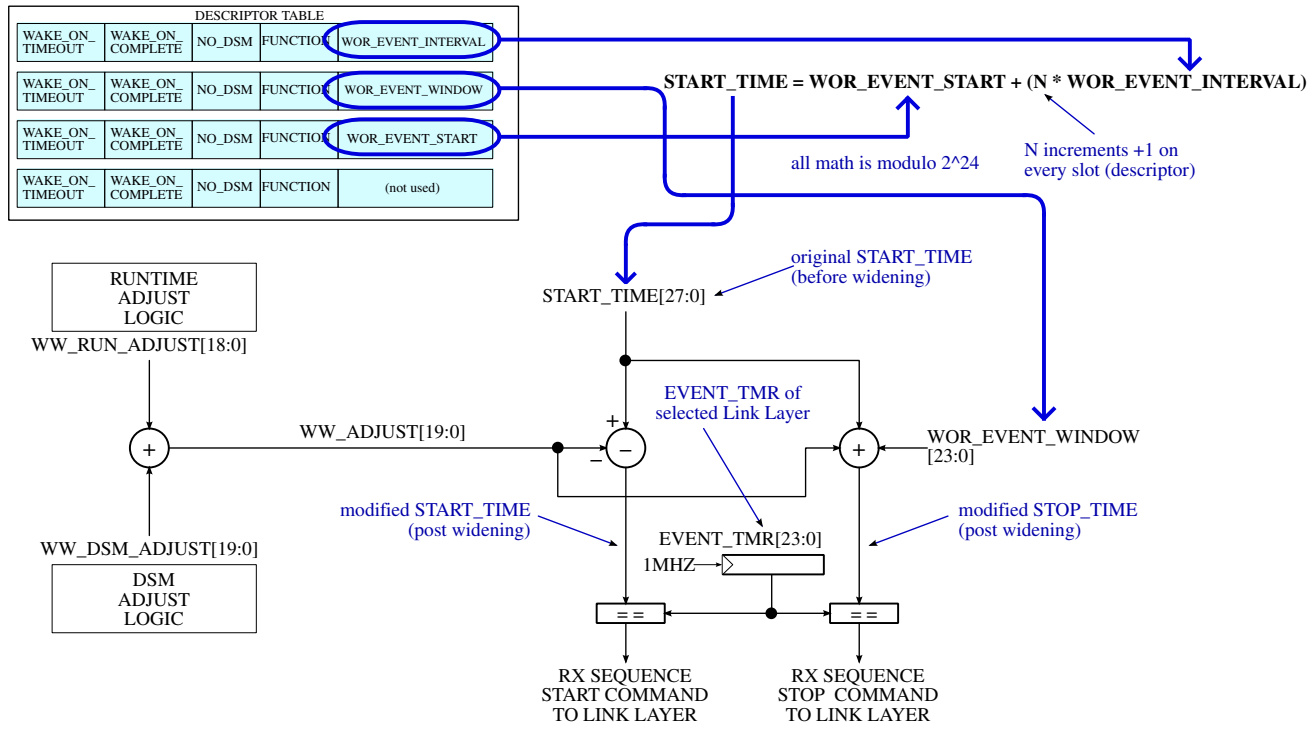


Figure 246. APPLYING WINDOW WIDENING (CONSTANT-INTERVAL MODE)

NOTE

When dealing with the combination of high SCA (Sleep Clock Accuracy) as well as long requested DSM intervals, it is the responsibility of software to guarantee that the accumulation of window widening does not become excessively large. Window widening should not be allowed to accumulate to such an extent that the adjusted START_TIME is advanced so much that it moves to the left of the *previous* slot's START_TIME, or that the TIMEOUT is retarded so much that is moves to the right of the *next* slot's START_TIME, especially in scenarios where no good packets are received for an extended period of time. Software should periodically monitor window widening accumulation using mandatory MCU wakeups after every third or fourth slot (timeslot mode), or WAKE_ON_NTH_SLOT (constant interval mode), and use caution when scheduling long DSM intervals (distance between START_TIME on consecutive slots) with high WW_DSM_FACTOR values.

55.4.6.2.3.6 Auto DSM Drift Calculation

Usually, the combination of SCA (Sleep Clock Accuracy) is hard to get, so for Window Widening, user has to set a big enough WW_DSM_FACTOR to cover the worst case. This makes the window larger enough to receive the packets but also wastes power. And also, the accumulation of window widening may become excessively large after many slots.

The "Auto Drift Calculation" feature is used to solve these shortcomings. This feature can only be used in Constant-Interval mode and it is enabled by register AUTO_CAL set when WW_EN is also enabled.

Its basic concept is that (the timestamp of the packet arrives - expected) is proportional to (the set drift - the actual drift). Based on this theory, hardware can calculate the actual drift after received a packet and then apply the calculated drift to the next slot to get the optimized window to save power. The detailed process is as follows.

1. Theory of Drift Calculation

Below figure shows a 4 slots receive case in Constant-Interval mode. After the reception of the first packet which is called the synchronization packet, software configures the WOR to the Constant-Interval mode and enables the Window Widening feature.

The term Rx(x=sync, 0~3) denotes the timestamp recorded in the link layer's Event Timer when the packet actually arrives, and Sx(x=0~3) denotes the expected timestamp which is calculated by the WOR hardware. RX_ENx(x=0~3) denotes the RX_EN signal assertion time.

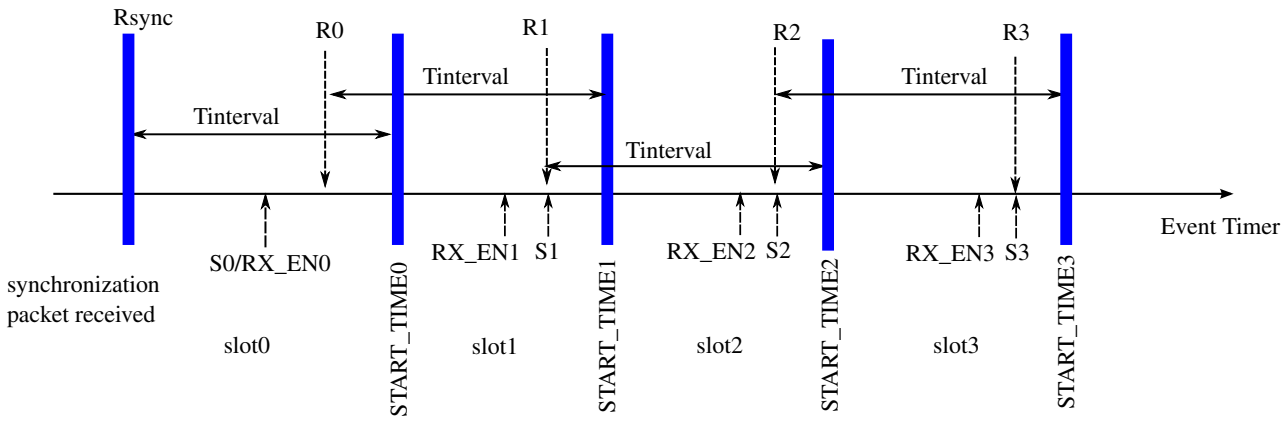


Figure 247. Auto Drift Calculation Algorithm

Software programs the START_TIME(WOR_EVENT_START) of the slot0 as

START_TIME0 = Rsync + Tinterval, Tinterval is WOR_EVENT_INTERVAL,

because if there is no clock drift, this is the next packet arrive time.

Since the START_TIME is the time which WOR hardware informs the Link Layer to assert the RX_EN signal, so it should consider the RX warm up time and the Preamble-SFD delay. The precise START_TIME should be

START_TIME0 = Rsync + Tinterval - AA_SFD_DLY,

where AA_SFD_DLY is a programmable regiter which counts for the RX warm up time and the Preamble-SFD delay.

AA_SFD_DLY does not effect the drift calculation, so for simplicity, it is ignored in the discussion below.

In this example, at slot0, user will set an initial WW_DSM_FACTOR to cover the worst case drift. If ignore the RUNTIME adjustment, the window will open WW_DSM_ADJUST0(Tsleep0 * drift_set0) earlier before the START_TIME. Due to the combination of SCA, the real packet arrive time is R0, it is (Tsleep0 * drift_real0) earlier than the START_TIME. That is to say,

the expected packet arrive time is:

S0 = START_TIME0 - WW_DSM_ADJUST0

= START_TIME0 - Tsleep0 * drift_set0 / 1000000, where drift_set0 = WW_DSM_FACTOR * 16 (ppm), Tsleep0 is the sleep time in slot0,

the real packet arrive time is:

R0 = START_TIME0 - Tsleep0 * drift_real0 / 1000000, where drift_real0 is the real drift during slot0, the unit is ppm

Hardware uses the delta time between S0 and R0, delta_time0, to get the drift_real0 as:

delta_time0 = R0 -S0

= Tsleep0 * (drift_set0 - drift_real0) / 1000000 (us)

drift_real0 = drift_set0 - (delta_time0 / Tsleep0 * 1000000) (ppm)

In the above formula, It is difficult for the hardware to calculate the division, so an approximate method is using Tinterval instead of Tsleep0:

drift_real0 = drift_set0 - (delta_time0 / Tinterval * 1000000) (ppm)

Convert drift_real0 to 16ppm unit as:

drift_real0_factor = drift_real0 / 16

= drift_set0_factor - (delta_time0 / Tinterval * 1000000 / 16) (LSB), where drift_set0_factor = WW_DSM_FACTOR

If user calculates the 1000000/Tinterval/16 and programs the result into register TINT_DIV_MILLION, hardware can simplify division into multiplication as,

drift_real0_factor = drift_set0_factor - (delta_time0 * TINT_DIV_MILLION) (LSB)

To cover the whole range of Tinterval, which is about 500 us to 2^28 us, the actual value programmed to the TINT_DIV_MILLION is

TINT_DIV_MILLION = (1000000 / Tinterval / 16) * 2^17

The hardware will take this extension into account and handle it accordingly.

The calculated drift_real_factor is readable by register **CAL_DSM_FACTOR** which is a signed register of two's complement value. A negetive value means the packet comes later than expected.

2. START_TIME Calculation for Each Slot

To calculate and track the drift in each slot, the START_TIME calculation is different with the normal Constant Interval Mode.

In normal Constant Interval Mode, the slot1's START_TIME is calculated as,

START_TIME1 = START_TIME0 +Tinterval

When AUTO_CAL is set, and if Link Layer received a packet in slot0, WOR calculates the next START_TIME as

START_TIME1 = R0 + Tinterval,

and the expected packet arrive time is:

S1 = START_TIME1 - WW_DSM_ADJUST1

= START_TIME1 - Tsleep1 * drift_set1 / 1000000, where drift_set1 = drift_real0, Tsleep1 is the sleep time in slot1,

and the real packet arrive time is:

R1 = START_TIME1 - Tsleep1 * drift_real1 / 1000000, where drift_real1 is the real drift during slot1, the unit is ppm

So, drift_real1 can be calculated as:

drift_real1 = drift_set1 - (delta_time1 / Tsleep1 * 1000000)

By this way, at each slot, hardware repeats to calculate and track the drift.

Ideally, the drift_real in slot0 and slot1 is same, so S1 is equal to R1. This means the packet may come at the exact time when the window opens. This achieves the longest sleep time.

But due to the drift calcuation error in slot0 or/and drift change between the slots, the packet may come a little bit earlier(like in slot2) or later than the expected time. To make sure that the packet is received, the real window open time is TIME_MGN earlier than the calculated point, Sx, where TIME_MGN is a user programmable register with us unit.

TIME_MGN can be calculated as:

TIME_MGN = (drift_calculation_err + inter_slot_drift_change) * Tinterval

The real Winodw Open time for slot 1 is

RX_EN1 = S1 - TIME_MGN

and the real STOP_TIME, i.e., Window Close time is

STOP1 = S1 + WOR_EVENT_WINDOW + TIME_MGN

For summary, for slot n, if there is a packet received in the previous slot n-1,

START_TIME(n) = R(n-1) + Tinterval

S(n) = START_TIME(n) - WW_DSM_ADJUST(n)

RX_EN(n) = S(n) - TIME_MGN

STOP(n) = S(n) + WOR_EVENT_WINDOW + TIME_MGN

3. Packet Missed Case

When a slot missed a packet due to some reasons, like drift sudden changes or RF signal is blocked, it will impact the drift calculation process.

Below figure shows the case that a packet is missed at slot 0. The basical idea is to apply the initial WW_DSM_FACTOR to slot 1, after received in slot 1, then apply the initial WW_DSM_FACTOR to slot2, after received in slot 2, the drift_real2 is calculated.

Since packet is missed in slot0, the initial WW_DSM_FACTOR is still used in slot1 and there is no margin time applied to START/STOP_TIME.

For slot1,

START_TIME1 = S0+Tinterval

RX_EN1 = START_TIME1 - WW_DSM_ADJUST1

STOP1 = RX_EN1 + WOR_EVENT_WINDOW + 2 * WW_DSM_ADJUST

WW_DSM_ADJUST is the accumulated DSM adjust,

WW_DSM_ADJUST = WW_DSM_ADJUST0 + WW_DSM_ADJUST1

WW_DSM_ADJUST0 is the DSM adjust of slot0,

WW_DSM_ADJUST0 = Tsleep0 * drift_set0 / 1000000, drift_set0 = WW_DSM_FACTOR * 16 (ppm)

WW_DSM_ADJUST1 is the DSM adjust of slot1,

WW_DSM_ADJUST1 = Tsleep1 * drift_set1 / 1000000, drift_set1 = WW_DSM_FACTOR * 16 (ppm)

For slot2,

START_TIME2 = R1+Tinterval

RX_EN2(S2) = START_TIME2 - WW_DSM_ADJUST2,

STOP2 = RX_EN2 + WOR_EVENT_WINDOW + 2 * WW_DSM_ADJUST2,

WW_DSM_ADJUST2 is the DSM adjust of slot2,

WW_DSM_ADJUST2 = Tsleep2 * drift_set2 / 1000000, drift_set2 = WW_DSM_FACTOR * 16 (ppm)

R2 = START_TIME2 - Tsleep2 * drift_real2 / 1000000

Using R2 and S2, we can get the drift_real2 and apply it to slot3.

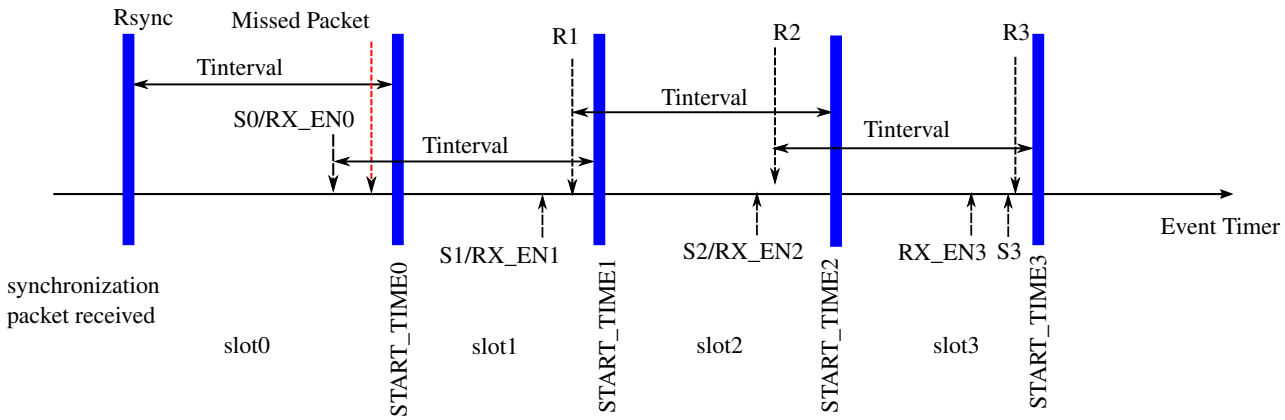


Figure 248. Handle of Missed Packet(Example 1)

As another example below, in slot0 packet is received, but in slot1, the packet is missed. The basical idea is to apply the drift_real0 to slot 2, after received in slot 2, then apply the initial WW_DSM_FACTOR to slot3, after received in slot 3, the drift_real3 is calculated..

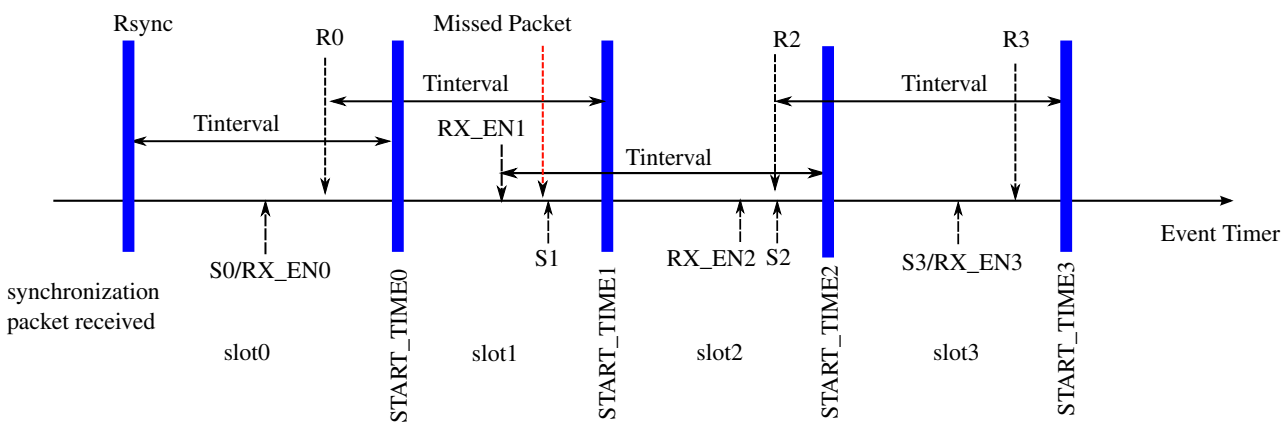


Figure 249. Handle of Missed Packet(Example 2)

Since packet is received in slot0, drift_real0 can be calculated.

For slot1,

START_TIME1 = R0+Tinterval

RX_EN1 = START_TIME1 - Tsleep1 * drift_real0 / 1000000 - TIME_MGN

STOP1 = RX_EN1 + WOR_EVENT_WINDOW + 2 * TIME_MGN

For slot2,

Then the START_TIME is calculated as:

START_TIME2 = RX_EN1+Tinterval

And hardware still applies the calculated drift_real0 to slot 2. Consider the drift change between slot0 and slot1, slot1 and slot2, the real window open point is TIME_MGN us before S2.

RX_EN2 = START_TIME2 - Tsleep2 * drift_real0 / 1000000 - TIME_MGN

and the real STOP_TIME, i.e., Window Close time is

STOP2 = RX_EN2 + WOR_EVENT_WINDOW + 4 * TIME_MGN

For summary, for slot n, if there is no packet received in the previous slot n-1,

START_TIME(n) = RX_EN(n-1) + Tinterval

RX_EN(n) = START_TIME(n) - Tsleep(n) * drift_set(n) / 1000000 - TIME_MGN

STOP(n) = RX_EN(n) + WOR_EVENT_WINDOW + 2 * (k+1) * TIME_MGN, where k is the number of missed slots since last reveived

In this example, in slot2, the packet is received. At this point, hardware does not calculate the drift_real for the next slot, because the calculation may be not correct since two slots passed. So hardware uses the initial WW_DSM_FACTOR for the next slot, slot3. and

START_TIME3 = R2 + Tinterval

So, if slot3 received a packet, the situation is like slot0, hardware get the drift_real for the next slot again.

S3 = START_TIME3 - Tsleep3 * drift_set3 / 1000000, where drift_set3 = WW_DSM_FACTOR * 16 (ppm)

R3 = START_TIME3 - Tsleep3 * drift_real3 / 1000000

drift_real3 = drift_set3 - (delta_time3 / Tsleep3 * 1000000) (ppm)

4. Slot with Function TX/TR/NO_RF

For slots with function TX/TR/NO_RF, they are not effected by the window widening. That is to say, for TX/TR, the TX_EN is asserted at the START_TIME_T:

START_TIME_T(n) = START_TIME_T(n-1) + Tinterval

START_TIME_T(0) = WOR_EVENT_START

For NO_RF, at the START_TIME_T, the WoR state machine jumps to next slot.

There is the case that TX/TR/NO_RF may be mixed with RX/RT function. In this case, the slot with TX/TR/NO_RF function is treated as "missed slot" since there is no packet received.

5. Handle of Missing Many Slots

In WoR operation, there may be some cases that many slots missed packets. The window will become larger and larger if missed so many packets and it is responsible for the WoR to wake up the MCU to check the fail condition and re-configure the radio block again.

WoR block provides a register **RX_SLOT_FAIL_THRESH**. When the number of missed slot is equal to this value plus 1, the WOR_RX_FAIL_IRQ is set. And when WOR_RX_FAIL_IRQ_ EN is set, the related interrupt will be sent to the MCU.

6. Software Do the Drift Calculation

Because the hardware uses Tinterval to approximate Tsleep, there is error. The WoR hardware provides a software interface that uses the MCU to calculate drift_real. When register **TIME_REC** is set with AUTO_CAL, some timing information is recorded to the Packet RAM. There can be at most 8 slots timing information can be recorded. WOR uses 160 bytes of Packet RAM for the timing information.

The timing information for one slot is as shown in below table.

Basically the drift_real can be calculated with:

drift_real_factor = drift_set_factor - (aa_match_time_delta_signed / Tsleep * 1000000 / 16) (LSB)

Where

Tsleep = rfmc_sleep_duration * 1000 / 32.768 (us)

Note, the calculation is only make sense when received a packet in a slot with the previous slot also received, or received a packet in the first slot.

This calculated drift factor can be programmed into register **SW_DRIFT_SET**, which is a two's complement value and each LSB is worth 16ppm.

Table 401. Timing Information for One Slot

Timing information word #	Timing information
Timing information word 0	{16'h0, packet_received, NO_DSM, 3'h0, wor_function[2:0], wor_slot_count[7:0]}
Timing information word 1	{{8{1'b0}}, rfmc_sleep_duration[23:0]}
Timing information word 2	TX_EN or RX_EN assert time
Timing information word 3	Packet arrived time, TimeStamp
Timing information word 4	{{11{1'b0}}, aa_match_time_delta_signed[20:0]} (aa_match_time_delta_signed is the delta time between Sx and Rx, two's complement value)

7. Calculation Error

There are many cases which will cause the calculation error. All these error can be covered by properly setting the register **TIME_MGN.**

7.1 Using Tinterval as Tsleep

Since the hardware are using Tinterval as Tsleep during the drift calculation, this will cause error.

7.2 Using 16ppm granularity

Since the hardware finally calculates the drift in 16ppm granularity, due to the round function, there may be 0.5 LSB(8 ppm) error

7.3 Drift change during slots

Due to the voltage or temperature change, the drift may change during slots.

7.4 The Event Timer Jitter after sleep

The event timer has a random ±1 us error after sleep. This error is caused and cannot be avoided by the fact that the 1MHz clock and the 32KHz clock are async. As shown in below waveform.

This will cause the delta_time ±1 us error, so will cause the calculation has error as:

When the Tinterval is 1ms, this 1us error will cause the calculated drift ±1000ppm error;

When the Tinterval is 10ms, ±100ppm error

When the Tinterval is 100ms, ±10ppm error

When the Tinterval is 1s, ±1ppm error

So the register TIME_MGN should be set to also consider this ±1 us error.

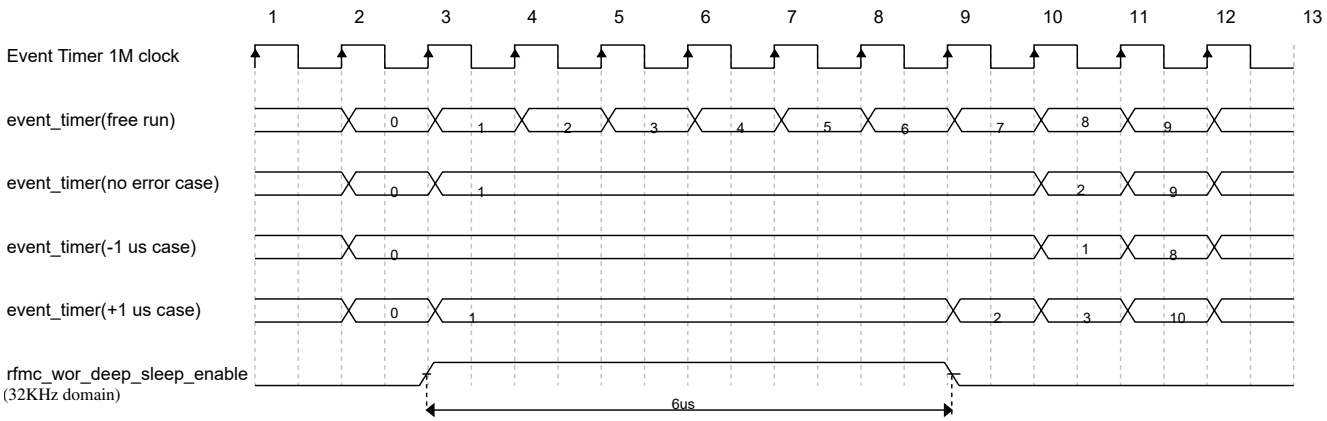


Figure 250. Event Timer Jitter

55.4.6.2.3.7 Timestamps

In Wake-On-Radio, every slot with FUNCTION=RX can capture a timeslot associated with any packet that is received in the slot. Any receive slot which results in an Access Address or an SFD detection will capture a timestamp into a register. Timestamps assist Link Layer software in maintaining and fine-tuning the network timebase across 2 or more networked devices.

Timestamp Capture

In Wake-On-Radio, each descriptor is associated with a slot. Any slot can be assigned an RX operation. RX operations require a timestamp to be captured. So, the timestamp can be thought of as being associated with the descriptor. The bit width of the timestamp is the same as the bit-width of the assigned Link Layer; i.e., 32 bits for GENERIC_FSK, and 28 bits for 802.15.4. Association of timestamps to descriptors is illustrated in the following diagram.

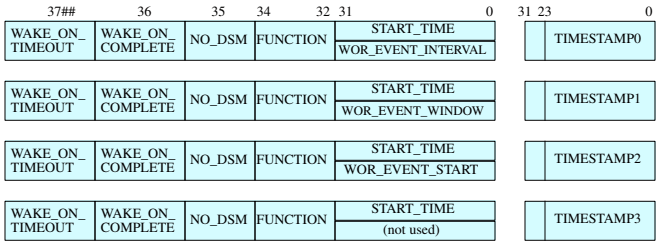


Figure 251. TIMESTAMP CAPTURE

Capture occurs at the instant AA or SFD is detected, and results in the current state of the assigned Link Layer's Event Timer being transferred to a TIMESTAMP register. WOR hardware does not apply any offset to the captured timestamp. A timestamp is captured for the last *N* RX operations, which result in (at least) one AA or SFD detection, where *N* = SLOTS_USED (typically 4).

Timestamp capture is area-intensive, due to:

- wide bit-width
- multiplicity (per descriptor)

Each of the supported protocol engines already has a TIMESTAMP register of the required bit-width. It is used for non-WOR (legacy) operations. To save area, when WOR is engaged, the TIMESTAMP register indigenous to the Link Layer, is used to capture the timestamp for WOR SLOT 0. Thus, in WOR mode (WOR_EN=1), the assigned Link Layer's TIMESTAMP register will only capture a timestamp when WOR SLOT=0, and will not change otherwise. If WOR is not engaged, the Link Layer TIMESTAMP register will capture all timestamps, the legacy behaviour. When WOR_EN, timestamps associated with SLOTS 1, 2, and 3 are captured in the TIMESTAMP1, TIMESTAMP2, and TIMESTAMP3 registers, respectively, which are new, and resident in the WOR scheduler. This distribution of timestamp register resources is illustrated in the following diagram.

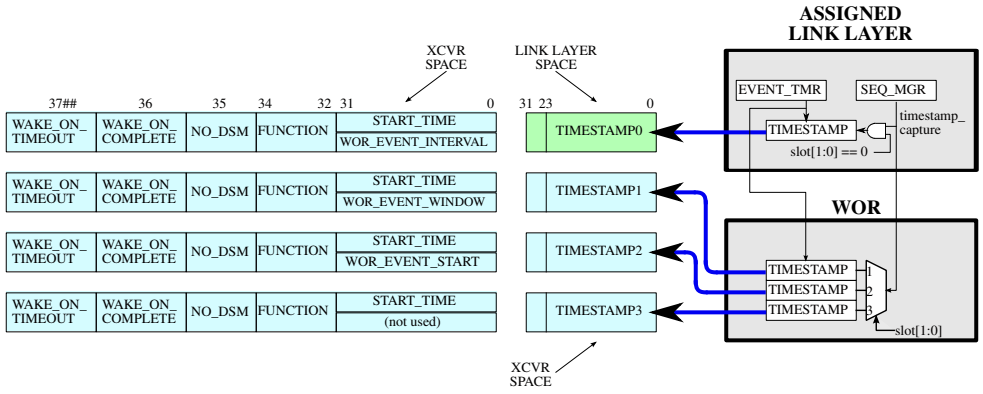


Figure 252. TIMESTAMP REGISTER DISTRIBUTION

Software should be aware of the fact that the designating the SLOT 0 timestamp to reside in the assigned Link Layer's TIMESTAMP register means that the TIMESTAMP regsters in Wake-On-Radio, are not all in one address space: TIMESTAMP0 resides in the assigned Link Layer's address space, while TIMESTAMP1-3 register reside in XVCR address space, alongside all the other WOR registers.

TIMESTAMP STATUS

A timestamp will be captured into a TIMESTAMPx register on any slot assigned to FUNCTION RX, whenever an Access Address or SFD is detected during the receive window. Whenever a timestamp is captured as described above, a 3-bit read-only status register is updated with information regarding status of the reception which occured during the slot. The timestamp status, TIMESTAMPx_STS[2:0], becomes valid, and final (for the current slot), at the end of the receive window, and is made available to MCU when it is next awakened, along with the TIMESTAMP itself. The status information can be used by software to ascertain the validity of the timestamp. The following diagram describes timestamp status.

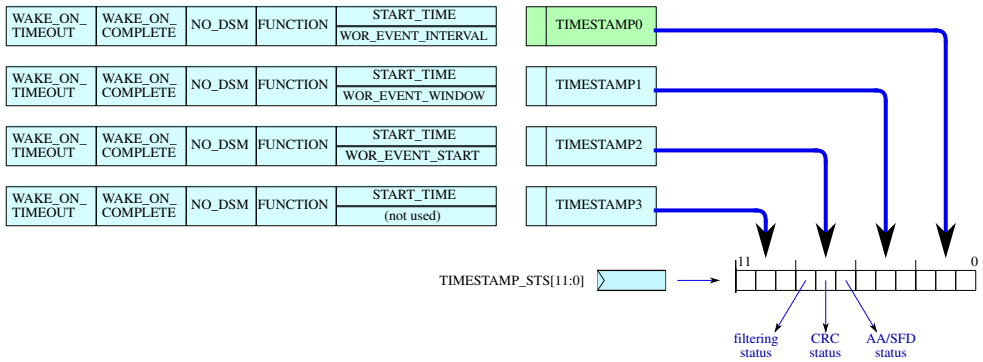


Figure 253. TIMESTAMP STATUS

A breakdown of the 3-bit status field, per timestamp, follows in the table below.

- 0: AA/SFD status
- 1: CRC status
- 2: Packet Filtering status

For any WOR slot assigned to RX, the 3-bit timestamp status field for that slot is cleared to 000 at the instant that WOR state WAIT_SEQ_START is reached. This is prior to the launch of the receive operation. (The bits are not cleared if the slot's function is not RX.) Once reception begins, the individual timestamp status bits are updated in realtime as the circumstances of the reception warrant. The 3-bit timestamp status is considered valid at the end of the receive window, and the bits reflect reception status per the following table.

TIMESTAMPx_STS bit	Meaning if 0	Meaning if 1
[0] AA/SFD status	No AA or SFD was detected during the receive window.	At least one AA or SFD was detected. RX recycles can lead to multiple occurrences.
[1] CRC status	For the packet associated with last (most recently) detected AA or SFD, CRC verification failed.	For the packet associated with last (most recently) detected AA or SFD, CRC verification passed.
[2] Packet Filter status	For the packet associated with last (most recently) detected AA or SFD, at least 1 packet filtering rule was violated.	For the packet associated with last (most recently) detected AA or SFD, all packet filtering rules passed.

Packet filtering rules, and hence the meaning of the TIMESTAMPx_STS[2] bit, vary across protocols, as summarized in the following table.

PROTOCOL	MEANING OF TIMESTAMP_STSx[2] (PACKET FILTER)
802.15.4	Reflects the Packet Processor FILTERFAIL flag
GENERIC_FSK	There are 3 packet filter criteria: 1. H0 matching 2. H1 matching 3. LENGTH_MAX compliance (maximum packet length verification)

Timestamp status bits are sticky. Once they become valid at the end of the receive window (end of WAIT_SEQ_END state), they remain valid until the *next* time the same slot number comes up again, and WOR reaches WAIT_SEQ_START state on that slot number.

In general, the MCU does not need to be notified on "bad packets." However, timestamp status may still be of value to WOR software even without Data Indication on any given slot. Notification policy can be modified at the protocol level, to enable RXIRQ on bad packets, mainly for debug purposes. If notification policy is modified by configuring the assigned Link Layer accordingly, this will affect Wake-On-Radio was well, as an rx_irq_trig will be generated on otherwise "bad" packets. See Section State Machine, specifically the WAIT_SEQ_END state, for details on how WOR responds to rx_irq_trig during this state, to determine how modifying notification rules would affect WOR response.

WOR software can use timestamp status to gauge the validity of the timestamp in any given slot. The following tables provides some guidelines.

TIMESTAMPx_STS	TIMESTAMPx REGISTER VALIDITY	MEANING
0b000	stale	No AA or SFD was received during this timeslot. The TIMESTAMPx register was not updated by HW and continues to hold the timestamp captured after the last AA or SFD detection on this slot number.
0bX01	invalid	At least 1 AA or SFD was detected during this timeslot. For the last (most recent) AA or SFD detected, CRC verification failed. TIMESTAMP register holds the EVENT_TMR captured for

Table continues on the next page...

Table continued from the previous page...

TIMESTAMPx_STS	TIMESTAMPx REGISTER VALIDITY	MEANING
		this reception. Because CRC failed, the TIMESTAMP should be deemed invalid, but this decision is left to the application.
0b011	valid	At least 1 AA or SFD was detected during this timeslot. For the last (most recent) AA or SFD detected, CRC verification passed. TIMESTAMP register holds the EVENT_TMR captured for this reception, and can be deemed valid. However Packet Filtering failed, so typically there would be no Data Indication.
0b111	valid	At least 1 AA or SFD was detected during this timeslot. For the last (most recent) AA or SFD detected, CRC verification passed. TIMESTAMP register holds the EVENT_TMR captured for this reception, and is valid. Packet Filtering passed.

55.4.6.2.3.8 Interrupt Management

WOR hardware utilizes the assigned Link Layer's interrupts for all MCU wakeups and notifications. In Wake-On-Radio, the MCU is in a low-power state most of the time. So, interrupts are allowed to assert only when it is mandatory to wake the MCU out of its low power state, to tend to required Link Layer or WOR tasks. WOR hardware modifies the interrupt activity of the assigned Link Layer by dynamically blocking and/or allowing Link Layer interrupts to propagate to the MCU, under control of the current slot descriptor specifications. Each of the 3 supported protocol engines have been modified to allow WOR to manage its RF-related interrupts. The RF-related interrupts have nearly identical functions and meanings across the 3 protocols:

- TXIRQ (or TX_IRQ)
- RXIRQ (or RX_IRQ)
- SEQ_END_IRQ (or SEQIRQ)

The modifications to the protocol engines, involve an AND gate inserted between each interrupt trigger, and the interrupt status bit, allowing or blocking the trigger from asserting the status bit. A gating signal, wor_allow_irq, is provided by WOR to each protocol engine to enable/disable the AND gate. The gating signal is updated dynamically by the WOR state machine based on state, RF operation status, and the current descriptor's WAKE_ON_* fields. See Section State Machine, specifically the description of the WAKE_SEQ_END state, for more details on how and when WOR allows and blocks RF-related interrupts. For protocols not assigned to WOR, interrupts are always allowed.

The raw interrupt triggers themselves (the inputs to the AND gates), namely, tx_irq_trig and rx_irq_trig, are passed directly to WOR hardware, so that WOR state machine can react to RF operation status in realtime, and sequence through the states as specified by the current slot's descriptor.

WOR manages one other Link Layer interrupt, for non-RF-related notifications. Each protocol engine has an existing interrupt, WAKE_IRQ, to notify MCU on wakeup from Manual DSM mode (not Wake-On-Radio). See Section Manual DSM for more details on this DSM control scheme. For any protocol, Wake-On-Radio and Manual DSM are mutually-exclusive control schemes, so when a protocol is assigned to WOR, its WAKE_IRQ is re-purposed, and re-named, to WOR_IRQ.

Like the "allow/block" gating WOR uses to manage the RF-related interrupts, WOR uses 2 signals to either block or *force* a WAKE_IRQ / WOR_IRQ:

- **wor_block_wake_irq:** prevents the (legacy) WAKE_IRQ from asserting at DSM exit, this is *mandatory* for Manual DSM, but is *conditional* for WOR.
- **wor_force_wake_irq:** WOR needs to assert WOR_IRQ under 3 conditions (see below)

There are 3 conditions for which WOR needs to compel the Link Layer to assert WOR_IRQ. WOR maintains 3 read-only status flags to indicate each of these conditions. The status flags exist in XCVR (WOR) address space, even though the WOR_IRQ resides in the protocol-specific address space. The 3 conditions, and the associated status flag for each, are described in the following table.

- **WOR_NO_RF_FLAG:** START_TIME has been reached on a slot with FUNCTION = NO_RF and WAKE_ON_COMPLETE=1
- **WOR_MAX_SLOT_FLAG:** In Constant-Interval scheduling mode, the WOR_SLOT_COUNT[7:0] has reached WAKE_ON_NTH_SLOT[7:0]
- **WOR_DSM_EXIT_FLAG:** An unscheduled, asynchronous DSM "Early Exit" has occurred

To service an WOR_IRQ, the interrupt service routine should query the 3 read-only status flags to determine the cause of the MCU wake up. At least one of these flags will be set. It is possible that more than 1 of the flags is set. For the WOR_NO_RF_FLAG, WAKE_ON_COMPLETE would typically not be set on a NO_RF slot, since the purpose of NO_RF is to allow "DSM cascading"; only for debug purpose has this flag been provided. Servicing of the other 2 flags is beyond the scope of this document.

Clear the WOR_IRQ by writing a 1 to this bit in the protocol-specific address space. However the 3 WOR flags are "sticky". Any flag which is set will remain set until a new WOR session has begun, which means taking WOR to DISABLED by setting SLOTS_USED=0, followed by re-enabling by setting SLOTS_USED>0.

In initializing the assigned Link Layer for WOR operation, it is *essential* that software enable (unmask) the 3 RF-related interrupts, as well as WAKE_IRQ / WOR_IRQ. It is equally essential that all non-WOR related interrupt sources are *disabled* when a protocol is assigned to WOR, to prevent unnecessary MCU wakeups. The following tables describe all the Link Layer interrupts, their controllability (or lack of) by WOR, and the required enabling/unmasking by software.

INTERRUPT CONFIGURATION (802.15.4)

INTERRUPT	WOR CONTROL	SW INIT	COMMENTS
TXIRQ	gated by wor_irq_allow	ENABLE	Asserted after packet transmission when: WAKE_ON_COMPLETE || WAKE_ON_TIMEOUT
RXIRQ	gated by wor_irq_allow	ENABLE	Asserted after good packet reception when: WAKE_ON_COMPLETE || WAKE_ON_TIMEOUT
SEQIRQ	gated by wor_irq_allow	ENABLE	1. Asserted after any good packet transmission or reception when: WAKE_ON_COMPLETE || WAKE_ON_TIMEOUT 2. Asserted after bad (or no) packet reception when: WAKE_ON_TIMEOUT
CCAIRQ	unaffected	DISABLE	Not used in WOR

Table continues on the next page...

Table continued from the previous page...

INTERRUPT	WOR CONTROL	SW INIT	COMMENTS
RXWTRMRKIRQ	unaffected	DISABLE	Not used in WOR
FILTERFAIL_IRQ	unaffected	DISABLE	Not used in WOR
PLL_UNLOCK_IRQ	unaffected	OPTIONAL	If enabled, would cause an immediate MCU LLWU on occurrence of a PLL unlock event.
WAKE_IRQ / WOR_IRQ	blocked by wor_block_wake_irq, forced by wor_force_wake_irq	ENABLE	1. Asserted when START_TIME reached on a slot with FUNCTION=NO_RF and WAKE_ON_COMPLETE=1 2. Asserted in CI mode when WOR_SLOT_COUNT==WAKE_ON_NTH_SLOT 3. Asserted after an unscheduled "DSM Early Exit"
TMRxIRQ	unaffected	DISABLE	Not used in WOR
TSM_IRQ	unaffected	DISABLE	Not used in WOR

INTERRUPT CONFIGURATION (non-802.15.4)

INTERRUPT	WOR CONTROL	SW INIT	COMMENTS
TX_IRQ	gated by wor_irq_allow	ENABLE	Asserted after packet transmission when: WAKE_ON_COMPLETE || WAKE_ON_TIMEOUT
RX_IRQ	gated by wor_irq_allow	ENABLE	Asserted after good packet reception when: WAKE_ON_COMPLETE || WAKE_ON_TIMEOUT
SEQ_END_IRQ	gated by wor_irq_allow	ENABLE	1. Asserted after any good packet transmission or reception when: WAKE_ON_COMPLET

Table continues on the next page...

Table continued from the previous page...

INTERRUPT	WOR CONTROL	SW INIT	COMMENTS
			E || WAKE_ON_TIMEOUT 2. Asserted after bad (or no) packet reception when: WAKE_ON_TIMEOUT
NTW_ADR_IRQ	unaffected	DISABLE	Not used in WOR
T1_IRQ	unaffected	DISABLE	Not used in WOR
T2_IRQ	unaffected	DISABLE	Not used in WOR
PLL_UNLOCK_IRQ	unaffected	OPTIONAL	If enabled, would cause an immediate MCU LLWU on occurrence of a PLL unlock event.
WAKE_IRQ / WOR_IRQ	blocked by wor_block_wake_irq, forced by wor_force_wake_irq	ENABLE	1. Asserted when START_TIME reached on a slot with FUNCTION=NO_RF and WAKE_ON_COMPLETE=1 2. Asserted in CI mode when WOR_SLOT_COUNT==WAKE_ON_NTH_SLOT 3. Asserted after an unscheduled "DSM Early Exit"
TSM_IRQ	unaffected	DISABLE	Not used in WOR
RX_WATERMARK_IRQ	unaffected	DISABLE	Not used in WOR

55.4.6.2.3.9 Channel Hop Table

Typically, in either Timeslot or Constant-Interval scheduling mode, RF operations on consecutive slots use different RF channels. Using Wake-On-Radio to program slot specifications for the next few RF operations in advance, would be of little value if the MCU had to wake up on every slot to program the assigned Link Layer's CHANNEL_NUM register. Thus, Wake-On-Radio provides hardware automation for channel hopping on a slot-by-slot basis.

A table of channel hop frequencies is maintained by hardware. The number of entries in the table (hop sequence length) is programmable. Prior to starting a WOR session, software will program the desired sequence of channel numbers into the table. After WOR is activated by setting SLOTS_USED>0, The WOR state machine controls the index (read pointer) into the table. The index is typically incremented at the end of every slot, when the slot counter is incremented. The increment is +1 modulo "hop sequence length". The RF operation on the current slot, is then carried out on the RF channel pointed to by the hop table index. If necessary, WOR or Link Layer software can update the hop table index at any time.

Rather than adding a channel number field to the descriptor, to save area, the hop table is decoupled from the descriptor table. The hop tables utilizes a small section of Packet RAM which is not otherwise used by the supported WOR protocols. WOR uses 128 bytes of Packet RAM for the channel hop table.

The channel information entered into the hop table, can use the (legacy) 7-bit "mapped channel" format, which allows channels to be specified on a 1MHz boundary; or one of 2 new 16-bit formats that allows channel frequency of higher resolution. The option for the legacy format, provides backwards compatability to existing software, which does not use WOR, to program RF channel information the way it does today, without any changes. The inclusion of the new formats allows additional flexibility for programming channels which deviate from a 1MHz boundary, coupled with the automation built in to WOR. The format selected, determines:

1. how the hop table is organized in RAM
2. how software programs the hop table
3. how the WOR parses the table during execution of the current slot
4. and how the PLL digital block interprets the frequency word extracted from the table.

Selecting the hop table configuration is by way of the HOP_TBL_CFG[2:0] register. Based on this setting, the source of channel information, can either be the CHANNEL_NUM[6:0] bit register that exists in each protocol engine today (legacy), or the new hop table, according to the following table:

Source of RF channel information, based on HOP_TBL_CFG[2:0]:

0: Assigned Link Layer's CHANNEL_NUM register (legacy)

1: WOR Hop Table

2: WOR Hop Table

3: WOR Hop Table

4-7: Reserved

To allow for the legacy option, the 3 protocol-specific CHANNEL_NUM registers are now routed to WOR, which multiplexes them with the hop table output based on HOP_TBL_CFG and WOR_PROTOCOL settings. WOR then outputs 3 16 bit frequencies words, one per protocol, to PLL Digital, conveying the required channel information.

Hop table organization is determined by HOP_TBL_CFG, as per the following table:

Hop Table Organization, based on HOP_TBL_CFG[2:0]:

0: N/A

1: Table size is 128 entries. Two 7-bit CHANNEL_NUM fields are packed into a 16-bit hop frequency word for RAM storage

2: Table size is 128 entries. Each 16-bit hop frequency word in RAM contains a specification for a single RF channel

3: Table size is 128 entries. Each 16-bit hop frequency word in RAM contains a specification for a single RF channel

4-7: Reserved

The content of the hop frequency word provided to PLL Digital, and how it is interpreted by that block, also varies with HOP_TBL_CFG, as per the following table:

PLL Digital Interpretation of Hop Frequency Word, based on HOP_TBL_CFG[2:0]:

0: Lower 7 bits of hop frequency word contain a mapped channel (legacy format). Upper 9 bits are 0

1: Lower 7 bits of hop frequency word contain a mapped channel (legacy format). Upper 9 bits are 0

2: All 16 bits are valid. See Chapter *PLL Digital* for interpretation

3: All 16 bits are valid. See Chapter *PLL Digital* for interpretation

4-7: Reserved

The following diagram summarizes the routing of channel hop signalling amongst the supported protocol engines, the WOR scheduler, the PLL Digital, and Packet RAM. It also shows the 2 hop table configurations, and hop frequency word generation, based on HOP_TBL_CFG. The existing (legacy) configuration is on the left, and the new, WOR-based configuration is on the right.

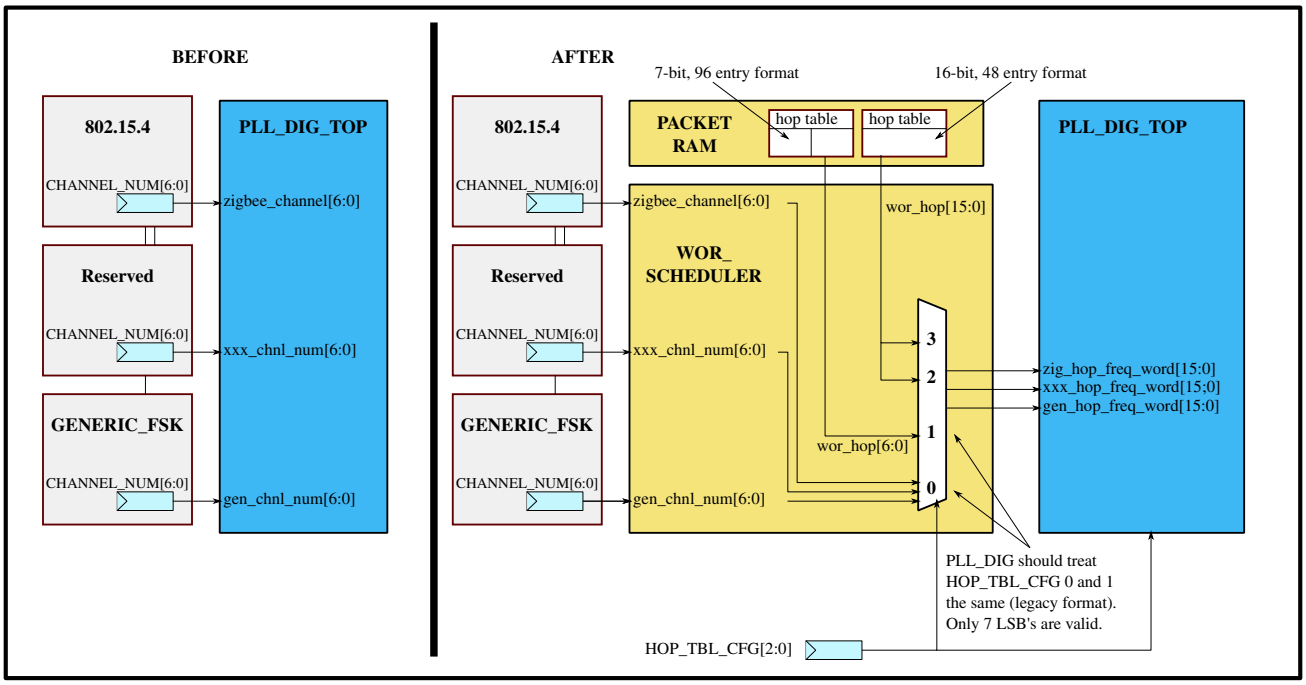


Figure 254. CHANNEL HOP SIGNAL ROUTING AND CONFIGURATION

The number of entries in the table is determined by programmable register HOP_SEQ_LENGTH[7:0]. If HOP_TBL_CFG=1, HOP_SEQ_LENGTH determines the number of 7-bit CHANNEL_NUM entries in the Packet RAM table, which are packed 2 per RAM word. HOP_SEQ_LENGTH must not exceed 128 in this case. If HOP_TBL_CFG=2 or HOP_TBL_CFG=3, HOP_SEQ_LENGTH determines the number of 16-bit format entries in the Packet RAM table, one per RAM word. HOP_SEQ_LENGTH must not exceed 128 in this case.

At the start of new WOR "session" (setting SLOTS_USED to a non-zero value), WOR hardware initalizes the table index (read pointer) to 0. (Software can modify this index, see below.) After slot execution completes, as the WOR state machine increments the slot counter to service the next slot, the hop table index is incremented by +1 (modulo HOP_SEQ_LENGTH). An exception is made to this rule, if the FUNCTION field of the slot descriptor is set to NO_RF, the hop index is *not* incremented when the slot counter is incremented to advance to the next slot. The NO_RF function is used for "DSM cascading".

Software can take control of the hop table index at any time, by writing a new desired index to the NEW_HOP_IDX[6:0], and with the UPDATE_HOP_IDX register set to 1. These 2 steps can be performed as a single, atomic IPS bus write. WOR hardware will update the index accordingly. From this point forward, the hop-table hardware incrementing rules, discussed in the previous paragraph, apply. This way, software can execute a one-time "jump" in the hop table indexing.

The read-only register WOR_HOP_IDX[6:0] allows software to read the instantaneous, hardware-controlled hop table index at any time. Similarly, the read-only register WOR_HOP_FREQ_WORD[15:0] allows software to read the instantaneous, hardware-generated frequency word sent to PLL digital. This word is the output of the hop table configuration selection mux shown in the diagram above, the protocol-specific word currently selected by WOR_PROTOCOL.

A diagram showing the RAM-based, 128-entry hop table (HOP_TBG_CFG=1), where it is situated in Packet RAM space, how the channel information is organized in the table, and how WOR hardware indexes the table, is shown below.

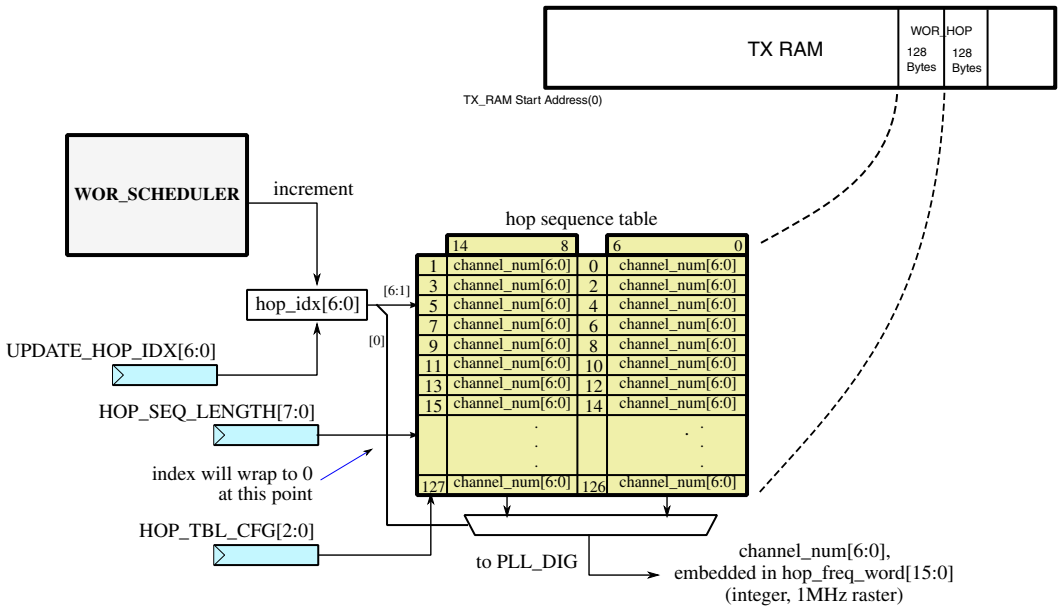


Figure 255. 128-ENTRY HOP TABLE

A diagram showing the RAM-based, 128-entry hop table (HOP_TBG_CFG=2 or 3), where it is situated in Packet RAM space, how the channel information is organized in the table, and how WOR hardware indexes the table, is shown below.

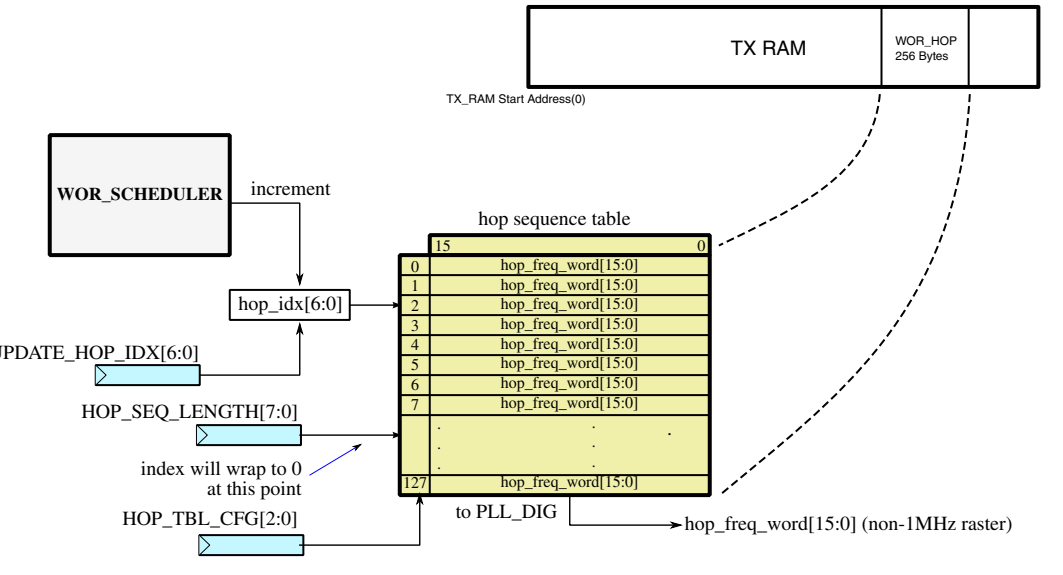


Figure 256. 128-ENTRY HOP TABLE

The 256 bytes of Packet RAM are reserved for the WOR hop table. The table is fixed in this position, and does not move based on HOP_TBL_CFG.

The WOR hop table should not be accessed (no writes or reads) while an RF operation is underway. There is no hardware to de-conflict RAM accesses with those originating in the protocol engines. Aside from this restriction, the hop table can be accessed at any time. A convenient time to modify the hop table, if required, would be during the WOR WAIT_RESUME state.

Follow the instructions below to program the WOR hop table.

1. Program the hop table

2. Program HOP_CTRL[HOP_SEQ_LENGTH] with the desired size of the hop sequence
3. Program HOP_CTRL[HOP_TBL_CFG] to select the desired table configuration and enable hopping during WOR

55.4.6.2.3.10 DSM Early Exit

As described in Section State Machine, for every slot, WOR hardware computes the duration of the upcoming DSM cycle based on descriptor START_TIME, current time, and window-widening, during the COMP_DURATION1 and COMP_DURATION2 states. WOR outputs the desired duration on wor_dsm_duration[23:0]. The Radio Mode Controller (RFMC) then executes this DSM for the prescribed number of 32.768KHz clock cycles, and when complete, sends the actual duration back to WOR using rfmc_sleep_duration[23:0]. Normally, these 2 durations will match precisely.

Conditions exist where the DSM cycle may occasionally need to be cut short, due to circumstances beyond the control of the radio. The SoC, starting with the MCU, must be awakened out of its low power state before the WOR DSM duration expires. When this happens, software handling the event which triggered the wakeup, will determine if the radio needs to be awakened as well. If this is the case, software will set the RFMC_WOR_WKUP bit. This will trigger an unscheduled wakeup process out of DSM by the RFMC deep sleep state machine. The wakeup will complete before the wor_dsm_duration[23:0] time has expired. RFMC will capture the *actual* DSM duration, cut short by the early exit, into rfmc_sleep_duration[23:0]

After DSM wakeup, WOR hardware always performs this check during the COMP_WW state:

rfmc_sleep_duration[23:0] == wor_dsm_duration[23:0]

If this check fails, WOR hardware declares an "DSM Early Exit", and takes the following steps:

1. Forces the assigned Link Layer to assert its WAKE_IRQ / WOR_IRQ (see Section Interrupt Management)
2. Sets the WOR_DSM_EXIT_FLAG in WOR space
3. Transitions to the WAIT_RESUME state and waits for software to service the WOR_IRQ.

Software should then take whatever steps are necessary to configure the radio to respond to the conditions which triggered the early exit. Servicing of a DSM early exit interrupt is beyond the scope of this document.

55.4.6.2.3.11 Manual DSM

Prior to Wake-On-Radio, implementation of the 2.4GHz used a software-driven, "manual" DSM. This DSM control scheme does not have any of the hardware automation associated with WOR, and software must perform such tasks as computing the Event Timer addback time, window-widening, etc.

Manual DSM still exists as an alternative to Wake-On-Radio. Manual mode can be of value in a multiprotocol setting. See Section Multiprotocol for more details.

Any of the 3 protocols supported by Wake-On-Radio, can utilize Manual DSM as an alternative to Wake-On-Radio. In short, Manual DSM works by programming a desired DSM start time into the ENTER_TIME register, and a desired DSM wake time into the WKUP_TIME register. The ENTER_TIME and WKUP_TIME registers are referenced to the 32.768KHZ TIMER, the radio's low-power (DGO) timer. ENTER_TIME, WKUP_TIME, and TIMER reside in RFMC address space.

When a match on TIMER to ENTER_TIME occurs, RFMC places the radio in deep sleep mode. The RF Oscillator is shut off, and the Link Layer's Event Timer is frozen in place. When a match on TIMER to WKUP_TIME occurs, the Event Timer resumes from where it left off. The RF Oscillator, is re-started as part of the DSM wake up process, and is running before WKUP_TIME is reached. At the instant the Event Timer resumes, the Link Layer asserts its WAKE_IRQ. The MCU should use the WAKE_IRQ ISR to compute the precise amount of time the radio was in DSM, convert this value to microseconds, and add this value to the current Link Layer Event Timer, to restore the timer to where it would have been, had DSM not occurred. There is a 1us loss-of-precision in the Event Timer timebase incurred on every DSM wakeup. This timebase error is random, and averages out over many DSM cycles.

Regarding the DSM wakeup process, there is additional programmability to configure the lead time for power-domain wakeup and oscillator re-start, relative to DSM exit (WKUP_TIME). This configurability is covered by the RFMC register. See the *Radio Mode Controller* chapter for more details.

55.4.6.2.3.12 Multiprotocol

The 2.4GHz radio supports 4 protocols, any 2 of which can be enabled for multiprotocol use on an alternating, time-shared basis. Both of the active protocols may require DSM. The following section describes the rules to be applied when assigning a protocol to a DSM control scheme.

The radio has 3 DSM control schemes:

1. Bluetooth LE-internal
2. Wake-On-Radio
3. Manual DSM

The Bluetooth LE-internal DSM controller applies to the Bluetooth LE protocol engine only, and is the only DSM option for Bluetooth LE.

Wake-On-Radio can be assigned to any one of the following protocols (see Section *Assign To Protocol*):

1. 802.15.4
2. GENERIC_FSK

Manual DSM can be assigned to any one of the following protocols (see Section *Manual DSM*):

1. 802.15.4
2. GENERIC_FSK

55.4.6.2.3.13 Hardware Multiplier

Wake-On-Radio requires 6 multiplies while executing each slot. The multiplications take place in the following WOR states (See Section State Machine):

1. COMP_DURATION1: to compute the upcoming DSM duration, excluding window widening
2. COMP_DURATION2 and COMP_DURATION3: to compute the subtractor to the upcoming DSM duration, for window widening
3. COMP_ADDBACK: to compute the Event Timer addback value based on RFMC-provided DSM duration
4. COMP_WW: to compute the DSM-component of window-widening incurred during the just-completed DSM cycle
5. DRIFT_CAL: to compute the clock drift

For maximum area and power saving, there is 1 hardware multiplier instance, which is timeshared for the 6 multiplies. Also, because the state machine has been architected to allow sufficient time to perform these computations before the results are needed, the multiplier uses a serial, add-and-accumulate construction. The multiplicands are 30 bits and 20 bits. The 30 bit operand is appropriately sized for the DSM_DURATION_NO_WW[23:0] * WW_DSM_FACTOR[5:0]. The 20 bit operand is appropriately sized to achieve the precision of the required calculations, as described in the earlier sections.

For each calculation, the multiplier requires 20 clocks (20 shifts). Each multiply takes place within the confines of a single state of duration 1µs. There are 32 periods of the reference oscillator during that period (26 for a 26MHz crystal), more than enough to complete the multiply. The clock to the multiplier runs at the reference oscillator rate, but runs only during the 6 1µs states listed above, to maximize power savings.

Depending on the calculation, the required precision, the positioning of the integer bits within the product, etc., the bits of the product tapped for each calculation varies. Also, depending on the calucation, combinational rounding or truncation of the product occurs, external to the multiplier.

55.4.6.2.3.14 Clocks and Gating

Because of the intensive interaction between Wake-On-Radio and 3 protocol engines it supports, the WOR hardware runs at 1MHz. This facilitates signalling between the WOR and the assigned Link Layers, with no additional pulse-stretching or signal-conditioning required. Like the protocol engines and the TSM, the CGM's cg1m_en gating signal is used to generate the common 1MHz, so the phasing of the clocks across all the communicating modules remains in sync.

All WOR clocks are conditioned on WOR_EN=1. This bit should be set, and remain set, whenever Wake-On-Radio is in use (it should not be toggled or modified dynamically by software, since module clock-gating is based on it). This bit should not be set if WOR is not in use; that way, the WOR hardware block will consume zero dynamic power.

In total, there are 4 clocks used by WOR hardware:

1. **ipg_clk_wor_1m**: Most of the WOR hardware runs off this 1MHz clock, including the state machine, descriptor processing, and hop table indexing. All link layer interface outputs are generated with this clock, and link layer inputs sampled by it.
2. **ipg_clk_wor_ww_1m**: Identical in timing to ipg_clk_wor_1m, but used only by functions that require window widening to be enabled. The additional clock gating granularity saves power when widening is disabled.
3. **ipg_clk_wor_comp**: This is the "compute clock" used by the serial multilier. This clock runs at the reference oscillator frequency (derivative of xcvr_ref_clk_gated), but only during the 4 1µs-long states (per slot) during which the multiplier is in use; See Section Hardware Multiplier)
4. **ipg_clk_wor_timestamp**: This is the "timestamp clock". Because the protocol engines determine the timestamp-capture timing, based on AA/SFD detection, the clock-gate enable for the capture is sent to the CGM, which routes the resulting timetamp clock itself to WOR so that it can update its TIMESTAMP1,2, and 3 registers. This clock will typically pulse once per RX slot, and not at all for slots of other types.

All gating for the WOR clocks is performed in CGM. WOR provides the gating-enable signal, and CGM provides the clock.

All WOR clocks are in the Reference Oscillator domain. There are no clock domain crossings inside the module.

No WOR critical timing paths are anticipated, nor any timing exceptions (e.g., multicycle paths).

55.4.6.3 Radio Bit Manipulation Engine

55.4.6.3.1 Introduction

This section provides an introduction to the Radio Bit Manipulation Engine (RBME) module. The RBME module is a group of configurable bit manipulation modules.

55.4.6.3.1.1 Features

- FEC: Forward Error Correction
  - Support for Bluetooth LE and Bluetooth LE LR
    - Requires a two pass FEC, over the AA+CI and over the payload + CRC
    - These are called FECBlock1 and FECBlock2 respectively
- Spreading
  - Hard Spreader
  - Inclusive of hard Manchester encode and decode
- Configurable CRC and Whiten (CRCW)
  - CRC
    - Supports direct method only
    - Supports input byte reflection
    - Final XOR and checksum byte reflection
    - Multiple CRCs per packet is not currently supported
    - Programable polynomial and seed values
  - Whitening Block
    - Hard whitening and de-whitening modes only
    - Programable polynomial and seed values

- Galois or Fibonacci type LFSR

— CRC can switch on before or after the second whitening module

- RAM interface

— Debug

— Frame processing

NOTE

The RBME is fully functional with GEN-LL, while totally bypassed when 15p4-LL is activated. If the NBU is the active link-layer, the RBME only be enabled for receiving long-range (coded) packets. In this case, the RBME responsible for despreading and do FEC decode. The de-whitening and CRC calculation should be done inside the NBU

55.4.6.3.1.2 TX-RBME Diagram

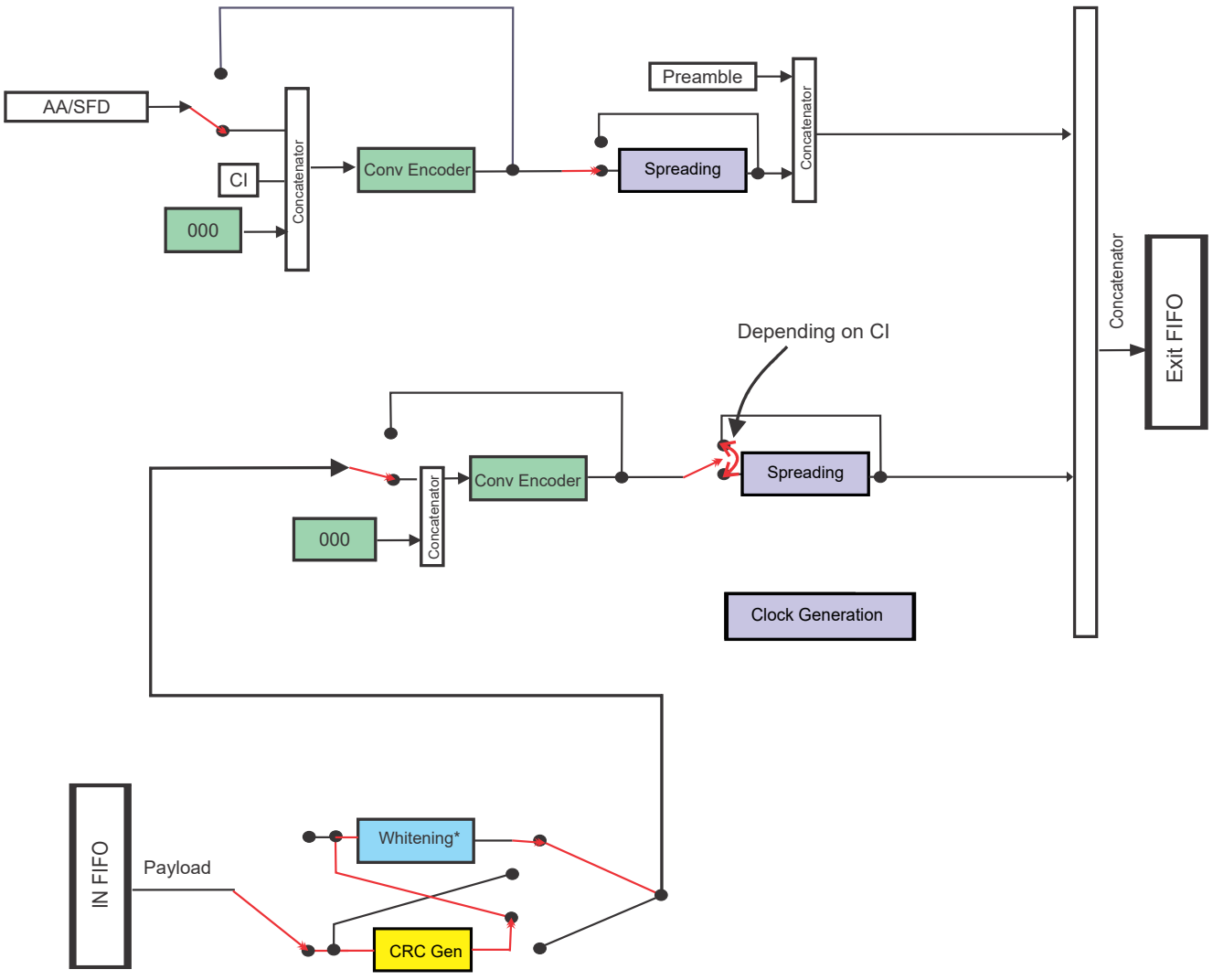


Figure 257. TX Diagram

55.4.6.3.1.3 RX-RBME Diagram

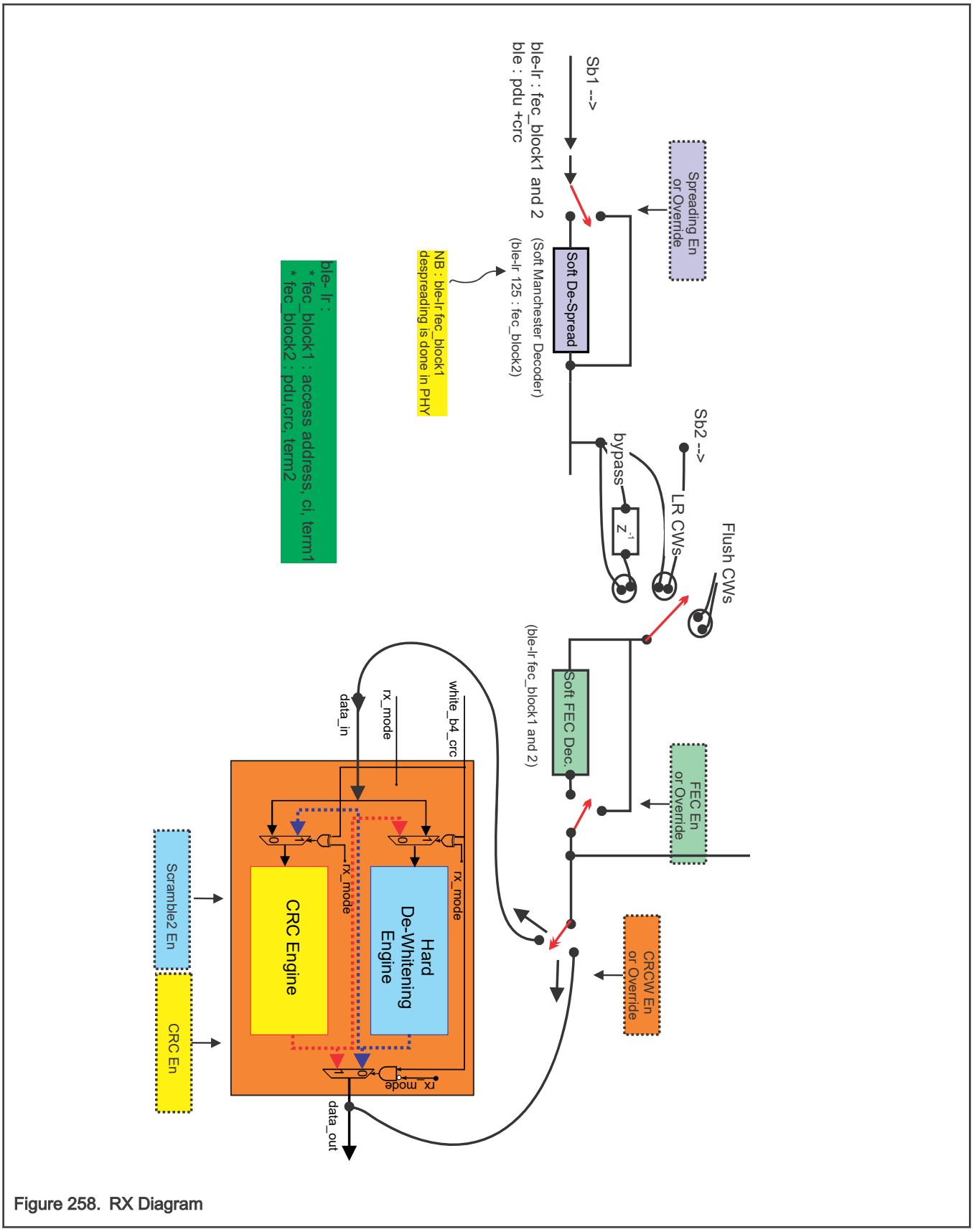


Figure 258. RX Diagram

55.4.6.3.2 Processing Order

TX-RBME Diagram shows the order of processing for the Transceiver (TX). The Receiver (RX) has the reverse order of processing shown in RX-RBME Diagram. The order of processing and the number of modules involved depends on:

1. The enable register for the block
   - Used to configure a block statically
   - Allows for configurability for a pre-known setup of the RBME
   - All TX modules only require a static enable as the configurations are known in advance
2. Override of the block (RX only)
   - Allows a dynamic override of the block enabling it regardless of the enable register
   - It requires that the appropriate override enable register be asserted
   - It does not disable the block: if a block is to be enabled dynamically
     - Set enable register off
     - Set override enable on
3. Bypass of the (RX only)
   - It has the highest priority
   - It is used only internally
   - It is dynamic

55.4.6.3.2.1 RX Bluetooth LE LR Example

When the Coding Indicator (CI) bits are demodulated and decoded, the mode of Bluetooth LE LR is known. Therefore, dynamic override needs to be used for de-spreading. For this reason, the spreading is disabled and the override is enabled for the RX. Furthermore, the receive for Bluetooth LE LR requires special handling as the FEC is required twice. The following steps describe the RX flow.

1. RBME input mode from link layer has been set to 1 to let RBME prepare for a long range (LR) configuration before AAFound.
2. RBME_en is asserted once AAFound, from the PHY acquisition, has been asserted.
3. After reading the decoded CI bits, the PHY sends a one-bit CI indication.
   - CI_indicator NOT asserted for 125 ksps reception
   - CI_indicator asserted for 500 ksps reception
4. The FEC processing must have the packet length from the link layer before the end of the packet for the CRC to be applied correctly. The processing counts the soft bits till Cnt = FECCounter2.
   a. FECCounter2 equals MaxPacketSize register until the LL updates the Frame length
   b. FECCounter2 = FrameLengthBits + NPHRBits + NCRCBits + FineTuneFECFlushBits
      i. FrameLengthBits is the number of bits from the link layer
      ii. NPHRbits = 16 for Bluetooth LE LR
      iii. NCRCbits = length of the CRC
      iv. FineTuneFECFlushBits is a register that allows for customization of the decoder length

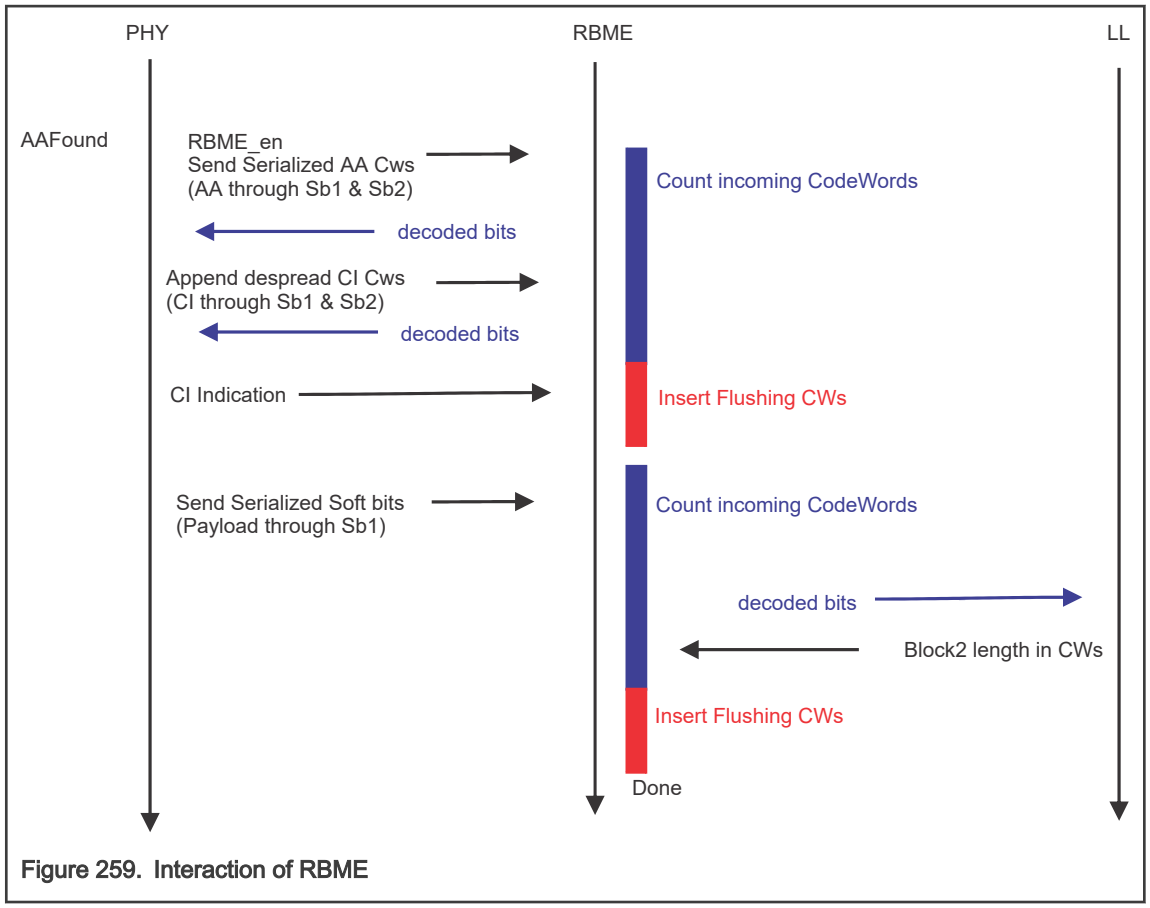


Figure 259. Interaction of RBME

The above figure (Interaction of RBME) shows the interaction among link layer, RBME, and PHY. The following table illustrates the Bluetooth LE LR packet format that RBME RX must account for. Note the separation into FEC Block1 and FEC Block2. Other important aspects are:

- AA+CI are spread with a spreading pattern of length 4
- AA+CI+TERM1 are combined as FECBlock1 as they need to be decoded together
- Header+Payload+CRC+TERM2 are combined as FECBlock2 as they need to be decoded together
- The de-spreading of AA+CI is done in the PHY for speed
- FECBlock2 may or may not need de-spreading depending on whether 125 ksps or 500 ksps are transmitted

Table 402. Standard Bluetooth LE LR Frame Format

10 Octets	4 Octets	2 bits	3 bits	Variable in Octets			24 bits	3 bits
Pre	AA	CI	TERM1	PDU			CRC	TERM2
				Header (16 bits)		Payload		
					Length (8 bits) L0:L7 Payload field length in octets. LSB first	ADV => 1:255 octets Data => 0:251 octets		
	FEC BLOCK 1			FEC BLOCK 2				

55.4.6.3.2.2 Frame delimiter before CRC Protocol (FCP) Example

An example of FCP frame packet is shown in the table below. It has the distinct characteristic that the Start Frame Delimiter (SFD) is passed through the CRC. This complicates the processing for the RX as there are two possibilities:

1. The ideal SFD is assumed for processing the CRC.
2. The decoded SFD is provided by the PHY.

   RBME assumes option 1, that is, the CRC presumes an ideal SFD. That ideal SFD is injected at the CRC buffer for processing, so no SFD bit errors will ever be caught. Another important feature of FCP is that whitening from a TX perspective is done before the CRC.

Table 403. FCP Frame Packet

1 octet	2 octets	5 octets		Variable in octets (8 -25)	2 octets
		RBME PHR		RBME Payload	
Pre	Net add	Device Address	Control (1 octet)	Data	CRC

55.4.6.3.3 RBME RAM And Interface

55.4.6.3.3.1 RAM Interface for RBME

RAM interface for RBME is important for debug and pre/post process frame. The only interrupt of RBME is generted by RAM interface. When packet RAM is filled completely with payload, the interrupt is sent to MCU in case of RAM_IF_IE set. The following figure (RBME TX RAM Interface) shows all the options, however, for simplicity the implementation of the RAM interface is in two developmental stages:

1. RAM-RBME-RAM only

   This interface is the simplest as it presumes that data into RBME and out of RBME comes and returns to the packet RAM.
2. RAM-RBME TX

   This interface writes single bit data from the packet RX RAM into RBME TX. The interface is like that of the generic LL using fast clock implicitly for the data rate and then switching internally to OTA derived clocks. For example, if FEC encoding alone is done, the input buffer is normally read at twice the OTA rate. TX_DIG normally dictates the OTA rate that is desired for the implementation. For the RAM equivalent, the read rate can be higher. The rate of RBME output may be higher than the equivalent OTA rate for the use case but must be smaller or equal to ref_clk/encoding_factor. The values for encoding factor are for example:

   a. For FEC encoding only

   b. 32 if a spreading factor of 16 was required and FEC was also required

   In Step2 of next figure (RBME RX RAM Interface), the data from TX RAM pre-processed is sent to link layer as normal case without RAM interface. Then RBME does the same way as usual.

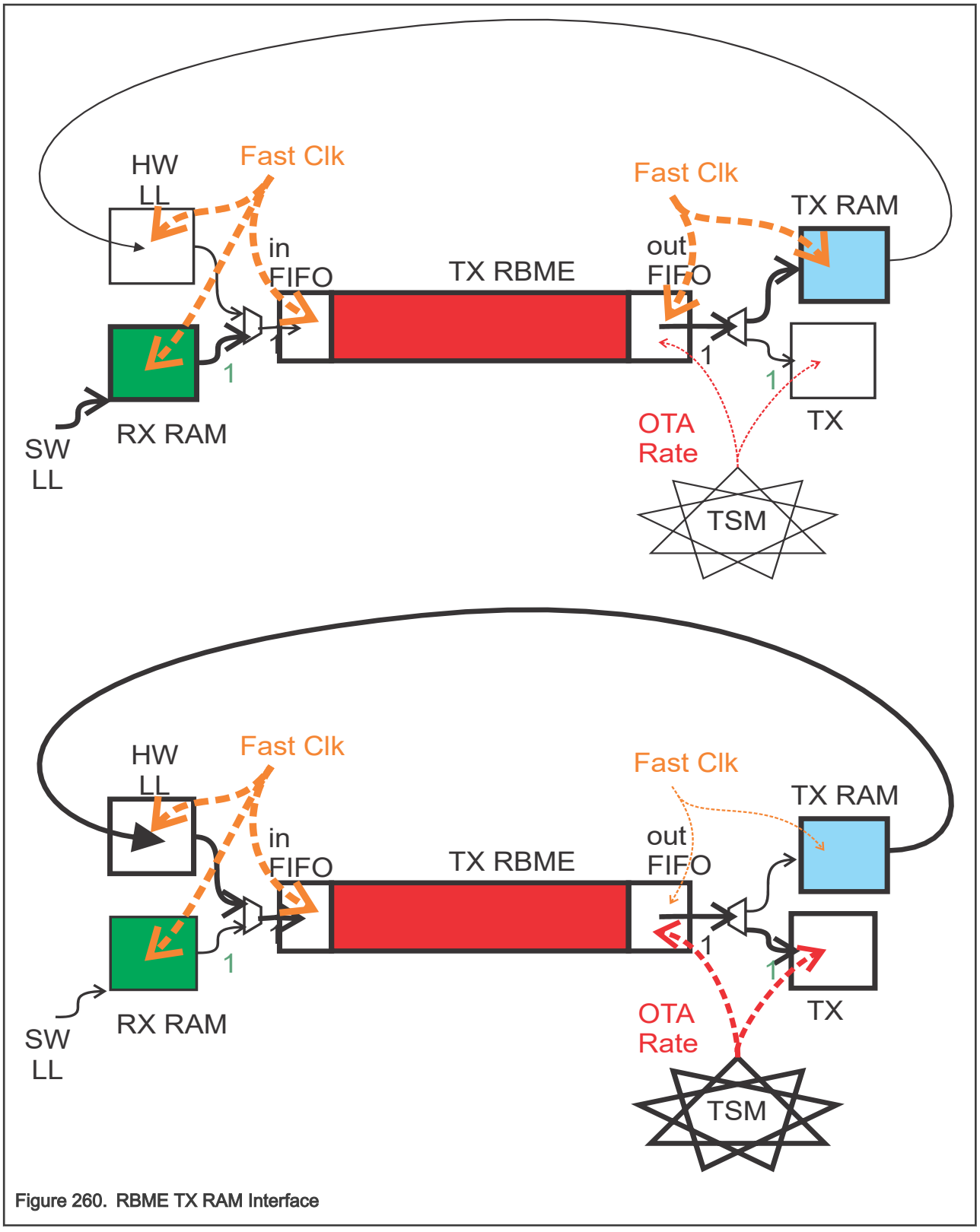


Figure 260. RBME TX RAM Interface

3. RAM-RBME RX

RBME RX processing: PHY ->[Despreading->FEC decoding->CRC & hard de-whitening]->LL Example: unsupported hard de-whitener with a different polynomial, interleaving and FEC plus soft de-spreading and CRC processing can be done in RBME RX.

1. First Pass: Use the PHY to RBME interface to process the data as normal but save the data to TX packet RAM before the de-whitener
    a. FEC decoding is enabled
    b. CRCW is bypassed
    c. Use software processing to do the soft de-whitening data
2. Second-Pass: Use the RAM interface to write 7-bit softbit words into RBME
    a. Do CRC decoding in RBME
    b. Use the RX RAM interface to read the CRC processed hard bit data

NOTE

A bypass from the PHY directly into packet RAM is needed if the de-spreader must be bypassed.

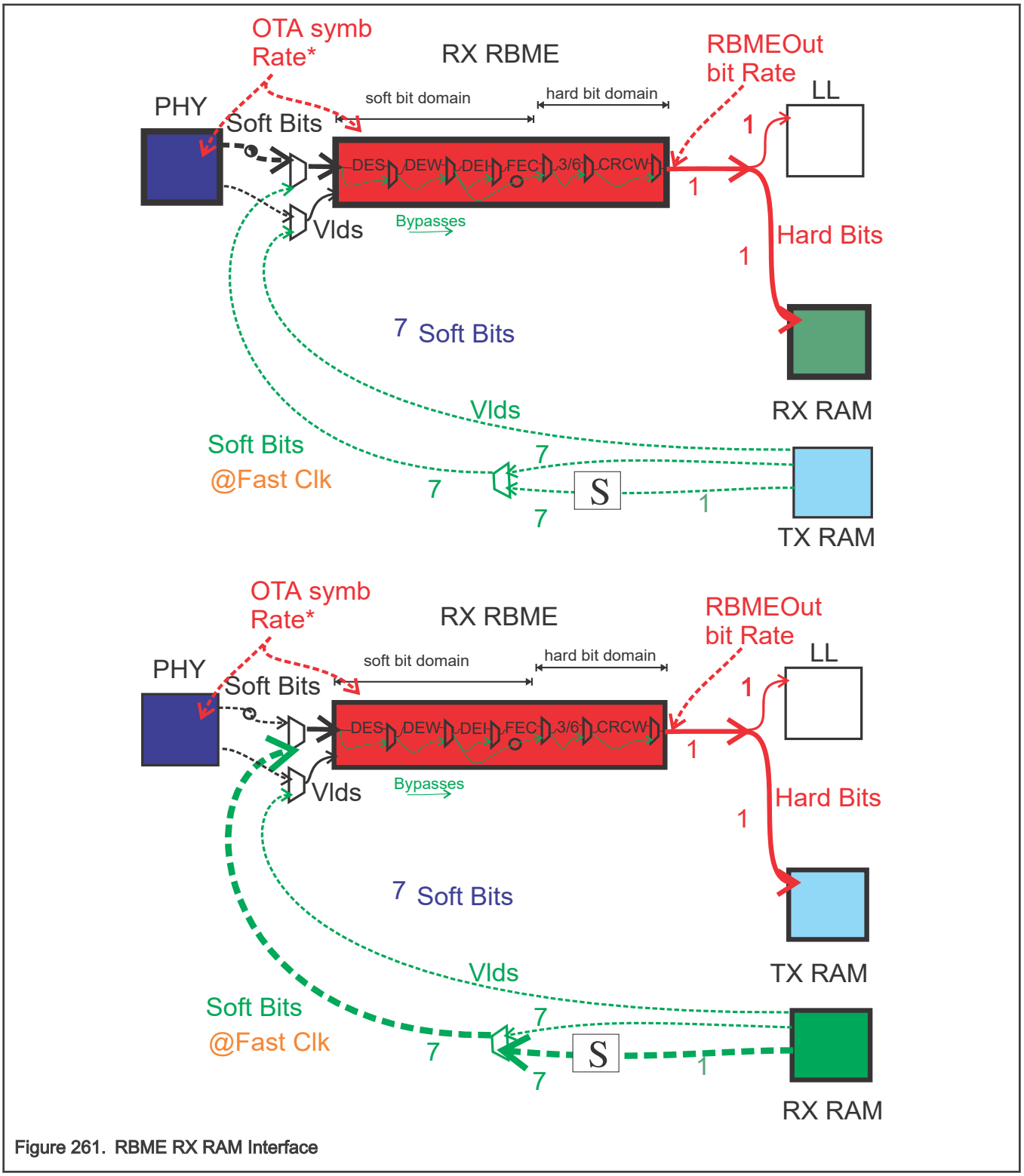


Figure 261. RBME RX RAM Interface

55.4.6.3.3.2 Internal Interfaces, Word Sizes, and Internal State Machines

For the TX, data and dataVld are single bits; for the RX and before the FEC, the data is in 7-bit words to describe softbits. The data format for CRCW for both TX and RX needs to be Boolean. In terms of clock valid management, every module internally

controls its output clock valid (as opposed to a central control). This methodology makes the modules easier to interact with each other. Examples:

1. When bypassing a module, the data valid is equally bypassed.
2. If de-spreading by four is used, the output data valid(s) are four times slower at the output.

The FEC Decoder input is in codeword (CW) pairs. This is normal for a rate ½ FEC. For Bluetooth LE LR, the de-spreading is done at the PHY so that the input to RBME is CW pairs at FrefFreq/2 for speed. This means that a second seven-bit input word needs to be separately added to the input. In summary, the input to the RBME RX is either a seven-bit input or two seven-bit inputs (both associated with the same input data valid signal). For Bluetooth LE LR only, the two seven-bit inputs are expected during the decoding of the AA and CI. For all other cases, there is a single seven-bit input. The RBME may also decode data from external memory, such data are expected to be serialized outside the RBME. The single Sb1 input can be used for this, alternatively, both Sb1 and Sb2 inputs can be used for faster processing. The enable control for a module is tabulated in Table 404. Note that a bypass signal has priority. Block bypasses are internal to the RBME. External override access on the other hand, that is, for the Soft-Whitening module and the FEC module, is allowed as long as the corresponding override enable register is set.

Table 404. RBME Module Enable Truth Table

Enable (Register)	OverrideEnable (Register)	Override	Bypass*	Block Enabled
True	True	True	True	False
True	True	True	False	True
True	True	False	True	False
True	True	False	False	True
True	False	True	True	False
True	False	True	False	True
True	False	False	True	False
True	False	False	False	True
False	True	True	True	False
False	True	True	False	True
False	True	False	True	False
False	True	False	False	False
False	False	True	True	False
False	False	True	False	False
False	False	False	True	False
False	False	False	False	False

NOTE

Bypass signal is uesd internally, user does not need to program it.

55.4.6.3.4 RBME register descriptions

55.4.6.3.4.1 RBME Base Address memory map

RBME base address: 48A0_6200h

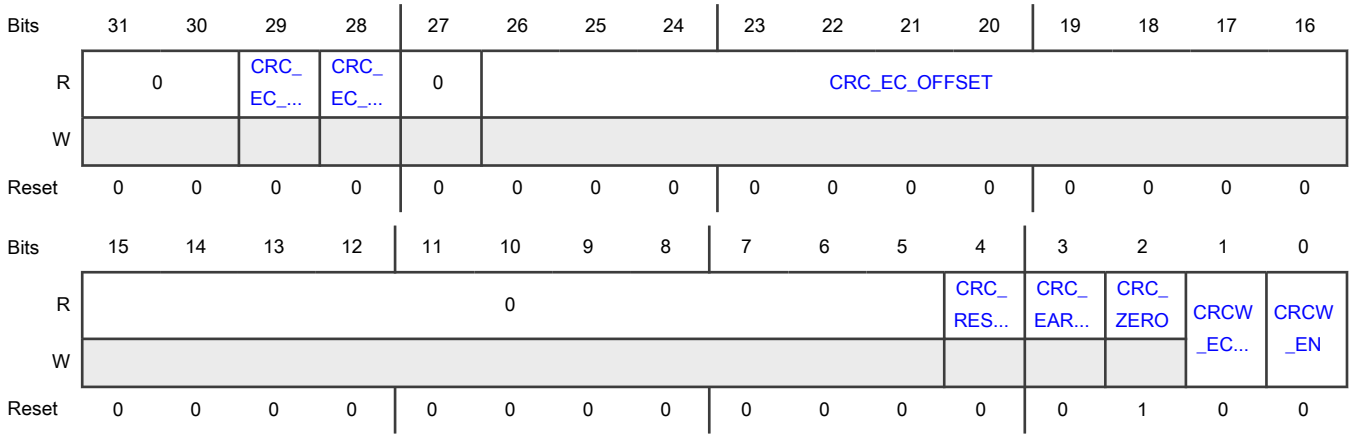
Offset	Register	Width (In bits)	Access	Reset value
0h	CRC/WHITENER CONFIG REGISTER (CRCW_CFG)	32	RW	0000_0004h
4h	CRC ERROR CORRECTION MASK (CRC_EC_MASK)	32	R	0000_0000h
8h	CRC RESULT (CRC_RES_OUT)	32	R	0000_0000h
Ch	CRC/WHITENER CONFIG 2 REGISTER (CRCW_CFG2)	32	RW	0000_0000h
10h	CRC CONFIGURATION (CRCW_CFG3)	32	RW	0000_0002h
14h	CRC INITIALIZATION (CRC_INIT)	32	RW	0000_0000h
18h	CRC POLYNOMIAL (CRC_POLY)	32	RW	1021_0000h
1Ch	CRC XOR OUT (CRC_XOR_OUT)	32	RW	0000_0000h
20h	WHITENER CONFIGURATION (WHITEN_CFG)	32	RW	01FF_0918h
24h	WHITENER POLYNOMIAL (WHITEN_POLY)	32	RW	0000_0021h
28h	WHITENER SIZE THRESHOLD (WHITEN_SZ_THR)	32	RW	0000_0800h
2Ch	FEC CONFIG REGISTER 1 (FEC_CFG1)	32	RW	0000_0300h
30h	RBME SOFT RESET REGISTER (RBME_RST)	32	RW	0000_0000h
34h	FEC CONFIG REGISTER 2 (FEC_CFG2)	32	RW	003F_7F10h
3Ch	SPREADER CONFIG REGISTER (SPREAD_CFG)	32	RW	000C_0200h
40h	WHITEN CONFIG REGISTER (WHT_CFG)	32	RW	0200_1000h
44h	PACKET SIZE REGISTER (PKT_SZ)	32	RW	0022_4000h
48h	LENGTH OF PHR CONFIG REGISTER (CRC_PHR_SZ)	32	RW	0000_0003h
4Ch	FCP SUPPORT CONFIG REGISTER (FCP_CFG)	32	RW	0000_0000h
50h	FRAME OVERRIDE SIZE REGISTER (FRAME_OVER_SZ)	32	RW	0000_0000h
54h	OVERRIDE OF FEC BLOCK SIZE REGISTER (FEC_BSZ_OV_B4SP)	32	RW	0000_0000h
58h	LEG0 CONFIG REGISTER (LEG0_CFG)	32	RW	F202_DD00h
5Ch	OVERRIDE PAYLOAD LENGTH REGISTER (NPAYL_OVER_SZ)	32	RW	0000_0000h
64h	PACKET RAM SOURCE ADDRESS (RAM_S_ADDR)	32	RW	0000_0000h
68h	PACKET RAM DESTINATION ADDRESS (RAM_D_ADDR)	32	RW	0000_0000h
6Ch	PACKET RAM INTERFACE CONFIG REGISTER (RAM_IF_CFG)	32	RW	0000_0000h

55.4.6.3.4.2 CRC/WHITENER CONFIG REGISTER (CRCW_CFG)

Offset

Register	Offset
CRCW_CFG	0h

Diagram



Fields

Field	Function
31-30 —	Reserved
29 CRC_EC_FAIL	CRC error correction fail This signal is cleared when *crc_init* is asserted. It is set after the completion of a CRC error correction calculation that does not find a valid data correction solution or if error correction was disabled and the CRC check found an error.
28 CRC_EC_DONE	CRC error correction done This signal is cleared when *crc_init* is asserted. It is set after the error correction logic completes processing the syndrome. It is immediately set after a packet is received if no error was detected or if error correction is disabled. Otherwise, it can take up to N system clocks after the packet is received to assert, with N = number of bits used in the CRC calculation, including the CRC value.
27 —	Reserved
26-16 CRC_EC_OFFSET	CRC error correction offset This value provides the byte offset within the data packet to which the CRC error correction mask should be XOR-ed. This value is valid if the *crc_ec_done* signal is asserted and the *crc_ec_fail* signal is not asserted. The offset includes only the bytes used in the CRC calculation, including the CRC value.
15-5 —	Reserved
4 CRC_RES_OUT_VLD	CRC result output valid CRC result output valid
3	CRC error correction fail

Table continues on the next page...

Table continued from the previous page...

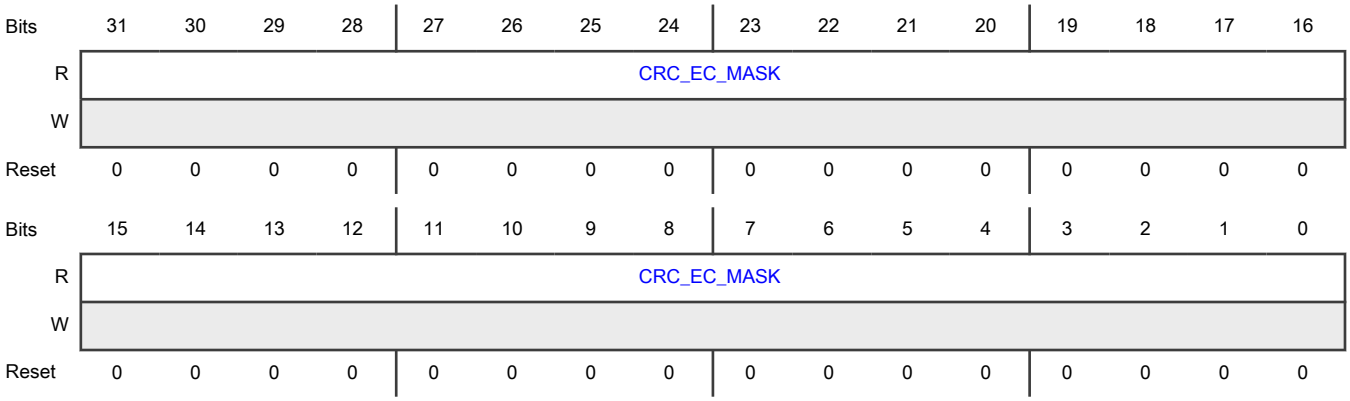
Field	Function
CRC_EARLY_FAIL	This signal is cleared when *crc_init* is asserted. It is set after the completion of a CRC error correction calculation that does not find a valid data correction solution or if error correction was disabled and the CRC check found an error.
2 CRC_ZERO	CRC zero This signal is asserted at any time when the CRC shift register contains a value of zero.
1 CRCW_EC_EN	CRC Error Correction Enable CRC Error Correction Enable
0 CRCW_EN	CRC calculation enable Input data bits loaded with this signal asserted are used in the CRC calculation.

55.4.6.3.4.3 CRC ERROR CORRECTION MASK (CRC_EC_MASK)

Offset

Register	Offset
CRC_EC_MASK	4h

Diagram



Fields

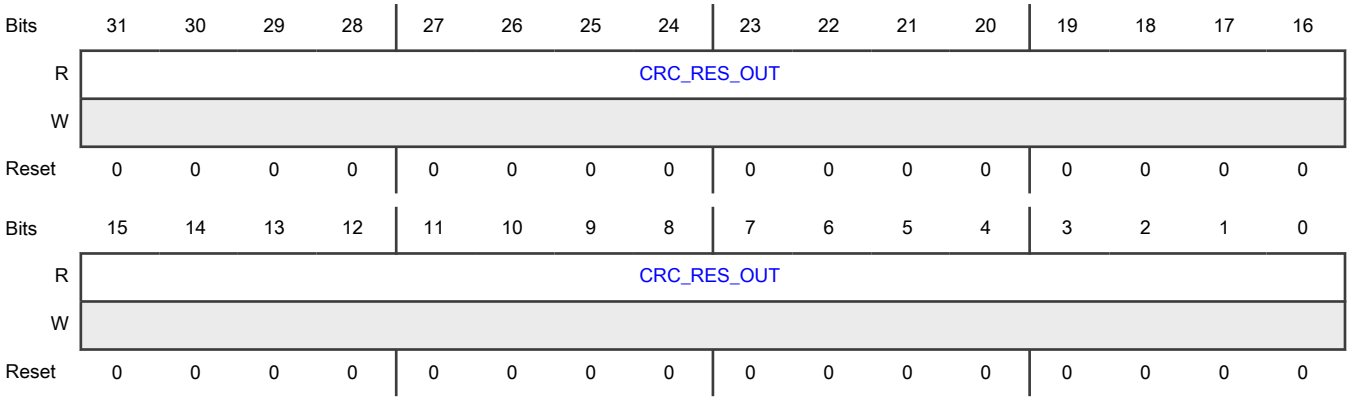
Field	Function
31-0 CRC_EC_MASK	CRC error correction mask This value provides a 32-bit XOR mask that must be applied to the input data packet to correct the burst errors detected by the error correction calculation. This value is valid if the *crc_ec_done* signal is asserted and the *crc_ec_fail* signal is not asserted.

55.4.6.3.4.4 CRC RESULT (CRC_RES_OUT)

Offset

Register	Offset
CRC_RES_OUT	8h

Diagram



Fields

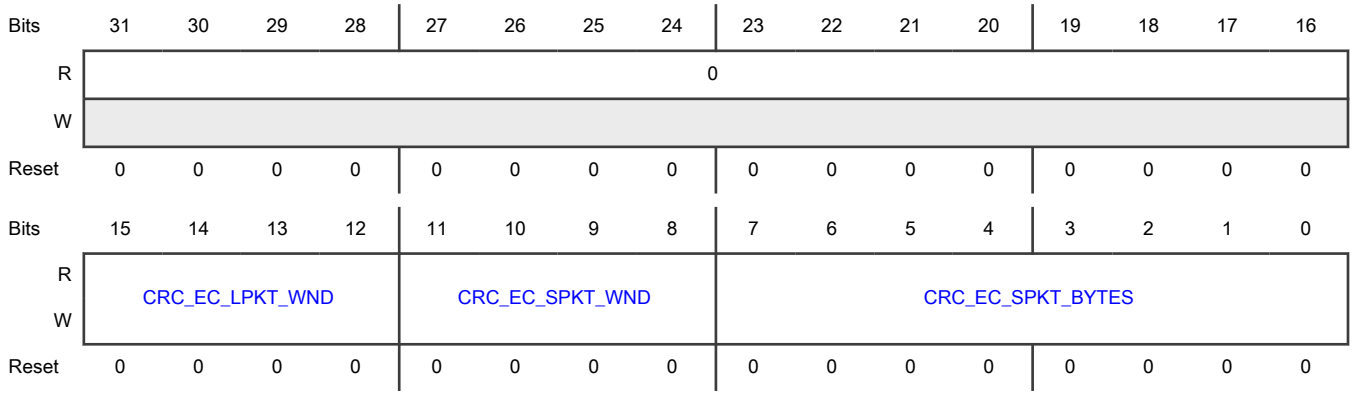
Field	Function
31-0 CRC_RES_OUT	CRC result output This bus provides the instantaneous value of the CRC shift register.

55.4.6.3.4.5 CRC/WHITENER CONFIG 2 REGISTER (CRCW_CFG2)

Offset

Register	Offset
CRCW_CFG2	Ch

Diagram



Fields

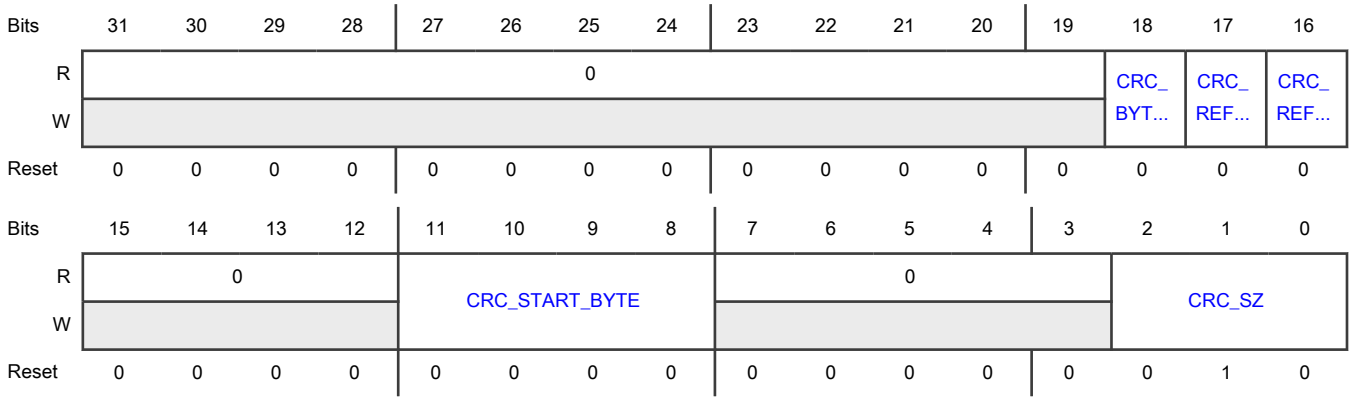
Field	Function
31-16 —	Reserved
15-12 CRC_EC_LPKT_WND	Error correction long packet burst error window This value determines the largest burst error corrected when long packets are processed. A value of zero will disable long packet error correction. The maximum value is 15.
11-8 CRC_EC_SPKT_WND	Error correction short packet burst error window This value determines the largest burst error corrected when short packets are processed. A value of zero will disable short packet error correction. The maximum value is 15.
7-0 CRC_EC_SPKT_BYTES	Error Correction Short Packet Bytes Short packets are defined as those where the number of bytes used in the CRC calculation, including the CRC value, do not exceed this value. The maximum value is 255.

55.4.6.3.4.6 CRC CONFIGURATION (CRCW_CFG3)

Offset

Register	Offset
CRCW_CFG3	10h

Diagram



Fields

Field	Function
31-19 —	Reserved
18 CRC_BYTE_ORD	CRC Byte Order 0b - LS Byte First 1b - MS Byte First
17 CRC_REF_OUT	CRC Reflect Out 0b - Does not manipulate CRC result 1b - CRC result is to be reflected bitwise (operated on entire word)
16 CRC_REF_IN	CRC Reflect In 0b - Does not manipulate input data stream 1b - reflect each byte in the input stream bitwise
15-12 —	Reserved
11-8 CRC_START_BYTE	Configure CRC Start Point Starts CRC with this byte position. Byte #0 is the first byte of Sync address.
7-3 —	Reserved
2-0 CRC_SZ	CRC Size (in octets) Number of CRC Octets = CRC_SZ, CRC_SZ from 0 to 4.

55.4.6.3.4.7 CRC INITIALIZATION (CRC_INIT)

Offset

Register	Offset
CRC_INIT	14h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CRC_SEED															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CRC_SEED															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CRC_SEED	CRC Seed Value Initial Value for CRC LFSR

55.4.6.3.4.8 CRC POLYNOMIAL (CRC_POLY)

Offset

Register	Offset
CRC_POLY	18h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CRC_POLY															
Reset	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CRC_POLY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

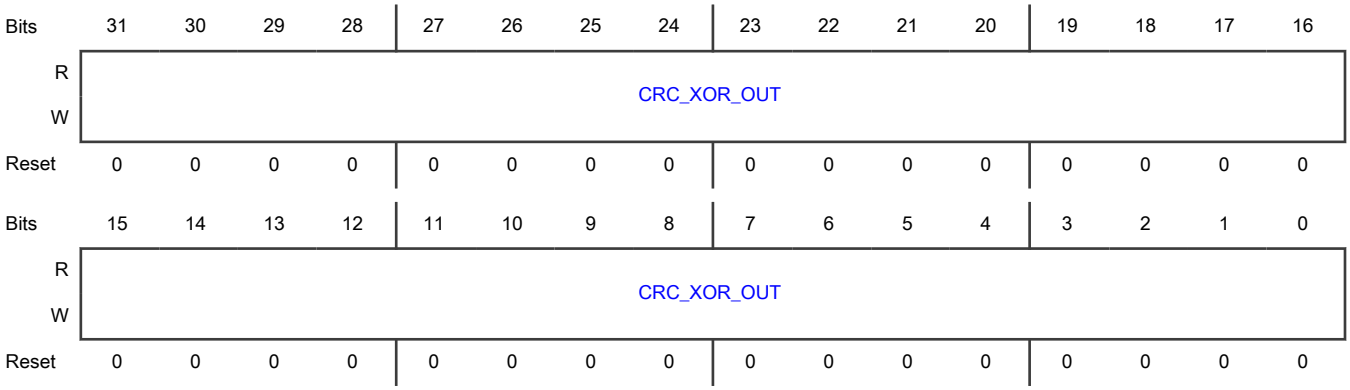
Field	Function
31-0 CRC_POLY	CRC Polynomial.

55.4.6.3.4.9 CRC XOR OUT (CRC_XOR_OUT)

Offset

Register	Offset
CRC_XOR_OUT	1Ch

Diagram



Fields

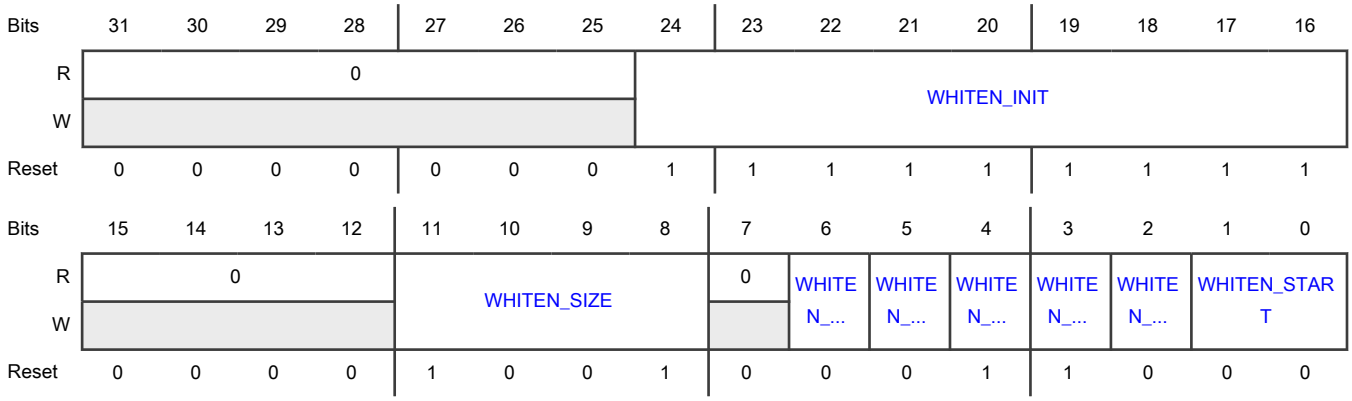
Field	Function
31-0 CRC_XOR_OUT	CRC XOR OUT Register XOR mask for CRC result (for no mask, should be 0)

55.4.6.3.4.10 WHITENER CONFIGURATION (WHITEN_CFG)

Offset

Register	Offset
WHITEN_CFG	20h

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 WHITEN_INIT	Initialization value for whitening/de-whitening
15-12 —	Reserved
11-8 WHITEN_SIZE	Length of Whitener LFSR
7 —	Reserved
6 WHITEN_PAYLOAD_REINIT	Configure for Whitener re-initialization 0b - Does not re-initialize Whitener LFSR at start-of-payload 1b - Re-initialize Whitener LFSR at start-of-payload
5 WHITEN_REF_IN	Whiten Reflect Input The input data stream is reflected, bit-wise, per byte, if this register bit is asserted. Bit 7 becomes bit 0, bit 6 becomes bit 1, etc. This register bit only causes the reflection of the payload data bits as they are used in the whiten calculation and does not cause any change in the output bit order.
4 WHITEN_POLY_TYPE	Whiten Polynomial Type A Fibonacci type LFSR is used with the whiten polynomial if this register bit is asserted. Otherwise, a Galois type LFSR is used.
3	Congifure for Whitening-before-CRC Sets the order of Bit Stream Processing for TX and RX.

Table continues on the next page...

Table continued from the previous page...

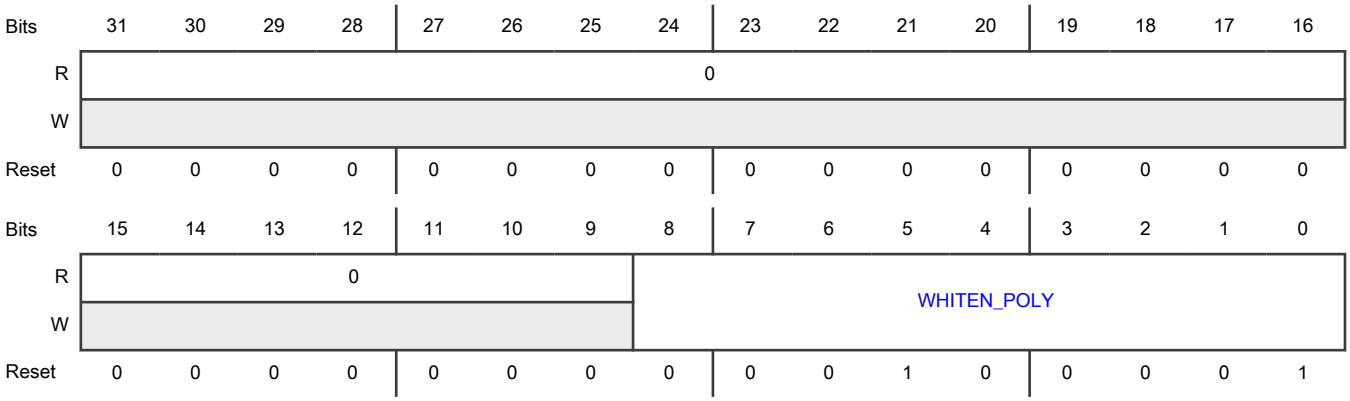
Field	Function
WHITEN_B4_CRC	0b - CRC before whiten/de-whiten 1b - Whiten/de-whiten before CRC
2 WHITEN_END	Configure end-of-whitening 0b - end whiten at end-of-payload 1b - end whiten at end-of-crc
1-0 WHITEN_START	Configure Whitener Start Point 00b - no whitening 01b - start whitening at start-of-H0 10b - start whitening at start-of-H1 but only if LENGTH > WHITEN_SZ_THR 11b - start whitening at start-of-payload but only if LENGTH > WHITEN_SZ_THR

55.4.6.3.4.11 WHITENER POLYNOMIAL (WHITEN_POLY)

Offset

Register	Offset
WHITEN_POLY	24h

Diagram



Fields

Field	Function
31-9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8-0 WHITEN_POLY	Whitener Polynomial The 9-bit polynomial used in the whiten calculation. The polynomial value must be right-justified if smaller than 9-bits.

55.4.6.3.4.12 WHITENER SIZE THRESHOLD (WHITEN_SZ_THR)

Offset

Register	Offset
WHITEN_SZ_THR	28h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				WHITEN_SZ_THR											
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Fields

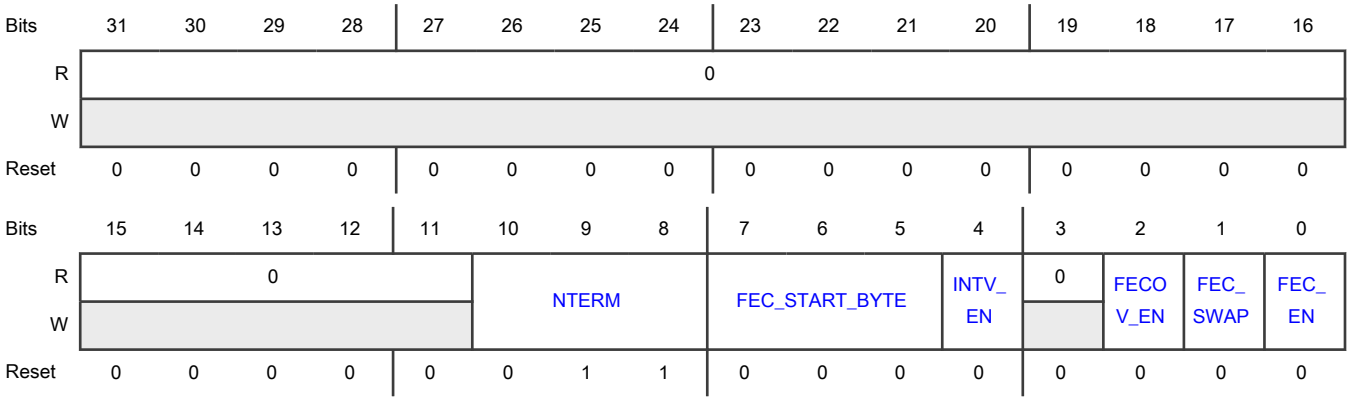
Field	Function
31-12 —	Reserved
11-0 WHITEN_SZ_THR	Whitener Size Threshold Minimum Packet Length (extracted LENGTH field) required to enable whiten. Requires WHITEN_START=2 or 3

55.4.6.3.4.13 FEC CONFIG REGISTER 1 (FEC_CFG1)

Offset

Register	Offset
FEC_CFG1	2Ch

Diagram



Fields

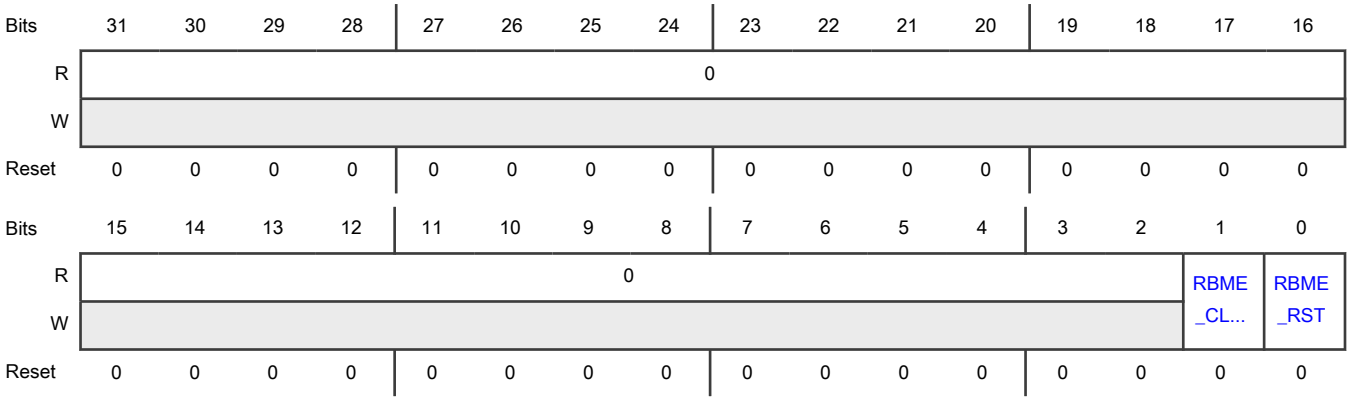
Field	Function
31-11 —	Reserved
10-8 NTERM	Number of term bits Size of term bits, from 0 to 7 bits length can support. In Bluetooth LE, the default value is 3.
7-5 FEC_START_BYTE	FEC Start Byte
4 INTV_EN	Enable interleaver reigster Should keep 0 for this field
3 —	Reserved
2 FECOV_EN	Enable dynamic overide of FEC 0b - Disable FEC override 1b - The override of FEC is only used in Bluetooth LE LR cases, dynamically depending on the LR AA detected
1 FEC_SWAP	FEC output swap Should keep 0 for this field
0 FEC_EN	FEC enable 0b - Disable FEC encoder and decoder 1b - Enable FEC encoder and decoder

55.4.6.3.4.14 RBME SOFT RESET REGISTER (RBME_RST)

Offset

Register	Offset
RBME_RST	30h

Diagram



Fields

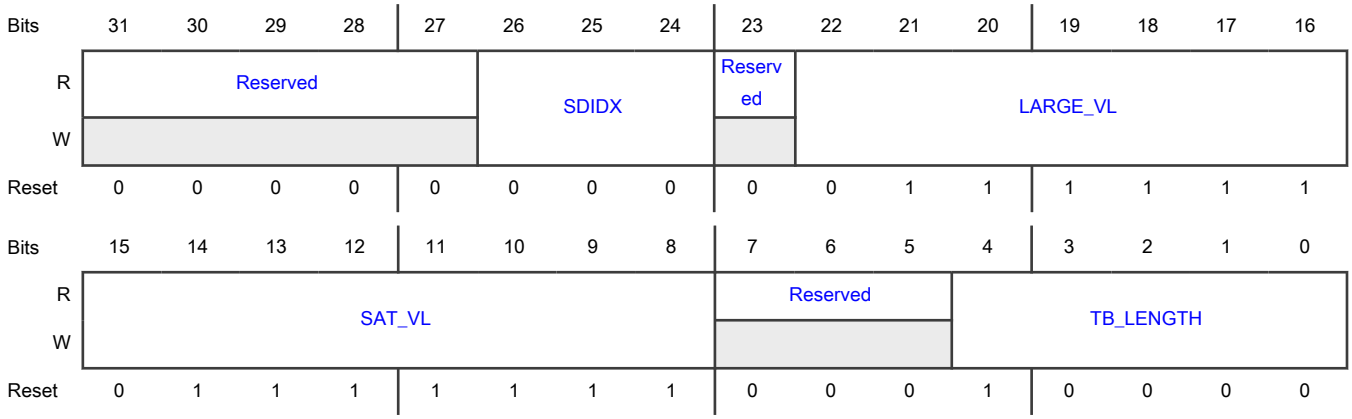
Field	Function
31-2 —	Reserved
1 RBME_CLK_EN_OVRD	RBME Clock Enable override Set this bit to enable RBME internal clock manually.
0 RBME_RST	RBME reset signal 0b - Disable soft reset 1b - Enable soft reset. When this bit is write to 1, the soft reset to RBME happens immediately. Then all internal registers and functions will be reset.

55.4.6.3.4.15 FEC CONFIG REGISTER 2 (FEC_CFG2)

Offset

Register	Offset
FEC_CFG2	34h

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 SDIDX	Index of startup state. PM(startStIdx)=0 Keep default values in Bluetooth LE mode.
23 —	Reserved
22-16 LARGE_VL	Large value used at startup phase, assigned to the initial PMs. Keep default values in Bluetooth LE mode.
15-8 SAT_VL	Saturation value for PM Keep default values in Bluetooth LE mode.
7-5 —	Reserved
4-0 TB_LENGTH	Trace-back length Viterbi Trace back length from 1 to 16, the longer trace back length will get better BER in lower SNR environment. Keep default values in Bluetooth LE mode.

55.4.6.3.4.16 SPREADER CONFIG REGISTER (SPREAD_CFG)

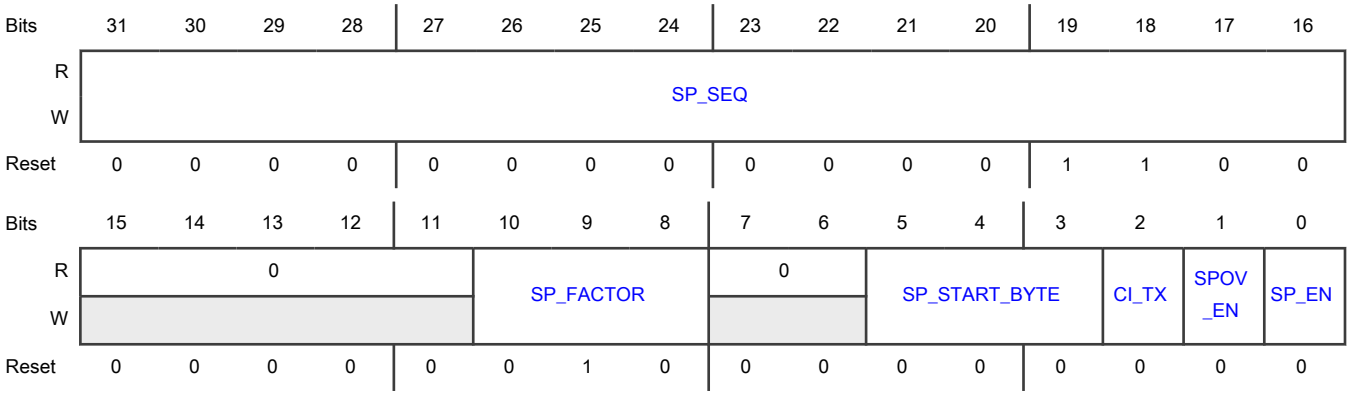
Offset

Register	Offset
SPREAD_CFG	3Ch

Function

Soft/hard spreader/de-spreader config register

Diagram



Fields

Field	Function
31-16 SP_SEQ	Spreading Bit Sequence Bit sequence used for spreading and Manchester encoding
15-11 —	Reserved
10-8 SP_FACTOR	Spreading Factor Spreading and despreading factor selection 000b - Factor = 1(No spreading and despreading) 001b - Factor = 2 010b - Factor = 4 011b - Factor = 8 100b - Factor = 16
7-6 —	Reserved
5-3 SP_START_BYTE	Spread Start Byte
2 CI_TX	Bluetooth LE The CI field determines which coding scheme is used for FEC block 2. It is only used in GENFSK working as active Bluetooth LE link layer TX mode. 0b - FEC Block 2 coded using S=8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - FEC Block 2 coded using S=2
1 SPOV_EN	Spreader Override Enable Allows overwrite of the spreading, the override is automatically done by hardware in GLL working as active Bluetooth LE link layer. So in normal Bluetooth LE case, there is no need to set this enable bit. 0b - Does not allow active override of the spreading enable 1b - Allows active override of the spreading enable
0 SP_EN	Spreader Enable bit Enable bit for Bluetooth LE spreader or Manchester block. If any protocol has spreader or Manchester function then SP_EN should be enabled. The time for Bluetooth LE 500 or Bluetooth LE 125 spreader is controlled by hardware override. 0b - Disable spreader 1b - Enable spreader

55.4.6.3.4.17 WHITEN CONFIG REGISTER (WHT_CFG)

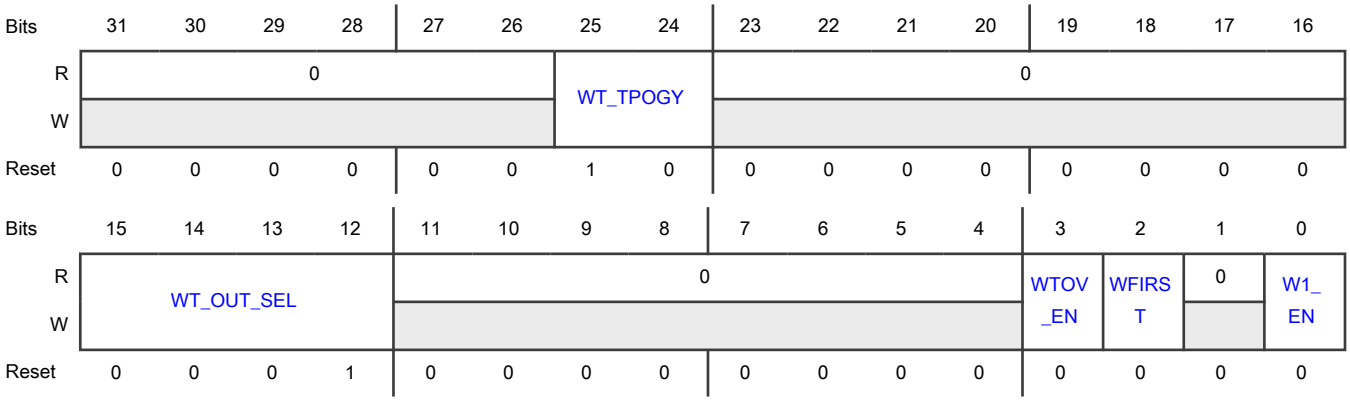
Offset

Register	Offset
WHT_CFG	40h

Function

The WHT_CFG register is used to soft-whiten 1 configuration register.

Diagram



Fields

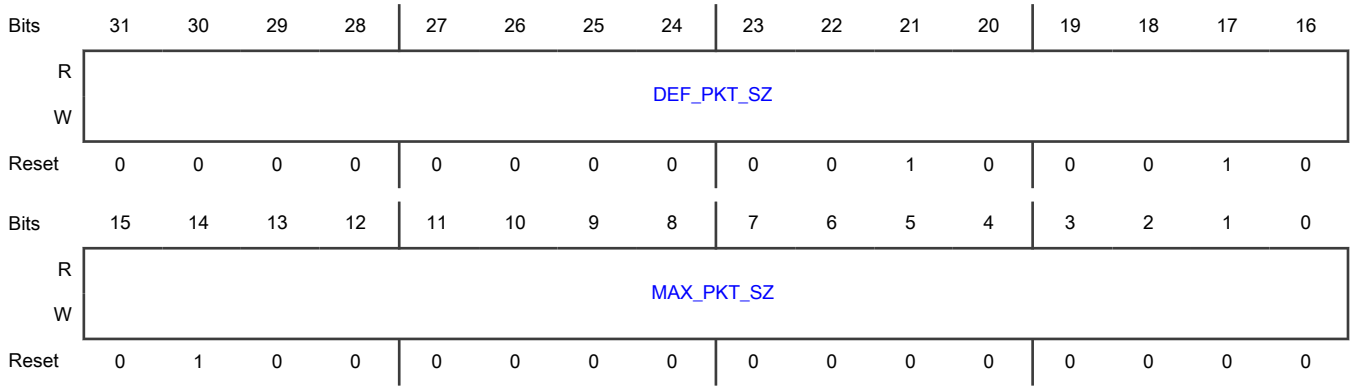
Field	Function
31-26 —	Reserved
25-24 WT_TPOGY	Whiten 1 Polynomial Type Fibonacci =1, Galois = 2, Fibonacci FCP = 3.
23-16 —	Reserved
15-12 WT_OUT_SEL	Selected Output
11-4 —	Reserved
3 WTOV_EN	Allows overwrite of the whitening
2 WFIRST	Whitens before CRC
1 —	Reserved
0 W1_EN	Enable first whitener

55.4.6.3.4.18 PACKET SIZE REGISTER (PKT_SZ)

Offset

Register	Offset
PKT_SZ	44h

Diagram



Fields

Field	Function
31-16 DEF_PKT_SZ	Default Packet Size AA length bits + CI length bits + TERM1 length bits
15-0 MAX_PKT_SZ	Maximum Packet Size In Bits Large value is required as placeholder until the link layer provides the packet length

55.4.6.3.4.19 LENGTH OF PHR CONFIG REGISTER (CRC_PHR_SZ)

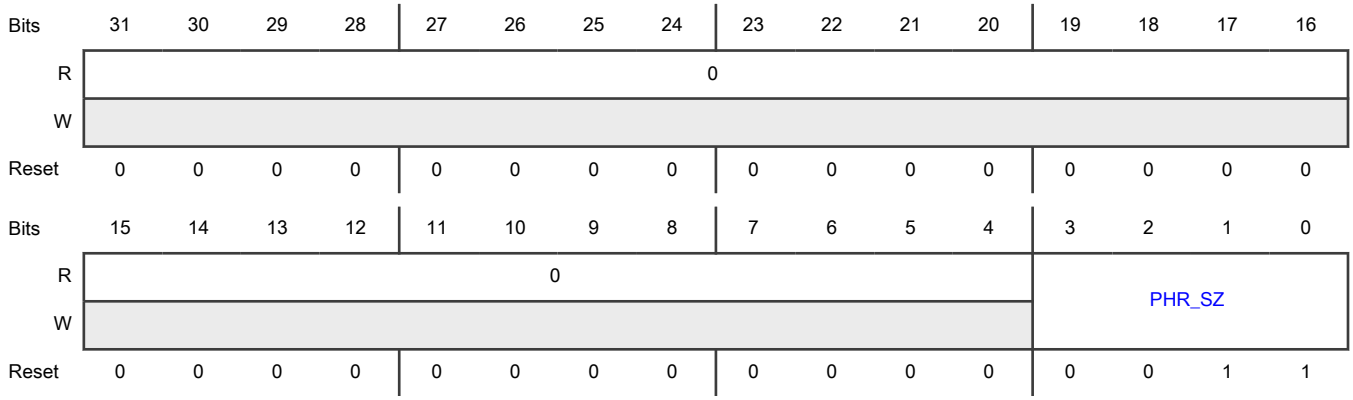
Offset

Register	Offset
CRC_PHR_SZ	48h

Function

Length of PHR in bytes.

Diagram



Fields

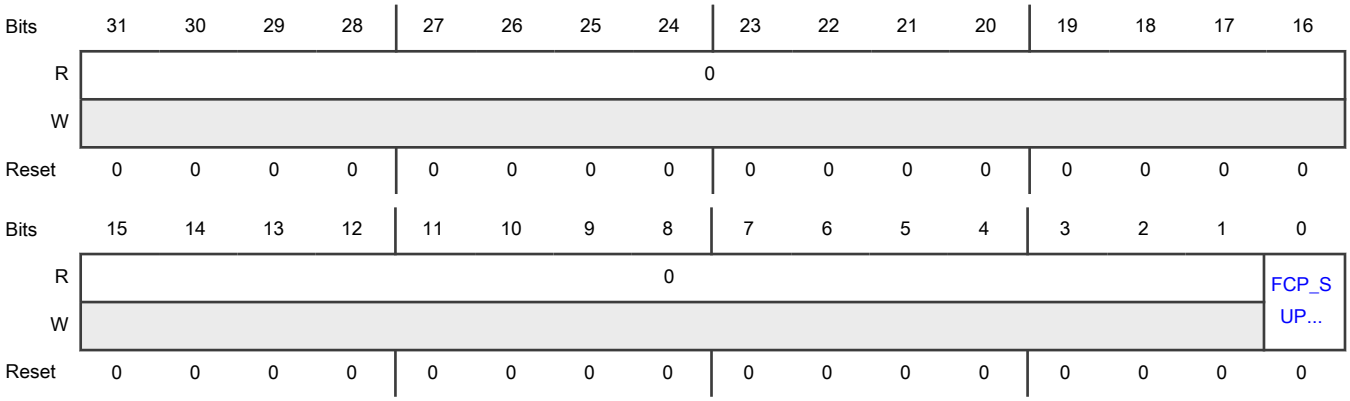
Field	Function
31-4 —	Reserved
3-0 PHR_SZ	PHR Size Config For Bluetooth LE LR mode(either NBU or GEN-LL is activated), this field should be 2. For other protocols, this field has no effect.

55.4.6.3.4.20 FCP SUPPORT CONFIG REGISTER (FCP_CFG)

Offset

Register	Offset
FCP_CFG	4Ch

Diagram



Fields

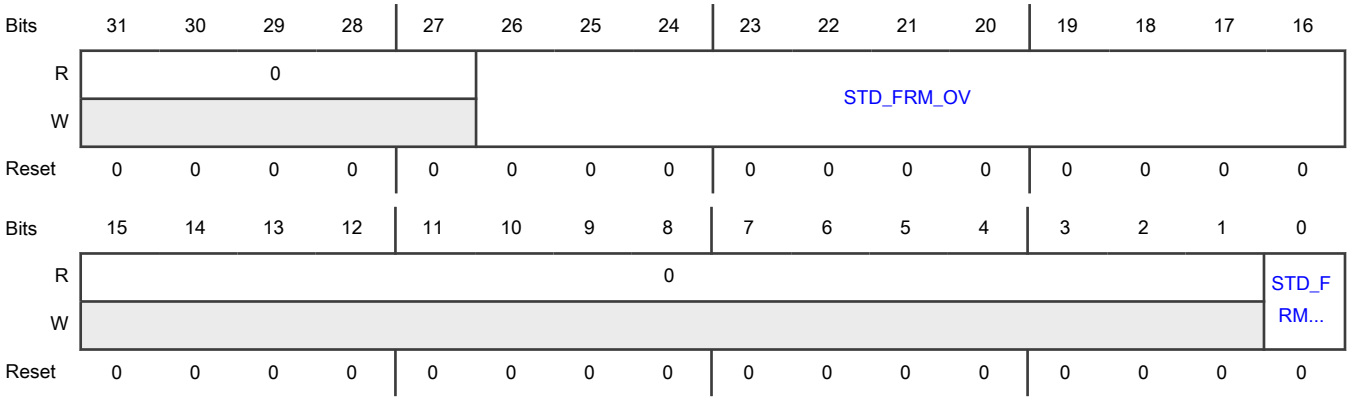
Field	Function
31-1 —	Reserved
0 FCP_SUPPORT	FCP Suppport 0b - Disable FCP support 1b - Enable FCP support

55.4.6.3.4.21 FRAME OVERRIDE SIZE REGISTER (FRAME_OVER_SZ)

Offset

Register	Offset
FRAME_OVER_SZ	50h

Diagram



Fields

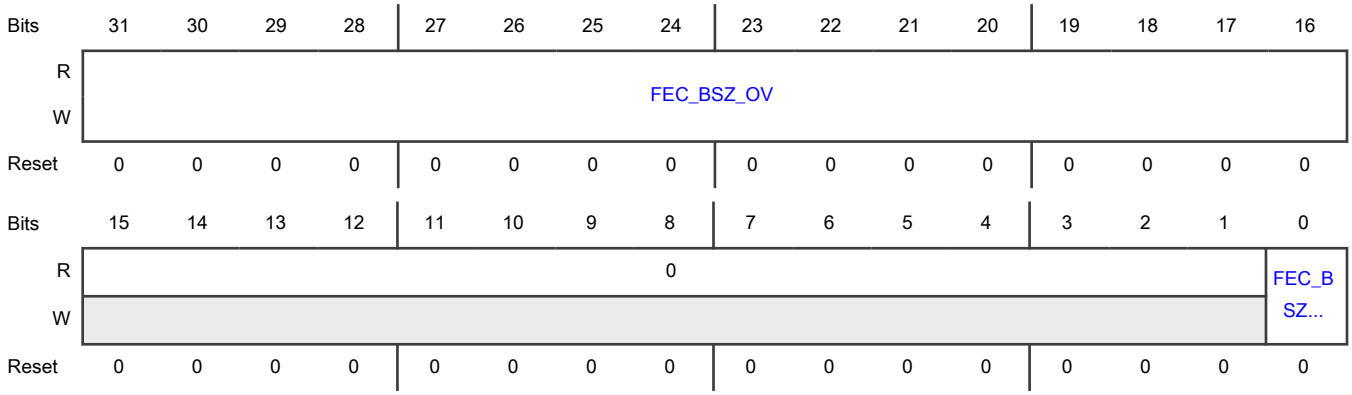
Field	Function
31-27 —	Reserved
26-16 STD_FRM_OV	Value to overide the STD frame length (bits)
15-1 —	Reserved
0 STD_FRM_OV_EN	Overrides actvie STD frame length from link layer enable bit 0b - Disable override actvie STD frame length from link layer 1b - Enable override actvie STD frame length from link layer

55.4.6.3.4.22 OVERRIDE OF FEC BLOCK SIZE REGISTER (FEC_BSZ_OV_B4SP)

Offset

Register	Offset
FEC_BSZ_OV_B4SP	54h

Diagram



Fields

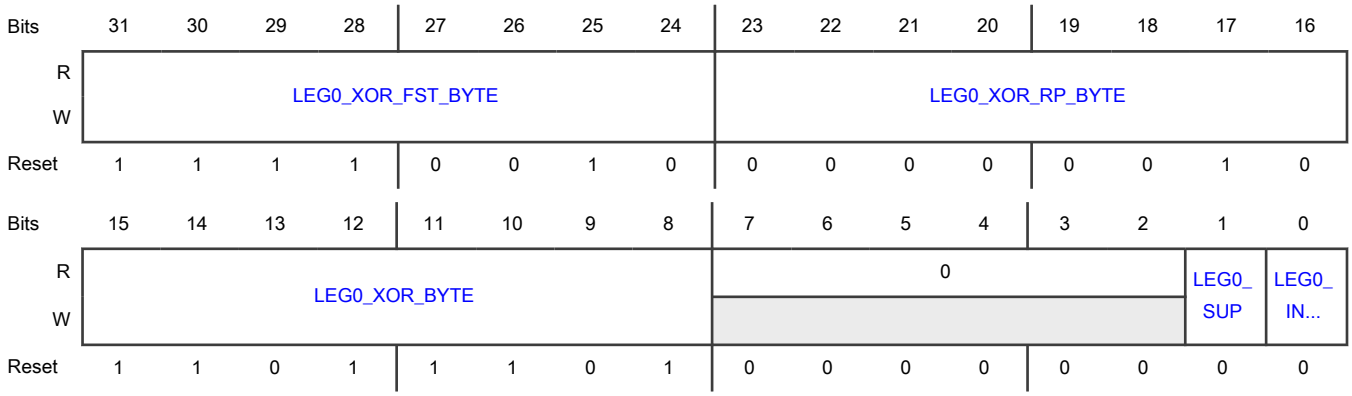
Field	Function
31-16 FEC_BSZ_OV	Value of the override in bits. It is for test purpose.
15-1 —	Reserved
0 FEC_BSZ_OV_B4SP_EN	Override of the FEC block size for data 0b - Disable Override actvie STD frame length from link layer 1b - Enable Override actvie STD frame length from link layer

55.4.6.3.4.23 LEG0 CONFIG REGISTER (LEG0_CFG)

Offset

Register	Offset
LEG0_CFG	58h

Diagram



Fields

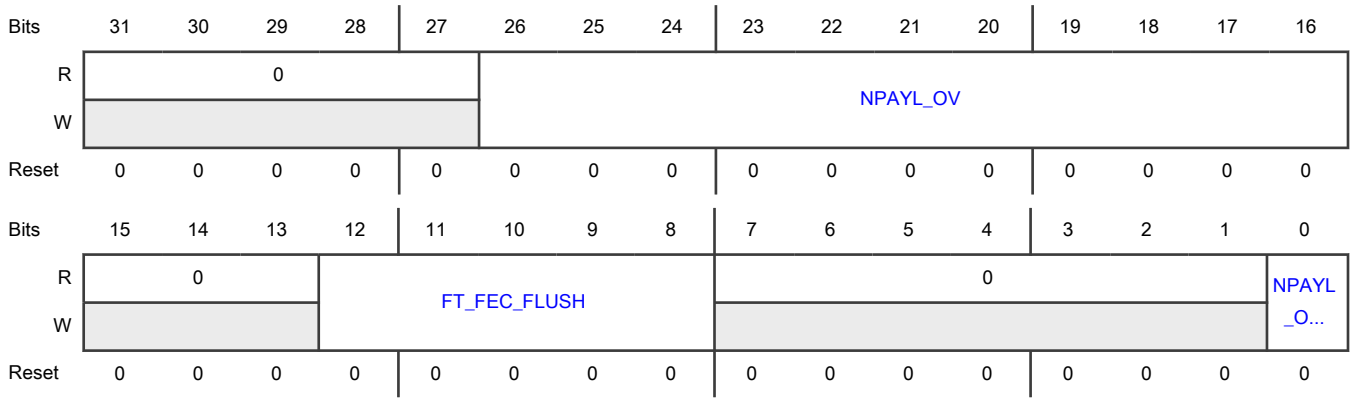
Field	Function
31-24 LEG0_XOR_FST_BYTE	FEC first byte masking
23-16 LEG0_XOR_RP_BYTE	LEG0 repeat bytes masking
15-8 LEG0_XOR_BYTE	LEG0 whitening masking byte
7-2 —	Reserved
1 LEG0_SUP	LEG0 support register 0b - Disable LEG0 support 1b - Enable LEG0 support
0 LEG0_INV_EN	Whiten invert enable bit 0b - Disable whiten invert for LEG0 1b - Enable whiten invert for LEG0

55.4.6.3.4.24 OVERRIDE PAYLOAD LENGTH REGISTER (NPAYL_OVER_SZ)

Offset

Register	Offset
NPAYL_OVER_SZ	5Ch

Diagram



Fields

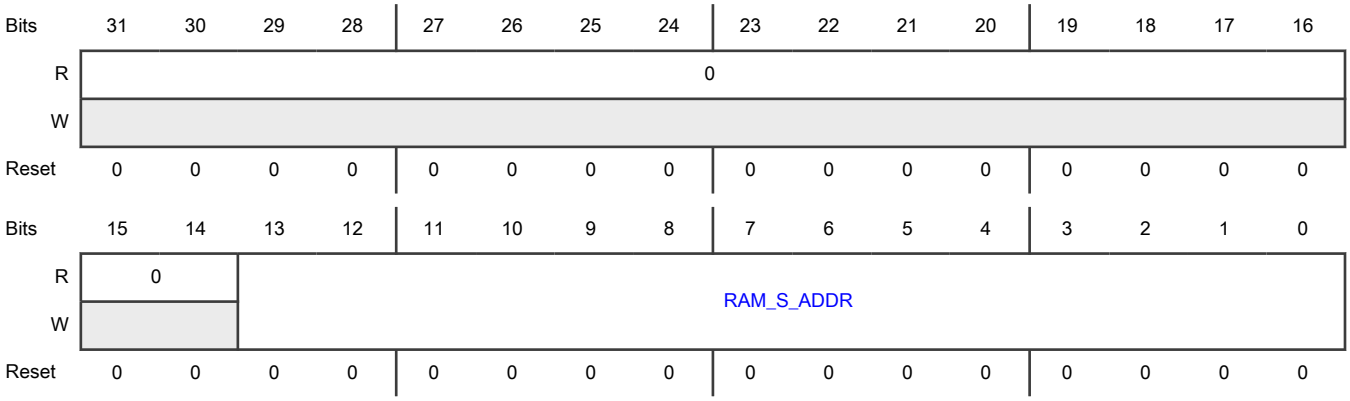
Field	Function
31-27 —	Reserved
26-16 NPAYL_OV	
15-13 —	Reserved
12-8 FT_FEC_FLUSH	Value to overide the payload length (bits) For test purpose only
7-1 —	Reserved
0 NPAYL_OV_EN	Override the internal payload length computation 0b - Disable override the internal payload length 1b - Enable override the internal payload length

55.4.6.3.4.25 PACKET RAM SOURCE ADDRESS (RAM_S_ADDR)

Offset

Register	Offset
RAM_S_ADDR	64h

Diagram



Fields

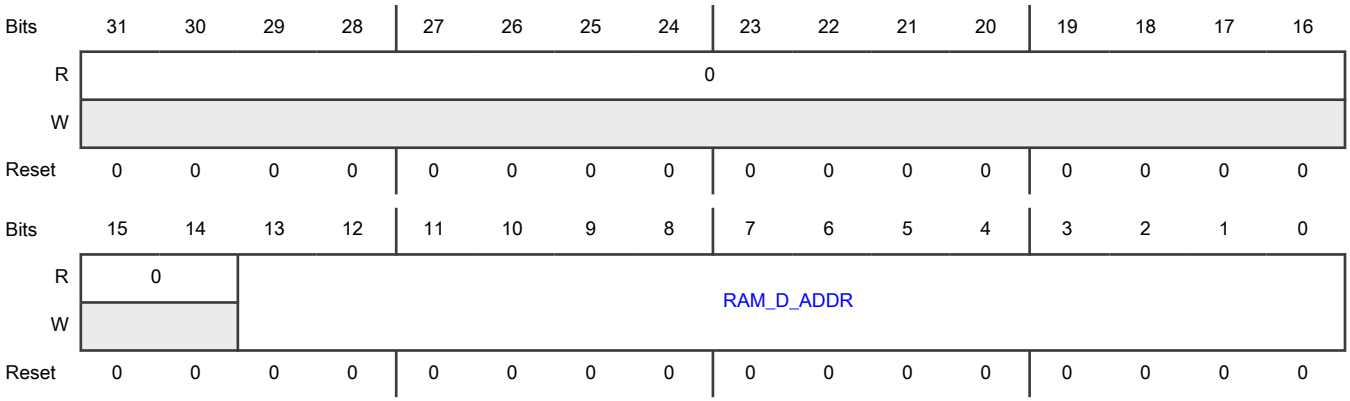
Field	Function
31-14 —	Reserved
13-0 RAM_S_ADDR	Packet RAM source address. This address is ram physical address.

55.4.6.3.4.26 PACKET RAM DESTINATION ADDRESS (RAM_D_ADDR)

Offset

Register	Offset
RAM_D_ADDR	68h

Diagram



Fields

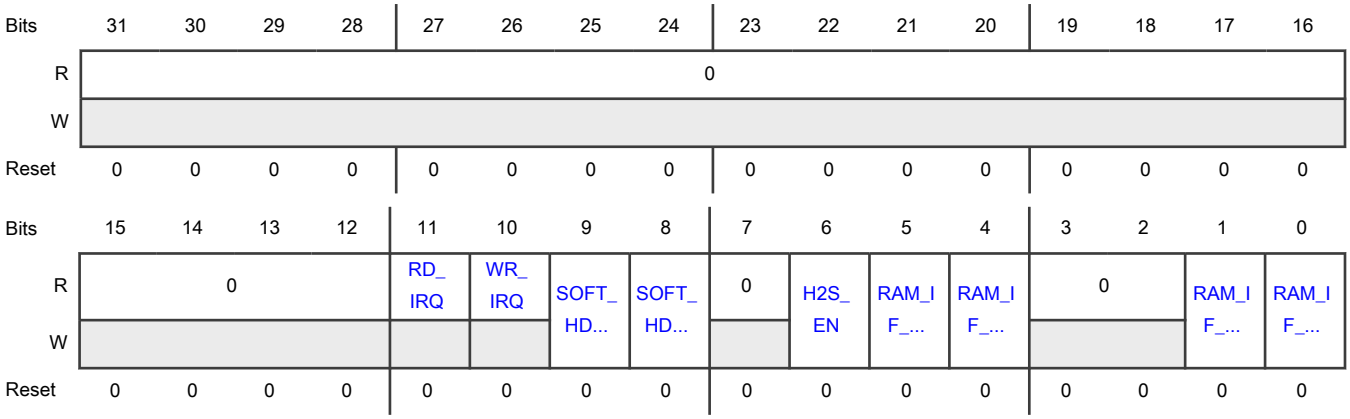
Field	Function
31-14 —	Reserved
13-0 RAM_D_ADDR	Packet RAM destination address, this address is ram physical address.

55.4.6.3.4.27 PACKET RAM INTERFACE CONFIG REGISTER (RAM_IF_CFG)

Offset

Register	Offset
RAM_IF_CFG	6Ch

Diagram



Fields

Field	Function
31-12 —	Reserved
11 RD_IRQ	Read to RAM complete flag 0b - Reading to RAM not complete 1b - Reading to RAM complete
10 WR_IRQ	Write to RAM complete flag 0b - Writing to RAM not complete 1b - Writing to RAM complete
9 SOFT_HD_SEL_WR	Soft and hard bit selection of read operation 0b - Hard bit selection of read operation 1b - Soft bit selection of read operation
8 SOFT_HD_SEL_RD	Soft and hard bit selection of write operation 0b - Hard bit selection of write operation 1b - Soft bit selection of write operation
7	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 H2S_EN	Hard bit convert to soft bit enable 0b - Disable hard bit to soft bits coversion 1b - Enable hard bit to soft bits coversion
5 RAM_IF_IC	RAM interface interrupt clear 0b - To do nothing to RAM interface interrupt 1b - To clear RAM interface interrupt
4 RAM_IF_IE	RAM interface interrupt enable bit 0b - Disable RAM interface interrupt 1b - Enable RAM interface interrupt
3-2 —	Reserved
1 RAM_IF_RX_EN	RAM interface RX enable 0b - Disable RAM interface RX 1b - Enable RAM interface RX
0 RAM_IF_TX_EN	RAM interface TX enable bit 0b - Disable RAM interface TX 1b - Enable RAM interface TX

55.4.6.3.5 FEC

The initial FEC originates from BL5 specifications related to the long-range feature. It is a part of 2.4 GHz Radio IP. The convolutional FEC encoder uses:

1. A non-systematic, non-recursive design
2. Code rate ½ with constraint length 4
3. The generator polynomials are:

$$G0(x) = 1 + x + x^2 + x^3$$

$$G1(x) = 1 + x^2 + x^3$$

55.4.6.3.5.1 Two-Pass FEC

The FEC encoder and decoder can run twice on a packet. For the TX and Bluetooth LE long range, it must encode the AA, CI + termination bits (FECBlock1), and then encode the payload and CRC (FECBlock2), see section "RX Bluetooth LE LR Example".

For the RX, it must decode the two differently. To reduce latency, FECBlock1 is decoded with a fast clock once the AA is detected. FECBlock2 is decoded at the symbol rate.

55.4.6.3.5.2 TX FEC

FEC Encoder: Bluetooth LE-LR use case A shift register based representation of convolutional encoder is shown in the following figure.

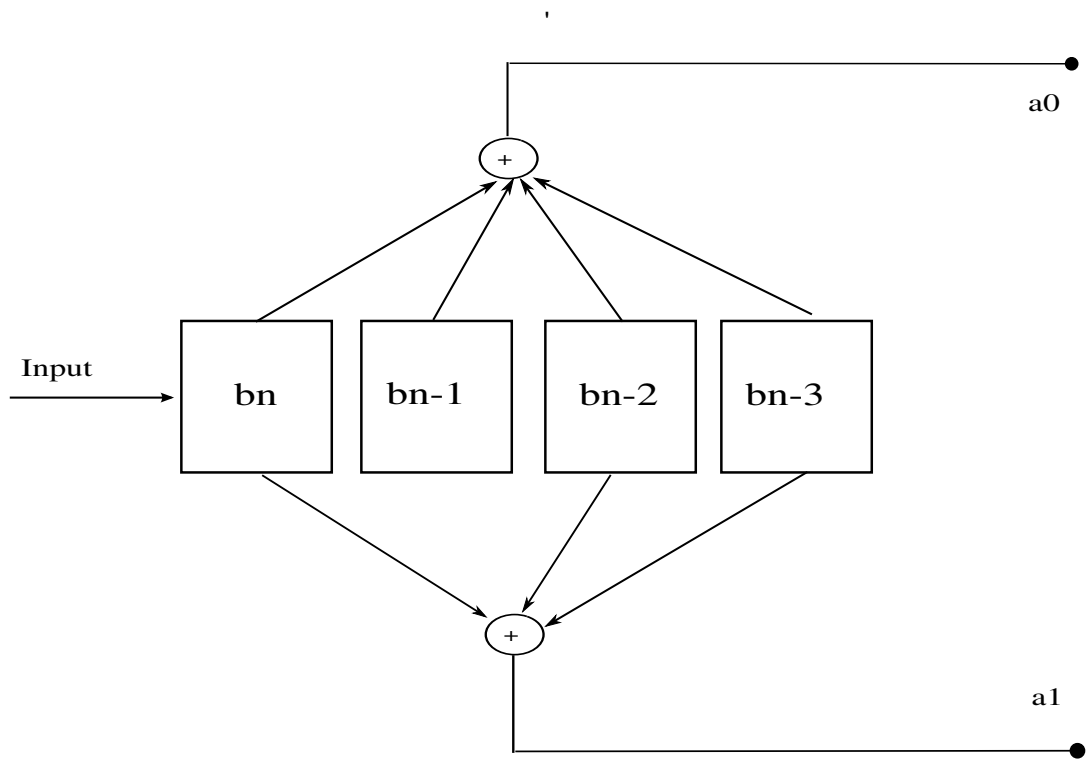


Figure 262. Bluetooth LE-LR FEC Encoder

Where:

- bn denotes the current bit, this is the input for the shift register.
- bn-i (i=1..3) are the previous bits sequence, they are referred to as the state or the memory of the encoder. The encoder produces two bits for each input bit.

The bit from generator polynomial G0 (a0) is transmitted first; the bit from generator polynomial G1 (a1) is transmitted second. The initial state of the convolutional FEC encoder is set to all zeros. An input sequence of three consecutive zeros always brings the convolutional FEC encoder back to its original state. This sequence is known as the termination sequence.

55.4.6.3.6 Spreader

TX Spreading

The spreading maps each input bit to P output bits (P = 2,4,8,16) symbols. The following table shows an example of Bluetooth LE LR where the choice for M is 4, depending on the coding scheme of protocols in use.

Table 405. Pattern Mapper inputs and outputs for Bluetooth LE LR spreading

Input from FEC encoder	Output sequence when P=1 (no spreading)	Output sequence when P=4
0	0	0011*
1	1	1100*
*The entries in the table are transmitted MSB first		

Manchester encoding is a subset of spreading with just two bits per input bit — see below figure. As the Manchester example illustrates, the clock rates (actually, clock valid signal rates) are affected by the Manchester encoding and decoding. The same applies to the spreader and de-spreader modules but with data rates changing by P = 2, 4, 8 or 16. Up to 16 are necessary to support all the 802.15.4 standards. The use of spreading means that output bits are serialized at P times the rate of the input bits. This is an important consideration as a clock valid on the output must be P times more often. That is, the data rate after the spreader is P times higher than that before the spreader. The data rate after the de-spreader in the RX is up to 16x smaller.

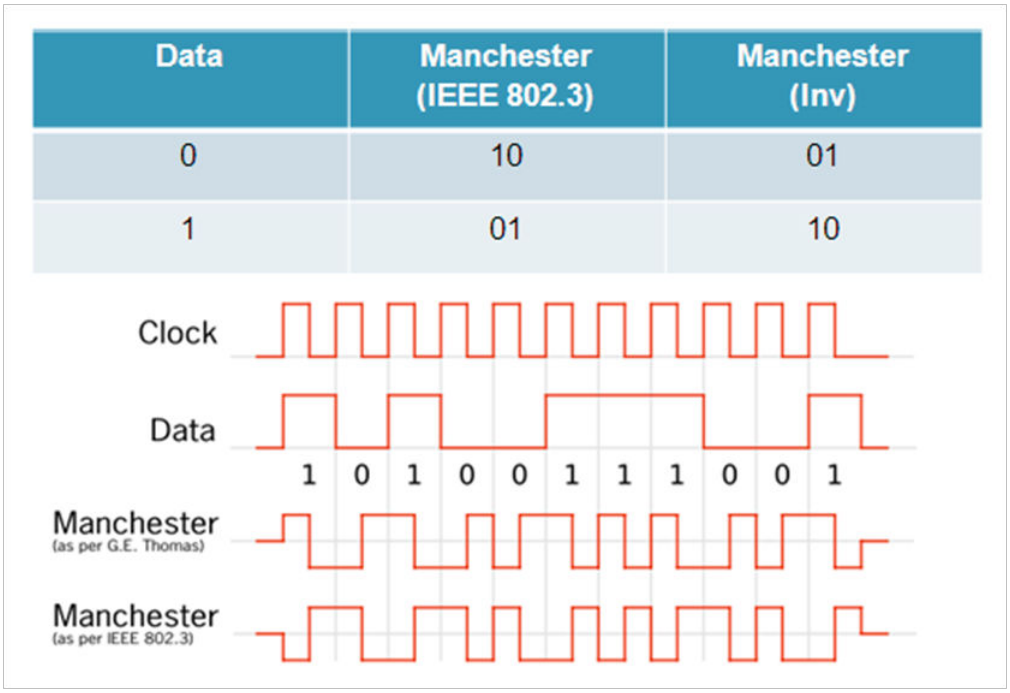


Figure 263. Manchester Encoding option for the Spreading Module

RX DE-Spreading

De-spreading is the dual of the spreading operation. De-spreading means correlating the soft bits provided by the demodulator with the reference sequence. This correlation needs to be done in a timely manner, that is, it has specific timing. For example, with a de-spreading of 4, four samples of the input generate the first output, the next four samples generate the second output and so on. The boundaries of de-spreading are fixed in the PHY.

55.4.6.3.7 Operation

The CRCW module contains the functionality to perform transmit CRC insertion, receive CRC checking, and data whitening. Additionally, the module can perform data error correction on received packets using the CRC syndrome if configured appropriately.

55.4.6.3.7.1 CRC Calculation

The CRC value is calculated using a configurable LFSR as depicted in Figure 264. The CRC calculation polynomial value is loaded into register *P*. The seed is loaded into register *SR* at initilization. The result is in register *SR* after all bits have been processed.

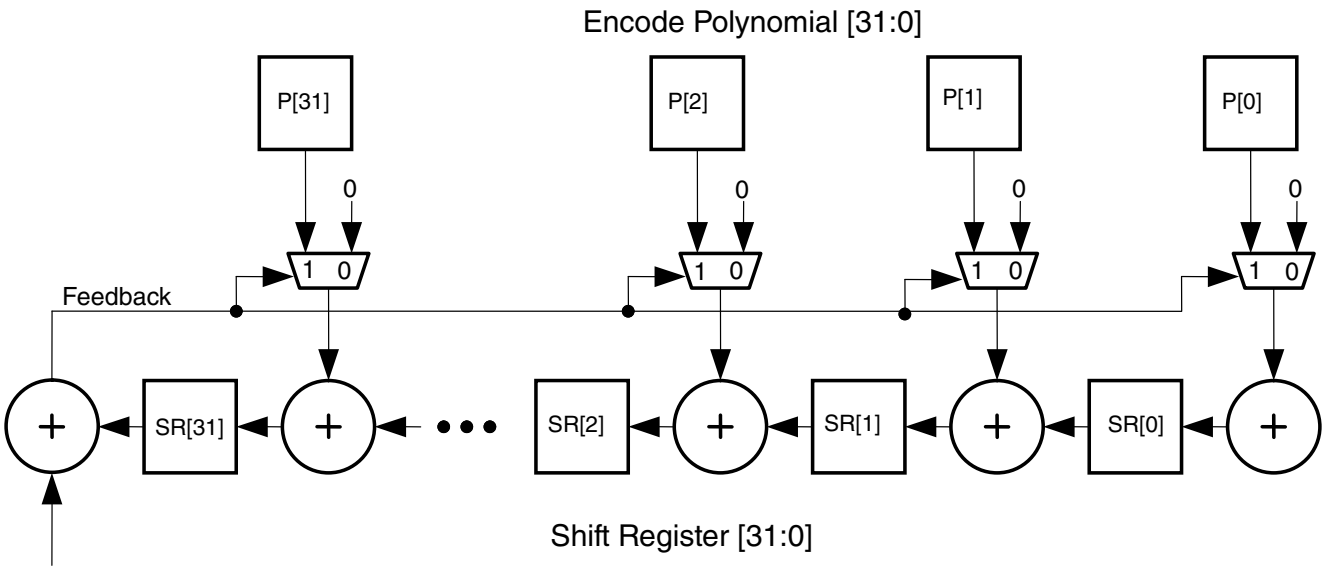
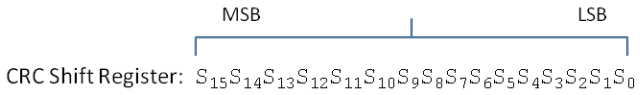


Figure 264. CRC Calculation Shift Register

Data flows from the link layer function to the CRCW module during transmit before being sent to the radio physical layer. All bits in each packet are provided by the link layer function, including placeholder bits for the CRC value. The link layer function marks the CRC calculation start and stop points in the bit stream. The link layer also indicates the start of the CRC placeholder bits. The CRCW module overwrites the placeholder CRC bits with the CRC result, ordered as specified by the configuration options. Figure 265 is a list of example 16-bit CRC output modes.



REF IN	BYTE ORDER	REF OUT	Order of Transmitted Bits (shifted out first ..... shifted out last)
X	0	0	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $S_{15}S_{14}S_{13}S_{12}S_{11}S_{10}S_9S_8S_7S_6S_5S_4S_3S_2S_1S_0$
X	0	1	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $S_8S_9S_{10}S_{11}S_{12}S_{13}S_{14}S_{15}S_0S_1S_2S_3S_4S_5S_6S_7$
X	1	0	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $S_7S_6S_5S_4S_3S_2S_1S_0S_{15}S_{14}S_{13}S_{12}S_{11}S_{10}S_9S_8$
X	1	1	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $S_0S_1S_2S_3S_4S_5S_6S_7S_8S_9S_{10}S_{11}S_{12}S_{13}S_{14}S_{15}$

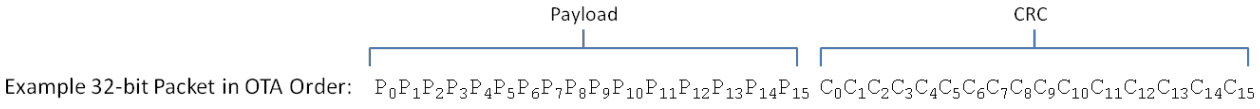
Figure 265. CRC Output Bit Order

55.4.6.3.7.2 CRC Check

During reception, data flows from the radio physical layer to the CRCW module before being sent to the link layer function. All bits in each packet are provided by the radio physical layer. Just as in transmit, the link layer function marks the CRC calculation start and stop points in the bit stream. The link layer also indicates the start of the received CRC bits.

The CRCW calculates the expected CRC result just as in transmit, but compares the received CRC bits as they arrive, ordered as specified by the configuration options, instead of inserting them into the packet. An early fail signal is asserted if a CRC bit miscompare is detected.

The CRC register contains the expected CRC when either of the *config_crc_byte_order* or *config_crc_ref_in* CRC configuration options are set. The CRC register contains the syndrome of the CRC error calculation when both options are not set.



REF IN	BYTE ORDER	REF OUT	Order of Bits Used in CRC Calculation (shifted in first ..... shifted in last)
0	0	0	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $C_0C_1C_2C_3C_4C_5C_6C_7C_8C_9C_{10}C_{11}C_{12}C_{13}C_{14}C_{15}$
0	0	1	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $C_7C_6C_5C_4C_3C_2C_1C_0C_{15}C_{14}C_{13}C_{12}C_{11}C_{10}C_9C_8$
0	1	0	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $C_8C_9C_{10}C_{11}C_{12}C_{13}C_{14}C_{15}C_0C_1C_2C_3C_4C_5C_6C_7$
0	1	1	$P_0P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$ $C_{15}C_{14}C_{13}C_{12}C_{11}C_{10}C_9C_8C_7C_6C_5C_4C_3C_2C_1C_0$
1	0	0	$P_7P_6P_5P_4P_3P_2P_1P_0P_{15}P_{14}P_{13}P_{12}P_{11}P_{10}P_9P_8$ $C_0C_1C_2C_3C_4C_5C_6C_7C_8C_9C_{10}C_{11}C_{12}C_{13}C_{14}C_{15}$
1	0	1	$P_7P_6P_5P_4P_3P_2P_1P_0P_{15}P_{14}P_{13}P_{12}P_{11}P_{10}P_9P_8$ $C_7C_6C_5C_4C_3C_2C_1C_0C_{15}C_{14}C_{13}C_{12}C_{11}C_{10}C_9C_8$
1	1	0	$P_7P_6P_5P_4P_3P_2P_1P_0P_{15}P_{14}P_{13}P_{12}P_{11}P_{10}P_9P_8$ $C_8C_9C_{10}C_{11}C_{12}C_{13}C_{14}C_{15}C_0C_1C_2C_3C_4C_5C_6C_7$
1	1	1	$P_7P_6P_5P_4P_3P_2P_1P_0P_{15}P_{14}P_{13}P_{12}P_{11}P_{10}P_9P_8$ $C_{15}C_{14}C_{13}C_{12}C_{11}C_{10}C_9C_8C_7C_6C_5C_4C_3C_2C_1C_0$

Figure 266. CRC Input Bit Order

55.4.6.3.7.3 Data Whitening

The data whitener behaves identically in both transmit (whiten) and receive (de-whiten) operating modes. The LFSR used in this operation is designed to use either Fibonacci or Galois types of calculations. In either case, the LSB of the shift register is used to whiten each data bit as they are streamed through the CRCW.

An initialization signal for the data whiten logic loads the shift register with the seed value. The initialization signal can be used to re-initialize the shift register during a packet. A protocol-specific state machine controls a signal to mark the start and stop points for data whitening in the data stream.

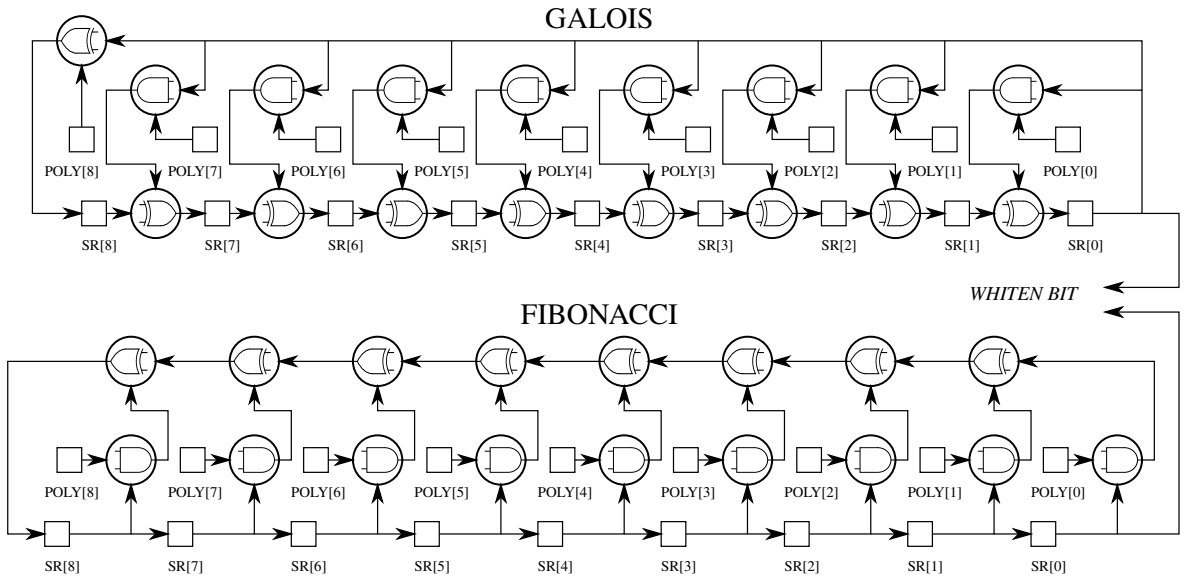


Figure 267. Data Whitening Shift Register

55.4.6.4 SFA32K(Signal Frequency Analyser of 32KHz Clock)

55.4.6.4.1 Overview

The 32KHz clock source is used as the sleep timer clock source inside RFMC and Bluetooth LE Link Layer. So accuracy of the sleep timer depends on the accuracy of the 32KHz clock in the chip. In order to achieve a higher accuracy 32KHz clock source, the SFA module is used to measure the 32KHz clock source with RF OSC clock. The measurement result can be used by software to calibrate the 32KHz clock source.

The radio SFA module can be configured to measure the 32KHz clock source by a hardware trigger when radio exists from low power mode.

55.4.6.4.2 Radio SFA

Refer to SoC Signal Frequency Analyser (SFA) chapter.

55.4.6.5 Basic Radio Integrated Crypto (BRIC)

55.4.6.5.1 BRIC

55.4.6.5.1.1 Introduction

The Basic Radio Integrated Crypto (BRIC) module is a wrapper, with control logic around the Low Tier Crypto (LTC) block and PKB (Private Key Bus) Slave that interfaces with EdgeLock Secure Enclave. This module provides the legacy interface (IPS + DMA) to the LTC module, contains additional logic to enable direct hardware interfacing for simple AES functions and a PKB interface with the EdgeLock Secure Enclave to receive KEYS securely over the PKB interface. Essentially, the logic takes control of the IPS interface so that another HW entity can easily interface.

55.4.6.5.1.1.1 Overview

The BRIC module allows for legacy behavior of the LTC. The IPS bus and relevant DMA signals are brought out to be used by the system. When in isolated mode, the control of these signals comes from the internal logic using a dedicated hardware interface. For example, RPA checking can be performed by a direct hardware connection as opposed to using a bus master to interface with the LTC.

55.4.6.5.1.1.2 Features

- Supports all LTC functions (See LTC block guide)
- IPS and DMA interface for use by the system as needed
- Dedicated HW interface that overrides the IPS interface
- HW interface simplified for easy integration
- PKB Slave that interfaes directly with EdgeLock Secure Enclave for secure KEY Transfer
- No RAM footprint

55.4.6.5.1.2 External Signal Descriptions

Signal	Width	I/O	Function
ipg_clk	1	I	IP Bus Clock, Free-running
ipg_hard_async_reset_b	1	I	asynchronous HW reset
ipt_se_gatedclk	1	I	Scan enable for gated clocks
ipg_soft_reset	1	I	synchrounous soft reset
ipg_stop	1	I	IPG stop mode
ipg_sop_ack	1	O	IPG stop acknowledge
bric_busy	1	O	BRIC/LTC busy
IPS Bus Interface			
ips_module_en	1	I	IPS module enable
ips_rwb	1	I	IPS read/write
ips_addr [15:0]	16	I	IPS address
ips_wdata [31:0]	32	I	IPS write data
ips_byte_en [3:0]	4	I	IPS byte enable
ips_rdata [31:0]	32	O	IPS read data
ips_xfr_wait	1	O	IPS transfer wait
ips_xfr_err	1	O	IPS transfer error
DMA/Interrupt			
ipd_done_idata	1	I	IPD done in data
ipd_done_odata	1	I	IPD done out data
ipd_req_idata	1	O	IPD request in data
ipd_req_odata	1	O	IPD request out data
ipi_bric_int	1	O	Interrupt
Direct HW Interface			
key_in [127:0]	128	I	Key input

Table continues on the next page...

Table continued from the previous page...

Signal	Width	I/O	Function
prand [23:0]	24	I	PRAND input for creating hash
hash [23:0]	24	O	Hash output from AES
lock_bric	1	I	Lock LTC IPS interface for direct Hardware access (Multiple RPA resolution) • **0 :** Legacy IP interface can be used to access LTC • **1 :** Legacy IPS bus disabled, direct HW Interface is active
hi_mode	1	I	Request hash operation (high pulse of one clock cycle) through Hardware interface. key_in and prand inputs has to be valid at the time of assertion hi_mode. hi_ready will be 1 while a hash operation is performed. • **0 :** Legacy IPS interface is available, HW Interface is not • **1 :** IPS bus disabled, direct HW Interface to be used
hi_ready	1	O	HW interface ready • **0 :** HW Interface ready to accept command, safe to change hi_mode • **1 :** HW Interface busy, no inputs should be changed
hash_req	1	I	Not used
hash_comp	1	O	Hash operation complete. Asserts for one clock cyclewhen 'hash' is valid and can be read.
Private Key Bus (PKB) Interface			
pkb_psel	1	I	PKB APB select
pkb_penable	1	I	PKB APB Enable
pkb_paddr	32	I	PKB APB address
pkb_pwrite	1	I	PKB APB write enable (1= write, read not supported)
pkb_pwdata	32	I	PKB APB write data
pkb_pwdata_mask	32	I	PKB APB write data mask
pkb_pready	1	O	PKB APB ready
pkb_pslverr	1	O	PKB APB slave error

55.4.6.5.1.3 Block Diagram

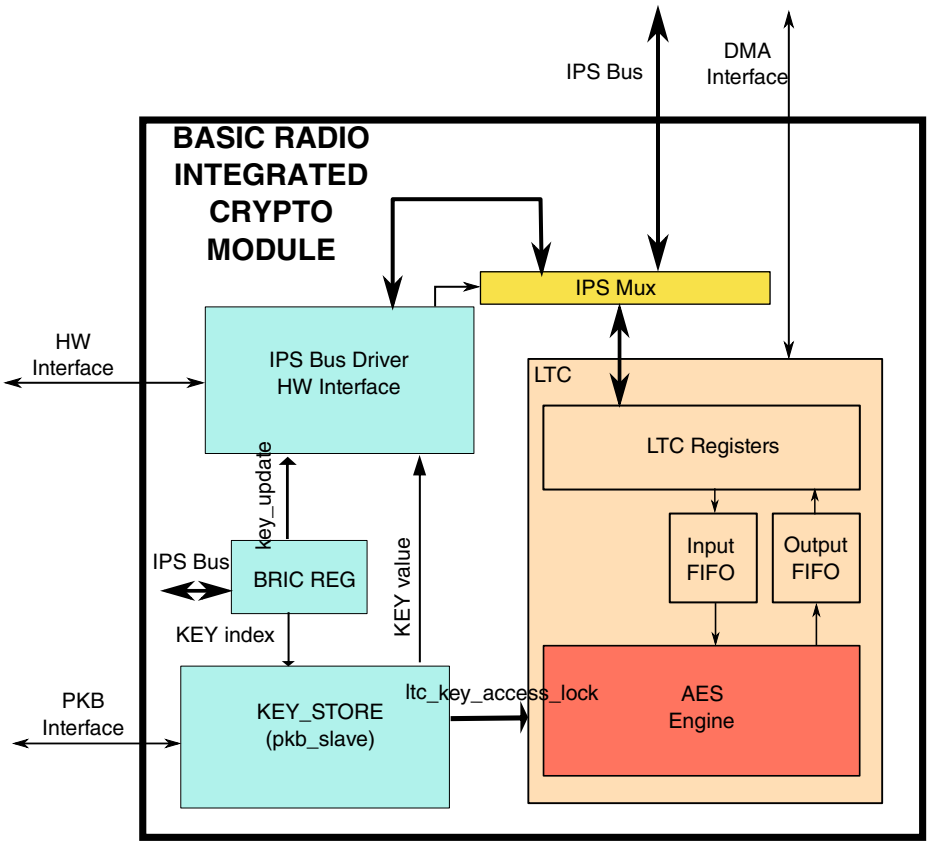


Figure 268. BRIC Block Diagram

55.4.6.5.1.4 Functional Description

The BRIC module provides following functionalities:

- Legacy LTC mode
- Hardware Interface mode
- Secure key transfer

55.4.6.5.1.4.1 Legacy LTC mode

If the HW interface mode is not selected. The BRIC module simply passes through the IPS and DMA signals. Is this configuration the core, or any bus master for that matter, can access the LTC in the traditional manner and use its capabilities as described in the LTC block guide.

55.4.6.5.1.4.2 Hardware interface mode

If the HW interface is selected, the BRIC module will take over control of the LTC.

The first part of this is to check that no process is pending by checking the AB bit in LTC_STA. If this bit is clear, the IPS module enable is then checked to confirm that no accesses are in progress on the IPS bus. When the module enable is low, the xfr_wait will be asserted until the HW interface is disabled. This will effectively block any bus master from accessing the LTC until the HW interface has completed its activity. **NOTE: There is a possibility that SW could be interrupted while configuring the LTC. The BRIC module will not maintain such configuration and SW should take care to resolve this situation if it happens by re-configuring.**

In addition to blocking the system IPS bus, the DMA signals will also be blocked to avoid any spurious requests during HW interface operation. This completes the interface isolation and hi_ready will be asserted.Once the external interfaces have been isolated as per above, the HW interface will begin operation according to the following sequence (HWM indicates HW master below):

1. HWM waits for hi_ready = 0
2. HWM asserts lock_bric for continuous BRIC access for multiple key resolution
3. HWM sets key_in[127:0] and prand[23:0] to desired values
4. HWM asserts hi_mode
5. BRIC uses key_in[127:0] and prand[23:0] to configure and execute ah(key_in,prand) using the LTC as per the Bluetooth Low Energy privacy specification.
6. BRIC sets resultant hash value on hash[23:0] when the LTC is complete, then asserts hash_comp.
7. HWM waits until hash_comp is asserted
8. HWM captures the hash
9. HWM repeats steps **3-8** as needed
10. HWM de-asserts lock_bric

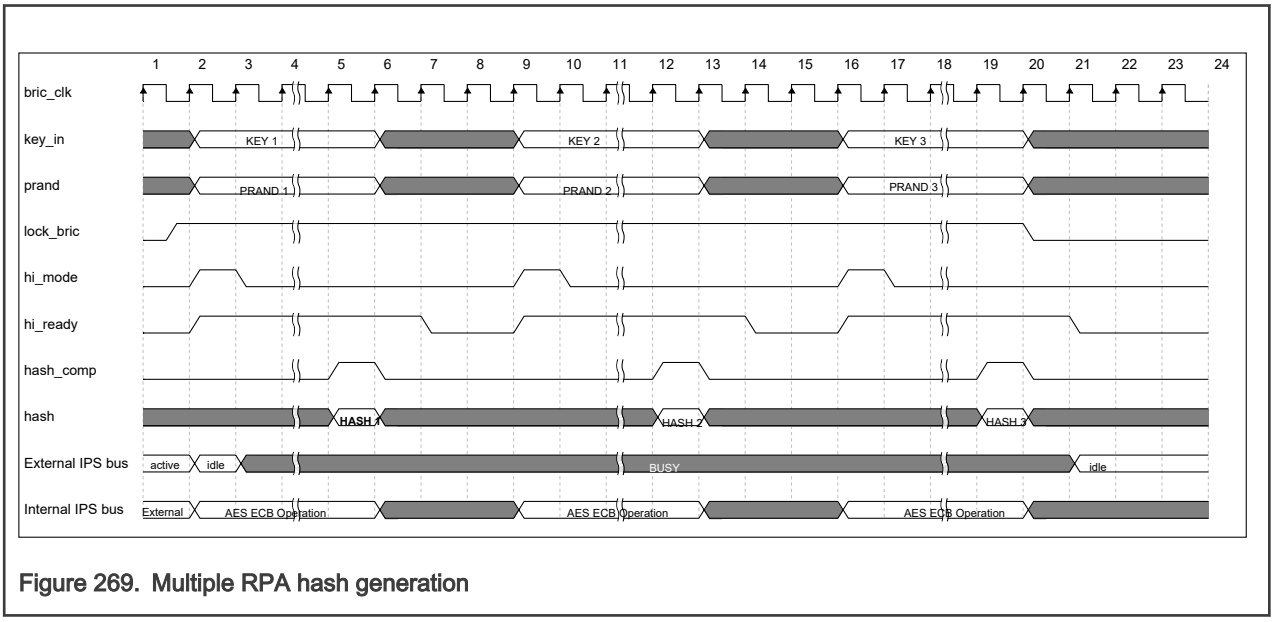


Figure 269. Multiple RPA hash generation

55.4.6.5.1.4.3 Software Access duing HW mode

To avoid any unexpected results due to software access through IPS bus during HW mode the following response methods are implemented.

IPS_XFR_WAIT is asserted through out the HW mode to inform that SW that BRIC is busy. Hence any SW access wull have to wait till the HW access is over.

IPS_XFR_ERR is asserted whenever there is a SW access during ongoign HW mode. This is to avoid SW waiting for longer time till the end of HW operation.

Both the above signal behavior can be controlled individually through register bits in BRIC_CONFIG register.

A few SW-HW access scenarios are listed in below timing diagrams.

1. HW request is asserted while SW access is ongoing - HW waits for SW access to be over
2. SW access start and HW request at the same clock - HW waits for SW access to be over

3. SW access during HW mode - respond with WAIT/ERROR/WAIT & ERROR based on the configuration

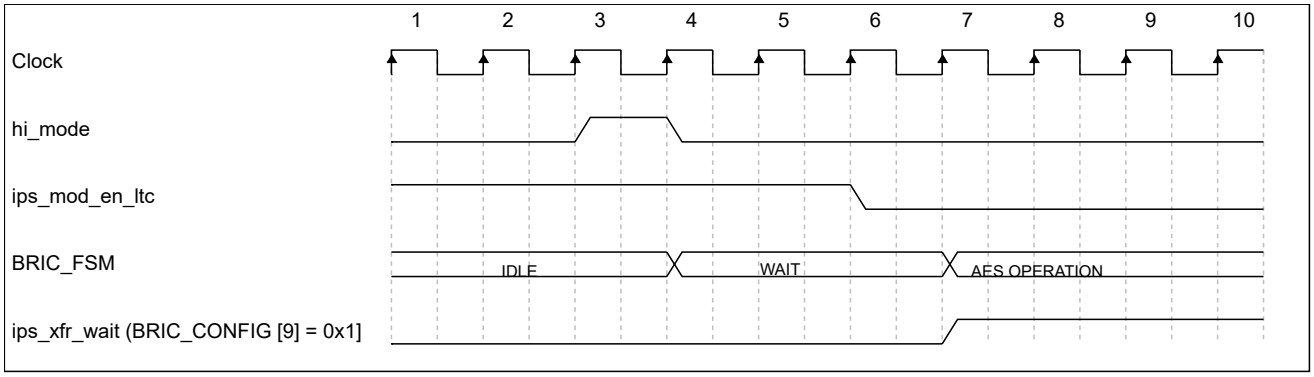
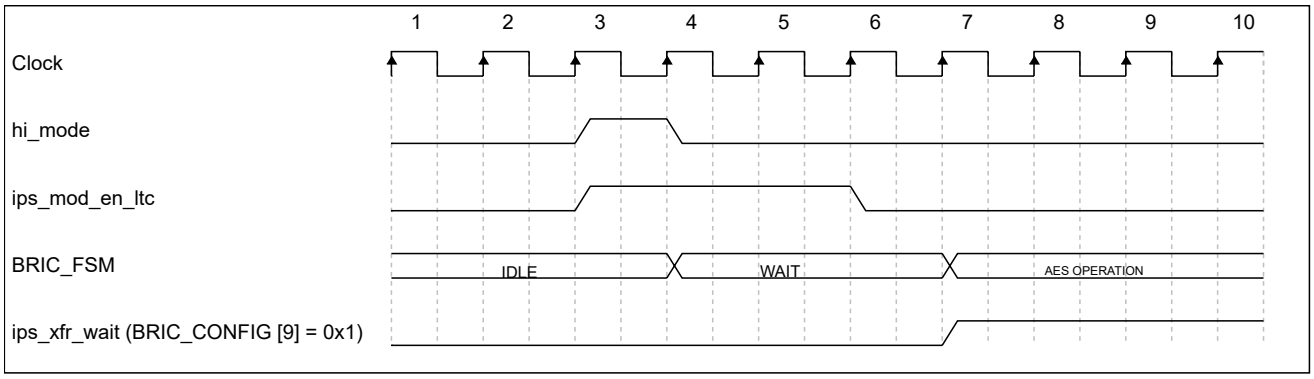


Figure 270. HW request asserted during SW access

Figure 271. SW access and HW request at same clock



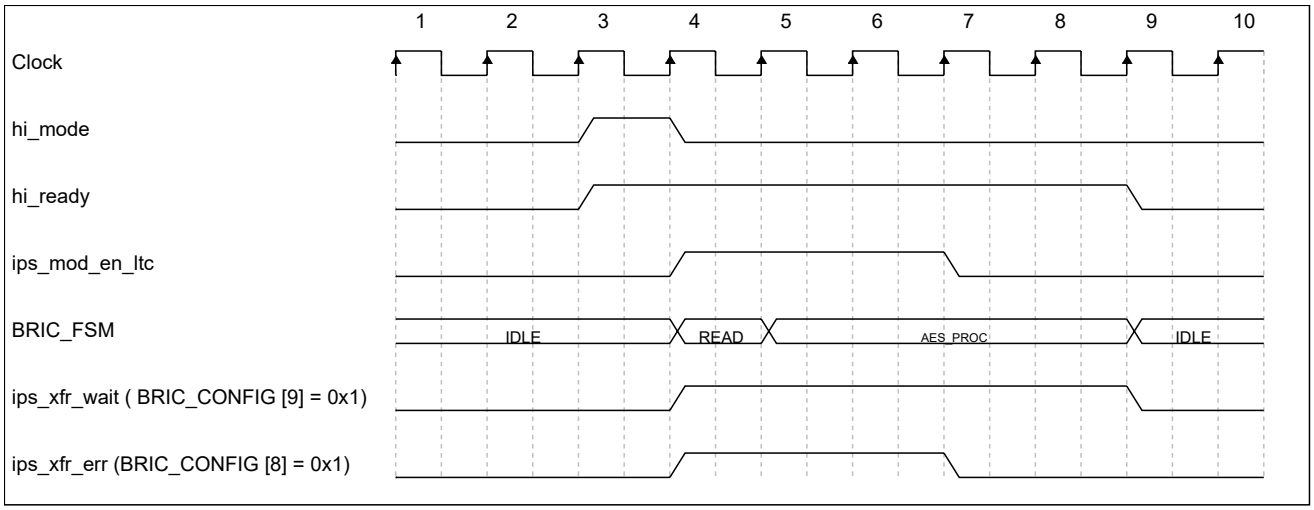


Figure 272. SW access during HW mode

55.4.6.5.1.4.4 Secure KEY transfer

The PKB master (EdgeLock Secure Enclave) can write the KEY securely over a PKB interface to any of the two KEY registers present in the BRIC. Any one of these KEYs (transferred over PKB interface) can be selectively transferred to the LTC by writing the index 0/1 to the KEY_INDEX_REGISTER.

The BRIC key_store module can accept and store two 128 bit keys from security module over the secure PKB interface. The written KEYs are no more readable from BRIC.

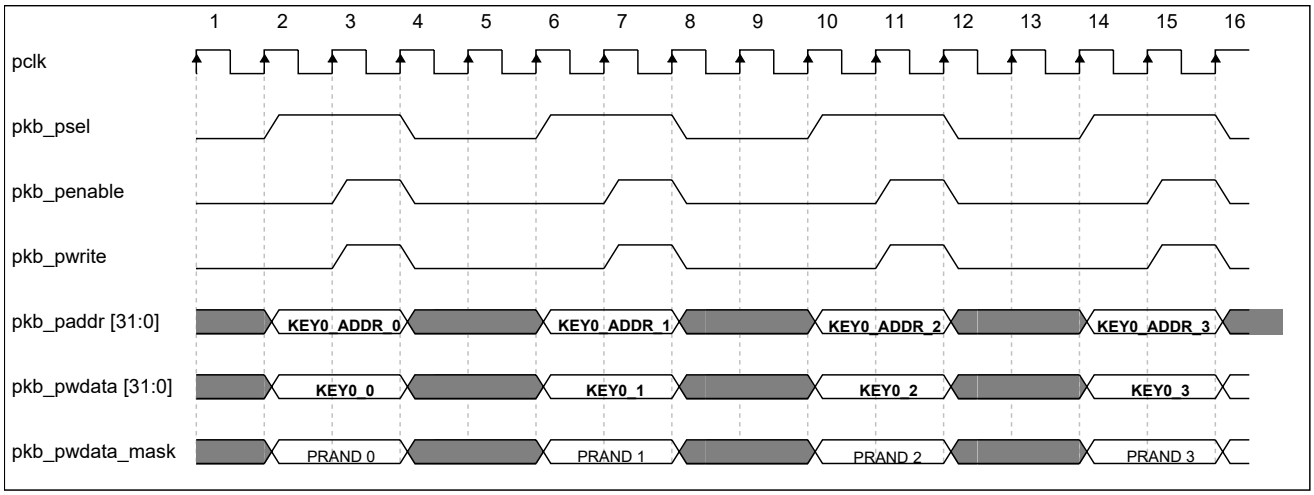


Figure 273. KEY write through PKB

Once KEYs are written by the secutiry IP t to the BRIC the procesor can initiate a KEY transfer by writing a valid value to the BRIC KEY_INDEX register. The below transactions indicate the steps involved in transferring a KEY from PKB master (security IP) to LTC module.

1. Write KEY value to BRIC key_store module through PKB interface securely. PKB bus is write only.
2. To transfer the KEY from BRIC key_store to LTC update the KEY_INDEX register with intended value
3. KEY_INDEX register update triggers KEY value write to LTC through the HW IPS interface

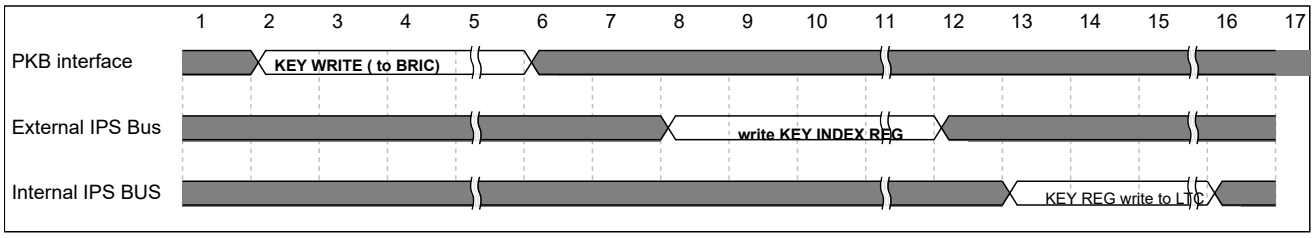


Figure 274. PKB KEY transfer to LTC

55.4.6.5.1.4.5 LTC KEY read access

LTC legacy module has a provisoin to disable the KEY read access by asserting the input signal "ltc_key_access_lock". In the BRIC this input signal has to be tied to "high" (KEY read is locked/disabled) always. To ensure a read path for the LTC KEY for special requirements like debug a different method is implemented and available for the debug software. This is by writing a key value of all 1's to KEY0 REGISTER or KEY1 REGISTER in the BRIC key store through the PKB interface. The KEY register in LTC will be readable as long as the value is retained in any of the KEYx Registers.

55.4.6.5.1.4.6 Interrupts

The LTC interrupt is passed through to the BRIC output in all modes of operation except HW interface mode.

During HW interface mode the LTC module generates a done_interrupt once encryption is complete. BRIC state machine clears this itnerrupt internally and hence this interrupt is masked from the output during HW interface mode.

55.4.6.5.1.5 Memory Map and register definition

The BRIC memory map and register description is included in the following section.

55.4.6.5.1.5.1 BRIC register descriptions

55.4.6.5.1.5.1.1 BRIC Base Address memory map

BRIC base address: 48A0_6700h

Offset	Register	Width (In bits)	Access	Reset value
0h - Ch	KEY0 Registers (PKB) (KEY0_0 - KEY0_3)	32	W	0000_0000h
10h - 1Ch	KEY1 Registers (PKB) (KEY1_0 - KEY1_3)	32	W	0000_0000h
20h	BRIC CONFIG register (BRIC_CONFIG)	32	RW	0000_0200h

55.4.6.5.1.5.1.2 KEY0 Registers (PKB) (KEY0_0 - KEY0_3)

Offset

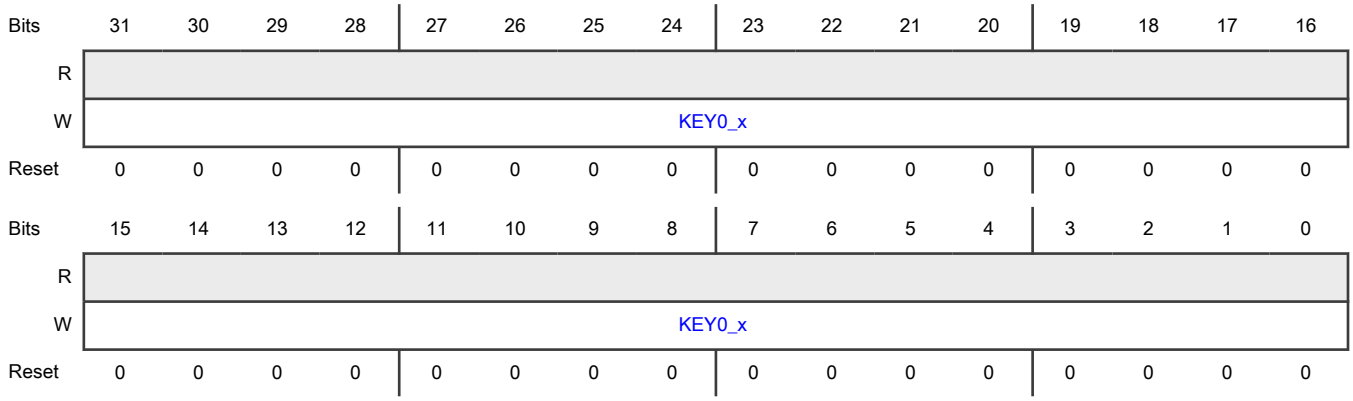
Register	Offset
KEY0_0	0h
KEY0_1	4h
KEY0_2	8h
KEY0_3	Ch

Function

The KEY0 Register normally holds the left-aligned key for the internal crypto engine written by the EdgeLock Secure Enclave. The MSB "KEY0_3" is in offset 0Ch. The Key is 128 bits in length and is written as four PKB masked write access.

The value in the Key0 Register can be transferred to the crypto engine by writing the KEY_INDEX_CONFIG register with value 00h.

Diagram



Fields

Field	Function
31-0 KEY0_x	KEY0 written through PKB interface

55.4.6.5.1.5.1.3 KEY1 Registers (PKB) (KEY1_0 - KEY1_3)

Offset

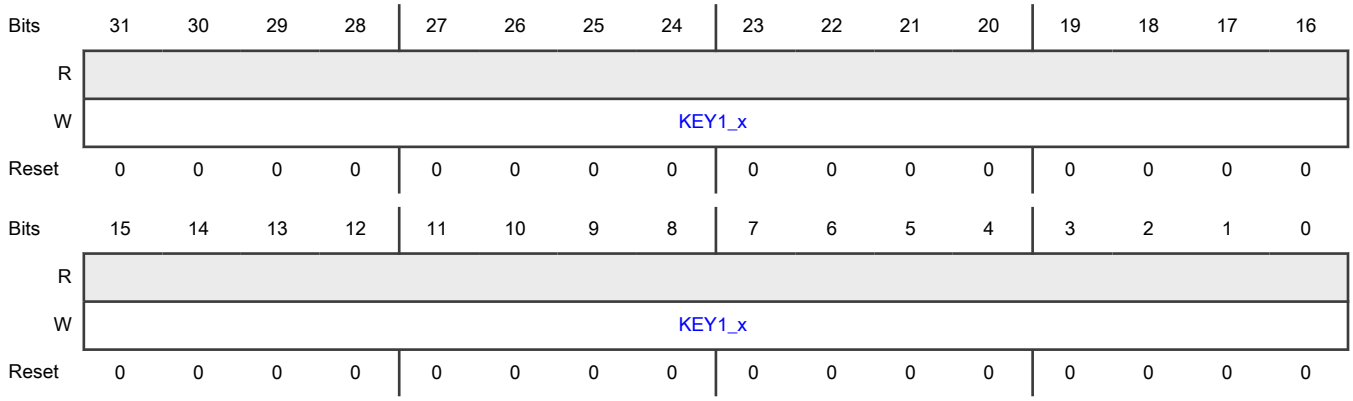
Register	Offset
KEY1_0	10h
KEY1_1	14h
KEY1_2	18h
KEY1_3	1Ch

Function

The KEY1 Register normally holds the left-aligned key for the internal crypto engine written by the EdgeLock Secure Enclave. The MSB "KEY1_3" is in offset 1Ch. The Key is 128 bits in length and is written as four PKB masked write access.

The value in the "KEY1" Register can be transferred to the crypto engine by writing the KEY_INDEX register with value 01h.

Diagram



Fields

Field	Function
31-0 KEY1_x	KEY1 written through PKB interface

55.4.6.5.1.5.1.4 BRIC CONFIG register (BRIC_CONFIG)

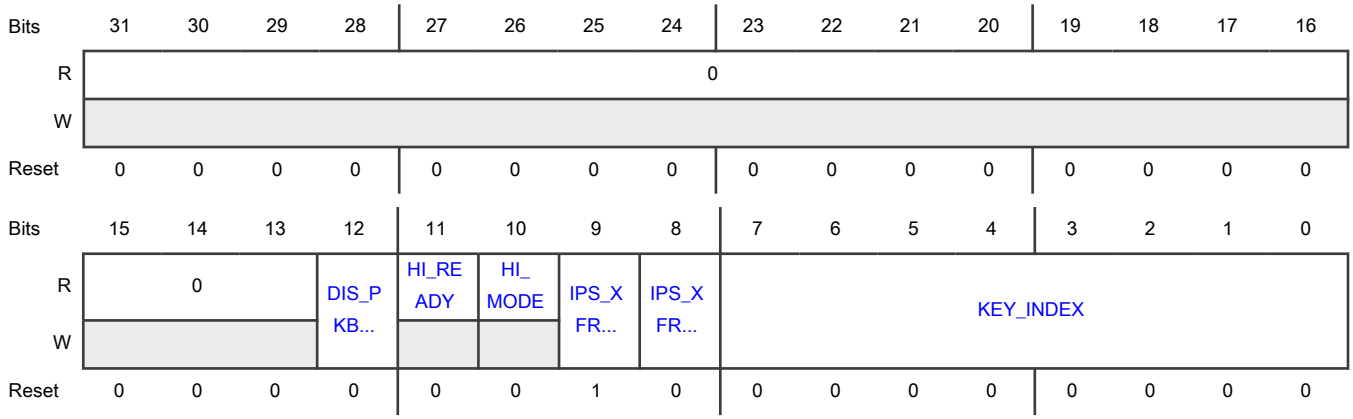
Offset

Register	Offset
BRIC_CONFIG	20h

Function

Configuration and Status register in BRIC

Diagram



Fields

Field	Function
31-13 —	This bit is reserved and should not be set.
12 DIS_PKB_ERR_RESP	Disable (0x1) PKB error response (Error response will be forced to zero)
11 HI_READY	Indicates HW mode is in progress and any SW access during this time is responded with an IPS_ERROR or IPS_WAIT depending on the configuration
10 HI_MODE	Indicates HW mode request to BRIC is active. BRIC may be in HW mode or waiting for a SW access to be over to start the HW mode when this bit is 0x1
9 IPS_XFR_WAIT_EN	Enable (0x1) ips_xfr_wait generation on IPS access during HW mode
8 IPS_XFR_ERR_EN	Enable (0x1) ips_xfr_err generation on IPS access during HW mode
7-0 KEY_INDEX	KEY INDEX KEY INDEX value for selecting the KEY from the PKB slave key array to be transferred to LTC. Writing this field with a valid value (0x0 or 0x1) triggers the KEY transfer from the PKB Slave KEY array to the LTC. To initiate KEY transfer to LTC a write has to be done to this register even if there is no change in KEY value

55.4.6.5.2 LTC

55.4.6.5.2.1 LP Trusted Cryptography Block Diagram

LP Trusted Cryptography is an architecture that allows multiple cryptographic hardware accelerator engines to be instantiated and share common registers. This version of LTC only supports AES. The following figure presents a top-level diagram of the LP Trusted Cryptography module with an AES engine.

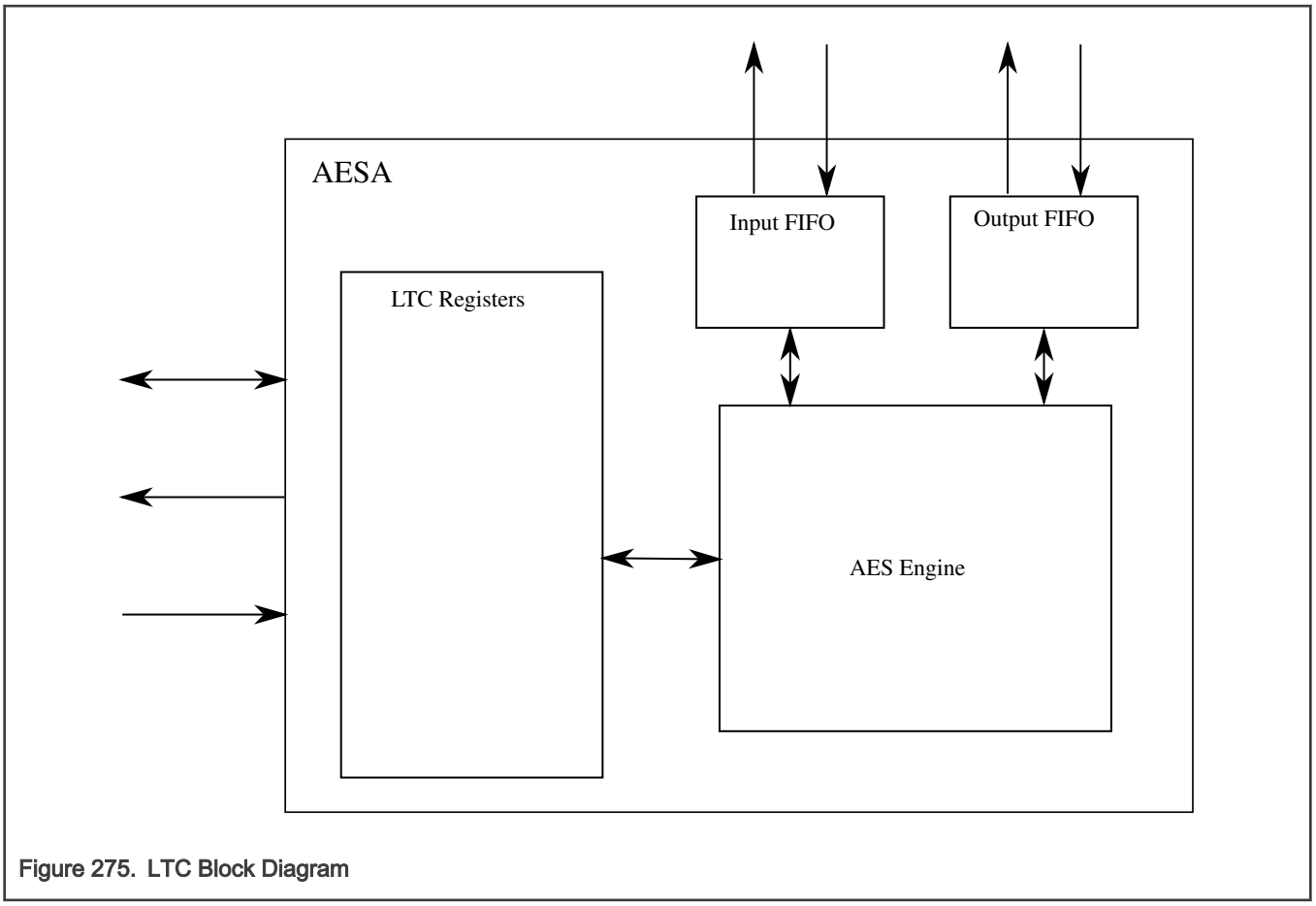


Figure 275. LTC Block Diagram

55.4.6.5.2.2 Feature summary

LTC includes the following actives:

- Cryptographic authentication
  - Message authentication codes (MAC)
    - AES-CMAC
    - AES-XCBC-MAC
- Authenticated encryption algorithms
  - AES-CCM (counter with CBC-MAC)
- Symmetric key block ciphers
  - AES (128-bit keys)
  - Cipher modes
    - ECB, CBC, CTR for AES
- Secure Scan

55.4.6.5.2.3 AES accelerator (AESA) functionality

The advanced encryption standard accelerator (AESA) module is a hardware co-processor capable of accelerating the advanced encryption standard (AES) cryptographic algorithm.

55.4.6.5.2.3.1 Differences between the AES encrypt and decrypt keys

The decrypt form of the key is different from the encrypt form of the key, because AES successively modifies the cryptographic key during the steps of the cryptographic operation. The decryption operation yields the correct result only if the modified form of the key (the decrypt key) is used at the beginning of the decryption operation. Unless told otherwise (via the DK bit in the Mode Register), AES assumes that a key loaded from memory is the encrypt key, that is, the form appropriate for encryption. If a decryption operation is specified and DK = 0, AES first goes through the steps required to derive the decrypt key from the encrypt key, and then performs the decryption operation. If a decryption operation is specified and DK = 1, the steps required to derive the decrypt key are skipped and the decryption operation is performed immediately, significantly improving performance for small data blocks.

Note that the difference between the encrypt key and the decrypt key must be taken into account when sharing keys between jobs. When an AES decryption job loads a key from memory, it is probably an encrypt key, so the DK bit in the Mode Register should be set to 0 so that AES derives the decrypt key from the encrypt key before beginning the decryption operation. But when a subsequent AES decryption job shares the key from a previous decryption job, the key that is shared is a decrypt key. In that case, the DK bit should be set to 1, which tells AES to skip the key derivation steps. If DK were set to 0 in this case, the decrypt key would be modified as if it were an encrypt key, and consequently, the wrong key value would be used in the decryption operation.

55.4.6.5.2.3.2 AESA modes of operation

The following modes are supported by AESA:

- Electronic codebook (ECB)
- Cipher block chaining (CBC)
- Counter (CTR)
- Extended cipher block chaining message authentication code (XCBC-MAC)
- Cipher-based MAC (CMAC)
- CTR and CBC-MAC (CCM)

AES modes can be classified into these categories:

- Confidentiality (ECB, CBC, CTR)
- Authenticated Confidentiality (CCM, CCM*)
- Authentication (XCBC-MAC, CMAC)

CBC Mode can also be viewed as an authentication mode when used to encrypt data, because it provides CBC-MAC in the context registers.

55.4.6.5.2.3.3 AESA use of registers

Note the following regarding the AESA's use of registers:

- For all modes, if AES is selected and the mode code written to the Mode Register does not correspond to any of the implemented AES modes, the illegal-mode error is generated.
- If ICV-only(Integrity Check Value, Final MAC) jobs are created (no data to be processed, only ICV to be checked) in modes that support ICV check, the AS mode field should be reset.

55.4.6.5.2.3.4 AES ECB mode

The electronic codebook (ECB) mode is a confidentiality mode that features, for a given key, the assignment of a fixed, ciphertext block to each plaintext block, analogous to the assignment of code words in a codebook. In ECB encryption, the forward cipher function is applied directly and independently to each block of the plaintext. The resulting sequence of output blocks is the ciphertext. In ECB decryption, the inverse cipher function is applied directly and independently to each block of the ciphertext. The resulting sequence of output blocks is the plaintext.

55.4.6.5.2.3.4.1 AES ECB mode use of the Mode Register

AES ECB mode uses the Mode Register as follows:

- The Encrypt (ENC) field should be 1 for ECB encryption and 0 for ECB decryption.
- The Algorithm State (AS) field in not used in ECB mode.
- The Additional Algorithm Information (AAI) field must be set with value 20h that activates ECB mode. Setting the MSB in the AAI field (interpreted as the Decrypt Key or DK bit for AES operations) specifies that the key loaded to the Key Register is the decryption form of the key, rather than the encryption form of the key. If DK = 0, when a decryption operation is requested AES processes the content of the Key Register to yield the decryption form of the key. If DK = 1, AES skips this processing. The illegal-mode error is generated if DK = 1 and ENC=1.
- The Algorithm (ALG) field is used to activate AESA by setting it to 10h.

55.4.6.5.2.3.4.2 AES ECB mode use of the Context Register

ECB does not use Context Registers.

55.4.6.5.2.3.4.3 AES ECB Mode use of the Data Size Register

The length of the message to be processed in bytes must be written to the Data Size register. If this value is not divisible by 16, the Data Size error is generated.

55.4.6.5.2.3.4.4 AES ECB Mode use of the Key Register

ECB keys must be written to the Key Register and can have only 16 bytes.

55.4.6.5.2.3.4.5 AES ECB Mode use of the Key Size Register

The number of bytes in the ECB key must be written to the Key Size register. Any value other than 16 causes the key-size error to be generated.

55.4.6.5.2.3.5 AES CBC mode

The CBC mode is described in this table.

Table 406. AES CBC, OFB, CFB128 modes

Name	Abbreviation	Function
Cipher-block chaining mode	CBC	Confidentiality mode whose encryption process features the combining ("chaining") of the plaintext blocks with the previous ciphertext blocks. The CBC mode requires an IV (Initialization Vector) to combine with the first plaintext block **NOTE** CBC mode uses both forward and inverse AES cipher. OFB and CFB use only forward AES cipher.

55.4.6.5.2.3.5.1 AES CBC mode use of the Mode Register

The AES CBC mode use the Mode Register as follows:

- The Encrypt (ENC) field should be 1 for encryption and 0 for decryption
- The ICV/TEST bit is not used in these modes.
- The Algorithm State (AS) field is used only in CBC mode to prevent IV update in the context for the last data block when set to "Finalize" (2h).

- The Additional Algorithm Information (AAI) field must be set with value 10h that activates CBC mode. The Decrypt Key [DK] (AAI field MSB) bit specifies that the key loaded to the Key Register is the decrypt key. The illegal mode error is generated if DK=1 and ENC=1.
- The Algorithm (ALG) field is used to activate AESA by setting it to 10h.

55.4.6.5.2.3.5.2 AES CBC mode use of the Context Register

The AES CBC mode use the Context Register as follows:

- CBC use the Context Registers to provide IV, which is updated with every processed block of a message. When a message is split into chunks and processed in multiple sessions, the IV must be saved and later restored for the next chunk to be processed correctly. At the end of CBC processing, IV is also the MAC of the message.
- If the AS field of the Mode Register is set to "Finalize" (2h) in the CBC mode, the last IV update is not written to the context. This enables CBC encryption to effectively perform ECB encryption transformation of a single-block message located in the context in place of IV, and with an all-zero block provided as input data through the FIFO without overwriting the context.

Table 407. Context usage in CBC mode

Context Word	Definition
0	IV [127:96]
1	IV [95:64]
2	IV [63:32]
3	IV [31:0]

55.4.6.5.2.3.5.3 AES CBC mode use of the Data Size Register

The AES CBC mode use the Data Size Register as follows:

- The byte length of the message to be processed must be written to the Data Size Register.
- The first write to this register initiates processing. It can also be written during processing in which case the value written is accumulated to the current state of the register.
- After the Data Size Register is written for the last time, its value must be divisible by 16 in CBC mode, otherwise the data-size error is generated.

55.4.6.5.2.3.5.4 AES CBC mode use of the Key Register

The AES CBC mode use the Key Register as follows:

- A CBC key must be written to the Key Register.
- Keys must be 16 bytes.

55.4.6.5.2.3.5.5 AES CBC mode use of the Key Size Register

The AES CBC mode use the Key Size Register as follows:

- The number of bytes in a key must be written to the Key Size register.
- Any value other than 16 causes a key-size error to be generated.

55.4.6.5.2.3.6 AES CTR mode

The counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. Note that the counter value must be unique for each data block that is encrypted with the same key. CAAM uses a 128-bit counter to ensure that the counter value will not overflow and wrap around.

NOTE

It is the user's responsibility to ensure that the same key value is not used again following a reset.

55.4.6.5.2.3.6.1 AES CTR mode use of the Mode Register

The AES CTR mode uses the Mode Register as follows:

- The Additional Algorithm Information (AAI) field should be set to 00h to activate CTR mode. If the Decrypt Key [DK] (AAI field MSB) bit is set, the illegal-mode error is generated, because CTR uses only forward AES cipher requiring encryption rather than decryption keys.
- The Algorithm State (AS) field when set to "Finalize" (2h) prevents counter update in the context for the last data block.
- The Algorithm (ALG) field is used to activate AESA by setting it to 10h.

55.4.6.5.2.3.6.2 AES CTR mode use of the Context Register

The AES CTR mode uses the Context Register as follows:

- CTR uses context words 4,5,6 and 7 to provide initial counter value (CTR0). This value is incremented with every processed block of a message. When a message is split into chunks and processed in multiple sessions, the CTRi field of context has to be saved and later restored for the next chunk to be processed correctly.
- If the AS field of the Mode Register is set to Finalize (2h) in the CTR mode, the last counter update is not written to the context. This enables CTR encryption to effectively perform ECB encryption transformation of a single-block message located in the context words 4,5,6 and 7 in place of CTR0 and with all-zero block provided as input data through the FIFO without overwriting the context.

Table 408. Context usage in CTR mode

Context Word	Initial-input definition	Context-switching definition
0	-	-
1	-	-
2	-	-
3	-	-
4	CTR0 [127:96]	CTRi [127:96]
5	CTR0 [95:64]	CTRi [95:64]
6	CTR0 [63:32]	CTRi [63:32]
7	CTR0 [31:0]	CTRi [31:0]

55.4.6.5.2.3.6.3 AES CTR mode use of the Data Size Register

The byte-length of the message to be processed must be written to the Data Size register. CTR decrements the value in this register with every processed block.

55.4.6.5.2.3.6.4 AES CTR mode use of the Key Register

- CTR key must be written to the Key Register.
- The Key Register only supports 16 byte keys.

55.4.6.5.2.3.6.5 AES CTR mode use of the Key Size Register

The number of bytes in a key must be written to the Key Size register by the time that MODE and DATA SIZE have been written. Any value other than 16 will cause Key Size error to be generated.

55.4.6.5.2.3.7 AES XCBC-MAC and CMAC modes

The AES XCBC-MAC and CMAC modes are described together because of their similarities. They are extensions of the AES CBC mode that produces a key-dependent, one--way hash (or message authentication code (MAC)) in a secure fashion across messages of varying lengths. They also provide data-integrity and data-origin authentication regarding the original message source.

55.4.6.5.2.3.7.1 AES XCBC-MAC and CMAC modes use of the Mode Register

The AES XCBC-MAC and CMAC modes use the Mode Register as follows:

- The Encrypt (ENC) bit is ignored.
- The ICV bit must be set for computed MAC to be compared with the received MAC. The received MAC must be written to the Input Data FIFO after message data and the FIFO data type must be set to ICV. If this bit is not set, XCBC-MAC and CMAC do not expect received ICV to be supplied after message data.
- The Algorithm State (AS) field is defined for XCBC-MAC as shown in this table.

Table 409. Mode Register[AS] operation selections in AES XCBC-MAC

Operation	Description
INITIALIZE	Message is processed in multiple sessions and the current session is the first one. During initialization, derived keys K3 and K2 that are XOR-ed with the last message block are computed and stored in the context to be used in the last processing session. The derived key K1 used as an AES key is computed and written back to the Key Register over the original key
INITIALIZE/FINALIZE	Message is processed in a single XCBC session and the final MAC is computed
UPDATE	Message is processed in multiple sessions and the current session is neither the first nor the last. Derived keys K2 and K3 are provided in the context and the derived key K1 is provided in the Key Register. If decryption is requested, and data size is not written or is set to 0, and ICV bit is 1 - AS = UPDATE means that Check ICV (CICV) job is requested. The CICV-only job does not process any data, it just pops received ICV/MAC from the Input Data FIFO, and compares it to the computed MAC that is restored with the rest of the context from the previous session.
FINALIZE	Message is processed in multiple sessions and the current session is the last one. Derived keys K2 and K3 are provided in the context and the derived key K1 is provided in the Key Register. The final MAC is computed

- The Algorithm State (AS) field is defined for CMAC as shown in this table.

Table 410. Mode Register[AS] operation selections in CMAC

Operation	Function
INITIALIZE	Message is processed in multiple sessions and the current session is the first one. During initialization, the constant L = E(K, 0) is computed as encrypted block of zeros using key K and stored in the context to be used in the last processing session for derivation of keys K1 and K2. One of these keys will be XOR-ed with the last message block.
INITIALIZE/ FINALIZE	Message is processed in a single session and the final MAC is computed
UPDATE	Message is processed in multiple sessions and the current session is neither the first nor the last. The constant L used for key derivation is provided in the context. If decryption is requested, and data size is not written or is set to 0, and ICV bit is 1 - AS = UPDATE means that Check ICV (CICV) job is requested. The CICV-only job does not process any data, it just pops received ICV/MAC from the Input Data FIFO, and compares it to the computed MAC that is restored with the rest of the context from the previous session
FINALIZE	Message is processed in multiple sessions and the current session is the last one. The constant L used for key derivation is provided in the context. The final MAC is computed

- If the AS field is not set to either "Initialize/Finalize" or "Finalize" and the ICV bit is set to 1, the illegal-mode error is generated, except for CICV-only jobs.
- The Additional Algorithm Information (AAI) field must be set to 70h for XCBC and 60h for CMAC to be activated. Setting the DK bit (AAI field MSB) will cause the Illegal Mode error.
- The Algorithm (ALG) field is used to activate AESA by setting it to 10h.

55.4.6.5.2.3.7.2 AES XCBC-MAC and CMAC Modes use of the Context Register

The AES XCBC-MAC and CMAC modes use the Context Register as follows:

- No data needs to be provided in the context when starting a new XCBC or CMAC session.
- The computed MAC and the derived keys K2 and K3 are written back to the context by XCBC.
- The computed MAC and the constant L = E(K,0), computed as encrypted block of zeros using key K, are written back to the context by CMAC.
- When a message is split into chunks and processed in multiple sessions, these values need to be saved before context switch and restored before the next chunk of a message is to be processed. At the end of message processing the first 2 words of the context contain the MAC value.

Table 411. Context usage in XCBC-MAC and CMAC modes

Mode	Context word	Context-switching definition	Final-result definition
XCBC-MAC	0	MAC[127:96]	MAC[127:96]
	1	MAC[95:64]	MAC[95:64]
	2	MAC[63:32]	MAC[63:32]
	3	MAC[31:0]	MAC[31:0]

Table continues on the next page...

Table 411. Context usage in XCBC-MAC and CMAC modes (continued)

Mode	Context word	Context-switching definition	Final-result definition
	4	K3[127:96]	-
	5	K3[95:64]	-
	6	K3[63:32]	-
	7	K3[31:0]	-
	8	K2[127:96]	-
	9	K2[95:64]	-
	10	K2[63:32]	-
	11	K2[31:0]	-
CMAC	0	MAC[127:96]	MAC[127:96]
	1	MAC[95:64]	MAC[95:64]
	2	MAC[63:32]	MAC[63:32]
	3	MAC[31:0]	MAC[31:0]
	4	L[127:96]	-
	5	L[95:64]	-
	6	L[63:32]	-
	7	L[31:0]	-

55.4.6.5.2.3.7.3 AES XCBC-MAC and CMAC modes use of the ICV Size Register

The AES XCBC-MAC and CMAC modes use the ICV Size Register as follows:

- This ICV register is used to provide received ICV/MAC byte-size when it is other than 16 bytes.
- The computed ICV/MAC written to the context in the XCBC mode is always 16 bytes.
- In CMAC mode, this register determines also the computed MAC size-the remaining bytes are cleared.
- Supported values for ICV size are 4 to 16 bytes. If this register is 0, the size of ICV is 16 bytes.

55.4.6.5.2.3.7.4 AES XCBC-MAC and CMAC modes use of the Data Size Register

The AES XCBC-MAC and CMAC modes use the Data Size Register as follows:

- The byte-length of the message to be processed must be written to the Data Size register.
- XCBC-MAC and CMAC decrement the value in this register with every processed block.

55.4.6.5.2.3.7.5 AES XCBC-MAC and CMAC modes use of the Key Register

The AES XCBC-MAC and CMAC modes use the Key Register as follows:

- The key must be written to this register.
- For XCBC-MAC, if the AS mode field is set to either "Initialize" or "Initialize/Finalize", it is the original XCBC key (K) that must be written here. Otherwise, the derived key (K1) must be restored to this register. CMAC only uses original key K as an AES key.

55.4.6.5.2.3.7.6 AES XCBC-MAC and CMAC modes use of the Key Size Register

The AES XCBC-MAC and CMAC modes use the Key Size Register as follows:

- The total number of key bytes must be written to the Key Size register.
- For XCBC-MAC, any value other than 16 causes a key-size error to be generated. For CMAC, this error is generated only if any value other than 16 is written.

55.4.6.5.2.3.7.7 ICV checking in AES XCBC-MAC and CMAC modes

Automatic ICV checking is enabled by setting the ICV bit of the Mode Register to 1. When ICV is set to 1, the AS mode field must be set to either "Finalize" or "Initialize/Finalize"; otherwise the illegal-mode error is generated, except for CICV-only (Check-ICV-only) jobs.

The received ICV must be provided on the FIFO after the message data. The FIFO data type must be set to ICV when it is put on the FIFO. The size of the received and computed ICV is provided in the ICV Size register.

If the ICV check detects a mismatch between the decrypted received ICV and the computed ICV, the ICV error is generated.

55.4.6.5.2.3.8 AESA CCM and CCM modes*

CCM and CCM* consists of two related processes: generation encryption and decryption verification, which combine two cryptographic primitives: counter mode encryption (CTR) and cipher-block chaining based authentication (CBC-MAC). Only the forward cipher function of the block cipher algorithm is used within these primitives. Note that the counter value must be unique for each data block that is encrypted with the same key. AES uses a 128-bit counter to ensure that the counter value does not overflow and wrap around.

NOTE

It is the user's responsibility to ensure that the same key value is not used again following a reset.

55.4.6.5.2.3.8.1 Generation encryption

A cipher-block chaining is applied to the payload, the associated data (AAD), and the nonce to generate a message authentication code (MAC); then counter mode encryption is applied to the MAC and the payload to transform them into an unreadable form, called the ciphertext. Thus, CCM generation encryption expands the size of the payload by the size of the MAC.

55.4.6.5.2.3.8.2 Decryption verification

Counter-mode decryption is applied to the purported ciphertext to recover the MAC and the corresponding payload; then cipher block chaining is applied to the payload, the received associated data, and the received nonce to verify the correctness of the MAC.

55.4.6.5.2.3.8.3 AES CCM and CCM* mode use of the Mode Register

The AES CCM and CCM* mode uses the Mode Register as follows:

- The Encrypt (ENC) bit must be set to 1 for encryption and 0 for decryption.
- The ICV bit must be set for CCM and CCM* to compare computed MAC with the received MAC when decryption is requested.
- The received MAC must be written to the input-data FIFO after message data and the FIFO data type must be set to ICV.
- Setting the ICV bit causes the received MAC to be decrypted and compared with the computed MAC.
- The number of MSBs to be compared is defined by the MAC size in the CCM and CCM* IV (B_0) as described in the CCM specification.

- If the AS field is set to FINALIZE, but ICV = 0, AESA does not expect received ICV to be put on the input-data FIFO. In that case, MAC is computed and truncated to the specified size for decryption.
- For encryption, the computed MAC is encrypted and truncated to size. The illegal-mode error is generated if ICV = 1 and ENC = 1.
- If ICV = 1 and the decrypted received MAC do not match computed MAC, the ICV error is generated.
- The Algorithm State (AS) field is defined for CCM and CCM* as follows:

Table 412. Mode Register[AS] operation selections in AES CCM and CCM*

Operation	Description
INITIALIZE	Message is processed in multiple sessions and the current session is the first one. During initialization, the initial counter CTR0 is encrypted in the CTR mode and the B0 is processed with the CBC-MAC mode. The resulting values are stored in the context. Also, the size of MAC is decoded from B0 and written to the context. This AS setting must be used whenever the first part (or whole) AAD is being processed
INITIALIZE/ FINALIZE	Message is processed in a single CCM or CCM* session and the final MAC is computed and encrypted. The initial counter CTR0 and B0 must be provided in the context
UPDATE	Message is processed in multiple sessions and the current session is neither the first nor the last. All context data is restored from the previous session and the key is written to the Key Register. If decryption is requested, and data size is not written or is set to 0, and ICV bit is 1 - AS=UPDATE means that a CICV-only job is requested. The CICV-only job does not process any data, it just pops received ICV/MAC from the Input Data FIFO, decrypts it and compares it to the computed MAC that is restored with the rest of the context from the previous session
FINALIZE	Message is processed in multiple sessions and the current session is the last one. All context data is restored from the previous session and the key is written to the Key Register. The final MAC is computed and encrypted

- Whenever AS is set to Initialize or Initialize/Finalize, context registers must be zero.
- If the AS field is not set to either Initialize/Finalize or Finalize and the ICV bit is set to 1, the illegal-mode error is generated. This does not apply in case when only ICV check is requested as described for AS = UPDATE.
- The Additional Algorithm Information (AAI) field must be set to 80h for both CCM and CCM* to be activated. The C2K bit is used to select a key register. If C2K = 0, CCM and CCM* uses the key in the Key Register. Setting the DK bit causes the illegal-mode error.
- The Algorithm (ALG) field is used to activate AESA by setting it to 10h.

55.4.6.5.2.3.8.4 AES CCM and CCM* modes use of the Context Register

The AES CCM and CCM* mode uses the Context Register as follows:

- B0 and the initial counter CTR0 must be provided in the context before the first chunk of the message is to be processed. During initialization, the initial counter CTR0 is encrypted in the CTR mode and B0 (which functions like a CBC-MAC IV in CCM and CCM*) is processed with the CBC-MAC mode. The resulting values are stored in the context. Also, the size of MAC is decoded from B0 and written to context word 13.
- If there is AAD, the first block of it defines its size, and that value is decoded and written to context word 12. All of the context data must be restored before the next chunk of the message is to be processed in multi-session processing.
- For CCM and CCM* encryption, the ICV (encrypted final MAC) is written to context words 8-11. For CCM and CCM* decryption, the ICV (received MAC), which is always encrypted, is decrypted to words 8-11. The final computed MAC is written (in clear) to context words 0-3.

Table 413. Context usage in CCM and CCM* mode encryption

Context Word	Initial-input definition	Intermediate definition	Final-output definition
0	B0[127:96]	-	MAC[127:96]
1	B0[95:64]	-	MAC[95:64]
2	B0[63:32]	-	MAC[63:32]
3	B0[31:0]	-	MAC[31:0]
4	CTR0[127:96]	CTR[127:96]	-
5	CTR0[95:64]	CTR[95:64]	-
6	CTR0[63:32]	CTR[63:32]	-
7	CTR0[31:0]	CTR[31:0]	-
8	-	E(CTR0)[127:96][1]	E(MAC)[127:96]
9	-	E(CTR0)[95:64] [1]	E(MAC)[95:64]
10	-	E(CTR0)[63:32] [1]	E(MAC)[63:32]
11	-	E(CTR0)[31:0] [1]	E(MAC)[31:0]
12	-	AAD size; see Table 415	-
13	-	MAC size; see Table 416	-

1. E(x) means encrypted x

Table 414. Context usage in CCM and CCM* modes decryption

Context Word	Initial-input definition	Context-switching Definition	Final-result definition
0	B0[127:96]	-	MAC[127:96]
1	B0[95:64]	-	MAC[95:64]
2	B0[63:32]	-	MAC[63:32]
3	B0[31:0]	-	MAC[31:0]
4	CTR0[127:96]	CTR[127:96]	-
5	CTR0[95:64]	CTR[95:64]	-
6	CTR0[63:32]	CTR[63:32]	-
7	CTR0[31:0]	CTR[31:0]	-

Table continues on the next page...

Table 414. Context usage in CCM and CCM* modes decryption (continued)

Context Word	Initial-input definition	Context-switching Definition	Final-result definition
8	-	E(CTR0)[127:96]	Decrypted Received MAC[127:96]
9	-	E(CTR0)[95:64]	Decrypted Received MAC[95:64]
10	-	E(CTR0)[63:32]	Decrypted Received MAC[63:32]
11	-	E(CTR0)[31:0] [1]	Decrypted Received MAC[31:0]
12	-	AAD size; see Table 415	-
13	-	MAC size; see Table 416	-

Table 415. Format of Context Word 12 for AES-CCM and AES-CCM* mode

Bit 31	Bits 30-16	Bits 15-0
AAD Presence Flag	0	AAD Size

Table 416. Format of Context Word 13 for AES-CCM and AES-CCM* mode

Bits 31-3	Bits 2-0
0	Encoded MAC Size

55.4.6.5.2.3.8.5 AES CCM and CCM* mode use of the Data Size Register

The AES CCM and CCM* mode uses the Data Size Register as follows:

- The byte-length of the message to be processed must be written to the Data Size register.
- CCM and CCM* decrements the value in this register with every processed block.
- The content of the Data Size register must be divisible by 16 if the AS mode field is set to either "Update" or "Initialize". Otherwise, the data-size error is generated. In other words, message splitting can be done only on a 16-byte boundary.

55.4.6.5.2.3.8.6 AES CCM and CCM* mode use of the Key Register

CCM and CCM* key must be written to this register; it is always an encryption key.

55.4.6.5.2.3.8.7 AES CCM and CCM* mode use of the Key Size Register

The AES CCM and CCM* mode uses the Key Size Register as follows:

- The total number of key bytes must be written to the Key Size register.
- Any value other than 16 causes a key-size error to be generated.

55.4.6.5.2.3.8.8 AES CCM and CCM* mode use of the ICV check

The AES CCM and CCM* mode uses ICV checking as follows:

- Automatic ICV checking is enabled by setting the ICV bit of the Mode Register to 1. When ICV is set to 1, the AS mode field must be set to either "Finalize" or "Initialize/Finalize"-otherwise the illegal-mode error is generated, unless data size is 0 indicating ICV check is only requested. Also, if ICV = 1, the ENC bit must be 0.
- The received ICV(MAC) must be provided on the input data FIFO after the message data. In CCM and CCM*, received ICV(MAC) is always encrypted. The FIFO data type must be set to ICV when it is put on the FIFO. The size of the received and computed ICV(MAC) is for CCM and CCM* encoded in the B0.
- If the ICV check detects mismatch between the decrypted received ICV(MAC) and the computed ICV(MAC), the ICV error is generated.

55.4.6.5.2.4 Standalone AES Examples

Example AES ECB Operation:

1. Write key to Primary Key Register.
2. Write key size to Primary Key Size Register. (00000010h)
3. Write Mode to Primary Mode Register. (0010020Dh)
4. Write Size of data to encrypt/decrypt to Data Size Register.
5. Write data into the Input FIFO.
6. Read data from the Output FIFO.
7. Interrupt is generated after final word is pushed to output FIFO.

Example AES CTR Operation:

1. Write key to Primary Key Register.
2. Write key size to Primary Key Size Register. (00000010h)
3. Write initial counter value to Context Words 4-7 in the Context Register.
4. Write Mode to Primary Mode Register. (0010020Dh)
5. Write Size of data to encrypt/decrypt to Data Size Register.
6. Write data into the Input FIFO.
7. Read data from the Output FIFO.
8. Interrupt is generated after final word is pushed to output FIFO.

Example AES CCM or CCM* Operation:

1. Write key to Primary Key Register.
2. Write key size to Primary Key Size Register. (00000010h)
3. Write B0 value to Context Words 0-3 in the Context Register.
4. Write initial counter value to Context Words 4-7 in the Context Register.
5. Write Mode to Primary Mode Register. (0010080Dh) CCM and CCM* both use the same mode value.
   - CCM and CCM* both use the same mode value.
6. Write Size of Authentication Only Data to the AAD Size Register.
7. Write Authentication Only data to the Input FIFO.
   - Authentication data needs to be padded to a 16 byte boundary with zeros.
   - For example if there is 8 bytes of AAD then (00000008h) should be written to AAD Size register and 8 bytes of AAD data followed by 8 bytes of Zero should be written into the Input FIFO.

8. Write Size of data to encrypt/decrypt and authenticate to Data Size Register.
9. Write data into the Input FIFO.
10. Read data from the Output FIFO.
11. Interrupt is generated after final word is pushed to output FIFO.
12. Read MAC from Context Registers
    - MAC is read from Context Registers 0-3.
    - Encrypted MAC is read from Context Registers 8-11.

Example AES CCM or CCM* Authentication Only Operation:

1. Write key to Primary Key Register.
2. Write key size to Primary Key Size Register. (00000010h)
3. Write B0 value to Context Words 0-3 in the Context Register.
4. Write initial counter value to Context Words 4-7 in the Context Register.
5. Write Mode to Primary Mode Register. (0010080Dh).
    - CCM and CCM* both use the same mode value.
6. Write Size of Authentication Only Data to the AAD Size Register.
    - The AL bit needs to be set in the AAD Size Register. This tells the AES core engine that it will recieve only Authentication Data. Note for encryption only the mechanism is handled automatically.
7. Write Authentication Only data to the Input FIFO.
    - Authentication data needs to be padded to a 16 byte boundary with zeros.
    - For example if there is 8 bytes of AAD then (00000008h) should be written to AAD Size register and 8 bytes of AAD data followed by 8 bytes of Zero should be written into the Input FIFO.
8. Write data into the Input FIFO.
9. Interrupt is generated after final word is processed from input FIFO.
10. Read MAC from Context Registers
    - MAC is read from Context Registers 0-3.
    - Encrypted MAC is read from Context Registers 8-11.

Example AES CCM or CCM* Encryption Only Operation:

1. Write key to Primary Key Register.
2. Write key size to Primary Key Size Register. (00000010h)
3. Write B0 value to Context Words 0-3 in the Context Register.
4. Write initial counter value to Context Words 4-7 in the Context Register.
5. Write Mode to Primary Mode Register. (0010080Dh).
    - CCM and CCM* both use the same mode value.
6. Write Size of data to encrypt/decrypt to Data Size Register.
7. Write data into the Input FIFO.
8. Read data from the Output FIFO.
9. Interrupt is generated after final word is pushed into the output FIFO.

Writing and Reading Data from the FIFOs:

1. Writing and reading by polling operations.

- The FIFO Status Register(LTCFIFOSTA) shows the number of entries in both the input and output FIFOs. It also shows when the FIFOs are full.
- The input and output FIFOs support 4x32bit entries each.
- Whenever there is space in the input FIFO the user can write a word into the Input FIFO.
- Whenever there is a word in the output FIFO then the user can read a word from the Output FIFO.

2. Writing and reading FIFOs by DMA operations.
   - The on chip DMA will handle all reads and writes of the FIFOs.
   - The IDE and ODE bits in the Control Register must be written to enable the DMA handshake.
     - IDE will enable dma transfers to the input FIFO when there is space available.
     - ODE will enable dma transfers from the output FIFO when there are words in the FIFO.
   - The on chip DMA should then be programmed to write data to the input FIFO and read data from the output FIFO.

55.4.6.5.2.5 LTC register descriptions

All reads of write-only addresses always return zero. Writes to read-only addresses are ignored. LTC will generate a transfer error whenever an undefined address is read or written to on the register bus. Although many of the LTC registers hold more than 32 bits, the register addresses shown in the Memory Map below represent how these registers are accessed over the register bus as 32-bit words.

NOTE

The reset value of some registers differs between different versions of LTC. To ensure driver compatibility across different versions of LTC, when updating fields within registers, the registers should first be read, the required fields updated, and then the register should be written. This will avoid inadvertently changing the settings of other fields in the same register.

55.4.6.5.2.5.1 LTC memory map

LTC base address: 48A0_6800h

Offset	Register	Width (In bits)	Access	Reset value
0h	Mode Register (MD)	32	RW	0000_0000h
8h	Key Size Register (KS)	32	RW	0000_0010h
10h	Data Size Register (DS)	32	RW	0000_0000h
18h	ICV Size Register (ICVS)	32	RW	0000_0000h
30h	Command Register (COM)	32	RW	0000_0000h
34h	Control Register (CTL)	32	RW	0000_0000h
40h	Clear Written Register (CW)	32	RW	0000_0000h
48h	Status Register (STA)	32	RW	0000_0000h
4Ch	Error Status Register (ESTA)	32	R	0000_0000h
58h	AAD Size Register (AADSZ)	32	RW	0000_0000h
100h - 134h	Context Register (CTX_0 - CTX_13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
200h - 20Ch	Key Registers (KEY_0 - KEY_3)	32	RW	0000_0000h
4F0h	Version ID Register (VID1)	32	R	0034_0100h
4F4h	Version ID 2 Register (VID2)	32	R	0000_0101h
4F8h	CHA Version ID Register (CHAVID)	32	R	0000_0050h
7C0h	FIFO Status Register (FIFOSTA)	32	R	0000_0000h
7E0h	Input Data FIFO (IFIFO)	32	W	0000_0000h
7F0h	Output Data FIFO (OFIFO)	32	R	0000_0000h

55.4.6.5.2.5.2 Mode Register (MD)

Offset

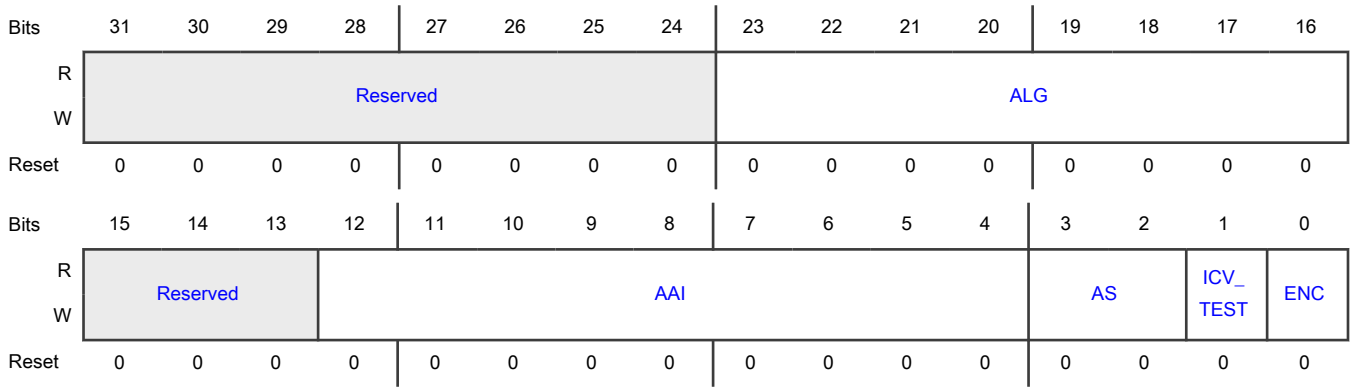
Register	Offset
MD	0h

Function

The Mode Register is used to tell the cryptographic engines which operation is being requested. The interpretation of this register will be unique for each CHA.

This section defines the format of the Mode Register when used with non-public-key algorithms and non-RNG operations.

Diagram



Fields

Field	Function
31-24	Reserved Must be 0.

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>23-16
ALG</td><td>Algorithm
This field specifies which algorithm is being selected.
0001_0000b - AES</td></tr>
<tr><td>15-13
—</td><td>Reserved Must be 0.</td></tr>
<tr><td>12-4
AAI</td><td>Additional Algorithm information
This field contains additional mode information that is associated with the algorithm that is being executed. See also the section describing the appropriate CHA.
NOTE
Some algorithms do not require additional algorithm information and in those cases this field should be all 0s.
Table 417. AAI Interpretation for AES Modes
<table>
<tr><th colspan="5">[For AES the MSB of AAI is the DK (Decrypt Key) bit.]</th></tr>
<tr><th>Code</th><th>Interpretation</th><th></th><th>Code</th><th>Interpretation</th></tr>
<tr><td>00h</td><td>CTR (mod 2^{128})</td><td></td><td>80h</td><td>CCM, CCM*</td></tr>
<tr><td>10h</td><td>CBC</td><td></td><td>90h</td><td>Reserved</td></tr>
<tr><td>20h</td><td>ECB</td><td></td><td>A0h</td><td>Reserved</td></tr>
<tr><td>30h</td><td>Reserved</td><td></td><td>B0h</td><td>CTR_XCBC_MAC</td></tr>
<tr><td>40h</td><td>Reserved</td><td></td><td>C0h</td><td>Reserved</td></tr>
<tr><td>50h</td><td>Reserved</td><td></td><td>D0h</td><td>Reserved</td></tr>
<tr><td>60h</td><td>CMAC</td><td></td><td>E0h</td><td>Reserved</td></tr>
<tr><td>70h</td><td>XCBC-MAC</td><td></td><td>F0h</td><td></td></tr>
<tr><td colspan="5">Setting the DK bit (i.e. ORing 100h with any AES code above) causes Key Register to be loaded with the AES Dcrypt key, rather than the AES Encrypt key.</td></tr>
</table></td></tr>
<tr><td>3-2
AS</td><td>Algorithm State
This field defines the state of the algorithm that is being executed. This may not be used by every algorithm.
00b - Update
01b - Initialize
10b - Finalize
11b - Initialize/Finalize</td></tr>
</table>

Table continues on the next page...

Field	Function
1 ICV_TEST	ICV Checking / Test AES fault detection. For algorithms other than AES ECB mode: ICV Checking This bit selects whether the current algorithm should compare the known ICV versus the calculated ICV. This bit will be ignored by algorithms that do not support ICV checking. 0 - Don't compare 1 - Compare For AES ECB mode: Test AES fault detection In AES ECB mode, this bit activates fault detection testing by injecting bit level errors into AES core logic as defined in the first 128 bits of the context. 0 - Don't inject bit errors 1 - Inject bit errors
0 ENC	Encrypt/Decrypt. This bit selects encryption or decryption. 0b - Decrypt. 1b - Encrypt.

55.4.6.5.2.5.3 Key Size Register (KS)

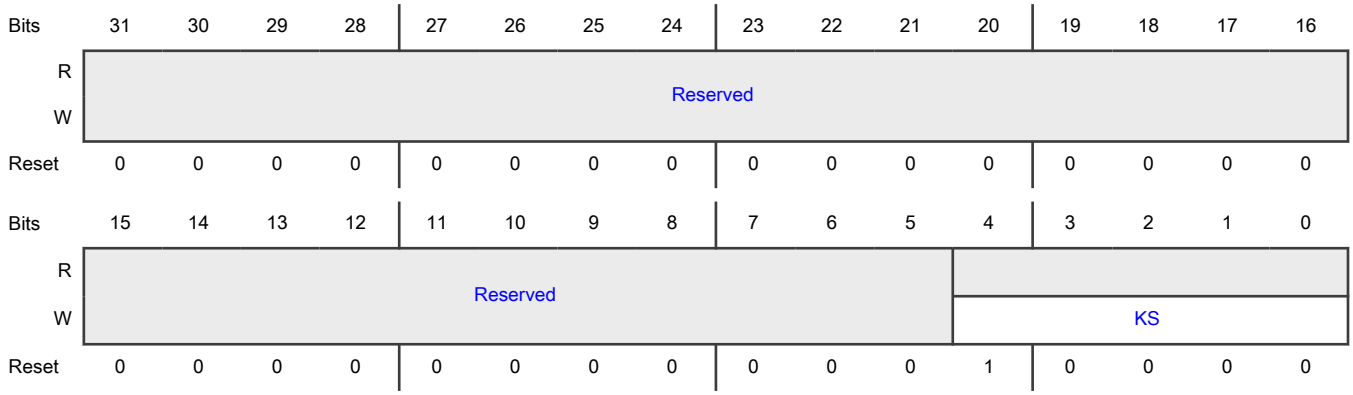
Offset

Register	Offset
KS	8h

Function

The Key Size Register is used to tell the crypto engine(AES) the size of the key that was loaded into the Key Register. The Key Size Register must be written after the key is written into the Key Register. Writing to the Key Size Register will prevent the user from modifying the Key Register. Only 16 byte keys are supported so this register will always read 16 bytes. This register is still required to be written to indicate to the AES engine that the key was loaded.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 KS	Key Size This is the size of a Key measured in bytes

55.4.6.5.2.5.4 Data Size Register (DS)

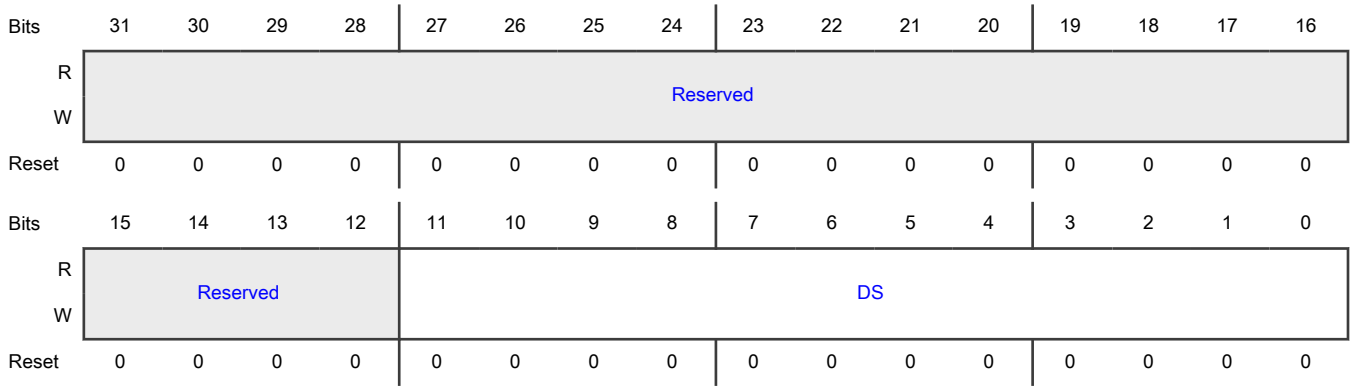
Offset

Register	Offset
DS	10h

Function

The Data Size Register is used to tell a crypto engine the amount of data that will be loaded into the Input Data FIFO. This register should only be written to once during a single operation. Note that writing to the AAD Size Register (AADSZ) , will cause this register to also update. When this register is then written directory to then the new value will be added to the previous value in the register. That is, if the DS field currently has the value 16, writing 2 to the least-significant half of the Data Size register (i.e. the DS field) will result in a value of 18 in the DS field. Note that AES decrements this register, so reading the register may return a value less than sum of the values that were written into it. This register is cleared whenever a key is decrypted or encrypted.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0	Data Size

Table continues on the next page...

Table continued from the previous page...

Field	Function
DS	This is the number of whole bytes of data that will be consumed by the CHA. Note that writing the AAD Size Register will result in this register also being written to.

55.4.6.5.2.5.5 ICV Size Register (ICVS)

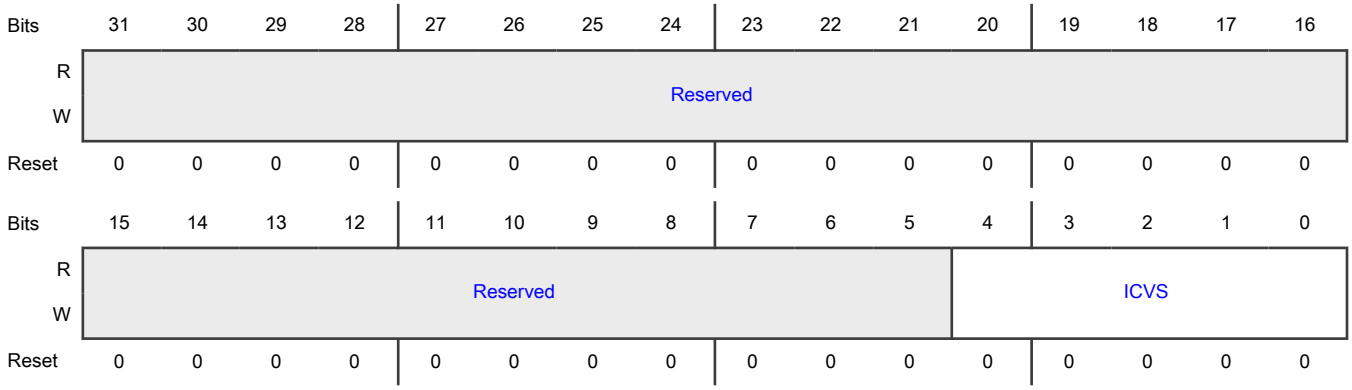
Offset

Register	Offset
ICVS	18h

Function

The ICV Size Register indicates how much of the last block of ICV is valid when performing AES integrity check modes (e.g. AES-CMAC, AES-XCBC-MAC). This register must be written prior to the corresponding word of data being consumed by AES. In practical terms, this means the register must be written prior to the corresponding data being written to the Input Data FIFO.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 ICVS	ICV Size, in Bytes

55.4.6.5.2.5.6 Command Register (COM)

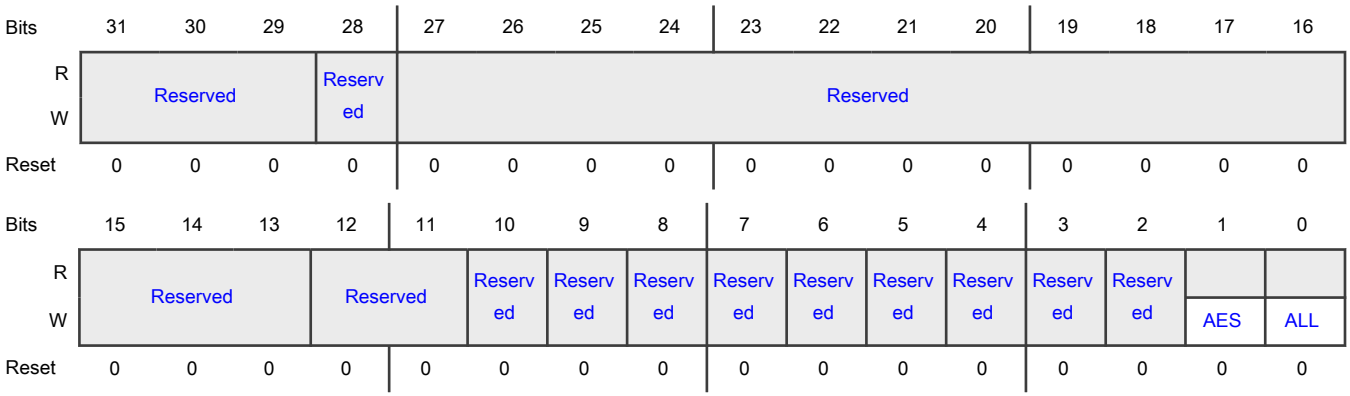
Offset

Register	Offset
COM	30h

Function

The Command Register is used to send control signals to the Crypto Engines.

Diagram



Fields

Field	Function
31-29 —	Reserved To preserve software compatibility with other versions of LTC, 0 should be written to all reserved bits.
28 —	Reserved
27-13 —	Reserved
12-11 —	Reserved
10 —	Reserved
9 —	Reserved
8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 AES	Reset AESA Writing a 1 to this bit resets the AES Accelerator core engine. 0b - Do Not Reset 1b - Reset AES Accelerator
0 ALL	Reset All Internal Logic Writing to this bit will reset all accelerator engines and as well as all the internal registers. 0b - Do Not Reset 1b - Reset all CHAs in use by this CCB.

55.4.6.5.2.5.7 Control Register (CTL)

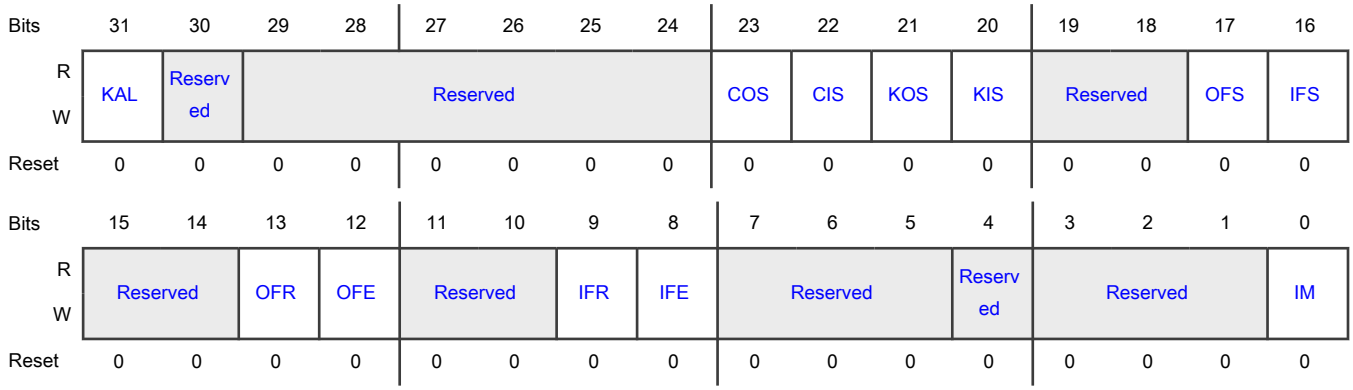
Offset

Register	Offset
CTL	34h

Function

This register is used for some of the internal controls of the LTC block.

Diagram



Fields

Field	Function
31 KAL	Key Register Access Lock Read access to the key register is blocked. Any reads of the key register will only return zero. Once this bit is set, it can only be cleared by hard reset. 0b - Key Register is readable. 1b - Key Register is not readable.
30 —	Reserved
29-24 —	Reserved
23 COS	Context Register Output Byte Swap Byte swap all data that is read from the context register. Data is byte swapped only within a single word. 0b - Do Not Byte Swap Data. 1b - Byte Swap Data.
22 CIS	Context Register Input Byte Swap Byte swap all data that is written to the context register. Data is byte swapped only within a single word. 0b - Do Not Byte Swap Data. 1b - Byte Swap Data.
21 KOS	Key Register Output Byte Swap Byte swap all data that is read from the key register. Data is byte swapped only within a single word. 0b - Do Not Byte Swap Data. 1b - Byte Swap Data.
20	Key Register Input Byte Swap

Table continues on the next page...

Table continued from the previous page...

Field	Function
KIS	Byte swap all data that is written to the key register. Data is byte swapped only within a single word. 0b - Do Not Byte Swap Data. 1b - Byte Swap Data.
19-18 —	Reserved
17 OFS	Output FIFO Byte Swap Byte swap all data that is read from the Onput FIFO. 0b - Do Not Byte Swap Data. 1b - Byte Swap Data.
16 IFS	Input FIFO Byte Swap Byte swap all data that is written to the Input FIFO. 0b - Do Not Byte Swap Data. 1b - Byte Swap Data.
15-14 —	Reserved
13 OFR	Output FIFO DMA Request Size The DMA request logic will only request data if the OUTPUT FIFO has enough data to satisfy the request. 0b - DMA request size is 1 entry. 1b - DMA request size is 4 entries.
12 OFE	Output FIFO DMA Enable 0b - DMA Request and Done signals disabled for the Output FIFO. 1b - DMA Request and Done signals enabled for the Output FIFO.
11-10 —	Reserved
9 IFR	Input FIFO DMA Request Size The DMA request logic will only request data if the INPUT FIFO has enough space for the request size. 0b - DMA request size is 1 entry. 1b - DMA request size is 4 entries.
8 IFE	Input FIFO DMA Enable 0b - DMA Request and Done signals disabled for the Input FIFO. 1b - DMA Request and Done signals enabled for the Input FIFO.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 IM	Interrupt Mask Once this bit is set, it can only be cleared by hard reset. 0b - Interrupt not masked. 1b - Interrupt masked

55.4.6.5.2.5.8 Clear Written Register (CW)

Offset

Register	Offset
CW	40h

Function

The Clear Written Register is used to clear many of the internal registers. All fields of this register are self-clearing.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			Reserved		Reserved											
W	CIF	COF														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved											Reserved			Reserved	
W										CKR	CCR		CICV	CDS		CM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 CIF	Clear Input FIFO Writing a 1 to this bit causes the Input Data FIFO.
30 COF	Clear Output FIFO Writing a 1 to this bit causes the Output FIFO to be cleared.
29-28 —	Reserved
27-16 —	Reserved
15-7 —	Reserved
6 CKR	Clear the Key Register Writing a one to this bit causes the Key and Key Size Registers to be cleared.
5 CCR	Clear the Context Register Writing a one to this bit causes the Context Register to be cleared.
4 —	Reserved
3 CICV	Clear the ICV Size Register Writing a one to this bit causes the ICV Size Register to be cleared.
2 CDS	Clear the Data Size Register Writing a one to this bit causes the Data Size Register to be cleared. This clears AAD Size as well.
1 —	Reserved
0 CM	Clear the Mode Register Writing a one to this bit causes the Mode Register to be cleared.

55.4.6.5.2.5.9 Status Register (STA)

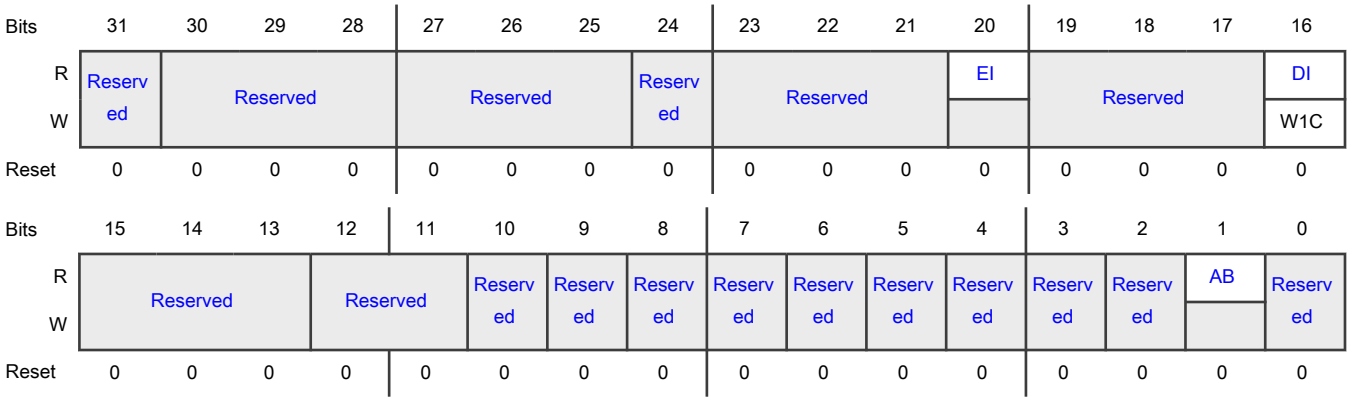
Offset

Register	Offset
STA	48h

Function

The Status Register shows the status of the internal Crypto engine and its internal registers.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27-25 —	Reserved
24 —	Reserved
23-21 —	Reserved
20 EI	Error Interrupt The Error Interrupt has been asserted. This error can only be cleared by resetting LTC. 0b - Not Error. 1b - Error Interrupt.
19-17 —	Reserved
16 DI	Done Interrupt The Done Interrupt has been asserted.

Table continued from the previous page...

Field	Function
	<table><tr><th>Value</th><th>Read</th><th>Write</th></tr><tr><td>0</td><td>No Done Interrupt</td><td>No change</td></tr><tr><td>1</td><td>Done Interrupt asserted</td><td>Clear the Done Interrupt</td></tr></table>
15-13 —	Reserved
12-11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 AB	AESA Busy This bit indicates that the AES Accelertor is busy. The CHA can either be busy processing data or resetting.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - AESA Idle 1b - AESA Busy.
0 —	Reserved

55.4.6.5.2.5.10 Error Status Register (ESTA)

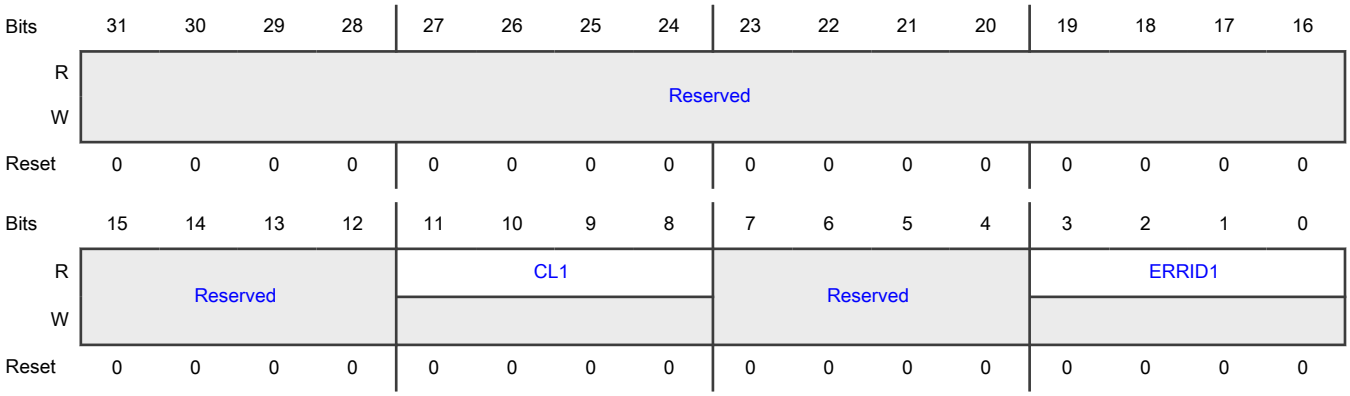
Offset

Register	Offset
ESTA	4Ch

Function

The Error Register shows the status of the internal Crypto Engine and its internal registers.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 CL1	algorithms The algorithms field indicates which algorithm is asserting an error. Others reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000b - General Error 0001b - AES
7-4 —	Reserved
3-0 ERRID1	Error ID 1 These bits indicate the type of error that was found while processing the Descriptor. The Algorithm that is associated with the error can be found in the CL1 field. Others reserved. 0001b - Mode Error 0010b - Data Size Error 0011b - Key Size Error 0110b - Data Arrived out of Sequence Error 1010b - ICV Check Failed 1011b - Internal Hardware Failure 1100b - CCM AAD Size Error (either 1. AAD flag in B0 =1 and no AAD type provided, 2. AAD flag in B0 = 0 and AAD povided, or 3. AAD flag in B0 =1 and not enough AAD provided - expecting more based on AAD size.) 1111b - Invalid Crypto Engine Selected

55.4.6.5.2.5.11 AAD Size Register (AADSZ)

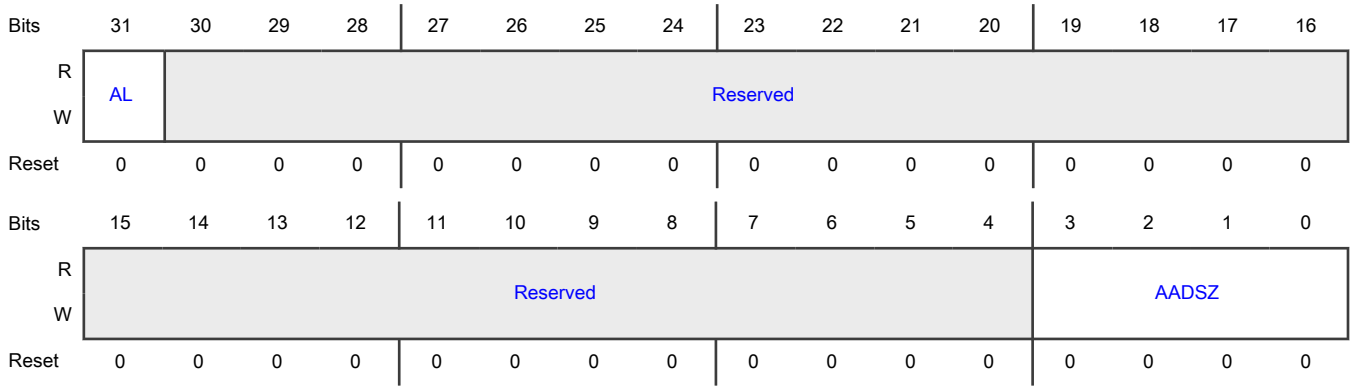
Offset

Register	Offset
AADSZ	58h

Function

The AAD Size Register is used by AESA to determine how much of the last block of AAD is valid. The write to this register should be the entire size of the AAD as it is also added directly to the Data Size Register. The size added to the Data Size Register is the AAD size rounded up to the next 16 byte boundary. For instance a size of 20 bytes written to the AAD size register will cause 32 bytes to be added to the Data Size Register. The size stored in the AADSZ field represents the number of bytes valid in the final block of AAD. However the entire size of AAD should be written to the AAD Size Register (AADSZ) Register address location. When authentication only is being done then the AL bit needs to be written to tell the AES engine that this is the last of the data.

Diagram



Fields

Field	Function
31 AL	AAD Last Only AAD data will be written into the Input FIFO.
30-4 —	Reserved
3-0 AADSZ	AAD size in Bytes, mod 16

55.4.6.5.2.5.12 Context Register (CTX_0 - CTX_13)

Offset

For a = 0 to 13:

Register	Offset
CTX_a	100h + (a × 4h)

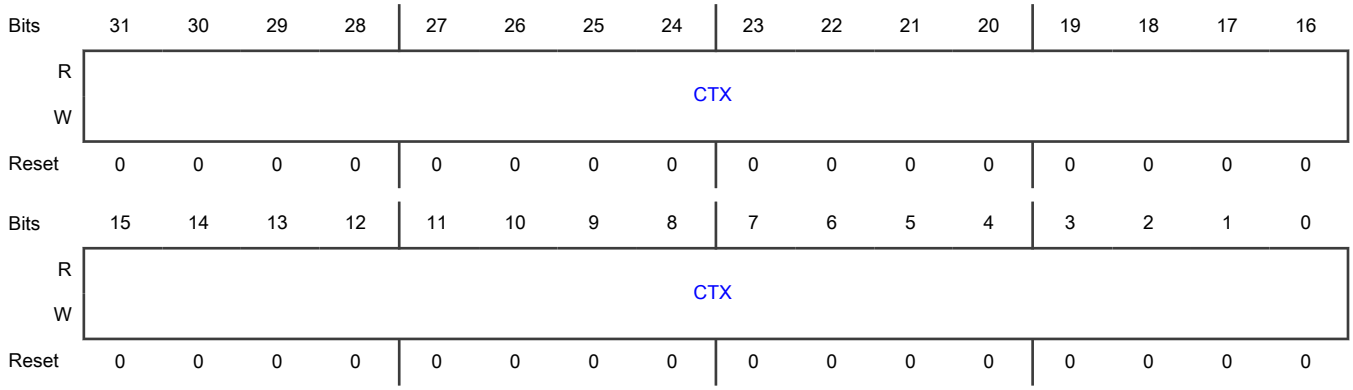
Function

The Context Register holds the context for the internal crypto engine. This register is 448 bits in length. The IP bus write to the Context Register is accessible only as full-word reads or writes to fourteen 32-bit registers. The MSB is located at offset 0100h with respect to the register page.

The bit assignments of this register are dependent on the algorithm, and in some cases the mode of that algorithm. See the appropriate section for the Context Register format used for that algorithm:

- AES ECB: Section AES ECB mode use of the Context Register
- AES CBC: Section AES CBC mode use of the Context Register
- AES CTR: Section AES CTR mode use of the Context Register
- AES CCM: Section AES CCM and CCM* modes use of the Context Register

Diagram



Fields

Field	Function
31-0 CTX	CTX

55.4.6.5.2.5.13 Key Registers (KEY_0 - KEY_3)

Offset

Register	Offset
KEY_0	200h
KEY_1	204h
KEY_2	208h
KEY_3	20Ch

Function

The Key Register normally holds the left-aligned key for the internal crypto engine. The MSB is in offset 200h. The Key Register is 128 bits in length. The IP bus write to the Context Register is accessible only as full-word reads or writes to four 32-bit registers.

Before the value in the Key Register can be used in a cryptographic operation, the size of the key must be written into the Key Size Register. Once the Key Size Register has been written, the Key Register cannot be written again until the Key Size Register has been cleared.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KEY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	KEY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 KEY	KEY

55.4.6.5.2.5.14 Version ID Register (VID1)

Offset

Register	Offset
VID1	4F0h

Function

This register contains the ID for LTC and major and minor revision numbers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJ_REV								MIN_REV							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 IP_ID	ID(0x0034).
15-8 MAJ_REV	Major revision number.
7-0 MIN_REV	Minor revision number.

55.4.6.5.2.5.15 Version ID 2 Register (VID2)

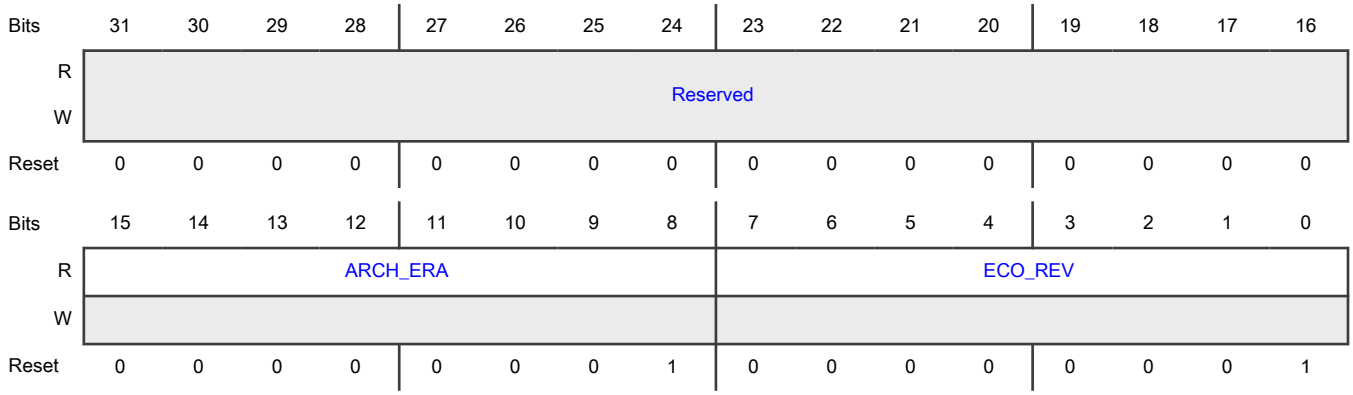
Offset

Register	Offset
VID2	4F4h

Function

This register contains the architectural era and eco revision numbers.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8	Architectural ERA.

Table continues on the next page...

Table continued from the previous page...

Field	Function
ARCH_ERA	
7-0 ECO_REV	ECO revision number.

55.4.6.5.2.5.16 CHA Version ID Register (CHAVID)

Offset

Register	Offset
CHAVID	4F8h

Function

This register contains the Version ID and Revision Number for the CHAs contained within LTC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								AESVID				AESREV			
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 AESVID	AES Version ID
3-0 AESREV	AES Revision Number

55.4.6.5.2.5.17 FIFO Status Register (FIFOSTA)

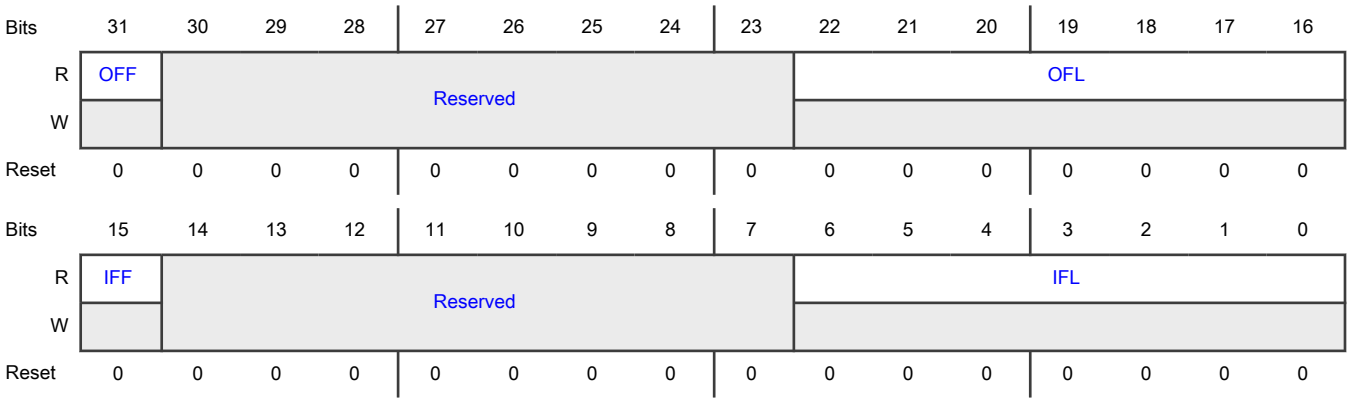
Offset

Register	Offset
FIFOSTA	7C0h

Function

The FIFO Status shows the current levels of the Input and Output FIFO.

Diagram



Fields

Field	Function
31 OFF	Output FIFO Full The Output FIFO is full and should not be written to.
30-23 —	Reserved
22-16	Output FIFO Level

Table continues on the next page...

Table continued from the previous page...

Field	Function
OFL	These bits indicate the current number of entries in the Output FIFO.
15 IFF	Input FIFO Full The Input FIFO is full and should not be written to.
14-7 —	Reserved
6-0 IFL	Input FIFO Level These bits indicate the current number of entries in the Input FIFO.

55.4.6.5.2.5.18 Input Data FIFO (IFIFO)

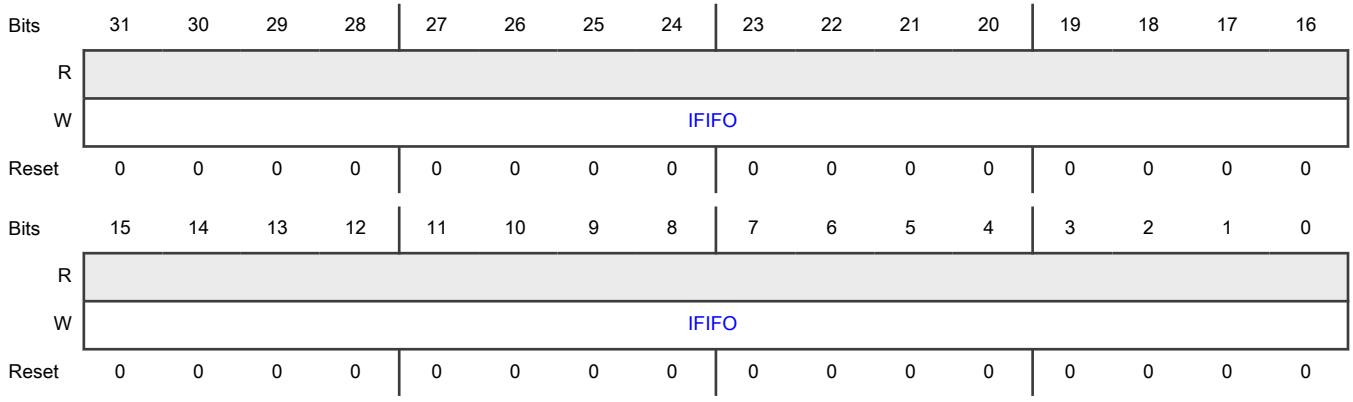
Offset

Register	Offset
IFIFO	7E0h

Function

Data to be processed by the various crypto engines is first pushed into the Input Data FIFO. The Input Data FIFO supports byte enables, allowing one to four bytes to be written to the IFIFO from the IP bus. The IFIFO is four entries deep, and each entry is four bytes. Care must be used to not overflow the Input Data FIFO. Reads from this address will always return 0x0.

Diagram



Fields

Field	Function
31-0 IFIFO	IFIFO

55.4.6.5.2.5.19 Output Data FIFO (OFIFO)

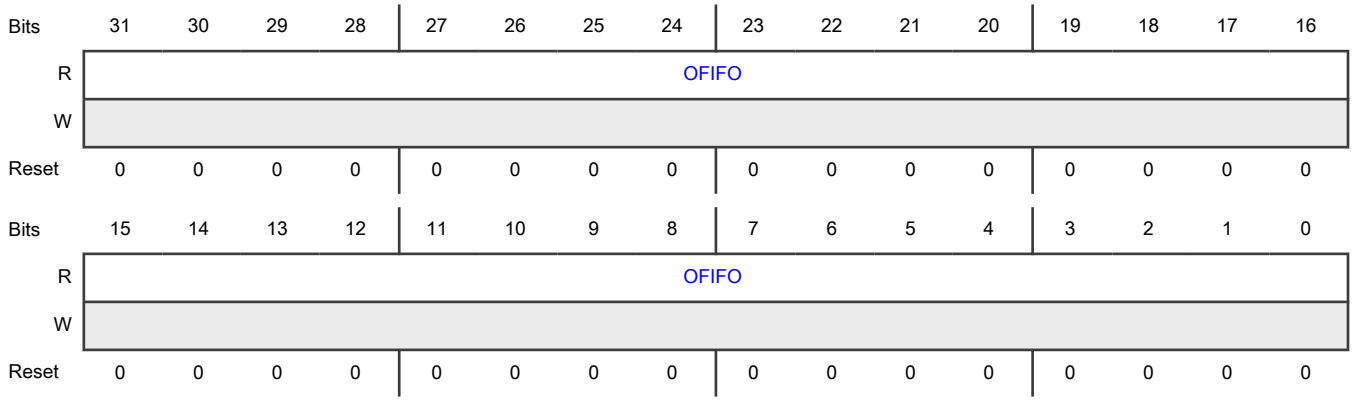
Offset

Register	Offset
OFIFO	7F0h

Function

Data that is output from the AES is pushed into the Output Data FIFO. The OFIFO is four entries deep, and each entry is four bytes. During normal operation, the AES will never overflow the Output Data FIFO. Writes to this register are ignored.

Diagram



Fields

Field	Function
31-0 OFIFO	Output FIFO

55.4.6.6 WiFi Coexistence Interface

The 2.4GHz radio architecture supports a flexible coexistence interface that enables the narrowband (NB) radio (e.g., Bluetooth, IEEE 802.15.4 and proprietary protocols) to coexist in the same space as a WiFi radio, which shares the same frequency band, or a different IC, with which active synchronization may be required to reduce either power consumption from battery or reduce electro-magnetic interference (EMI). As a simplification, this section primarily showcases the coexistence capabilities of the NB 2.4GHz radio in the context of WiFi radio coexistence. Note that an inter-processor communication (IPC) channel using a serial bus may also be used to augment the capabilities of the HW coexistence interface.

Typical coexistence schemes between a Bluetooth and WiFi designate the WiFi transceiver as the master, and the 2.4GHz transceiver as the slave. However in some cases, an application profile may dictate the opposite. The coexistence strategy is to prevent both the NB 2.4GHz radio and the WiFi radio from transmitting simultaneously; a configuration option exists to prevent the NB 2.4GHz radio from reception when the WiFi radio owns the RF channel. Yet another implementation choice is for the two radios to indicate the nature of their RF activity allowing the coexistence scheme to allow for simultaneous reception on both radios as well as other options when NB and WiFi channels have adequate separation.

55.4.6.6.1 Coexistence Pin Definition

The WiFi coexistence pins are described in the following table.

WiFi Coexist Function	Pin Direction	Pin Description
RF_ACTIVE (REQUEST)	O	An output which is asserted prior to any Radio event and remains asserted for the duration of the event.
RF_STATUS (DIRECTION)	O	An output which indicates when the Radio is in an RX or TX event, software can also control this signal directly.
RF_PRIORITY	O	An output which indicates to the external WiFi device that the Radio event is a high priority and it needs access to the 2.4 GHz antenna.
RF_NOT_ALLOWED (!GRANT)	I	External signal which causes the internal Radio to cease radio activity.
RF_TX_CONF	I	Signal from an external Radio which indicates the availability of the 2.4 GHz antenna to the internal Radio. Note that this signal is not connected to the Radio hardware. Radio software can use any interrupt-capable GPIO which the application assigns for this function.

Behavioral control information for the radio pins is provided below. Refer also to Radio Coexistence/FEM/LANT Connections which illustrates how the signals are connected and muxed in the radio.

55.4.6.6.1.1 RF_NOT_ALLOWED Behavioral Control

The WiFi IC generates a signal, RF_NOT_ALLOWED. If this signal is asserted, then 2.4GHz radio does not perform any communication. When this signal is de-asserted, then 2.4GHz radio is free to perform communications. In case when the signal RF_NOT_ALLOWED is asserted when 2.4GHz radio has already initiated the transmission/reception of a packet then the 2.4GHz radio must stop its activity immediately. Note that a software override of the RF_NOT_ALLOWED signal is also supported by the radio, which can be used for debug as well as enabling the capability to abort radio activity under software control (based on application layer arbitration between a multi-radio system).

The table below describes register bits in the RFMC and RADIO_MISC to enable, control and provide status for RF_NOT_ALLOWED

Field	Type	Description
RFNA_IBE[2:0]	RW	RFMC bitfield to select which (among several) SOC pin to use for RF_NOT_ALLOWED function
RF_NOT_ALLOWED_EN[3:0]	RW	RADIO_MISC (COEX_CTRL register) bitfield to choose which link layers will receive RF_NOT_ALLOWED signal
RF_NOT_ALLOWED_ASSERTED	W1C	RADIO_MISC (COEX_CTRL register) bitfield sets on assertion of RF_NOT_ALLOWED, cleared only by software.
RF_NOT_ALLOWED	R	RADIO_MISC (COEX_CTRL register) bitfield reflects the raw state of the selected RF_NOT_ALLOWED SOC pin
RF_NOT_ALLOWED_OVRD_EN	RW	RADIO_MISC (COEX_CTRL register) bitfield. If set, allows software to control the RF_NOT_ALLOWED signal to the link layers via the RF_NOT_ALLOWED_OVRD bitfield
RF_NOT_ALLOWED_OVRD	RW	RADIO_MISC (COEX_CTRL register) bitfield. If RF_NOT_ALLOWED_OVRD_EN is set, the RF_NOT_ALLOWED signal to the link layers is driven with the value of the RF_NOT_ALLOWED_OVRD bitfield
RF_NOT_ALLOWED_INV	RW	RADIO_MISC (COEX_CTRL register) bitfield. If set, the selected RF_NOT_ALLOWED pin is inverted before being used as the RF_NOT_ALLOWED signal to the link layers

55.4.6.6.1.2 RF_ACTIVE Behavioral Control

The NB radio provides the RF_ACTIVE (REQUEST) output to request coexistence access. This signal can be generated by either the TSM, one of the link layers, or the RFMC.

The table below describes register bits in the TSM, RFMC and RADIO_MISC to enable, control, and provide status for RF_ACTIVE. Refer to the link layer chapters for information on their REQUEST output.

Field	Type	Description
RF_ACTIVE_RX_HI[7:0]	RW	TSM bitfield to control when TSM RF_ACTIVE asserts in the RX sequence
RF_ACTIVE_RX_LO[7:0]	RW	TSM bitfield to control when TSM RF_ACTIVE de-asserts in the RX sequence
RF_ACTIVE_TX_HI[7:0]	RW	TSM bitfield to control when TSM RF_ACTIVE asserts in the TX sequence
RF_ACTIVE_TX_LO[7:0]	RW	TSM bitfield to control when TSM RF_ACTIVE de-asserts in the TX sequence
TSM_RF_ACTIVE_OVRD_EN	RW	TSM bitfield to provide override enable for TSM RF_ACTIVE output
TSM_RF_ACTIVE_OVRD	RW	TSM bitfield to provide override value for TSM RF_ACTIVE output
TSM_SPARE1_EXTEND[7:0]	RW	TSM bitfield to control when how long TSM RF_ACTIVE remains asserted after end of RX or TX sequence. This is intended to close any gaps which may occur between consecutive RX/TX or TX/RX operations
COEX_SEL	RW	RADIO_MISC (COEX_CTRL register) bitfield to select whether the TSM outputs (RF_ACTIVE, RF_STATUS, RF_PRIORITY) or the active Link Layer's outputs are input to the RFMC.
RFACT_SRC[1:0]	RW	RFMC bitfield to select whether RF_ACTIVE radio output is driven by RFMC (2'b00), TSM/LL output from the COEX_SEL mux (2'b01), or Bluetooth LE bt_clk_req signal (2'b10)
RFACT_WKUP_DLY[5:0]	RW	RFMC bitfield. If RFACT_SRC=2'b00, this configures number of 32kHz reference clocks following enable of the XO to assert the RF_ACTIVE pin for a MAN/WOR/Bluetooth LE wakeup event.
RFACT_IDIS	RW	RFMC bitfield. If RFACT_SRC=2'b00, this selects whether the RF_ACTIVE radio output de-asserts when the TSM becomes idle
RFACT_EN	RW	RFMC bitfield. If RFACT_SRC=2'b00, the RF_ACTIVE radio output is asserted while this bit is set
RFACT_FLAG	W1C	RFMC bitfield. Sets when the RF_ACTIVE radio output asserts
RFACT_IE	RW	RFMC bitfield. If set, the RFMC will generate an interrupt to CM33 when the RF_ACTIVE radio output asserts

55.4.6.6.1.3 RF_STATUS Behavioral Control

The NB radio provides the RF_STATUS (DIRECTION) output to indicate when the radio is in an RX or TX event. This signal can be generated by either the TSM or the active link layers.

The table below describes register bits in the TSM and RADIO_MISC to enable and control RF_STATUS. Refer to the link layer chapters for information on their DIRECTION output.

Field	Type	Description
RF_STATUS_RX_HI[7:0]	RW	TSM bitfield to control when TSM RF_STATUS asserts in the RX sequence

Table continues on the next page...

Table continued from the previous page...

RF_STATUS_RX_LO[7:0]	RW	TSM bitfield to control when TSM RF_STATUS de-asserts in the RX sequence
RF_STATUS_TX_HI[7:0]	RW	TSM bitfield to control when TSM RF_STATUS asserts in the TX sequence
RF_STATUS_TX_LO[7:0]	RW	TSM bitfield to control when TSM RF_STATUS de-asserts in the TX sequence
TSM_RF_STATUS_OVRD_EN	RW	TSM bitfield to provide override enable for TSM RF_STATUS output
TSM_RF_STATUS_OVRD	RW	TSM bitfield to provide override value for TSM RF_STATUS output
COEX_SEL	RW	RADIO_MISC (COEX_CTRL register) bitfield to select whether the TSM outputs (RF_ACTIVE, RF_STATUS, RF_PRIORITY) or the active Link Layer's outputs are input to the RFMC.

55.4.6.6.1.4 RF_PRIORITY Behavioral Control

The NB radio provides the RF_PRIORITY[1:0] outputs to indicate the radio access priority. This signal can be generated by either the TSM or the active link layers. Note that if the TSM is chosen, then RF_PRIORITY[1] is always 0.

The table below describes register bits in the TSM and RADIO_MISC to enable and control RF_PRIORITY. Refer to the link layer chapters for information on their PRIORITY output.

Field	Type	Description
RF_PRIORITY_RX_HI[7:0]	RW	TSM bitfield to control when TSM RF_PRIORITY asserts in the RX sequence
RF_PRIORITY_RX_LO[7:0]	RW	TSM bitfield to control when TSM RF_PRIORITY de-asserts in the RX sequence
RF_PRIORITY_TX_HI[7:0]	RW	TSM bitfield to control when TSM RF_PRIORITY asserts in the TX sequence
RF_PRIORITY_TX_LO[7:0]	RW	TSM bitfield to control when TSM RF_PRIORITY de-asserts in the TX sequence
TSM_RF_PRIORITY_OVRD_EN	RW	TSM bitfield to provide override enable for TSM RF_PRIORITY output
TSM_RF_PRIORITY_OVRD	RW	TSM bitfield to provide override value for TSM RF_PRIORITY output
COEX_SEL	RW	RADIO_MISC (COEX_CTRL register) bitfield to select whether the TSM outputs (RF_ACTIVE, RF_STATUS, RF_PRIORITY) or the active Link Layer's outputs are input to the RFMC.

55.4.6.6.2 Coexistence Signaling Options

The figure below illustrates a number of Bluetooth & WiFi coexistence signaling (adapted from WiFi alliance documentation). The interface definition between the two radio has evolved over time targeting improved spectral and antenna control efficiency. Most modern BT & WiFi interfaces utilize 3 or more signals.

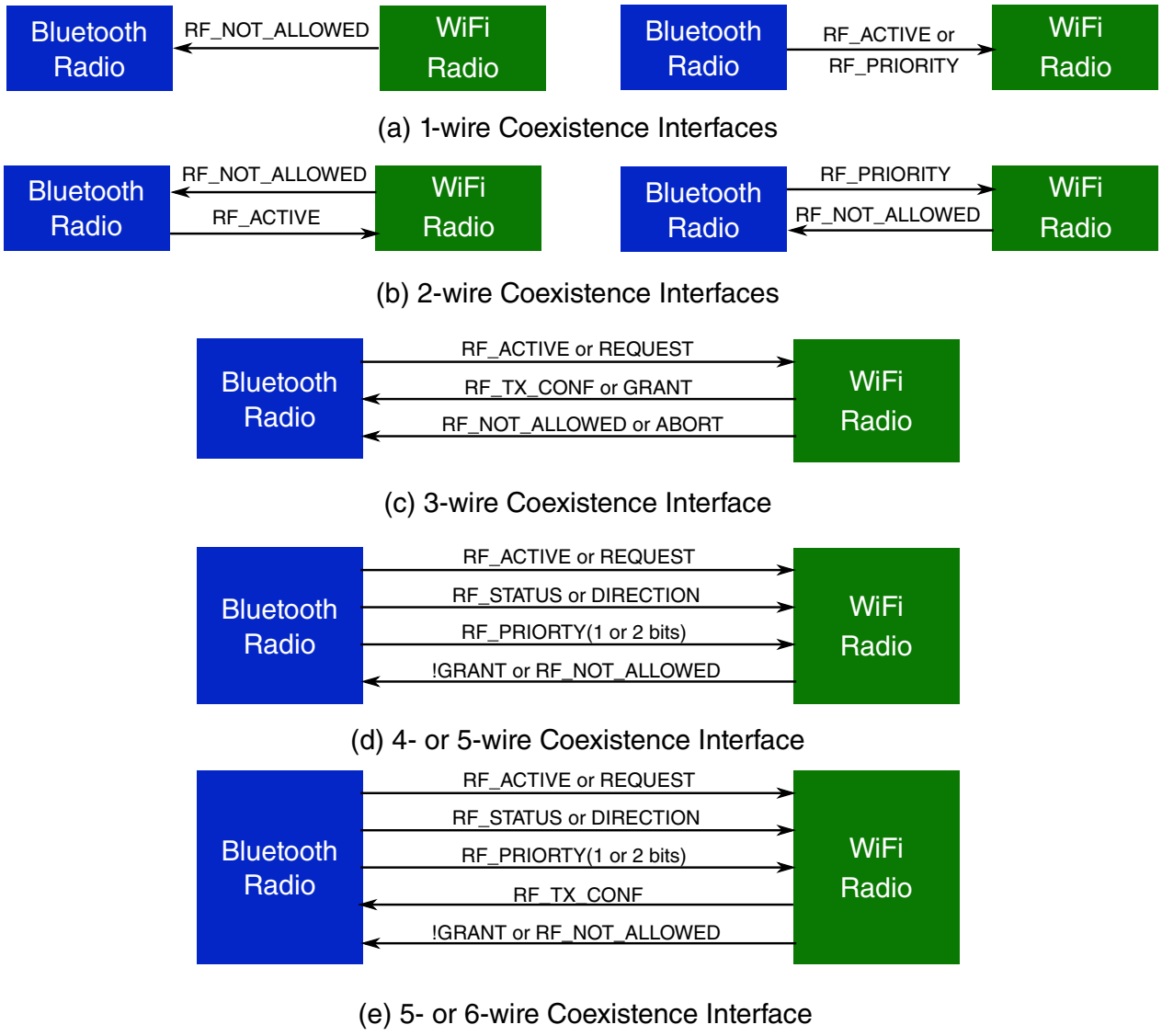


Figure 276. Coexistence signaling options between Bluetooth-LE and WiFi Radios

55.4.6.6.3 Example Coexistence Pin Mapping

The table below shows example mapping of legacy coexistence signals supported by the NB 2.4GHz radio for a few WiFi radios available in the market.

WiFi Co-existence Use Case	RF_ACTIVE	RF_STATUS	RF Priority	RF_NOT_ALLOWED	RF_TX_CONF	WiFi SoC
Direction	Output	Output	Output	Input	Input	
Functionality	Indicates a Radio Event; Needs to stay	Indicates RX/TX event. Optionally	Specifies RF_ACTIVE	Indicates an external command for	Indicates grant of antenna to	

Table continues on the next page...

Table continued from the previous page...

	asserted during a radio event; Deasserted if RF_NOT_ALLOW ED==1	RF_Priority signal can be mixed on it	priority to an external traffic arbiter	the NB radio to cease radio activity	TX; Need to sample if RF_priority requested; can be any GPIO	
1-Wire Interface, WLAN is Master				X		
1-Wire Interface, BT is Master	X					
2-Wire Interface, Cross Signaling between BT and WLAN	X			X		
2-Wire Interface, BT Priority and WLAN TX Signaling			X		X	
3-Wire: Third Party Interface	X			X	X	Broadcom, CSR
3-wire: Request, Grant and Status	X	X			X	BRCM43XX; Nokia 3-wire option
3-wire: Qualcomm	X		X	O	X	QCOM Dakota chipset
4-wire: Request, Grant, Status and Freq (Nokia 4-wire option)	X	X	X		X	BCM89xx; Nokia 4-wire option
4-wire: CSR Interface	X	X	X	O	X	CSR-BC41Bxx
4-wire: Phillips Interface	X	X		X	X	

55.4.6.7 Radio Coexistence/FEM/LANT Connections

The Radio provides flexible mux configuration for 4 coexistence outputs, 4 FEM control outputs and 4 Localization outpus. The detailed connections are shown in below diagram.

NOTE

For the Bluetooth LE use case using the NBU: once asserted the bt_grant_n input should remain asserted until 10us after the bt_request has been de-asserted.

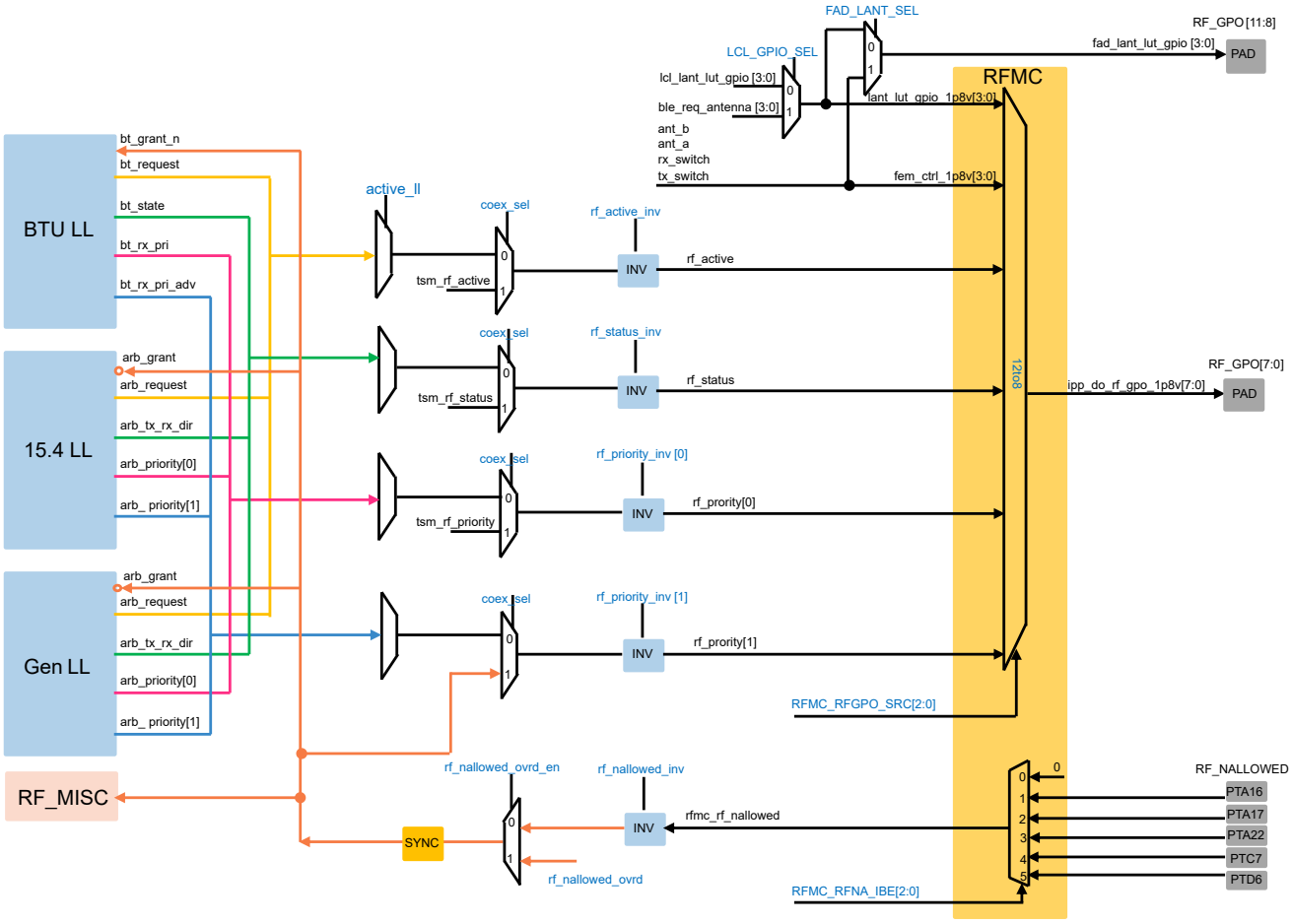


Figure 277. Radio Coexistence/FEM/LANT Mux diagram

55.4.7 Transceiver

55.4.7.1 Introduction

The transceiver uses a constant envelope direct launch transmitter and a low IF quadrature receiver. A high level figure illustrating the digital and analog components and their hierarchy is shown below.

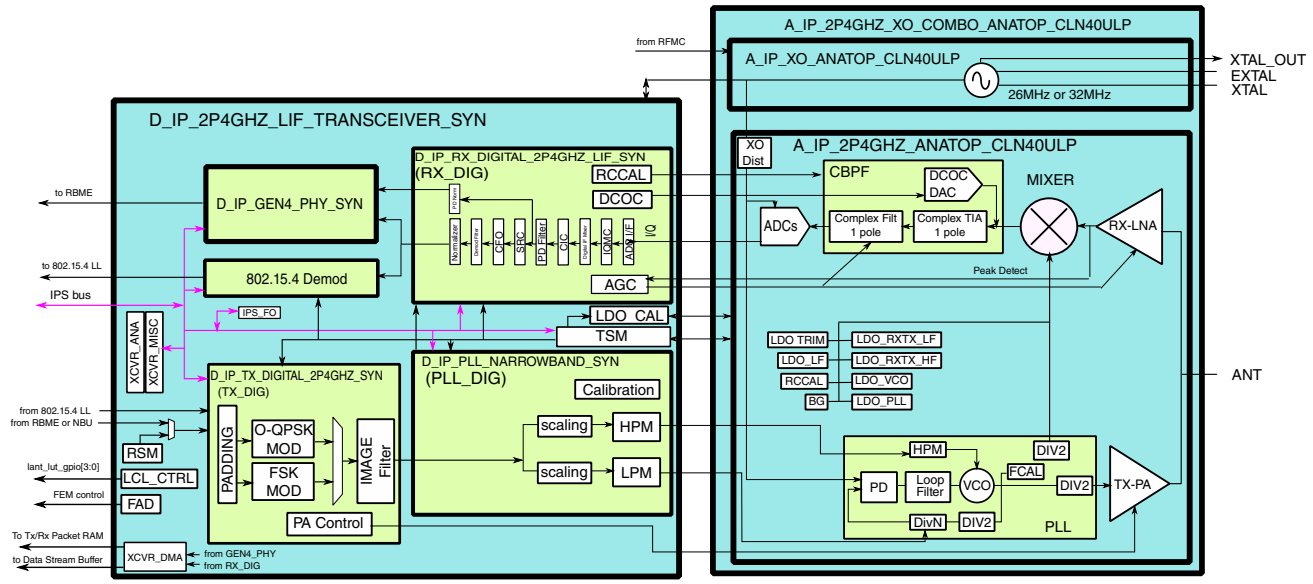


Figure 278. Transceiver Hierarchy

55.4.7.2 Transmitter Digital Module

55.4.7.2.1 About the Transmitter Digital

The Transmitter Digital processes the Transmission Data from the RF Protocol Link Layers and presents the processed data to the PLL Frequency Synthesizer as the Baseband Frequency Word.

PFC_en (Phase to Frequency Conversion) is set when zigbee symbols are to be transmitted.

55.4.7.2.1.1 Block diagram

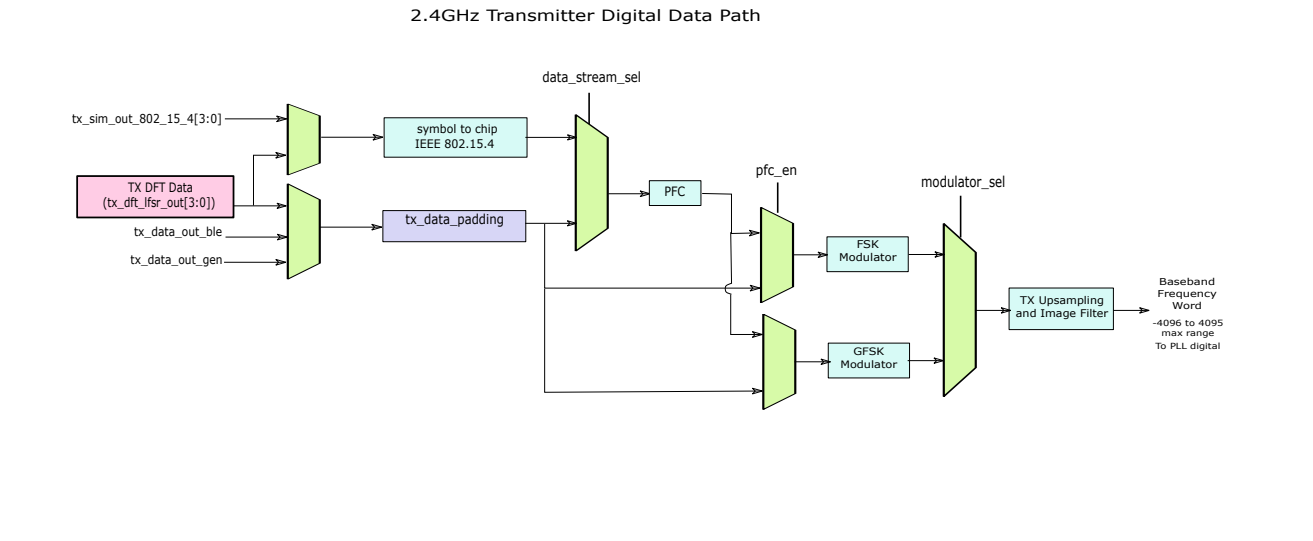


Figure 279. Transmitter Digital

modulator_sel	pfc_en	data_stream_sel	modulation type
0	0	-	GFSK
0	1	0	Invalid
1	0	-	FSK
1	1	0	MSK

55.4.7.2.1.2 Modulation Types

Modulation Type [1]	Data/Chip Rate (kbps) [2]	BT Product [3]	Modulation Index	Symbol Time (us)	Max Frequency Deviation (kHz)
GMSK/ GFSK,BT=0.5, h=0.5	2000	0.5	0.5	0.5	500
	1000	0.5	0.5	1	250
	500	0.5	0.5	2	125
	250	0.5	0.5	4	62.5
GFSK, BT=0.5, h=0.32	2000	0.5	0.32	0.5	320
	1000	0.5	0.32	1	160
	500	0.5	0.32	2	80
	250	0.5	0.32	4	40
GFSK, BT=0.5, h=0.7	2000	0.5	0.7	0.5	700
	1000	0.5	0.7	1	350
	500	0.5	0.7	2	175
	250	0.5	0.7	4	87.5
GFSK, BT=0.5, h=1.0	2000	0.5	1	0.5	1000
	1000	0.5	1	1	500
	500	0.5	1	2	250
	250	0.5	1	4	125
GMSK, BT=0.3	2000	0.3	0.5	0.5	300
	1000	0.3	0.5	1	150
	500	0.3	0.5	2	75
	250	0.3	0.5	4	37.5
GMSK, BT=0.7	2000	0.7	0.5	0.5	700
	1000	0.7	0.5	1	350
	500	0.7	0.5	2	175
	250	0.7	0.5	4	87.5
Generic MSK	2000	NA	0.5	0.5	500

Table continues on the next page...

Table continued from the previous page...

	1000	NA	0.5	1	250
	500	NA	0.5	2	125
	250	NA	0.5	4	62.5
Generic FSK	2000	NA	0.5	0.5	500
	1000	NA	0.5	1	250
	500	NA	0.5	2	125
	250	NA	0.5	4	62.5
OQPSK/Zigbee	2000	NA	0.5	0.5	500
	1000	NA	0.5	1	250
	500	NA	0.5	2	125
	250	NA	0.5	4	62.5

1. Modulation Type is set using TXDIG:MODULATION_TYPE.
2. Data Rate is set using XCVR_CTRL:DATA_RATE[2:0].
3. Bandwidth Time Product is changed by reprogramming the GFSK Gaussian Filter coefficients.

55.4.7.2.1.3 Data Rates

For other Radio protocols, the Transit Data Rate is set by the DATA_RATE bits in the XCVR_CTRL register.

The supported data rates are shown in the table below.

Table 418. Transmitter Data Rates

XCVR_CTRL: DATA_RATE[2:0] Identifies the data rate at the output of the PHY	TX Data Rate
000	2Mbps
001	1Mbps
010	500Kbps
011	250Kbps
100-111	Reserved

55.4.7.2.1.4 Over-Sampling

The TX Digital uses an Oversample Clock to process the transmission data.

The GFSK, GMSK, and MSK Oversample Ratios are summarized below.

Table 419. Oversample Ratios for GFSK, GMSK, and MSK Modulations

TX Data Rate (kb/s)	Ref Clk Freq (MHz)	OSR Ref Clk Divider	OSR Clk Rate (MHz)	Data Oversample Rate
2000	32	2	16	8
1000	32	4	8	8

Table continues on the next page...

Table 419. Oversample Ratios for GFSK, GMSK, and MSK Modulations (continued)

500	32	8	4	8
250	32	16	2	8
2000	26	1	26	13
1000	26	2	13	13
500	26	4	6.5	13
250	26	8	3.25	13

55.4.7.2.1.5 Frequency Word Adjust

The Transmitter has a frequency adjustment register, FREQ_WORD_ADJ, which is used as a signed 9-bit number that is added to the TX Digital output before it is presented to the PLL as the baseband frequency word. This allows the baseband frequency word to be adjusted, or skewed, by a range of -512 to +511.

This frequency adjustment is applied to the final modulation word from any source (GFSK, FSK, DFT) and there is no protection from a math overflow. So the baseband frequency word range of -4096 to 4095 must be considered when adding this frequency word adjustment.

55.4.7.2.2 Data Pre-Processing

Before the Transmission Data is processed by the GFSK or FSK Modulators, it can be inverted, and it can also have additional preamble-like data pre-pended.

55.4.7.2.2.1 Data Polarity

The Transmission Data that is received by the TX Digital module is mapped to a positive frequency deviation if the data is a 1, and is mapped to a negative frequency deviation if the data is a 0

This mapping can be reversed by setting the TX_CAPTURE_POL register bit.

55.4.7.2.2.2 Data Padding

Transmission Data Padding is done to avoid abrupt steps in On-Air frequency modulation and thereby minimize spectral transients during the transition from a low Power Amplifier setting to a high Power Amplifier setting prior to protocol data packet transmission.

The abrupt step in modulation is avoided by pre-pending frequency words onto the front end of the actual preamble transmission. The Power Amplifier ramp-up is done while modulating these additional frequency words, and then the modulating power ramp-up smoothly transitions into the packet transmission at the Target Power level.

The data-padding starts right after TxDig_en and is applied to the Tx modulator before and during PA ramp-up as well as PA settling time(PAD_DLY plus PA_PUP_ADJ).

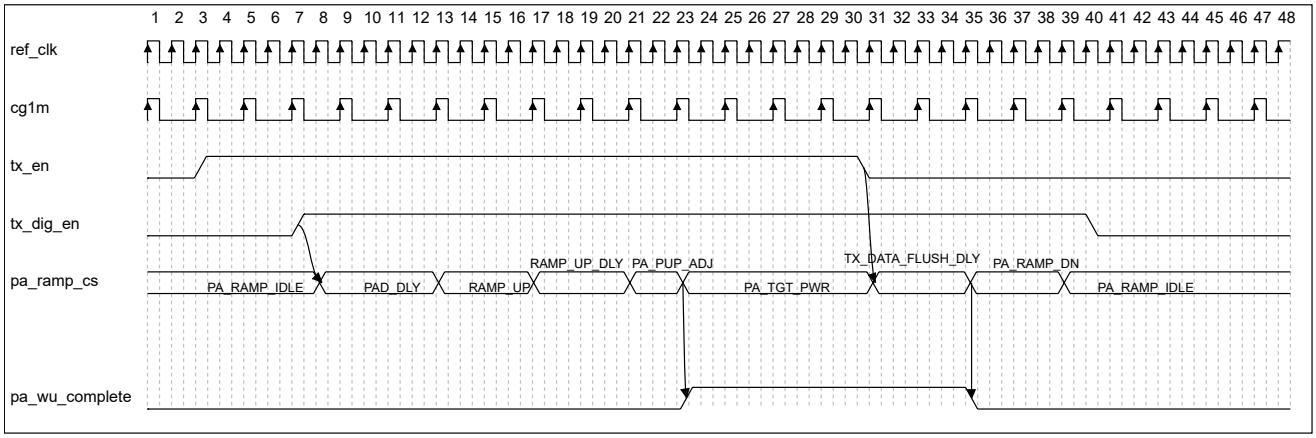


Figure 280. TX padding

There are basically three data padding steps:

- PAD_DLY, if enabled, it adds symbols before rampup.
- RAMP_UP_DLY, smooth transitions to the target power level.
- PA_PUP_ADJ, additional settling time, 2 times ref_clk cycles.
- TX_DATA_FLUSH_DLY, if enabled, it adds symbols before it starts ramping down.

DATA_PADDING_CTRL[PAD_DLY_EN],

- 1'b0, Pad delay is disabled.
- 1'b1:, Pad delay is enabled, and will insert 0-0-0-0... or 1-1-1-1..., based on data_padding_en settings.

DATA_PADDING_CTRL_1[RAMP_UP_DLY]

- It adds the number of cycles after PA ramp-up is completed, up to 31 cg1m cycles.

Table 420. Data Padding Selection

DATA_PADDING_EN[1:0]	Frequency word to PLL
00	13'h000
01	data_pad_pfdev
10	data_pad_mfdev
11	Reserved

55.4.7.2.2.3 Power Amplifier Ramping

PA ramping is included in the 2p4, to prevent abrupt transitions in PA (power amplifier) power, that could cause unwanted spectral transients. During a TX sequence, PA ramping graually ramps up PA power, between a programmable minimum, and the TX target power. The trajectory of the ramp is programmable, and ramping takes place over the course of 1, 2, or 4us (programmable).

During ramp-down of the Power Amplifier the TX Digital holds the TX Modulation Data at the final value until power reaches the minimum level.

Please refer to the section Power Amplifier Ramping in the Transceiver Sequence Manager chapter for more details on the Power Ramping topic.

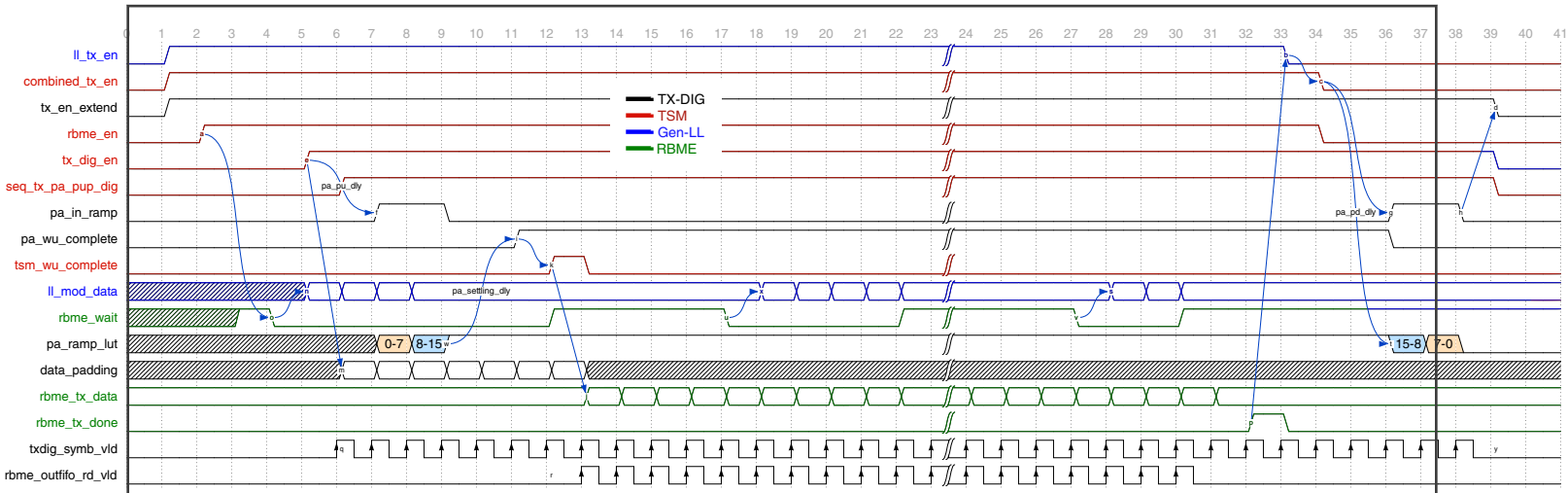


Figure 281. Power Amplifier Ramping

55.4.7.2.3 TX Warmup

When a Link Layer asserts *source*_tx_en (or software sets the FORCE_TX_EN bit) to start a TX burst, the TSM TX warmup sequence will begin. Near the end of the TSM TX warmup sequence, a TSM timing signal (tx_dig_en) will signal to the TXDIG to ramp-up the PA and start sending TX data.

The TXDIG warmup time is the time from tx_dig_en TSM signal assertion to when the TX is ready to send bits on air, which is the assertion of pa_wu_complete by the TXDIG.

The TXDIG warmup time depends on the following:

- pa_ramp_pad
- pa_ramp_up
- pa_ramp_settle

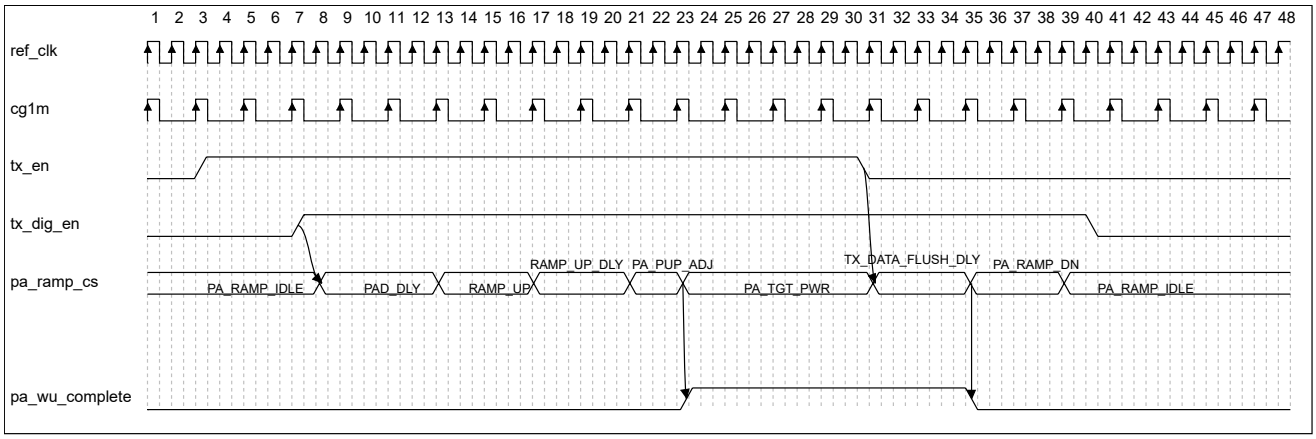


Figure 282. TX warmup delay

So the total TX warmup delay is composed of the following:

- pa_ramp_pad: DATA_PADDING_CTRL[PAD_DLY], number of cg1m cycles of padding before ramp-up.
- pa_ramp_up: PA_CTRL[PA_RAMP_SEL], 0us, 1us, 2us, or 4us.

- pa_ramp_settle: DATA_PADDING_CTRL_1[RAMP_UP_DLY], number of cg1m cycles plus DATA_PADDING_CTRL[PA_PUP_ADJ], 2 times the number of ref_clk cycles.

The total TX warmup delay can be then summarized with the following equation: DATA_PADDING_CTRL[PAD_DLY_EN]*DATA_PADDING_CTRL[PAD_DLY] (cg1m cycles, us).

PA_CTRL[PA_RAMP_SEL] 0x0: 0 us 0x1: 1 us 0x2: 2 us 0x3: 4 us DATA_PADDING_CTRL_1[RAMP_UP_DLY] (cg1m cycles, us). DATA_PADDING_CTRL_1[PA_UP_ADJ]*2 (ref_clk cycles).

55.4.7.2.4 GFSK Modulator

55.4.7.2.4.1 Gaussian Filter

GFSK Modulation is implemented using a Gaussian filter and a lookup table to implement the various reference frequencies, data rates, and modulation index values.

The Gaussian filter is a symmetric 16-tap FIR filter with 9-bit quantized coefficients and supports reference frequencies of 32 and 26 MHz at oversampling ratios of 8 and 13 respectively. There are 2 sets of pre-programmed filter coefficients available as shown in the table below.

Table 421. Gaussian Filter Coefficients BT=0.5

Tap Number(s)	32 MHz Ref	26 MHz Ref
0	9'h001	9'h017
1	9'h004	9'h029
2	9'h00d	9'h044
3	9'h029	9'h067
4	9'h063	9'h090
5	9'h0c0	9'h0ba
6	9'h12c	9'h0dc
7	9'h176	9'h0ef
8	9'h176	9'h0ef
9	9'h12c	9'h0dc
10	9'h0c0	9'h0ba
11	9'h063	9'h090
12	9'h029	9'h067
13	9'h00d	9'h044
14	9'h004	9'h029
15	9'h001	9'h017

The pre-programmed filter coefficients can be manually overridden using the gfsk_coeff_man bit to disable the GFSK Filter Lookup Table. In this case the filter coefficients will be taken from the TX GFSK Filter Coefficients register.

The following tables show the coefficients for BT=0.7 and BT=0.3

Table 422. Gaussian Filter Coefficients BT=0.7

Table continues on the next page...

Table 422. Gaussian Filter Coefficients BT=0.7 (continued)

Tap Number(s)	32 MHz Ref	26 MHz Ref
0	9'h000 (0)	9'h003 (3)
1	9'h000 (0)	9'h00A (10)
2	9'h001 (1)	9'h01B (27)
3	9'h007 (7)	9'h03E (62)
4	9'h025 (37)	9'h079 (121)
5	9'h08A (138)	9'h0C6 (198)
6	9'h14A (330)	9'h114 (276)
7	9'h1FE (510)	9'h145 (325)
8	9'h1FE (510)	9'h145 (325)
9	9'h14A (330)	9'h114 (276)
10	9'h08A (138)	9'h0C6 (198)
11	9'h025 (37)	9'h079 (121)
12	9'h007 (7)	9'h03E (62)
13	9'h001 (1)	9'h01B (27)
14	9'h000 (0)	9'h00A (10)
15	9'h000 (0)	9'h003 (3)

Table 423. Gaussian Filter Coefficients BT=0.3

Tap Number(s)	32 MHz Ref	26 MHz Ref
0	9'h019 (25)	9'h048 (72)
1	9'h02C (44)	9'h059 (89)
2	9'h046 (70)	9'h06B (107)
3	9'h069 (105)	9'h07D (125)
4	9'h091 (145)	9'h08D (141)
5	9'h0B8 (184)	9'h09A (154)
6	9'h0D8 (216)	9'h0A4 (164)
7	9'h0EA (234)	9'h0A9 (169)
8	9'h0EA (234)	9'h0A9 (169)
9	9'h0D8 (216)	9'h0A4 (164)
10	9'h0B8 (184)	9'h09A (154)
11	9'h091 (145)	9'h08D (141)
12	9'h069 (105)	9'h07D (125)

Table continues on the next page...

Table 423. Gaussian Filter Coefficients BT=0.3 (continued)

13	9'h046 (70)	9'h06B (107)
14	9'h02C (44)	9'h059 (89)
15	9'h019 (25)	9'h048 (72)

55.4.7.2.4.2 On-Air Frequency Deviation

By default, the GFSK Modulator presents the modulation word to the PLL starting with the Negative Frequency Deviation, this results in the PLL going On-Air at the Carrier Frequency minus the frequency deviation.

The GFSK Modulator can instead be configured to begin modulating at Zero Frequency Deviation, allowing the PLL to go On-Air at the Carrier Frequency. This configuration is used if the ZERO_FDEV bit in the TX Digital Control register is set.

55.4.7.2.5 FSK Modulator

55.4.7.2.5.1 FSK Modulation Word

The default FSK Modulation mapping translates the chips/bits to a frequency modulation of +Frequency Deviation/PLL Minimum Modulation Step for a data 1, and -Frequency Deviation/PLL Minimum Modulation Step for a data 0.

For a 32 MHz reference clock with a PLL Minimum Modulation Step Size of 244.14 Hz, the default FSK Modulation mapping translates to a +/- 500 kHz peak Frequency Deviation, which is a PLL Baseband Frequency Word of +2047/-2047.

55.4.7.2.5.2 802.15.4 Modes, DFT only.

In Gen4p5, the 802.15.4 mode is only implemented for DFT. No Symbol to chip is implemented, only Phase to Frequency Conversion takes place.

To enable 802.15.4 DFT mode the following settings are needed:

- data_stream_sel = 1'b0.
- radio_dft_mode = 4'b0010: for DFT pattern register.

  or radio_dft_mode = 4'b0011: for DFT LFSR register.
- pfc_en = 1'b1.
- modulator_sel = 1'b1.

55.4.7.2.5.3 FSK Modulator

The FSK Modulation is implemented by mapping directly to a frequency modulation.

For most cases, mFdev is equal to minus Fdev.

Fdev = mod_index * Symbol_Rate/2 * TX_Scale.

TX_Scale = LO_DIV/PLL_Scale

PLL_Scale = 2^PLL_Scale_exp

PLL_Scale_exp = floor(log2(LO_Div))

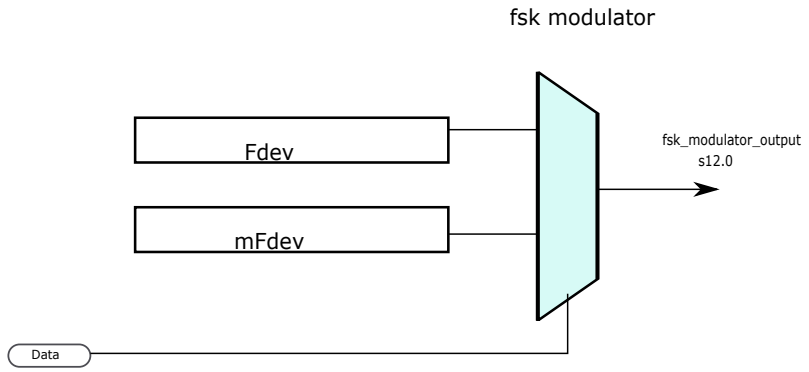


Figure 283. FSK Transmit Modulator

55.4.7.2.6 Data Post-Processing

After the Transmission Data is processed by the GFSK or FSK Modulators, it can be post-processed to remove TX modulation images.

55.4.7.2.6.1 Sample/Hold Images

Power efficient constant envelope transmit modulation is done at a multiple of the symbol clock (Over-Sample Rate), which is chosen as a fraction of the RF Osc reference clock frequency. The transmit modulation signal is then converted to the reference clock rate, at which it is presented to the PLL as the modulation word. This sample/hold (or zero-order hold, ZOH) operation results in TX modulation images that repeat at the fractional reference clock rate at used.

These modulation images are automatically filtered by the sinc ZOH transfer function and the integration (i.e., 1/s) operation of the VCO. These inherent filtering operations result in these modulation images not violating any of the transmit modulation frequency mask. However, to provide additional spectral margin to support using an external power amplifier to boost the transmit power, an image suppression set of FIR filters has been included in the transmit path to suppress the transmit sample/hold images by at least an additional 20 dB.

The transmit image filters are enabled by default in GFSK and GMSK modes and not enabled in IEEE 802.15.4 and MSK modulation modes, but their usage can be overridden using the TX_IMAGE_FILTER_OVRD_EN bit.

55.4.7.2.6.2 Image Filters

The Transmitter uses up to three image filters, by default they are enabled based upon the modulation type and data rate as shown below. There is an option to override the individual filters by programming: sync0_filter_en, sync1_filter_en, and image_fir_filter_en.

Table 424. Default Transmit Image Filter Usage

Ref Clk Freq (MHz)	ModulationType; Data Rate	TX Image Sync0	TX Image Sync1	TX Image FIR
32	GFSK/GMSK; 2000 kbps	0	0	1
32	GFSK/GMSK; 1000 kbps	1	1	1

Table continues on the next page...

Table 424. Default Transmit Image Filter Usage (continued)

32	GFSK/GMSK; 500 kbps	1	1	1
32	GFSK/GMSK; 250 kbps	1	1	1
26	GFSK/GMSK; 2000 kbps	0	0	1
26	GFSK/GMSK; 1000 kbps	0	1	1
26	GFSK/GMSK; 500 kbps	1	1	1
26	GFSK/GMSK; 250 kbps	1	1	1
32/26	FSK/MSK Modes; All data rates	0	0	0

55.4.7.2.6.3 Image Filters Frequency Response

The frequency response of the TX image suppression filters and the reduction in the level of TX images for the 1 Mbps GMSK (BluetoothLE) with the FIR filter are demonstrated in the plots below.

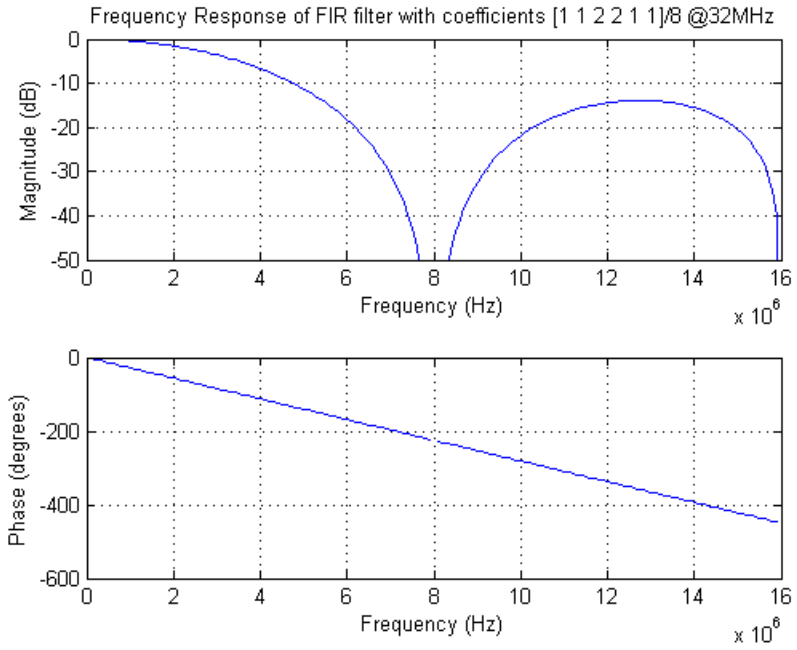


Figure 284. Frequency response of the 6-Tap FIR Transmit Image Suppression filter

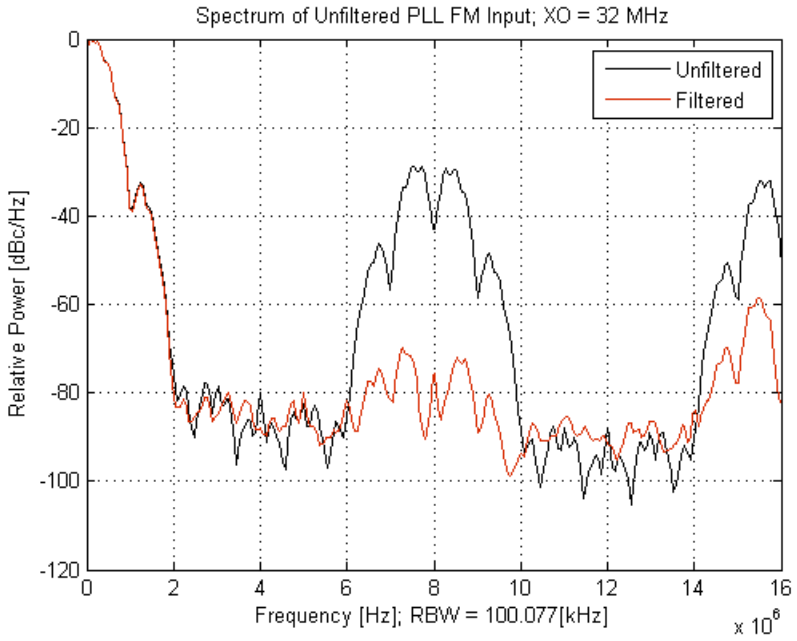


Figure 285. Comparison of Transmit Modulation Images at the PLL Input for BluetoothLE

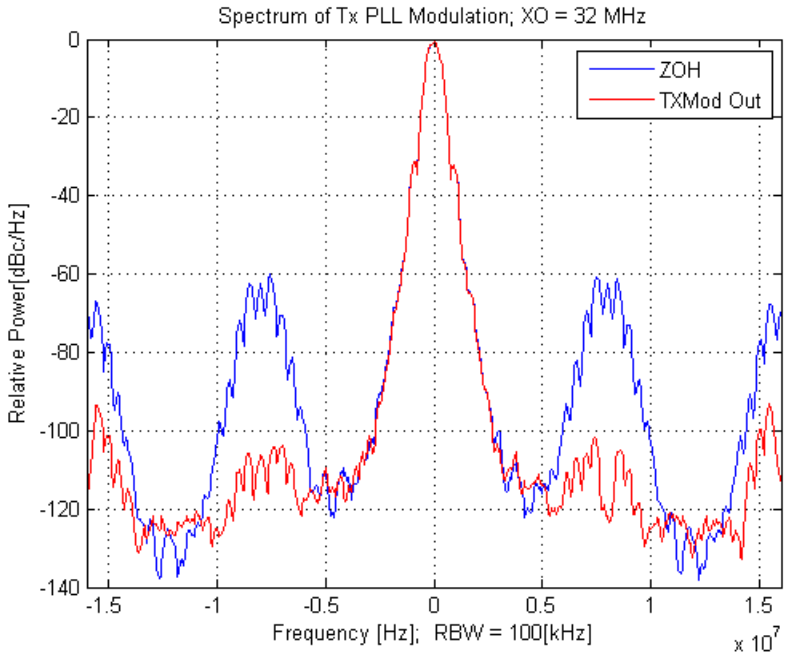


Figure 286. Comparison of Transmit Modulation Images at the PLL Output for BluetoothLE (1 Mbps GMSK)

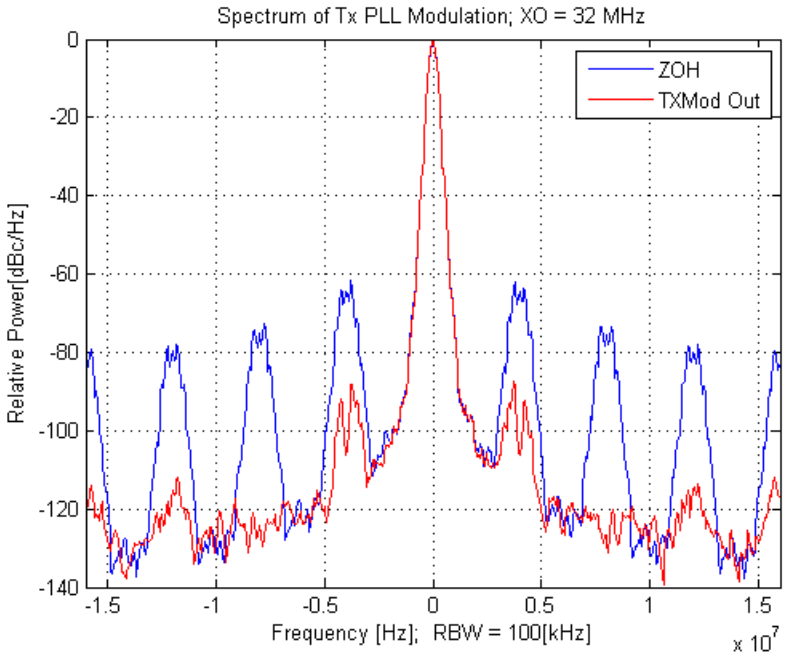


Figure 287. Comparison of Transmit Modulation Images at the PLL Output for 500 kbps GMSK

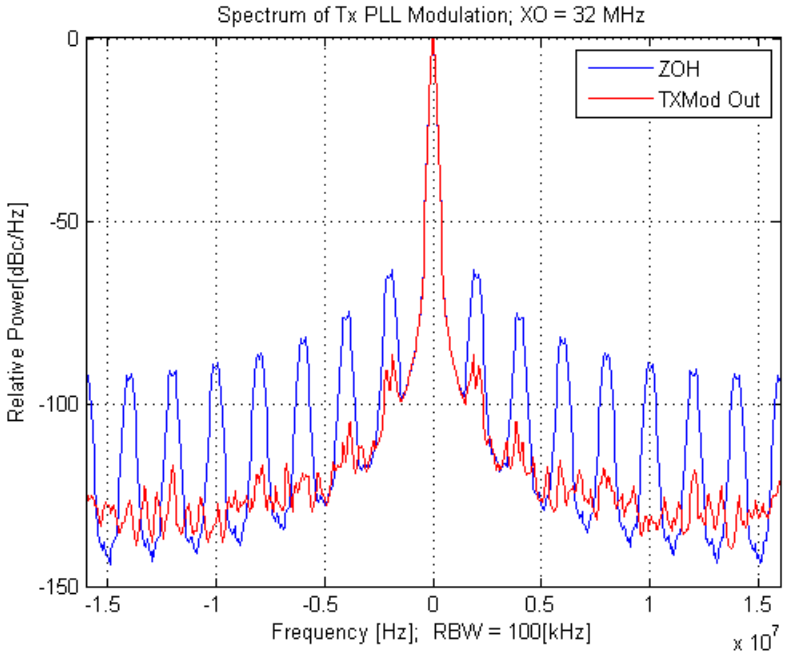


Figure 288. Comparison of Transmit Modulation Images at the PLL Output for 250 kbps GMSK

55.4.7.2.7 CTE (Continuous Tone Extension)

In BluetoothLE 5.1, Constant Tone Extension (CTE) consists of a constantly modulated series of unwhitened 1 symbols (however TX_DIG 2p4 has the option to transmitt 0 symbols as well). The Constant Tone Extension is not included in CRC or MIC calculations and it follows the CRC transmission in the packet. The Constant Tone Extension has a variable length with a minimum duration of 16us.

For direction finding applications, CTE is transmitted during various phases of the programmed antenna switching pattern that compromises of the guard period, the reference period and the switching/sampling slots.

- The Localization Control Module (lcl_ctrl) asserts tx_cte_on (based on cte_dur).
- When tx_en deasserts, TX_DIG looks at the state of tx_cte_on.

  If deasserted, the pa_ramp_cs behaves normally, it will go through the data flush state if a value other than zero is programmed, then it will ramp down.

  If tx_cte_on is asserted, then the pa_ramp_cs state machine keeps the state at target power, then it transmitts 0 if CTE_DATA = 1'b0 or 1 if CTE_DATA = 1'b1 (DATA_PADDING_CTRL[4]).

  When tx_cte_on is deassserted, then the state machine behaves as normal: it will go through data flush if a value other than zero is programmed and then it will enter ramp down.

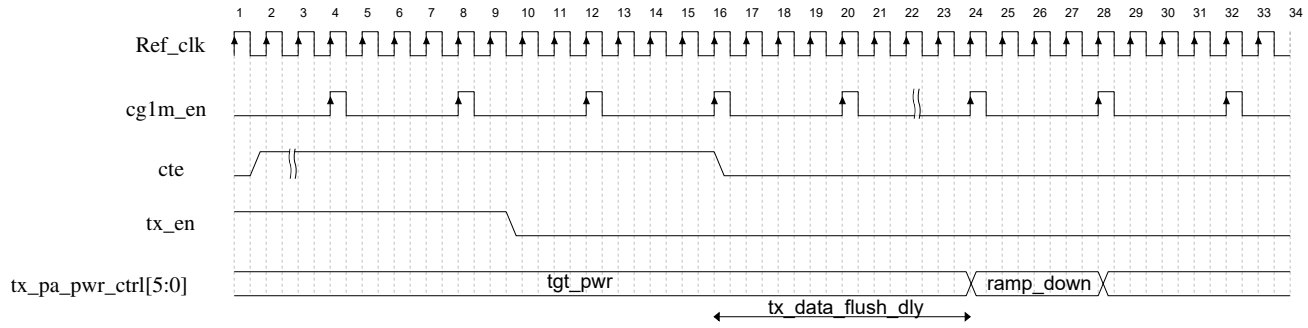


Figure 289. CTE

55.4.7.2.8 Memory Map and Register Definition

The Transmitter Digital memory map and detailed descriptions of all its registers are as follows.

55.4.7.2.8.1 XCVR_TX_DIG register descriptions

55.4.7.2.8.1.1 XCVR_TX_DIG_REGS memory map

XCVR_TX_DIG base address: 48A0_7200h

Offset	Register	Width (In bits)	Access	Reset value
0h	TXDIG_CTRL (TXDIG_CTRL)	32	RW	0000_0000h
4h	DATA_PADDING_CTRL (DATA_PADDING_CTRL)	32	RW	0000_0010h
8h	DATA_PADDING_CTRL_1 (DATA_PADDING_CTRL_1)	32	RW	0000_1000h
Ch	DATA_PADDING_CTRL_2 (DATA_PADDING_CTRL_2)	32	RW	0000_0000h
10h	FSK_CTRL (FSK_CTRL)	32	RW	0800_1800h
14h	GFSK_CTRL (GFSK_CTRL)	32	RW	0000_0000h
18h	GFSK_COEFF_0_1 (GFSK_COEFF_0_1)	32	RW	0000_0000h
1Ch	GFSK_COEFF_2_3 (GFSK_COEFF_2_3)	32	RW	0000_0000h
20h	GFSK_COEFF_4_5 (GFSK_COEFF_4_5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

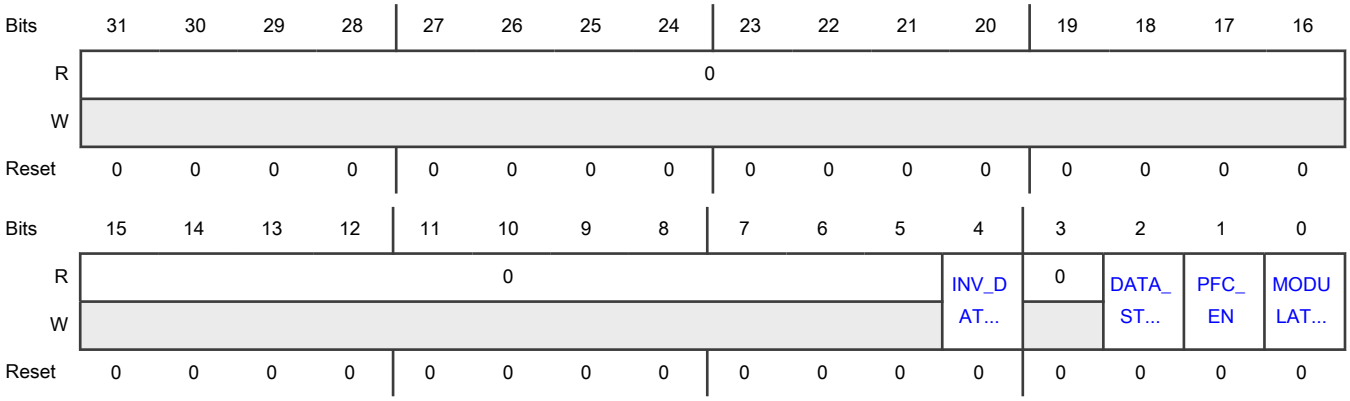
Offset	Register	Width (In bits)	Access	Reset value
24h	GFSK_COEFF_6_7 (GFSK_COEFF_6_7)	32	RW	0000_0000h
28h	IMAGE_FILTER_CTRL (IMAGE_FILTER_CTRL)	32	RW	0000_0000h
2Ch	PA_CTRL (PA_CTRL)	32	RW	0000_0000h
30h	PA_RAMP_TBL0 (PA_RAMP_TBL0)	32	RW	0604_0201h
34h	PA_RAMP_TBL1 (PA_RAMP_TBL1)	32	RW	1410_0C09h
38h	PA_RAMP_TBL2 (PA_RAMP_TBL2)	32	RW	2621_1C18h
3Ch	PA_RAMP_TBL3 (PA_RAMP_TBL3)	32	RW	3C38_322Ch
40h	SWITCH_TX_CTRL (SWITCH_TX_CTRL)	32	RW	0000_0000h
44h	RF_DFT_TX_CTRL0 (RF_DFT_TX_CTRL0)	32	RW	0000_0000h
48h	RF_DFT_TX_CTRL1 (RF_DFT_TX_CTRL1)	32	RW	2101_FFFFh
4Ch	RF_DFT_TX_CTRL2 (RF_DFT_TX_CTRL2)	32	RW	0000_0000h
50h	RF_DFT_PATTERN (RF_DFT_PATTERN)	32	RW	0000_0000h
54h	DATARATE_CONFIG_FSK_CTRL (DATARATE_CONFIG_FSK_CTRL)	32	RW	0000_0000h
58h	DATARATE_CONFIG_GFSK_CTRL (DATARATE_CONFIG_GFSK_CTRL)	32	RW	0000_0000h
5Ch	DATARATE_CONFIG_FILTER_CTRL (DATARATE_CONFIG_FILTER_CTRL)	32	RW	0000_0000h

55.4.7.2.8.1.2 TXDIG_CTRL (TXDIG_CTRL)

Offset

Register	Offset
TXDIG_CTRL	0h

Diagram



Fields

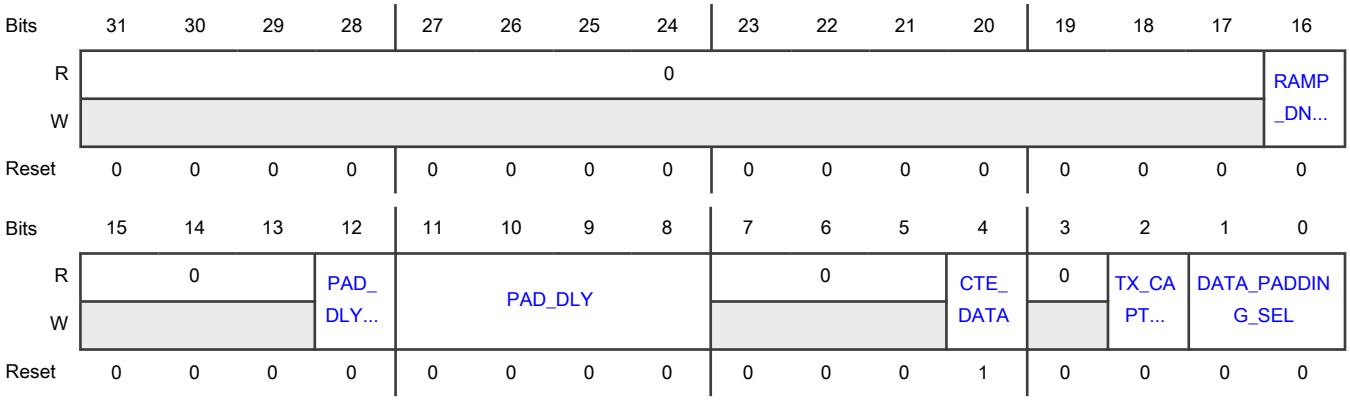
Field	Function
31-5 —	Reserved
4 INV_DATA_OUT	INV_DATA_OUT Inverts the data out.
3 —	Reserved
2 DATA_STREAM_SEL	DATA_STREAM_SEL Selects between symbol_to_chip and tx_data_padding. 1'b0: tx_data_padding 1'b1: symbol_to_chip
1 PFC_EN	PFC_EN Selects between the output of data_stream and tx_data_padding. PFC_en (Phase to Frequency Conversion) is set when zigbee symbols are to be transmitted. The frequency modulation (FM) information is obtained by differentiating the phase using a delay of 1 symbol (for Zigbee is a 1-chip delay). 1'b0: tx_data_padding 1'b1: data_stream
0 MODULATOR_SEL	MODULATOR_SEL Selects between GFSK and FSK modulator. 1'b0: GFSK 1'b1: FSK

55.4.7.2.8.1.3 DATA_PADDING_CTRL (DATA_PADDING_CTRL)

Offset

Register	Offset
DATA_PADDING_CTRL	4h

Diagram



Fields

Field	Function
31-17 —	Reserved
16 RAMP_DN_PAD_EN	RAMP_DN_PAD_EN Enables padding during ramp-down.
15-13 —	Reserved
12 PAD_DLY_EN	PAD_DLY_EN Enable padding before ramp-up.
11-8 PAD_DLY	PAD_DLY Number of Cg1m cycles of padding before ramp-up.
7-5 —	Reserved
4 CTE_DATA	CTE_DATA When CTE (Countinuous Tone Extension) selects the data to be transmitted.

Table continues on the next page...

Table continued from the previous page...

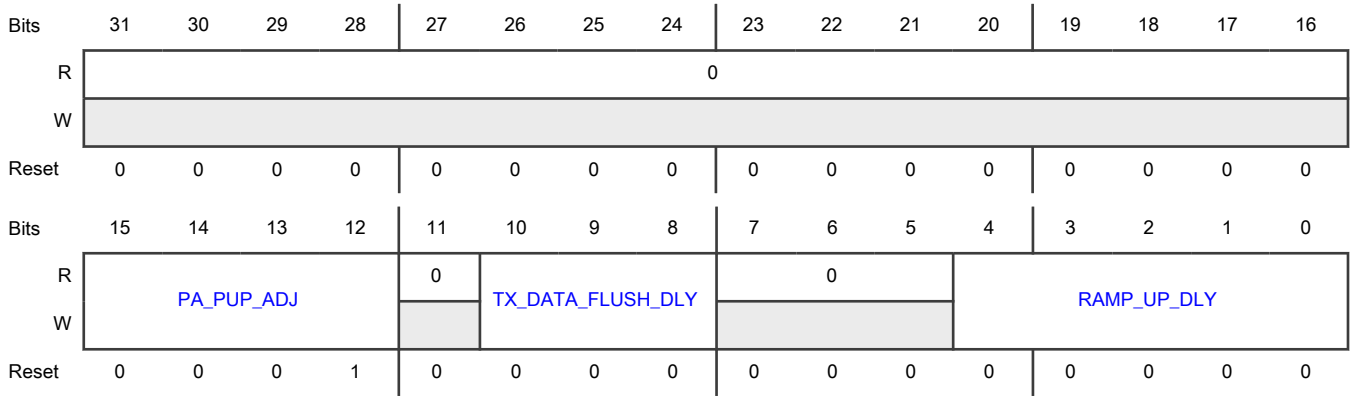
Field	Function
	1'b0: Transmitt 0. 1'b1: Transmitt 1.
3 —	Reserved
2 TX_CAPTURE_POL	TX_CAPTURE_POL Inverts the polarity of the transmitt data. 1'b0: Do not invert transmitt data. 1'b1: Inver transmitt data.
1-0 DATA_PADDING_SEL	DATA_PADDING_SEL Selects data to be used during padding. 2'b00: 0x0000 is sent to the PLL as frequency word (baseband). 2'b01: DATA_PAD_PFDEV is sent to the PLL (1). 2'b10: DATA_PAD_MFDEV is sent to the PLL (0). 2'b11: Reserved.

55.4.7.2.8.1.4 DATA_PADDING_CTRL_1 (DATA_PADDING_CTRL_1)

Offset

Register	Offset
DATA_PADDING_CTRL_1	8h

Diagram



Fields

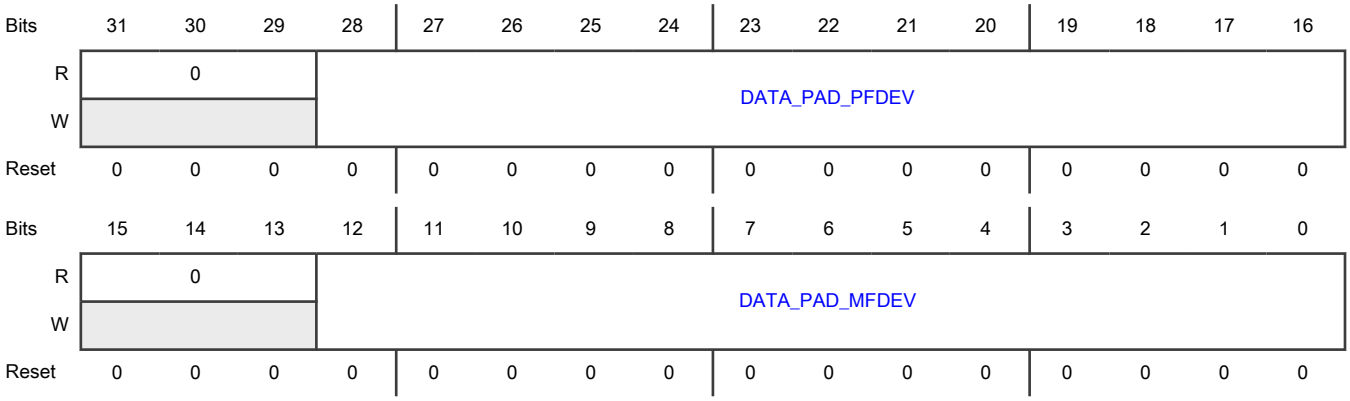
Field	Function
31-16 —	Reserved
15-12 PA_PUP_ADJ	PA_PUP_ADJ PA Power Up Adjust. This field tells the TX_DIG to assert tx_pa_pup after tx_dig_en is set, then waits for 2 times the number of ref clk cycles programmed in this field (2*PA_PUP_ADJ) before the PA_RAMP state machine enters the RAMP_UP state. If PAD_DLY_EN is set and PAD_DLY is different than zero, the TX_DIG pads for the number of cg1m cycles specified by PAD_DLY, then it asserts tx_pa_pup and waits for the number of ref clk cycles programmed (2xPA_PUP_ADJ) before entering RAMP_UP state.
11 —	Reserved
10-8 TX_DATA_FLUSH_DLY	TX_DATA_FLUSH_DLY TX Data Flush Delay This registers tells the number of cycles (Cg1m) to wait after tx_en de-asserts before ramp-down.
7-5 —	Reserved
4-0 RAMP_UP_DLY	RAMP_UP_DLY This registers indicates the number of cycles (cg1m) to wait after ramp-up before asserting pu_complete, if PA_PUP_ADJ is non zero, assertion of pu_complete takes place after 2 times the number of ref_clk cycles programmed in PA_PUP_ADJ after RAMP_UP_DLY. (ready to start receiving data from the link layer).

55.4.7.2.8.1.5 DATA_PADDING_CTRL_2 (DATA_PADDING_CTRL_2)

Offset

Register	Offset
DATA_PADDING_CTRL_2	Ch

Diagram



Fields

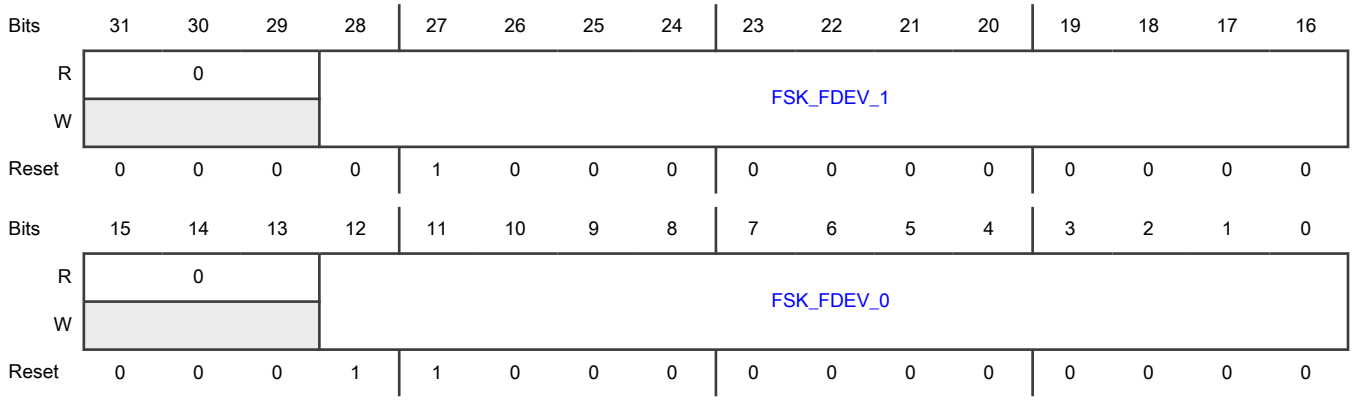
Field	Function
31-29 —	Reserved
28-16 DATA_PAD_PFDEV	DATA_PAD_PFDEV This register indicates the +1 signed frequency word to be used during padding.
15-13 —	Reserved
12-0 DATA_PAD_MFDEV	DATA_PAD_MFDEV This register indicates the -1 signed frequency word to be used during padding.

55.4.7.2.8.1.6 FSK_CTRL (FSK_CTRL)

Offset

Register	Offset
FSK_CTRL	10h

Diagram



Fields

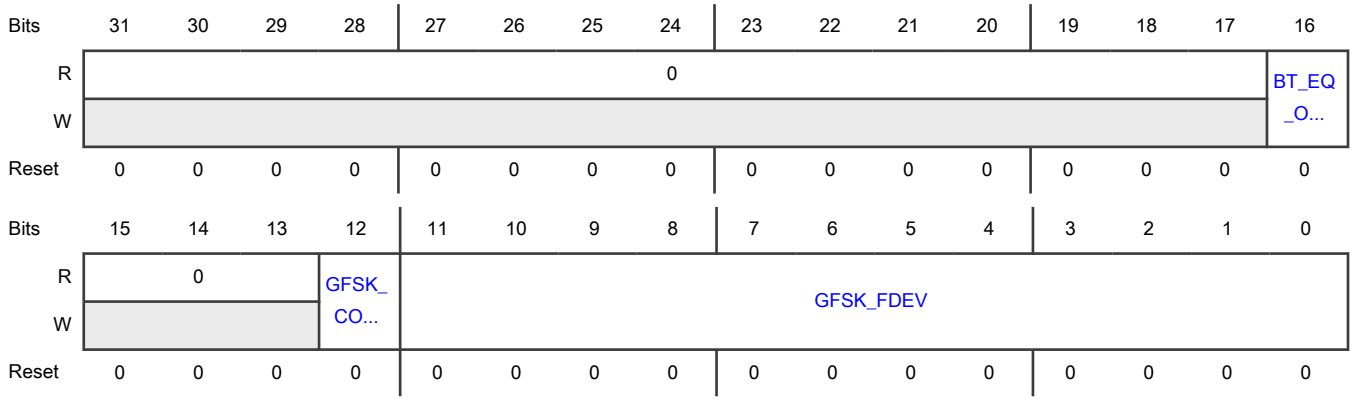
Field	Function
31-29 —	Reserved
28-16 FSK_FDEV_1	FSK_FDEV_1 Signed frequency word for a 1.
15-13 —	Reserved
12-0 FSK_FDEV_0	FSK_FDEV_0 Signed frequency word for a 0.

55.4.7.2.8.1.7 GFSK_CTRL (GFSK_CTRL)

Offset

Register	Offset
GFSK_CTRL	14h

Diagram



Fields

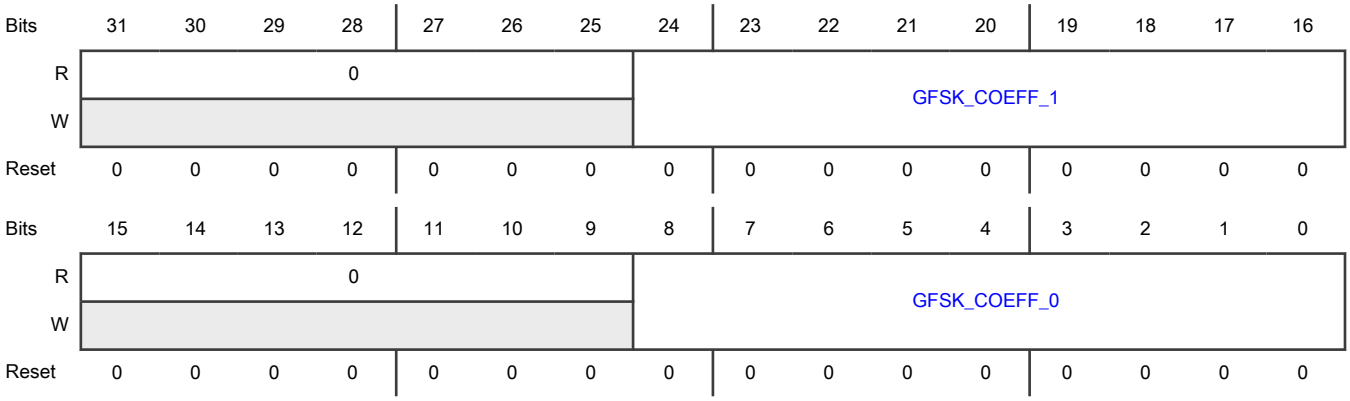
Field	Function
31-17 —	Reserved
16 BT_EQ_OR_GTR_ONE	BT_EQ_OR_GTR_ONE Indicates if the BT product is equal or grater than 1. Used to enable BT>= logic.
15-13 —	Reserved
12 GFSK_COEFF_MAN	GFSK_COEFF_MAN Enables manual selection of GFSK coefficients.
11-0 GFSK_FDEV	GFSK_FDEV GFSK frequency word.

55.4.7.2.8.1.8 GFSK_COEFF_0_1 (GFSK_COEFF_0_1)

Offset

Register	Offset
GFSK_COEFF_0_1	18h

Diagram



Fields

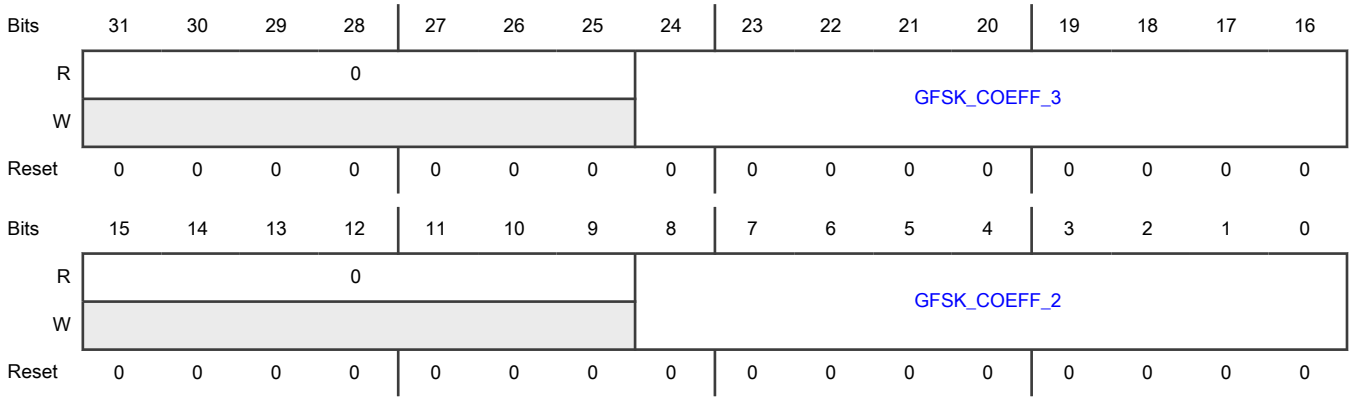
Field	Function
31-25 —	Reserved
24-16 GFSK_COEFF_1	GFSK_COEFF_1 GFSK_COEFF_1[8:0] GFSK filter coefficient 1, format u9.0
15-9 —	Reserved
8-0 GFSK_COEFF_0	GFSK_COEFF_0 GFSK_COEFF_0[8:0] GFSK filter coefficient 0, format u9.0

55.4.7.2.8.1.9 GFSK_COEFF_2_3 (GFSK_COEFF_2_3)

Offset

Register	Offset
GFSK_COEFF_2_3	1Ch

Diagram



Fields

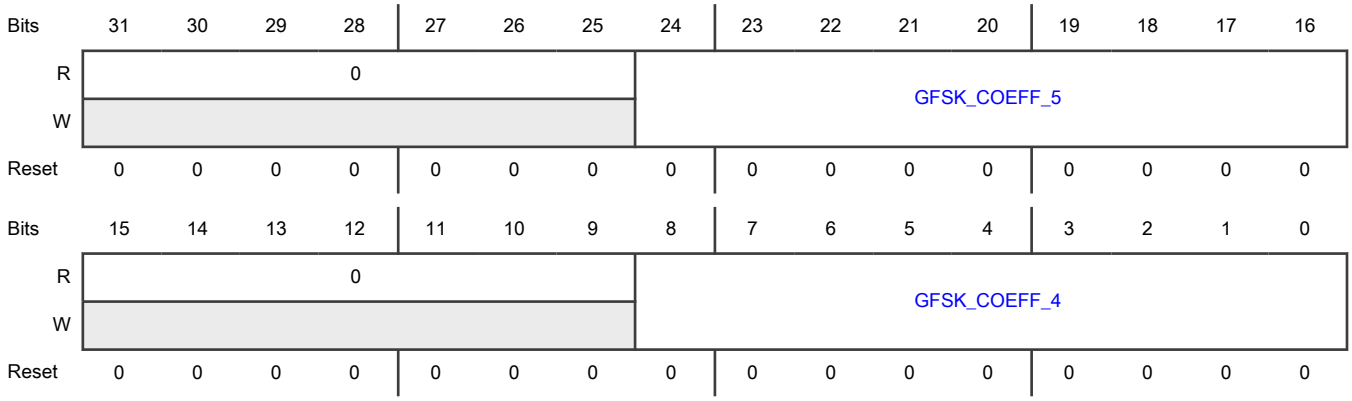
Field	Function
31-25 —	Reserved
24-16 GFSK_COEFF_3	GFSK_COEFF_3 GFSK_COEFF_3[8:0] GFSK filter coefficient 3, format u9.0
15-9 —	Reserved
8-0 GFSK_COEFF_2	GFSK_COEFF_2 GFSK_COEFF_2[8:0] GFSK filter coefficient 2, format u9.0

55.4.7.2.8.1.10 GFSK_COEFF_4_5 (GFSK_COEFF_4_5)

Offset

Register	Offset
GFSK_COEFF_4_5	20h

Diagram



Fields

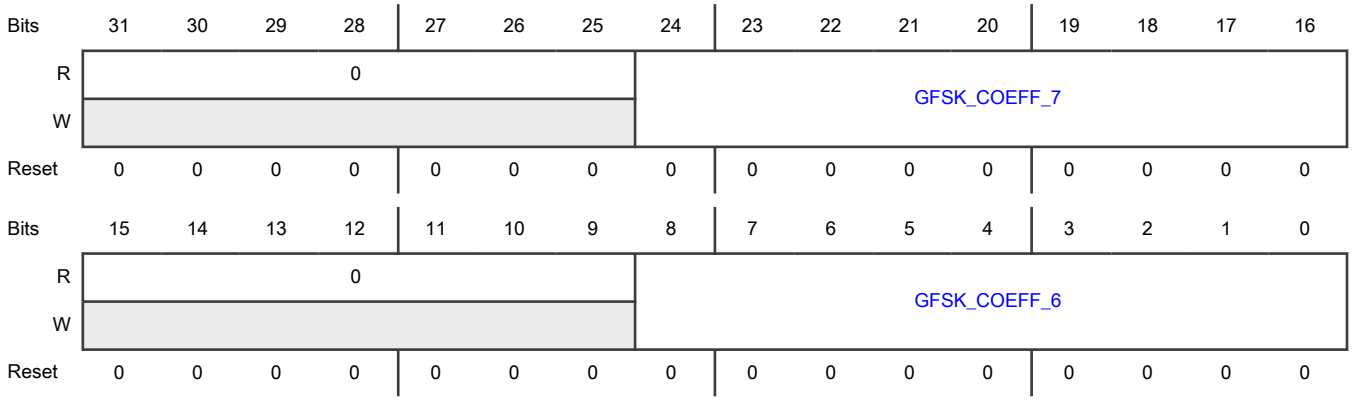
Field	Function
31-25 —	Reserved
24-16 GFSK_COEFF_5	GFSK_COEFF_5 GFSK_COEFF_5[8:0] GFSK filter coefficient 5, format u9.0
15-9 —	Reserved
8-0 GFSK_COEFF_4	GFSK_COEFF_4 GFSK_COEFF_4[8:0] GFSK filter coefficient 4, format u9.0

55.4.7.2.8.1.11 GFSK_COEFF_6_7 (GFSK_COEFF_6_7)

Offset

Register	Offset
GFSK_COEFF_6_7	24h

Diagram



Fields

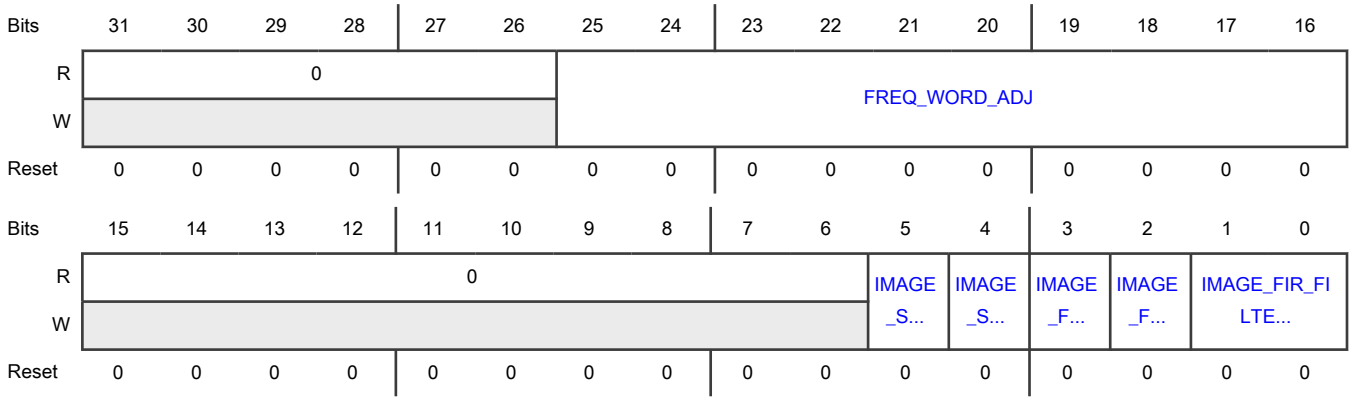
Field	Function
31-25 —	Reserved
24-16 GFSK_COEFF_7	GFSK_COEFF_7 GFSK_COEFF_7[8:0] GFSK filter coefficient 7, format u9.0
15-9 —	Reserved
8-0 GFSK_COEFF_6	GFSK_COEFF_6 GFSK_COEFF_6[8:0] GFSK filter coefficient 6, format u9.0

55.4.7.2.8.1.12 IMAGE_FILTER_CTRL (IMAGE_FILTER_CTRL)

Offset

Register	Offset
IMAGE_FILTER_CTRL	28h

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 FREQ_WORD_ADJ	FREQ_WORD_ADJ This register is a signed 9 bit number that is added to the GFSK modulator output. This allows the GFSK modulation to be adjusted, or skewed, by a range of -512 to +511.
15-6 —	Reserved
5 IMAGE_SYNC0_FILTER_OVRD	IMAGE_SYNC0_FILTER_OVRD Overrides the interpolator sync0 filter clock enable.
4 IMAGE_SYNC1_FILTER_OVRD	IMAGE_SYNC1_FILTER_OVRD Overrides the interpolator sync1 filter clock enable.
3 IMAGE_FIR_FILTER_OVRD	IMAGE_FIR_FILTER_OVRD If the TX_IMAGE_FILTER_OVRD_EN bit is set, then this bit turns on (=1) and off (=0) the Transmit Image Filter 0
2 IMAGE_FILTER_OVRD_EN	IMAGE_FILTER_OVRD_EN This bit enables the TX Image Filter Override Control bits.
1-0	IMAGE_FIR_FILTER_SEL Image FIR Filter Logic Select.

Table continues on the next page...

Table continued from the previous page...

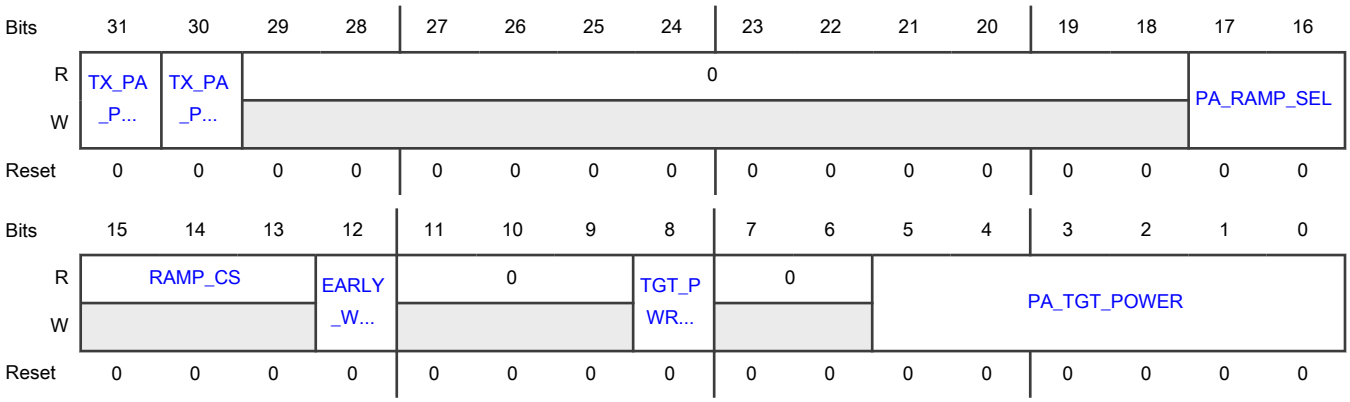
Field	Function
IMAGE_FIR_FILTER_SEL	2'b00: [11211] 2'b01: [12345654321] * (1/32 - 1/128) 2'b10: [12345677654321] * (1/64 - 1/512) 2'b11: Reserved

55.4.7.2.8.1.13 PA_CTRL (PA_CTRL)

Offset

Register	Offset
PA_CTRL	2Ch

Diagram



Fields

Field	Function
31 TX_PA_PUP_OVRD_EN	TX_PA_PUP_OVRD_EN This bit enables the TX PA_PUP Control bit.
30 TX_PA_PUP_OVRD	TX_PA_PUP_OVRD If the TX_PA_PUP_OVRD_EN bit is set, then this bit turns on (=1) and off (=0) the Transmit TX_PA_PUP bit.
29-18 —	Reserved
17-16	PA_RAMP_SEL

Table continues on the next page...

Table continued from the previous page...

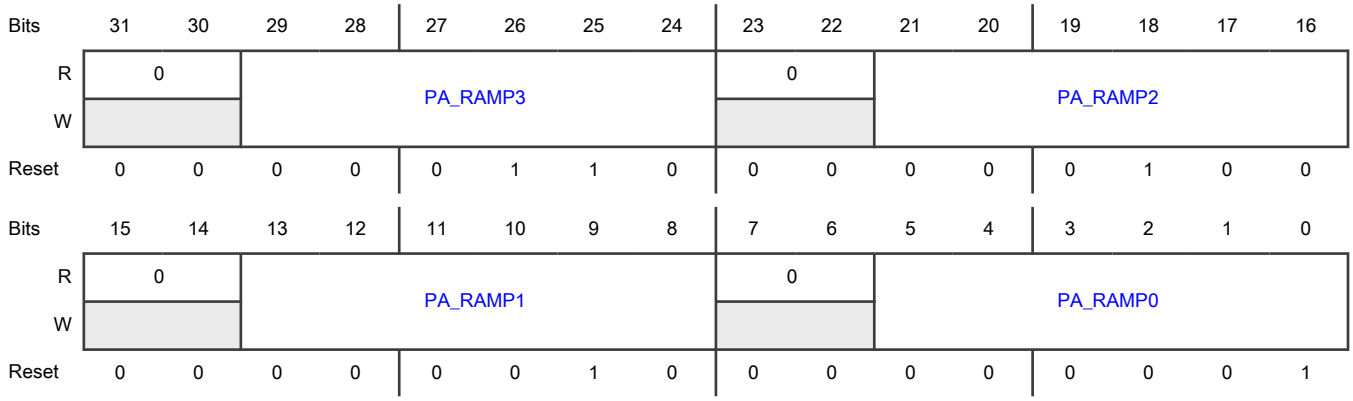
Field	Function
PA_RAMP_SEL	Indicates the ramp duration or no ramp. 2'b00: Ramp disabled. 2'b01: Ramp = 1us. 2'b10: Ramp = 2us. 2'b11: Ramp = 4us.
15-13 RAMP_CS	RAMP_CS PA Ramp Current State.
12 EARLY_WU_COMPLETE	EARLY_WU_COMPLETE When set, TXDIG 2p4 will assert pa_wu_complete after ramp-up without waiting for the settling time to expire (DATA_PADDING_CTRL_1[RAMP_UP_DLY]). If not set, the assertion of pa_wu_complete takes place normally, that is to say, after ramp-up the assertion of pa_wu_complete will happen after the number of cg1m cycles programmed in RAMP_UP_DLY have happened.
11-9 —	Reserved
8 TGT_PWR_SRC	TGT_PWR_SRC Indicated the source of the target power value. 1'b0: Target power specified in PA_TGT_POWER field. 1'b1: Target power specified by the link layer. .
7-6 —	Reserved
5-0 PA_TGT_POWER	PA_TGT_POWER The contents of this register are used as PA target power when TGT_PWR_SRC=0. The valid values for this fields are 0,1,2,4,6,8,...,62 (in increasing order of PA power). That is, above 1, only even numbers are permitted in the PA_TGT_POWER bitfield. Odd values of 3..63 are not permitted and will result in unexpected behavior.

55.4.7.2.8.1.14 PA_RAMP_TBL0 (PA_RAMP_TBL0)

Offset

Register	Offset
PA_RAMP_TBL0	30h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 PA_RAMP3	PA_RAMP3 PA_RAMP3 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the fourth ramp step. During PA ramp down, the contents of this register are the PA power value during the fourth-to-last ramp step. In both cases, PA_RAMP3 cannot exceed target power (enforced by PA ramping logic).
23-22 —	Reserved
21-16 PA_RAMP2	PA_RAMP2 PA_RAMP2 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the third ramp step. During PA ramp down, the contents of this register are the PA power value during the third-to-last ramp step. In both cases, PA_RAMP2 cannot exceed target power (enforced by PA ramping logic).
15-14 —	Reserved
13-8 PA_RAMP1	PA_RAMP1 PA_RAMP1 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the second ramp step. During PA ramp down, the contents of this register are the PA power value during the second-to-last ramp step. In both cases, PA_RAMP1 cannot exceed target power (enforced by PA ramping logic).
7-6 —	Reserved
5-0	PA_RAMP0

Table continues on the next page...

Table continued from the previous page...

Field	Function
PA_RAMP0	PA_RAMP0 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, when TSM tx_pa_en transitions low to high, and then for the duration of the first ramp step. During PA ramp down, the contents of this register are the PA power value during the final ramp step. In both cases, PA_RAMP0 cannot exceed target power (enforced by PA ramping logic). When PA ramping is enabled, the contents of PA_RAMP0 are also presented to the PA during sequenceidle conditions.

55.4.7.2.8.1.15 PA_RAMP_TBL1 (PA_RAMP_TBL1)

Offset

Register	Offset
PA_RAMP_TBL1	34h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		PA_RAMP7						0		PA_RAMP6					
W																
Reset	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		PA_RAMP5						0		PA_RAMP4					
W																
Reset	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	1

Fields

Field	Function
31-30 —	Reserved
29-24 PA_RAMP7	PA_RAMP7 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the eighth ramp step. During PA ramp down, the contents of this register are the PA power value during the eighth-to-last ramp step. In both cases, PA_RAMP7 cannot exceed target power (enforced by PA ramping logic).
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 PA_RAMP6	PA_RAMP6 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the seventh ramp step. During PA ramp down, the contents of this register are the PA power value during the seventh-to-last ramp step. In both cases, PA_RAMP6 cannot exceed target power (enforced by PA ramping logic).
15-14 —	Reserved
13-8 PA_RAMP5	PA_RAMP5 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the sixth ramp step. During PA ramp down, the contents of this register are the PA power value during the sixth-to-last ramp step. In both cases, PA_RAMP5 cannot exceed target power (enforced by PA ramping logic).
7-6 —	Reserved
5-0 PA_RAMP4	PA_RAMP4 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the fifth ramp step. During PA ramp down, the contents of this register are the PA power value during the fifth-to-last ramp step. In both cases, PA_RAMP4 cannot exceed target power (enforced by PA ramping logic).

55.4.7.2.8.1.16 PA_RAMP_TBL2 (PA_RAMP_TBL2)

Offset

Register	Offset
PA_RAMP_TBL2	38h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		PA_RAMP11						0		PA_RAMP10					
W																
Reset	0	0	1	0	0	1	1	0	0	0	1	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		PA_RAMP9						0		PA_RAMP8					
W																
Reset	0	0	0	1	1	1	0	0	0	0	0	1	1	0	0	0

Fields

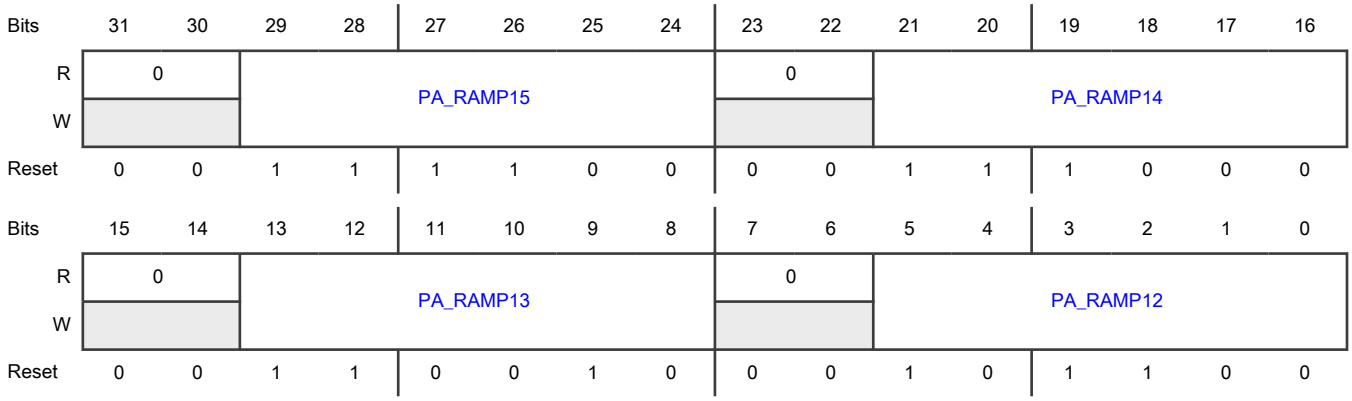
Field	Function
31-30 —	Reserved
29-24 PA_RAMP11	PA_RAMP11 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the twelvth ramp step. During PA ramp down, the contents of this register are the PA power value during the twelvth-to-last ramp step. In both cases, PA_RAMP11 cannot exceed target power (enforced by PA ramping logic).
23-22 —	Reserved
21-16 PA_RAMP10	PA_RAMP10 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the twelvth ramp step. During PA ramp down, the contents of this register are the PA power value during the twelvth-to-last ramp step. In both cases, PA_RAMP10 cannot exceed target power (enforced by PA ramping logic).
15-14 —	Reserved
13-8 PA_RAMP9	PA_RAMP9 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the tenth ramp step. During PA ramp down, the contents of this register are the PA power value during the tenth-to-last ramp step. In both cases, PA_RAMP9 cannot exceed target power (enforced by PA ramping logic).
7-6 —	Reserved
5-0 PA_RAMP8	PA_RAMP8 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the ninth ramp step. During PA ramp down, the contents of this register are the PA power value during the ninth-to-last ramp step. In both cases, PA_RAMP8 cannot exceed target power (enforced by PA ramping logic).

55.4.7.2.8.1.17 PA_RAMP_TBL3 (PA_RAMP_TBL3)

Offset

Register	Offset
PA_RAMP_TBL3	3Ch

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 PA_RAMP15	PA_RAMP15 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the sixteenth (final) ramp step. During PA ramp down, the contents of this register are the PA power value during the sixteenth-to-last (first) ramp step. In both cases, PA_RAMP15 cannot exceed target power (enforced by PA ramping logic).
23-22 —	Reserved
21-16 PA_RAMP14	PA_RAMP14 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the fifteenth ramp step. During PA ramp down, the contents of this register are the PA power value during the fifteenth-to-last ramp step. In both cases, PA_RAMP14 cannot exceed target power (enforced by PA ramping logic).
15-14 —	Reserved
13-8 PA_RAMP13	PA_RAMP13 If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the fourteenth ramp step. During PA ramp down, the contents of this register are the PA power value during the fourteenth-to-last ramp step. In both cases, PA_RAMP13 cannot exceed target power (enforced by PA ramping logic).
7-6 —	Reserved
5-0	PA_RAMP12

Table continues on the next page...

Table continued from the previous page...

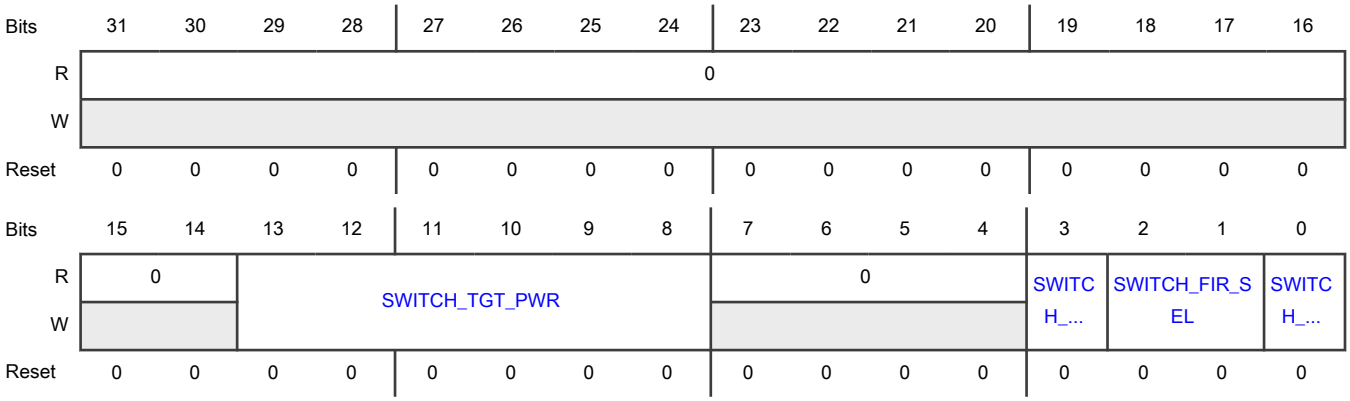
Field	Function
PA_RAMP12	If PA ramping is enabled (TSM_CTRL[PA_RAMP_SEL] > 00), the contents of this register are presented to the PA during PA ramping, for the duration of the thirteenth ramp step. During PA ramp down, the contents of this register are the PA power value during the thirteenth-to-last ramp step. In both cases, PA_RAMP12 cannot exceed target power (enforced by PA ramping logic).

55.4.7.2.8.1.18 SWITCH_TX_CTRL (SWITCH_TX_CTRL)

Offset

Register	Offset
SWITCH_TX_CTRL	40h

Diagram



Fields

Field	Function
31-14 —	Reserved
13-8 SWITCH_TGT_PWR	SWITCH_TGT_PWR The contents of this register are used as PA target power when PLL_DIG asserts switch_tx_config.
7-4 —	Reserved
3 SWITCH_GFSK_COEFF	SWITCH_GFSK_COEFF This bits tells the GFSK modulator to whether switch to the coefficients programmed in the manual cofficient registers when the PLL_DIG asserts switch_tx_config or use the default ones in the logic.

Table continues on the next page...

Table continued from the previous page...

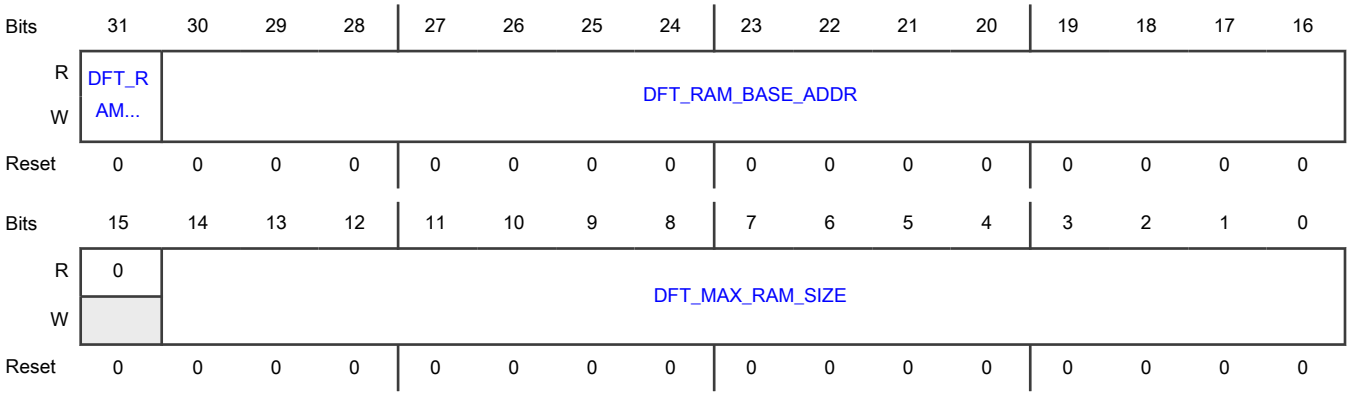
Field	Function
2-1 SWITCH_FIR_SEL	SWITCH_FIR_SEL If tx_image_filter_en is set and PLL_DIG asserts switch_tx_config, the image filter output is selected as follows: 2'b00: [11211] 2'b01: [12345654321] * (1/32 - 1/128) 2'b10: [12345677654321] * (1/64 - 1/512) 2'b11: [11211]
0 SWITCH_MOD	SWITCH_MOD This bit tells the TX Dig 2p4 to switch between GFSK and FSK modulators when PLL_DIG asserts switch_tx_config. 1'b0: GFSK 1'b1: FSK

55.4.7.2.8.1.19 RF_DFT_TX_CTRL0 (RF_DFT_TX_CTRL0)

Offset

Register	Offset
RF_DFT_TX_CTRL0	44h

Diagram



Fields

Field	Function
31	DFT_RAM_EN

Table continues on the next page...

Table continued from the previous page...

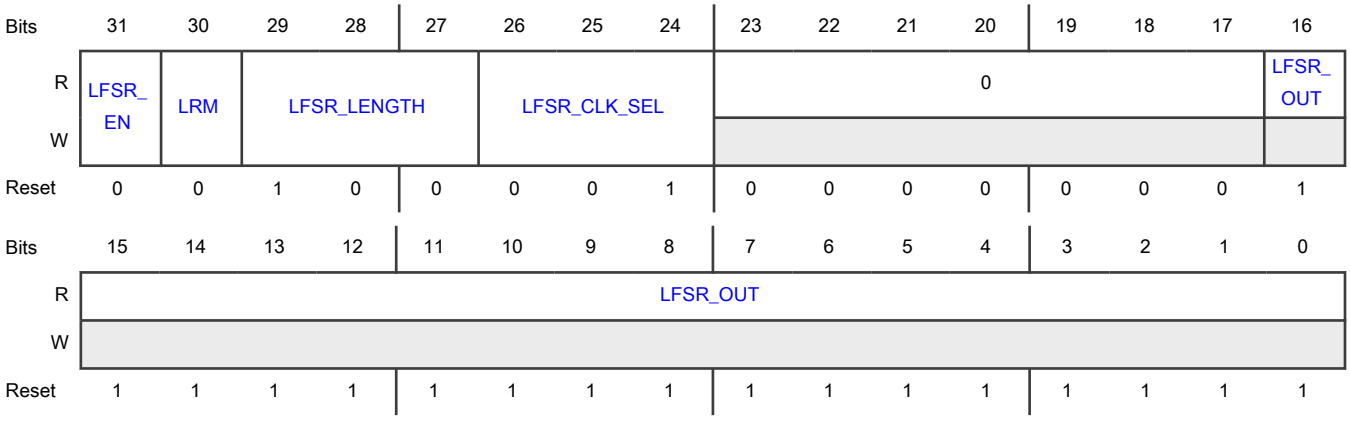
Field	Function
DFT_RAM_EN	Enables DFT RAM mode
30-16 DFT_RAM_BASE_ADDR	DFT_RAM_BASE_ADDR Base address in the RAM for DFT transmission.
15 —	Reserved
14-0 DFT_MAX_RAM_SIZE	DFT_MAX_RAM_SIZE RAM size in bytes to be used during DFT RAM transmission. Basically each word (16bit) is used as a frequency word. If negative value is stored, 2's complement shoud be stored in a word.

55.4.7.2.8.1.20 RF_DFT_TX_CTRL1 (RF_DFT_TX_CTRL1)

Offset

Register	Offset
RF_DFT_TX_CTRL1	48h

Diagram



Fields

Field	Function
31 LFSR_EN	LFSR_EN LFSR Enable. If the Radio is in a DFT LFSR mode, then this bit is used to turn on and off the LFSR that is used to generate the modulation. Note that the LFSR is clocked at the DFT Clock frequency.

Table continues on the next page...

Table continued from the previous page...

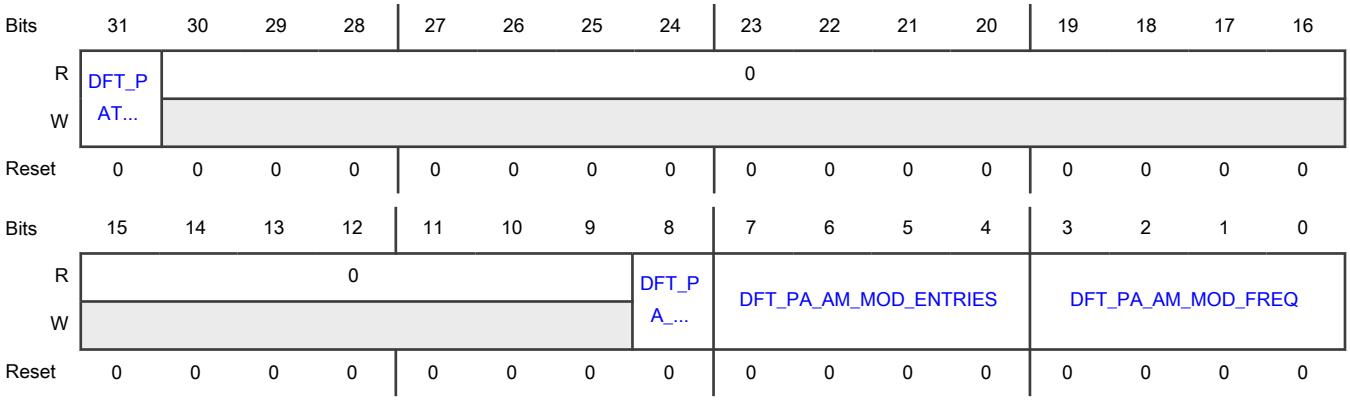
Field	Function
30 LRM	LRM LFSR Reset Mask When this bit is set the DFT LFSR will not be reset when LFSR_EN is cleared and will instead continue to repeat its sequence as defined by the DFT_LFSR_LEN bits when LFSR_EN is next set. When this bit is cleared the DFT LFSR will reset every time LFSR_EN is cleared.
29-27 LFSR_LENGTH	LFSR_LENGTH LFSR Length This register selects the length of the DFT LFSR and the associated LFSR Tap Mask. The Mask is in the form of [MSB...LSB] 000b - LFSR 9, tap mask 100010000 001b - LFSR 10, tap mask 1001000000 010b - LFSR 11, tap mask 11101000000 011b - LFSR 13, tap mask 1101100000000 100b - LFSR 15, tap mask 111010000000000 101b - LFSR 17, tap mask 11110000000000000 110b - Reserved 111b - Reserved
26-24 LFSR_CLK_SEL	LFSR_CLK_SEL DFT Clock SelectionThis register selects the frequency of the DFT clock that is used to shift out the DFT Modulation Pattern inDFT Pattern Register modes, and the same frequency is also used to clock the LFSR and generate thepseudo-random modulation in DFT LFSR modes. 000b - 62.5 kHz 001b - 125 kHz 010b - 250 kHz 011b - 500 kHz 100b - 1 MHz 101b - When this option is specified, the internal symbol clock (symb_clk_en) is selected. To ensure that 2 MHz is chosen, XCVR_CTRL[DATA_RATE] needs to be programmed to 3'b000. 110b - 4 MHz 111b - RF OSC Clock
23-17 —	Reserved
16-0 LFSR_OUT	LFSR_OUT LFSR Output This register can be read to observe the current value of the DFT LFSR, only bits [14:0] are available.

55.4.7.2.8.1.21 RF_DFT_TX_CTRL2 (RF_DFT_TX_CTRL2)

Offset

Register	Offset
RF_DFT_TX_CTRL2	4Ch

Diagram



Fields

Field	Function
31 DFT_PATTERN_EN	DFT_PATTERN_EN DFT pattern enable If the Radio is in a DFT Pattern mode, then this bit is used to turn on and off the pattern that is used to generate the modulation. Note that the DFT pattern is clocked at the DFT Clock frequency.
30-9 —	Reserved
8 DFT_PA_AM_MOD_EN	DFT_PA_AM_MOD_EN RF Power Amplifier Amplitude Modulation Enable This bit enables the DFT RF Power Amplifier Amplitude Modulation. As long as this bit is set the PA will have its amplitude modulated at the frequency selected with DFT_PA_AM_MOD_FREQ using the PA_RAMP table entries selected with DFT_PA_AM_MOD_ENTRIES.
7-4 DFT_PA_AM_MOD_ENTRIES	DFT_PA_AM_MOD_ENTRIES RF Power Amplifier Amplitude Modulation Table Entries This register selects the number PA_RAMP entries to use as the PA Amplitude Modulation Table Entries. The PA will be modulated using the Modulation Table Entries in a continuous loop. 0000b - Reserved 0001b - 2 entries 0010b - 3 entries

Table continues on the next page...

Table continued from the previous page...

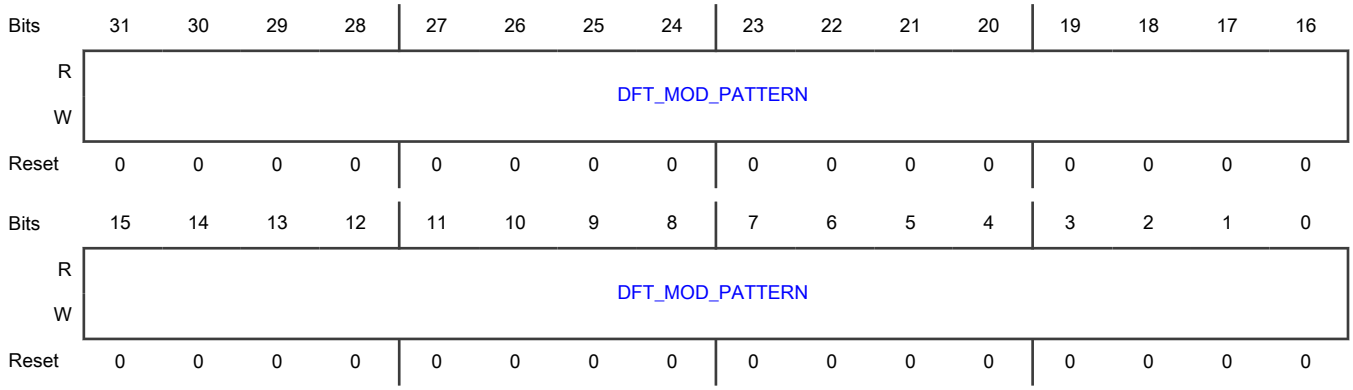
Field	Function
	0011b - 4 entries 0100b - 5 entries 0101b - 6 entries 0110b - 7 entries 0111b - 8 entries 1000b - 9 entries 1001b - 10 entries 1010b - 11 entries 1011b - 12 entries 1100b - 13 entries 1101b - 14 entries 1110b - 15 entries 1111b - 16 entries
3-0 DFT_PA_AM_MOD_FREQ	DFT_PA_AM_MOD_FREQ This register selects the Power Amplifier Amplitude Modulation Frequency. The PA will be modulated atthis frequency using the Modulation Table Entries in a continuous loop. 0000b - 32 MHz 0001b - 16 MHz 0010b - 8 MHz 0011b - 4 MHz 0100b - 2 MHz 0101b - 1 MHz 0110b - 500 kHz 0111b - 250 kHz 1000b - 125 kHz 1001b - 62.5 kHz 1010b - 1111b: Reserved

55.4.7.2.8.1.22 RF_DFT_PATTERN (RF_DFT_PATTERN)

Offset

Register	Offset
RF_DFT_PATTERN	50h

Diagram



Fields

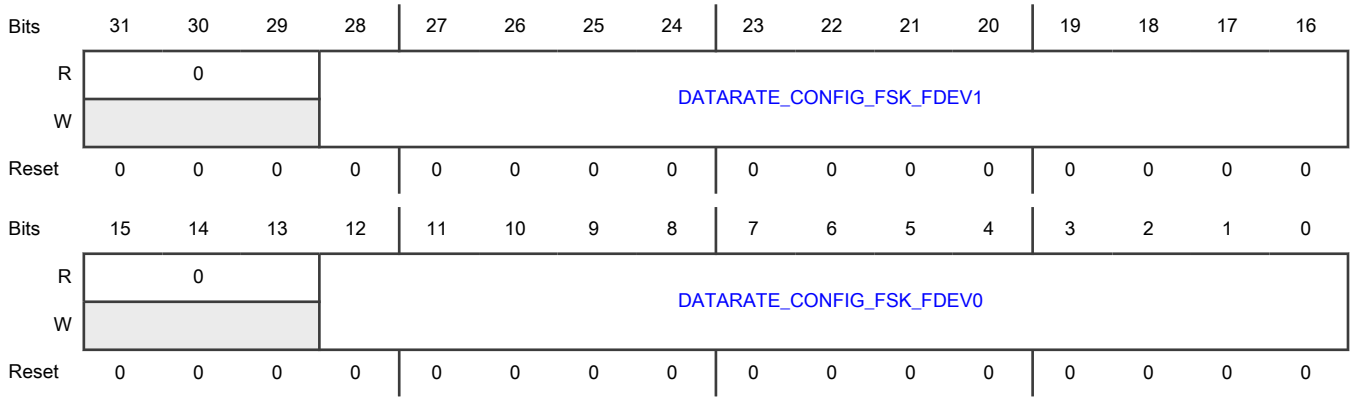
Field	Function
31-0 DFT_MOD_PATTERN	DFT_MOD_PATTERN DFT Modulation Pattern. The DFT Pattern Register can be written to any bit pattern.

55.4.7.2.8.1.23 DATARATE_CONFIG_FSK_CTRL (DATARATE_CONFIG_FSK_CTRL)

Offset

Register	Offset
DATARATE_CONFIG_FSK_CTRL	54h

Diagram



Fields

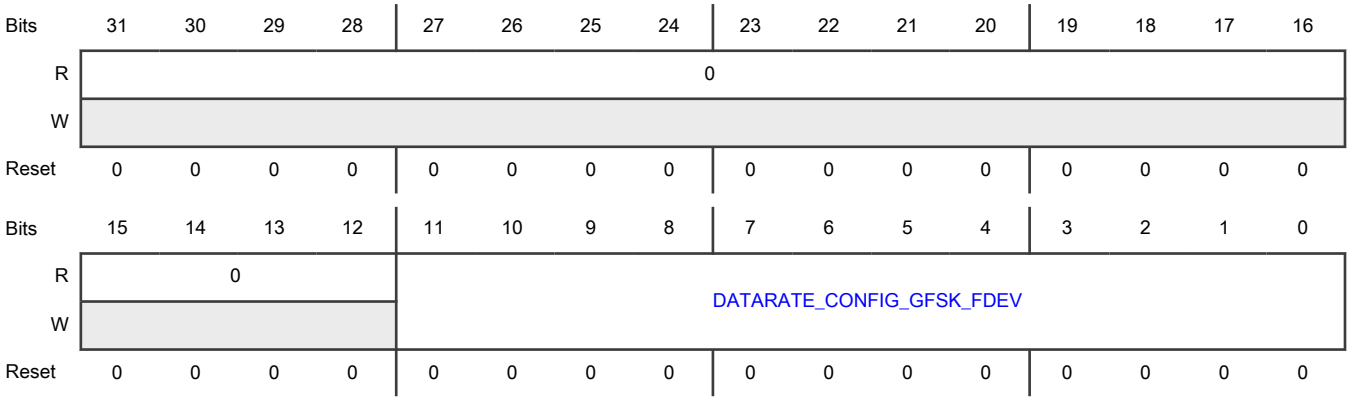
Field	Function
31-29 —	Reserved
28-16 DATARATE_CONFIG_FSK_FDEV1	DATARATE_CONFIG_DATA_PAD_PFDEV This register indicates the +1 signed frequency word to be used when datarate_config_sel is asserted.
15-13 —	Reserved
12-0 DATARATE_CONFIG_FSK_FDEV0	DATARATE_CONFIG_DATA_PAD_MFDEV This register indicates the -1 signed frequency word to be used when datarate_congig_sel is asserted.

55.4.7.2.8.1.24 DATARATE_CONFIG_GFSK_CTRL (DATARATE_CONFIG_GFSK_CTRL)

Offset

Register	Offset
DATARATE_CONFIG_GFSK_CTRL	58h

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

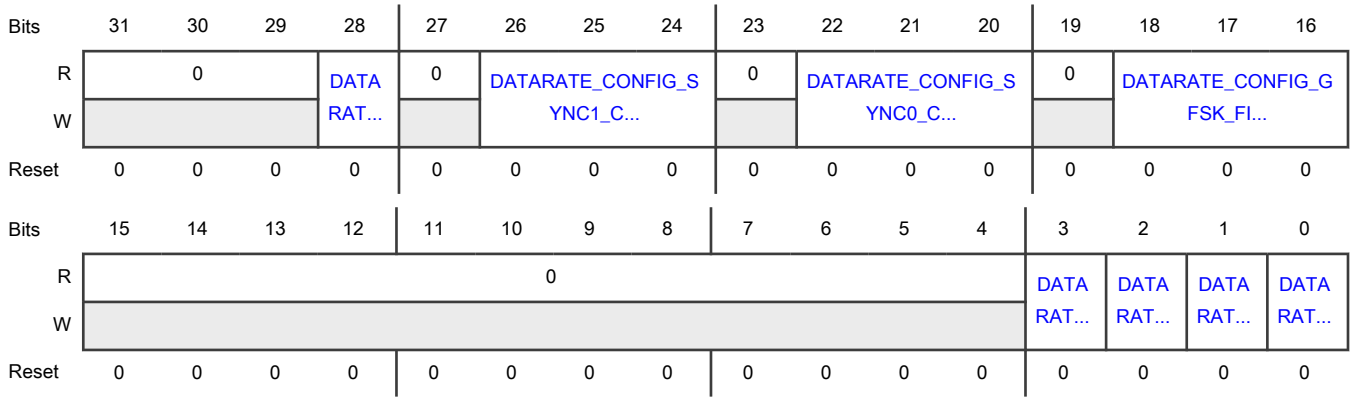
Field	Function
—	
15-12 —	Reserved
11-0 DATARATE_CONFIG_GFSK_FDEV	DATARATE_CONFIG_GFSK_FDEV This register indicates the signed frequency deviation word to be used when datarate_config_sel is asserted.

55.4.7.2.8.1.25 DATARATE_CONFIG_FILTER_CTRL (DATARATE_CONFIG_FILTER_CTRL)

Offset

Register	Offset
DATARATE_CONFIG_FILTER_CTRL	5Ch

Diagram



Fields

Field	Function
31-29 —	Reserved
28	DATARATE_CONFIG_IMAGE_FIR_CLK_SEL DATARATE_CONFIG_IMAGE_FIR_CLK_SEL If datarata_config_sel = 1'b1:

Table continues on the next page...

Table continued from the previous page...

Field	Function
DATARATE_CONFIG_IMAGE_FIR_CLK_SEL	1'b0: ref_clk 1'b1: ref_clk div_by_2
27 —	Reserved
26-24 DATARATE_CONFIG_SYNC1_CLK_SEL	DATARATE_CONFIG_IMAGE_SYNC1_CLK_SEL DATARATE_CONFIG_IMAGE_SYNC1_CLK_SEL If ref_clk = 32MHz: 3'b000: ref_clk 3'b001: ref_clk div_by_2 3'b010: ref_clk div_by_4 3'b011: ref_clk div_by_8 3'b100: ref_clk div_by_16 3'b101: ref_clk div_by_32 3'b110: ref_clk div_by_64 3'b111: ref_clk div_by_128 If ref_clk = 26MHz: 3'b000: ref_clk 3'b001: ref_clk div_by_2 3'b010: ref_clk div_by_4 3'b011: ref_clk div_by_13 3'b100: ref_clk div_by_26 3'b101: ref_clk div_by_52 3'b110: ref_clk div_by_104 3'b111: reserved
23 —	Reserved
22-20 DATARATE_CONFIG_SYNC0_CLK_SEL	DATARATE_CONFIG_IMAGE_SYNC0_CLK_SEL DATARATE_CONFIG_IMAGE_SYNC0_CLK_SEL If ref_clk = 32MHz: 3'b000: ref_clk 3'b001: ref_clk div_by_2 3'b010: ref_clk div_by_4 3'b011: ref_clk div_by_8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	3'b100: ref_clk div_by_16 3'b101: ref_clk div_by_32 3'b110: ref_clk div_by_64 3'b111: ref_clk div_by_128 If ref_clk = 26MHz: 3'b000: ref_clk 3'b001: ref_clk div_by_2 3'b010: ref_clk div_by_4 3'b011: ref_clk div_by_13 3'b100: ref_clk div_by_26 3'b101: ref_clk div_by_52 3'b110: ref_clk div_by_104 3'b111: reserved
19 ——	Reserved
18-16 DATARATE_CONFIG_GFSK_FILT_CLK_SEL	DATARATE_CONFIG_GFSK_FILT_CLK_SEL DATARATE_CONFIG_GFSK_FILT_CLK_SEL If datarata_config_sel = 1'b1: 3'b000: ref_clk 3'b001: ref_clk div_by_2 3'b010: ref_clk div_by_4 3'b011: ref_clk div_by_8 3'b100: ref_clk div_by_16 3'b101: ref_clk div_by_32 3'b110: ref_clk div_by_64 3'b111: ref_clk div_by_128
15-4 ——	Reserved
3 DATARATE_CONFIG_SYNC1_FILTER_OVRD	DATARATE_CONFIG_SYNC1_FILTER_OVRD DATARATE_CONFIG_SYNC1_FILTER_OVRD
2	DATARATE_CONFIG_SYNC0_FILTER_OVRD

Table continues on the next page...

Table continued from the previous page...

Field	Function
DATARATE_CONFIG_SYNC0_FILTER_OVRD	DATARATE_CONFIG_SYNC0_FILTER_OVRD
1 DATARATE_CONFIG_FIR_FILTER_OVRD	DATARATE_CONFIG_FIR_FILTER_OVRD DATARATE_CONFIG_FIR_FILTER_OVRD
0 DATARATE_CONFIG_IMAGE_FILTER_OVRD_EN	DATARATE_CONFIG_IMAGE_FILTER_OVRD_EN DATARATE_CONFIG_FIR_FILTER_OVRD_EN

55.4.7.3 PLL Digital Module

55.4.7.3.1 About the PLL Frequency Synthesizer

55.4.7.3.1.1 Introduction

The PLL Frequency Synthesizer is a Digital Module that -

1. Selects the Radio Carrier Frequency
2. Coarse Tunes the VCO and Calibrates the High Port DAC
3. Controls the PLL Loop Divider value to lock the VCO at the Carrier Frequency
4. Applies the Baseband Frequency Word as High Port Modulation (HPM) to the VCO High Port DAC
5. Applies the Baseband Frequency Word as Low Port Modulation (LPM) to the PLL Loop Divider
6. Monitors the PLL lock status and flags unlocked conditions
7. Provides DFT features for evaluating the PLL Frequency Synthesizer during factory validation and testing

55.4.7.3.1.2 Block diagram

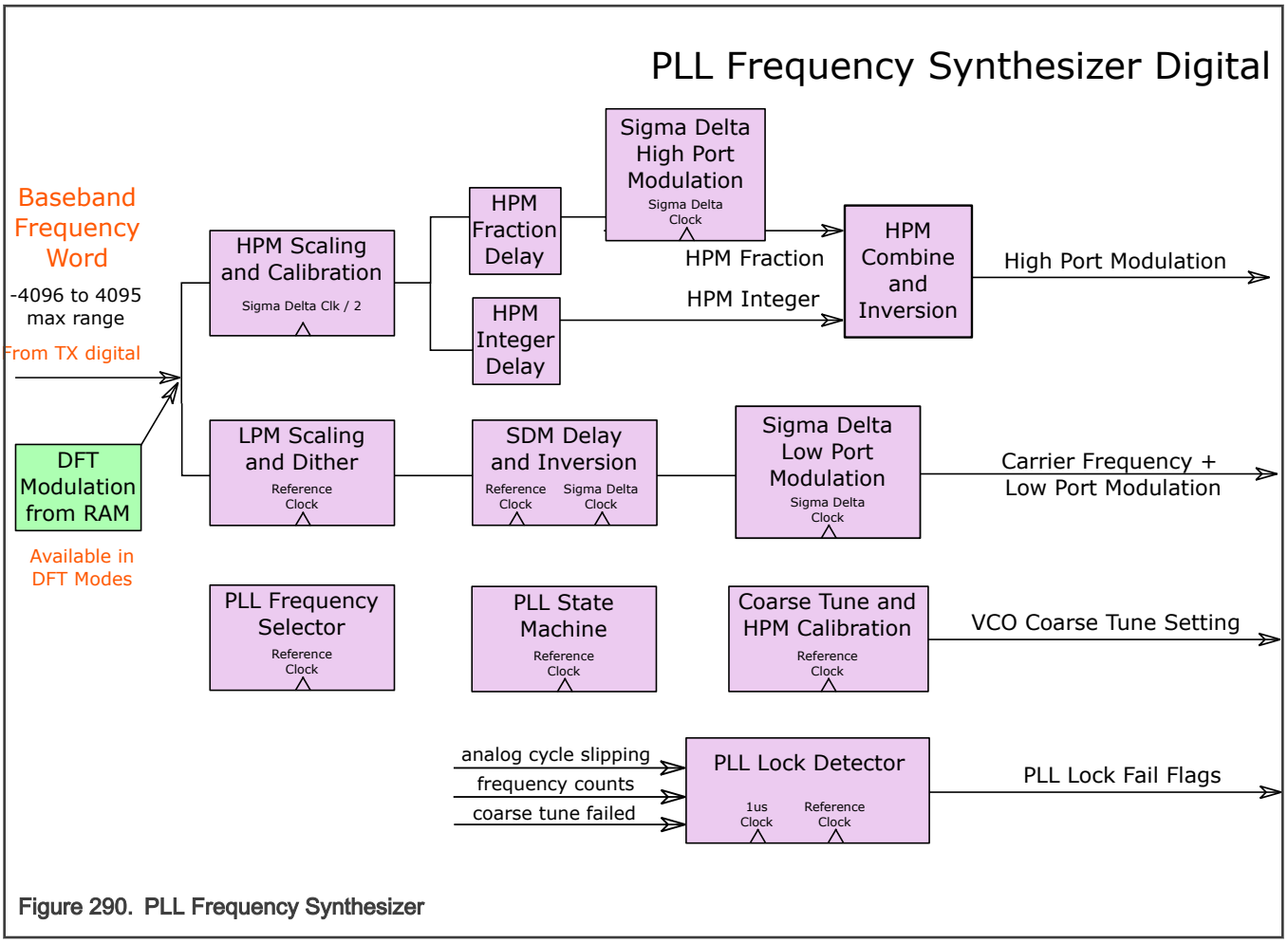


Figure 290. PLL Frequency Synthesizer

55.4.7.3.1.3 Baseband Frequency Word

The Transmitter Digital presents the PLL Frequency Synthesizer with a signed 13-bit Baseband Frequency word to modulate onto the RF Carrier Frequency. The Baseband Frequency Word range is -4096 to 4095, which provides a modulation range of +/- 1000 kHz when using a 32 MHz reference frequency with a PLL Minimum Frequency Step Size of 244.14 Hz. Note, this range can be adjusted using LPM_CTRL[LPM_SCALE] along with LPM_CTRL[HPM_CAL_SCALE] to adjust the PLL Minimum Frequency Step Size.

55.4.7.3.1.3.1 Manual Frequency Word

The Baseband Frequency word can be overridden by software if the MOD_DISABLE bit is set. In this case the PLL Digital will use the MODULATION_WORD_MANUAL register as the source of the modulation.

Note that due to the speeds at which the High Port and Low Port react to changes in the modulation word, and the speed at which software is able to effect such changes, for all practical purposes this can only support low frequency modulation.

55.4.7.3.2 Carrier Frequency Tuning

55.4.7.3.2.1 Radio Frequency Selection

The PLL Digital has hardware support for 128 RF channels in 1 MHz increments spanning the range from 2.360 GHz to 2.487 GHz.

Each supported carrier frequency is derived by Fractional-N Frequency Synthesis using the Low Port Sigma Delta Modulator. Note, the frequency selection registers should be programmed from the perspecive of TX operation. If the PLL is operating in

an RX mode then the CHAN_MAP_S3[NUM_OFFSET] will be utilized to derive the RX channel frequency. The NUM_OFFSET register field represents the IF frequency difference between RX and TX.

These 128 RF channels are mapped to the various supported RF Protocols through two lookup tables as shown in the figure below.

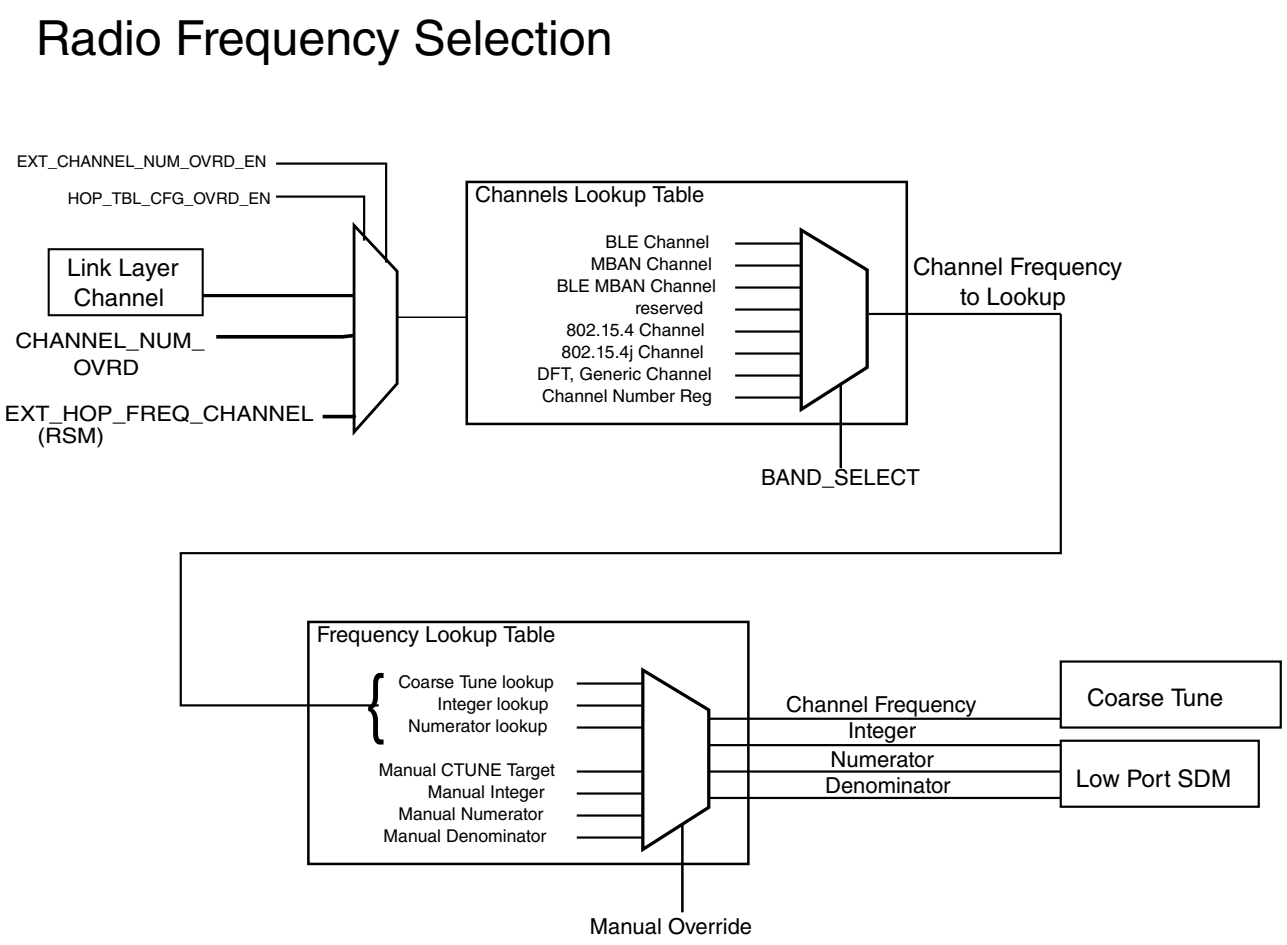


Figure 291. Radio Frequency Selection

55.4.7.3.2.2 RF Channels Supported

The channel frequencies for each protocol are shown in the tables below. The channel mapping is selected based on the register CHAN_MAP[BAND_SELECT].

Table 425. Radio Protocols (1 to 5) Channel Frequencies

Radio Protocols Supported, Channel Frequency in MHz													
Bluetooth Low Energy		MBAN		Bluetooth Low Energy/MBAN		Reserved				802.15.4		802.15.4j MBAN	
0		1		2		3				4		5	
Chan	MHz	Chan	MHz	Chan	MHz	Chan	MHz	Chan	MHz	Chan	MHz	Chan	MHz
0	2,402	0	2,360	0	2,402	0	2,400	43	2,443	11	2,405	0	2,363

Table continues on the next page...

Table 425. Radio Protocols (1 to 5) Channel Frequencies (continued)

1	2,404	1	2,361	1	2,404	1	2,401	44	2,444	12	2,410	1	2,368
2	2,406	2	2,362	2	2,406	2	2,402	45	2,445	13	2,415	2	2,373
3	2,408	3	2,363	3	2,408	3	2,403	46	2,446	14	2,420	3	2,378
4	2,410	4	2,364	4	2,410	4	2,404	47	2,447	15	2,425	4	2,383
5	2,412	5	2,365	5	2,412	5	2,405	48	2,448	16	2,430	5	2,388
6	2,414	6	2,366	6	2,414	6	2,406	49	2,449	17	2,435	6	2,393
7	2,416	7	2,367	7	2,416	7	2,407	50	2,450	18	2,440	7	2,367
8	2,418	8	2,368	8	2,418	8	2,408	51	2,451	19	2,445	8	2,372
9	2,420	9	2,369	9	2,420	9	2,409	52	2,452	20	2,450	9	2,377
10	2,422	10	2,370	10	2,422	10	2,410	53	2,453	21	2,455	10	2,382
11	2,424	11	2,371	11	2,424	11	2,411	54	2,454	22	2,460	11	2,387
12	2,426	12	2,372	12	2,426	12	2,412	55	2,455	23	2,465	12	2,392
13	2,428	13	2,373	13	2,428	13	2,413	56	2,456	24	2,470	13	2,397
14	2,430	14	2,374	14	2,430	14	2,414	57	2,457	25	2,475	14	2,395
15	2,432	15	2,375	15	2,432	15	2,415	58	2,458	26	2,480		
16	2,434	16	2,376	16	2,434	16	2,416	59	2,459				
17	2,436	17	2,377	17	2,436	17	2,417	60	2,460				
18	2,438	18	2,378	18	2,438	18	2,418	61	2,461				
19	2,440	19	2,379	19	2,440	19	2,419	62	2,462				
20	2,442	20	2,380	20	2,442	20	2,420	63	2,463				
21	2,444	21	2,381	21	2,444	21	2,421	64	2,464				
22	2,446	22	2,382	22	2,446	22	2,422	65	2,465				
23	2,448	23	2,383	23	2,448	23	2,423	66	2,466				
24	2,450	24	2,384	24	2,450	24	2,424	67	2,467				
25	2,452	25	2,385	25	2,452	25	2,425	68	2,468				
26	2,454	26	2,386	26	2,454	26	2,426	69	2,469				
27	2,456	27	2,387	27	2,456	27	2,427	70	2,470				
28	2,458	28	2,388	28	2,458	28	2,428	71	2,471				
29	2,460	29	2,389	29	2,460	29	2,429	72	2,472				
30	2,462	30	2,390	30	2,390	30	2,430	73	2,473				
31	2,464	31	2,391	31	2,391	31	2,431	74	2,474				
32	2,466	32	2,392	32	2,392	32	2,432	75	2,475				
33	2,468	33	2,393	33	2,393	33	2,433	76	2,476				

Table continues on the next page...

Table 425. Radio Protocols (1 to 5) Channel Frequencies (continued)

34	2,470	34	2,394	34	2,394	34	2,434	77	2,477				
35	2,472	35	2,395	35	2,395	35	2,435	78	2,478				
36	2,474	36	2,396	36	2,396	36	2,436	79	2,479				
37	2,476	37	2,397	37	2,397	37	2,437	80	2,480				
38	2,478	38	2,398	38	2,398	38	2,438	81	2,481				
39	2,480	39	2,399	39*	2,480	39	2,439	82	2,482				
						40	2,440	83	2,483				
						41	2,441						
						42	2,442						

* The Bluetooth Low Energy MBAN Channel Remap bit (BMR) controls the frequency mapping for channel 39 in Bluetooth Low Energy/MBAN, 2480 MHz is the default, but this channel can be remapped to 2399 MHz by setting BMR to a 1.

Table 426. Radio Protocols (6 to 7) Channel Frequencies

Radio Protocols Supported, Channel Frequency in MHz					
DFT, Generic					
6,7					
Channel	MHz	Channel	MHz	Channel	MHz
0	2,360	43	2,403	86	2,446
1	2,361	44	2,404	87	2,447
2	2,362	45	2,405	88	2,448
3	2,363	46	2,406	89	2,449
4	2,364	47	2,407	90	2,450
5	2,365	48	2,408	91	2,451
6	2,366	49	2,409	92	2,452
7	2,367	50	2,410	93	2,453
8	2,368	51	2,411	94	2,454
9	2,369	52	2,412	95	2,455
10	2,370	53	2,413	96	2,456
11	2,371	54	2,414	97	2,457
12	2,372	55	2,415	98	2,458
13	2,373	56	2,416	99	2,459
14	2,374	57	2,417	100	2,460
15	2,375	58	2,418	101	2,461
16	2,376	59	2,419	102	2,462

Table continues on the next page...

Table 426. Radio Protocols (6 to 7) Channel Frequencies (continued)

17	2,377	60	2,420	103	2,463
18	2,378	61	2,421	104	2,464
19	2,379	62	2,422	105	2,465
20	2,380	63	2,423	106	2,466
21	2,381	64	2,424	107	2,467
22	2,382	65	2,425	108	2,468
23	2,383	66	2,426	109	2,469
24	2,384	67	2,427	110	2,470
25	2,385	68	2,428	111	2,471
26	2,386	69	2,429	112	2,472
27	2,387	70	2,430	113	2,473
28	2,388	71	2,431	114	2,474
29	2,389	72	2,432	115	2,475
30	2,390	73	2,433	116	2,476
31	2,391	74	2,434	117	2,477
32	2,392	75	2,435	118	2,478
33	2,393	76	2,436	119	2,479
34	2,394	77	2,437	120	2,480
35	2,395	78	2,438	121	2,481
36	2,396	79	2,439	122	2,482
37	2,397	80	2,440	123	2,483
38	2,398	81	2,441	124	2,484
39	2,399	82	2,442	125	2,485
40	2,400	83	2,443	126	2,486
41	2,401	84	2,444	127	2,487
42	2,402	85	2,445		

55.4.7.3.2.3 Channel Selector

During typical Radio usage, the channel selection is done automatically by the Link Layer.

Channel selection can be overridden using the PLL_CHAN_MAP register bits HOP_TBL_CFG_OVRD_EN and HOP_TBL_CFG_OVRD. If HOP_TBL_CFG_OVRD_EN is set then the CHANNEL_NUM_OVRD register selects the Radio Channel Number instead of the Link Layer. In addition, the HOP_TBL_CFG_OVRD selection will determing how the CHANNEL_NUM_OVRD is interpreted.

55.4.7.3.2.4 Manual Carrier Frequency

The internal frequency mapping can be bypassed and software can directly control the Low Port Sigma Delta inputs to select any channel frequency in the VCO's range (in increments of the Low Port Minimum Frequency Step Size).

The Radio Channel Frequency override is manually selected by setting the SDM_MAP_DISABLE bit along with the LPM_INTG, LPM_NUM, and LPM_DENOM register values.

Any modulation from the TX Digital will be added to this manual frequency selection.

The Manual carrier frequency selected can be calculated using the formula below:

Radio Carrier Frequency = ((Reference Clock Frequency x 2) x (LPM_INTG + (LPM_NUM / LPM_DENOM))

WARNING : The fraction (LPM_NUM / LPM_DENOM) must be in the range of -0.55 to +0.55 for valid Sigma Delta Modulator operation.

55.4.7.3.3 VCO Calibration

55.4.7.3.3.1 Coarse Tune

The VCO Coarse Tune is carried out by a Successive Approximation Register method that computes the best coarse tune setting for the VCO in successive steps from MSB to LSB.

This SAR method uses a Frequency Meter count value that is successively compared to expected counts to determine the best coarse tune setting for the VCO.

The coarse tune setting can be overridden for test and validation purposes, and the individual frequency counts used at each successive step for the coarse tune setting are available for viewing in DTEST mode and also as register values in the PLL register space.

55.4.7.3.3.2 High Port DAC Calibration

The objective of the HPM DAC Calibration is to determine the DAC Step Size and estimate the ratio of this DAC unit step to the Low Port Minimum Frequency Step Size.

Once the step size of the High Port DAC is known, a scaling factor is applied to correctly map the baseband frequency word to the High Port modulation.

The length of the calibration time is set using the HPM_CAL_TIME register and the size of the DAC array used is selected using the HPM_CAL_ARRAY_SIZE register.

The register HPM_COUNT_ADJUST[3:0] provides a (-8/+7) range of adjustment to the HPM Calibration Count Difference for the HPM Calibration lookup table to allow for any count error bias that may be present. This adjusted Count Difference is used with a Lookup Table to choose the HPM Calibration Factor

55.4.7.3.3.2.1 HPM CAL Factor

The HPM_CAL_FACTOR register shows the value that is currently being used by High Port Modulation Multiplier. This value is from the most recent High Port calibration unless it is being overridden using the HP_CAL_DISABLE and HPM_CAL_FACTOR_MANUAL registers.

55.4.7.3.4 High Port Modulation

The High Port Modulator multiplies the HPM Calibration Factor and the PLL Baseband Frequency Word to get the High Port Modulation Word.

The High Port Modulation Word is represented by both an Integer and a Fractional portion. The Fraction is processed by the High Port Sigma Delta Modulator and the result is combined with the Integer and used to control the VCO High Port DAC.

55.4.7.3.4.1 High Port DAC

The High Port Digital to Analog Converter is designed to take the High Port Modulation Word and convert it to a control voltage that is applied as modulation to the VCO High Port.

55.4.7.3.4.1.1 HPM DAC Settings

The HPM DAC settings during calibration and normal operation are shown in the table below.

Warning : TX operation with On-Air modulation ranges greater than +/- 250 kHz (+/- 500 kHz at the VCO) require that the VCO Kvm be adjusted using the PLL_VCO_TRIM_KVM[2:0] register.

Table 427. Nominal HPM DAC parameters in various modes

Radio TX Mode		DAC max swing (mV)	DAC output common mode (mV)	DAC swing during operation	Vcm (mV) / HPM_VCM	OPAMP gain setting (R2/R1) / HPM_FDB_RES	Common mode at varactor (mV)	Swing at Varactor input (mV)	Kvm (MHz/Kv)	Deviation at VCO (KHz)[1]
802.15.4/ 2000kHz	HPM calibration	+/-82	82	+/-82	288 / 3'b001	2/ 2'b01	700	+/-164	20	+/-3280
	TX operation	+/-82	82	+/-51.2	391 / 3'b000	1/ 2'b00	700	+/51.2	20	+/-1024
Bluetooth Low Energy/ 1000kHz	HPM calibration	+/-82	82	+/-82	288 / 3'b001	2 / 2'b01	700	+/-164	10	+/-1640
	TX operation	+/-82	82	+/-51.2	391 / 3'b000	1/ 2'b00	700	+/51.2	10	+/-512
500kHz	HPM calibration	+/-82	82	+/-82	288 / 3'b001	2/ 2'b01	700	+/-164	10	+/-1640
	TX operation	+/-82	82	+/-25.6	391 / 3'b000	1/ 2'b00	700	+/25.6	10	+/-256
250kHz	HPM calibration	+/-82	82	+/-82	288 / 3'b001	2/ 2'b01	700	+/-164	10	+/-1640
	TX operation	+/-82	82	+/-12.8	391 / 3'b000	1/ 2'b00	700	+/12.8	10	+/-128

1. On-Air modulation frequency is a divide by 2 of the VCO frequency.

55.4.7.3.4.1.2 HPM DAC Bump

The HPM DAC Calibration is normally done at 2X the On-Air Modulation Range, this is accomplished by writing the PLL HPM Analog Bump Control (PLL_HPM_BUMP) register bits:

- HPM_VCM_CAL, value applied during HPM DAC Calibration
- HPM_FDB_RES_CAL, value applied during HPM DAC Calibration
- HPM_VCM_TX, value applied during Radio Transmissions
- HPM_FDB_RES_TX, value applied during Radio Transmissions

If the registers are written in such a way that the DAC range is not doubled, then the HPM_CAL_NOT_BUMPED bit must be set to change the math in the HPM Calibration.

55.4.7.3.4.1.3 HPM DAC 1 MHz Range

TX operation with On-Air modulation ranges greater than +/- 250 kHz (+/- 500 kHz at the VCO) require that the VCO Kvm be adjusted using the PLL_VCO_TRIM_KVM[2:0] register.

This VCO Kvm setting should be made prior to HPM DAC calibration.

55.4.7.3.4.2 High Port Sigma Delta

The High Port fractional modulation is implemented using a 2nd-order MASH Sigma Delta Modulator (SDM), which converts the 6-bit fractional portion of the high port modulation word to a 3-state thermometric code {-1,0,+1}. The resulting code is then added to the integer portion of the high port modulation word and the final result is applied to the HPM DAC.

55.4.7.3.4.2.1 HPM SDM Dither

In addition to the nominal SDM functionality, the HPM SDM design includes a zero-mean dithering to mitigate idle tones that can occur when modulating a string of ones or zeros.

The Dither is implemented using a Linear-Feedback Shift Register (LFSR) which provides a 1-bit pseudo-randomization that adds or subtracts 1-bit to the 6-bit fractional portion before it is processed by the Sigma Delta Modulator.

The length of the LFSR can be selected using the HPM_LFSR_SIZE register, and the scale can be changed using the HPM_DTH_SCL bit. The Dither can also be disabled by clearing the HPM_DTH_EN bit.

55.4.7.3.4.3 High Port Delay, Disable, Invert

The HPM Integer and Fraction portions each have an array of programmable delays available to allow the HPM modulations to be matched to each other and to the LPM modulation and the delay through the PLL Loop Divider and onto the VCO feedback loop.

The HPM delay steps are in increments of the PLL Sigma Delta Clock divided by 2, and the LPM SDM delay steps are in increments of the PLL Sigma Delta Clock. All three delay step settings are in the range of {0 to 15}.

There are several inversion bits that can be used if needed:

- HPM_CAL_INVERT, Invert the High Port Modulator during Calibration
- HPM_MOD_IN_INVERT, Invert the input word to the High Port Modulator
- HPM_INTEGER_INVERT, Invert the High Port Modulation Integer portion
- HPM_SDM_OUT_INVERT, Invert the HPM SDM Fractional Output

High Port Modulation can be disabled by setting the HPM_MOD_DISABLE bit, in this case the HPM_MOD_MANUAL register value is applied to the VCO High Port. Disabling high port modulation can be useful depending on the modulation range required for the desired data rate. For use cases where no high port modulation is required, the TX warm-up time can be reduced by skipping the HPM Calibration. To skip HPM CAL, the register bit HPMCAL_CTRL[HPM_CAL_SKIP] must be set. In this case, the HPM_MOD_MANUAL register should be set to 0x80. Note, the TSM default warm-up sequence must be overridden if HPM_CAL_SKIP is set to recover the HPM CAL warm-up time.

55.4.7.3.5 Low Port Modulation

The Low Port Modulator uses a 3rd order Sigma-Delta to produce a fractional division (Fractional-N Frequency Synthesis) to control the PLL Loop Divider and tune the VCO to the Radio Carrier Frequency.

The Low Port Baseband Frequency Word is added as modulation to the Sigma-Delta input.

55.4.7.3.5.1 Low Port Resolution

The table below shows the PLL Minimum Frequency Step Sizes for Modulation and Carrier Frequency tuning. Note, the PLL Minimum Carrier Frequency Step size is equivalent to the PLL Numerator LSB resolution. This step size is measuered at the PLL VCO. This table is calculated using the default numerator value of Numerator = 0x400_0000.

Note that the minimum step size for carrier frequency tuning is a divide by 256 of the modulation step sizes.

Table 428. PLL Minimum Frequency Step Size

Ref Clk Freq (MHz)	32	26

Table continues on the next page...

Table 428. PLL Minimum Frequency Step Size (continued)

PLL Minimum Modulation Step (Hz)	244.14	198.36
PLL Minimum Carrier Frequency Step (Hz)	0.9536	0.7749

55.4.7.3.5.2 Low Port Sigma-Delta

The Integer, Numerator, and Denominator are presented at the input to the Sigma-Delta Modulator.

A 3-level quantized Sigma-Delta output signal {-1, 0, +1} is generated and combined with the Integer divide value to produce the Modulus Divide Ratio for the PLL Loop Divider. The resulting RF Frequency can be calculated using the formula below:

RF Frequency = ((Reference Clock Frequency x 2) x (LPM_INTG + (LPM_NUM / LPM_DENOM))

WARNING : The fraction (LPM_NUM / LPM_DENOM) must be in the range of -0.55 to +0.55 for valid Sigma Delta Modulator operation.

55.4.7.3.5.2.1 LPM SDM Dither

While in Transmit mode the input to the SDM is naturally dithered by the Low Port modulation, which keeps the SDM from generating a pseudo-periodic output. However, in Receive mode, the SDM can potentially generate idle tones (dominant frequency modes at its output) when the input to the SDM is a constant. And in either Transmit or Receive mode the LPM SDM can generate idle tones if the Fraction is near zero.

To mitigate these impairments, the Numerator input to the SDM is dithered whenever the Numerator is in the near zero range { -64 to 63 }, or whenever the Radio is in Receive Mode.

This dither operation can also be overridden using the LPM_D_OVRD and LPM_D_CTRL bits.

55.4.7.3.5.3 Low Port Delay, Disable, Invert

The Low Port Modulator has an array of programmable delays available to allow the HPM modulation to be matched to the LPM modulation and the delay through the PLL Loop Divider and onto the VCO feedback loop.

The Low Port Modulation can be delayed at the PLL Loop Divider by delaying the input to the LPM SDM (LPM_SDM_DELAY).

There is an option to invert the Numerator (LPM_SDM_INV) before it is applied to the SDM in order to invert the sigma-delta modulation to the PLL.

The Low Port Modulation of the Loop Divider can be disabled by setting the LPM_DISABLE bit. When the modulation is disabled the LPM SDM will continue to tune the VCO to the Radio Carrier Frequency.

55.4.7.3.6 Lock Detector

The PLL Lock Detector has two methods to measure the stability of the PLL and check that it is tuned to the selected Radio Frequency.

The real-time results of the PLL Lock Detect module can be read as register bits, and flags are set and can be cleared to provide software long-term access to any PLL unlock events.

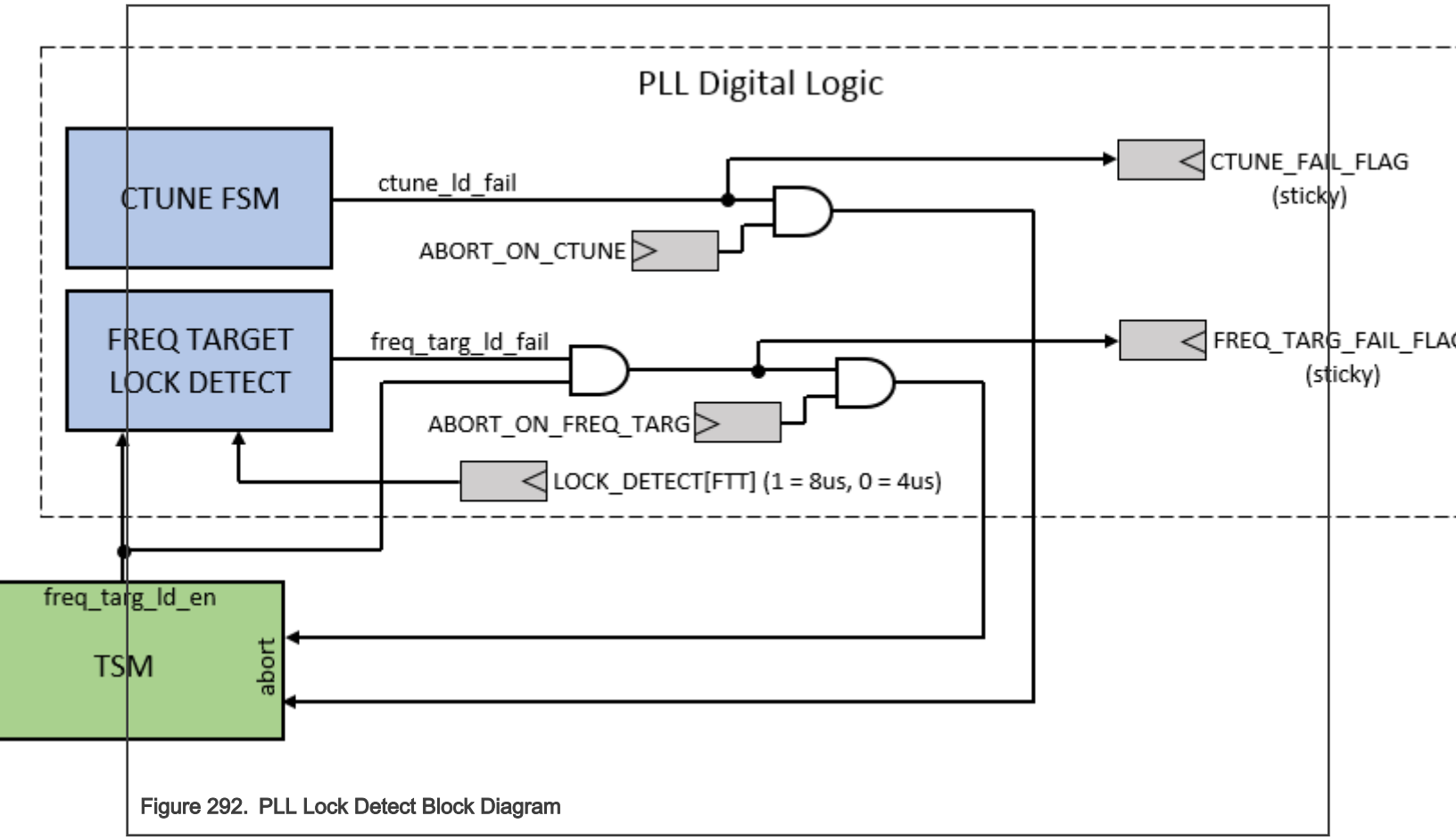


Figure 292. PLL Lock Detect Block Diagram

55.4.7.3.6.1 Coarse Tune Fail

During every Radio start-up sequence the PLL does a Coarse Tune Calibration to select a VCO setting that will allow the PLL to correctly tune and lock the VCO when the PLL/VCO loop is closed.

During the coarse tune calibration a count is maintained that indicates how close the coarse tune is to the targeted Radio Frequency, if this count is outside of the threshold selected by the Coarse Tune Lock Detect Fail Level register (LOCK_DETECT[CTUNE_LDF_LEV]), then the Coarse Tune Failure Flag bit (CTFF) will be set.

The CTFF bit is cleared by writing a 1 to it.

The CT_FAIL bit always shows the real time status of the latest Coarse Tune Calibration. If the absolute count difference was out of the range selected by CTUNE_LDF_LEV, then this bit will be set.

55.4.7.3.6.2 Fine Tune Fail

An additional lock detect method, Fine Tune Lock Detect, can be added to the end of a start-up sequence. The PLL turns on the PLL Frequency Meter for a time that is requested by the Frequency Target Window time select register bits, separately selected for RX (FTW_RX) and TX (FTW_TX).

By default, the pll frequency meter will be turned off after the standard warm-up routine (HPM CAL in TX Mode, CTUNE in RX Mode). THe Fine Tune Lock Detect sequence required that the frequency meter remain on after standard warm-up. To override the default behaviour, the LOCK_DETECT[FCAL_HOLD_EN] bit must be set. This will keep the frequency meter on until the Fine Tune Lock Detect sequence has been ran. Note, to trigger the fine tune lock detect sequence, the TSM will also need to be programmed to insert a lock detect at the end of warm-up.

These bits select either a 4us or an 8us count period and if the frequency counter is outside of the threshold selected by the Frequency Target Fail Threshold registers, FTF_RX_THRSH and FTF_TX_THRSH, then the Frequency Target Fail Flag bit (FTFF) will be set.

The FTFF bit is cleared by writing a 1 to it

The FT_FAIL bit always shows the real time status of the latest Fine Tune Frequency Measurement. If the absolute count difference was out of the range selected by Frequency Target Fail Threshold registers, then this bit will be set.

55.4.7.3.7 Memory Map and Register Definition

The PLL Digital memory map and detailed descriptions of all its registers are as follows.

55.4.7.3.7.1 XCVR_PLL_DIG register descriptions

55.4.7.3.7.1.1 XCVR_PLL_DIG_ADDR memory map

XCVR_PLL_DIG base address: 48A0_7300h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL HPM Analog Bump Control (HPM_BUMP)	32	RW	0044_1012h
4h	PLL Modulation Control (MOD_CTRL)	32	RW	0000_0000h
8h	PLL Channel Mapping (CHAN_MAP)	32	RW	0000_0000h
10h	PLL Channel Mapping Extended (CHAN_MAP_EXT)	32	RW	0010_0000h
18h	PLL Lock Detect Control (LOCK_DETECT)	32	RW	0060_6800h
1Ch	PLL High Port Modulator Control (HPM_CTRL)	32	RW	0084_0000h
20h	PLL High Port Calibration Control (HPMCAL_CTRL)	32	RW	0000_2221h
24h	PLL High Port Calibration Result 1 (HPM_CAL1)	32	R	4430_0000h
28h	PLL High Port Calibration Result 2 (HPM_CAL2)	32	R	0210_0000h
2Ch	PLL High Port Sigma Delta Results (HPM_SDM_RES)	32	RW	0100_0000h
30h	PLL Low Port Modulator Control (LPM_CTRL)	32	RW	0800_0800h
34h	PLL Low Port Sigma Delta Control 1 (LPM_SDM_CTRL1)	32	RW	0000_0000h
38h	PLL Low Port Sigma Delta Control 2 (LPM_SDM_CTRL2)	32	RW	0200_0000h
3Ch	PLL Low Port Sigma Delta Control 3 (LPM_SDM_CTRL3)	32	RW	0400_0000h
40h	PLL Low Port Sigma Delta Result 1 (LPM_SDM_RES1)	32	R	0E20_0000h
44h	PLL Low Port Sigma Delta Result 2 (LPM_SDM_RES2)	32	R	0400_0000h
48h	PLL Delay Matching (DELAY_MATCH)	32	RW	0000_0204h
4Ch	Tuning Cap Settings in Transmit Mode (TUNING_CAP_TX_CTRL)	32	RW	006D_B6DBh
50h	Tuning Cap Settings in Receive Mode (TUNING_CAP_RX_CTRL)	32	RW	006D_B6DBh
58h	Max Transmit Frequency For TX Configuration 1 (MAX_TX_CFG1_FREQ)	32	RW	0000_0FFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
5Ch	PLL Coarse Tune Control (CTUNE_CTRL)	32	RW	0000_0000h
60h	PLL Data Rate Override Control (DATA_RATE_OVRD_CTRL1)	32	RW	0000_0188h
64h	PLL Data Rate Override Control (DATA_RATE_OVRD_CTRL2)	32	RW	0018_0000h
84h	PLL Coarse Tune Results (CTUNE_RES)	32	R	0000_0000h
A0h	PLL HPM Calibration Timing Attributes (HPM_CAL_TIMING)	32	RW	0000_0000h
A4h	PLL Offset Control (PLL_OFFSET_CTRL)	32	RW	0000_0000h
A8h	PLL Data Rate Switch Control (PLL_DATARATE_CTRL)	32	RW	0000_6612h

55.4.7.3.7.1.2 PLL HPM Analog Bump Control (HPM_BUMP)

Offset

Register	Offset
HPM_BUMP	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									PLL_VCO_TRIM_KVM_CAL			0	PLL_VCO_TRIM_KVM_TX		
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		HPM_FDB_RES_CAL		0		HPM_FDB_RES_TX		0	HPM_VCM_CAL			0	HPM_VCM_TX		
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0

Fields

Field	Function
31-23 —	Reserved
22-20 PLL_VCO_TRIM_KVM_CAL	reg_vco_trim_kvm_dig[2:0] for calibration kv mod control for for calibration (for modulation index control).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - 10MHz/V 100b - 20MHz/V 110b - 30MHz/V 111b - 40MHz/V
19 —	Reserved
18-16 PLL_VCO_TRIM_KVM_TX	reg_vco_trim_kvm_dig[2:0] for transmitt kv mod control for transmitt (for modulation index control). 000b - 10MHz/V 100b - 20MHz/V 110b - 30MHz/V 111b - 40MHz/V
15-14 —	Reserved
13-12 HPM_FDB_RES_CAL	a_ip_2p4ghz_reg_dac_trim_rfbk_dig[1:0] during Calibration This is the value applied to the a_ip_2p4ghz_reg_dac_trim_rfbk_dig[1:0] port during High Port Calibration. This register sets the HPM DAC feedback resistor to increase the modulation gain during calibration. 00b - 38.0k (1.0) 01b - 76.0k (0.5) 10b - 32.5k (1.14) 11b - 25.3k (1.4)
11-10 —	Reserved
9-8 HPM_FDB_RES_TX	a_ip_2p4ghz_reg_dac_trim_rfbk_dig[1:0] during Transmission This is the value applied to the a_ip_2p4ghz_reg_dac_trim_rfbk_dig[1:0] port during Radio Transmissions. This register sets the HPM DAC feedback resistor during transmissions. 00b - 38.0k (1.0) 01b - 76.0k (0.5) 10b - 32.5k (1.14) 11b - 25.3k (1.4)
7	Reserved

Table continues on the next page...

Table continued from the previous page...

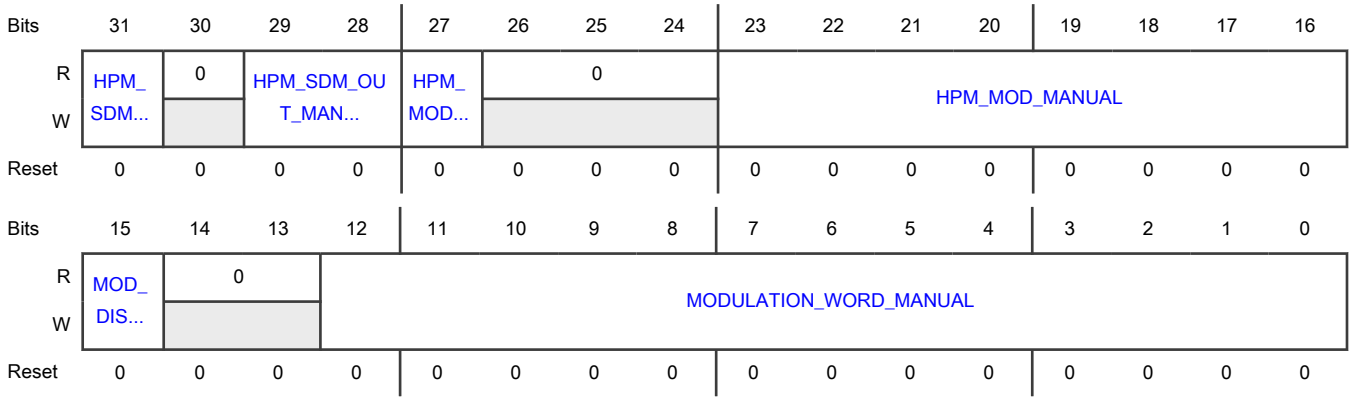
Field	Function
—	
6-4 HPM_VCM_CAL	a_ip_2p4ghz_reg_dac_trim_vcm_dig[2:0] during Calibration This is the value applied to the a_ip_2p4ghz_reg_dac_trim_vcm_dig[2:0] port during High Port Calibration. This field sets the HPM DAC Op-Amp reference voltage during calibration. Trim of amplifier reference voltage (amplifier output voltage for dac_code = 127). 000b - 0.120 (0.122) 001b - 0.153 (0.189) 010b - 0.182 (0.247) 011b - 0.209 (0.300) 100b - 0.234 (0.348) 101b - 0.258 (0.393) 110b - 0.279 (0.434) 111b - 0.318 (0.509)
3 —	Reserved
2-0 HPM_VCM_TX	a_ip_2p4ghz_reg_dac_trim_vcm_dig[2:0] during Transmission This is the value applied to the a_ip_2p4ghz_reg_dac_trim_vcm_dig[2:0] port during Radio Transmissions. This register sets the HPM DAC Op-Amp reference voltage during transmissions. Trim of amplifier reference voltage (amplifier output voltage for dac_code = 127). 000b - 0.120 (0.122) 001b - 0.153 (0.189) 010b - 0.182 (0.247) 011b - 0.209 (0.300) 100b - 0.234 (0.348) 101b - 0.258 (0.393) 110b - 0.279 (0.434) 111b - 0.318 (0.509)

55.4.7.3.7.1.3 PLL Modulation Control (MOD_CTRL)

Offset

Register	Offset
MOD_CTRL	4h

Diagram



Fields

Field	Function
31 HPM_SDM_OUT_DISABLE	Disable HPM SDM out If this bit is set, the High Port Sigma Delta Modulator output is disabled, and the High Port Fractional value applied to the VCO comes from the HPM_SDM_OUT_MANUAL register.
30 —	Reserved
29-28 HPM_SDM_OUT_MANUAL	Manual HPM SDM out If HPM_SDM_OUT_DISABLE is set, this register is the Fractional value that is applied to the VCO High Port.
27 HPM_MOD_DISABLE	Disable HPM Modulation If this bit is set, the High Port Modulation is disabled, and the High Port Modulation value applied to the VCO comes from the HPM_MOD_MANUAL register.
26-24 —	Reserved
23-16 HPM_MOD_MANUAL	Manual HPM Modulation If HPM_MOD_DISABLE is set, this register is the modulation value that is applied to the VCO High Port.
15 MOD_DISABLE	Disable Modulation Word If this bit is set, any modulation from the TX Digital is disabled and the source of the Baseband Frequency Word is the MODULATION_WORD_MANUAL register.
14-13 —	Reserved
12-0	Manual Modulation Word

Table continues on the next page...

Table continued from the previous page...

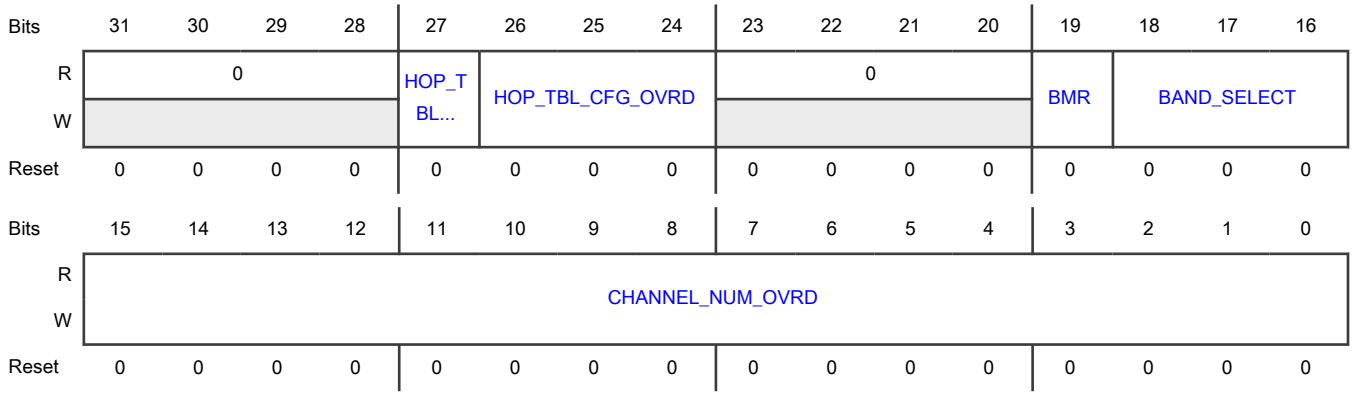
Field	Function
MODULATION_WORD_MANUAL	If MOD_DISABLE is set, the signed 12 bit value that is represented by this register is the Baseband Frequency Word.

55.4.7.3.7.1.4 PLL Channel Mapping (CHAN_MAP)

Offset

Register	Offset
CHAN_MAP	8h

Diagram



Fields

Field	Function
31-28 —	Reserved
27 HOP_TBL_CFG_OVRD_EN	Hop Table Configuration Override Enable If this bit is set then the HOP_TBL_CFG_OVRD and CHANNEL_NUM_OVRD Registers will be used to control the Coarse Tune, Channel Number, and Hop Table for the PLL Carrier Frequency Selection.
26-24 HOP_TBL_CFG_OVRD	Hop Table Configuration Override If the HOP_TBL_CFG_OVRD_EN bit is set, then the CHANNEL_NUM_OVRD Register will be used to control the Hop Table for the PLL Carrier Frequency Selection. 000b-001b - CHANNEL_NUM_OVRD[6:0] is used as the mapped channel number. CHANNEL_NUM_OVRD[15:7] is unused.

Table continues on the next page...

Table continued from the previous page...

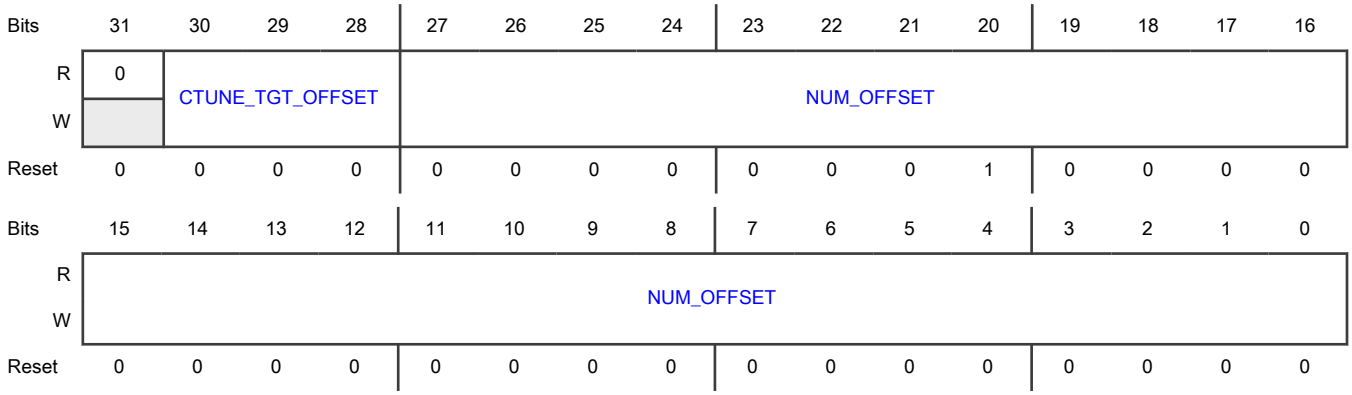
Field	Function
	010b - CHANNEL_NUM_OVRD[15:7] is signed Numerator offset to CHANNEL_NUM_OVRD[6:0] mapped channel number 011b - CHANNEL_NUM_OVRD[15:1] is selected as the signed Numerator, CHANNEL_NUM_OVRD[0] is integer selection
23-20 —	Reserved
19 BMR	Bluetooth Low Energy MBAN Channel Remap This bit controls the mapping of Bluetooth Low Energy channel 39 in Radio Protocol 2, Bluetooth Low Energy overlap MBAN mode. 0b - Bluetooth Low Energy channel 39 is mapped to Bluetooth Low Energy channel 39, 2.480 GHz 1b - Bluetooth Low Energy channel 39 is mapped to MBAN channel 39, 2.399 GHz
18-16 BAND_SELECT	Channel Mapping Band Select This register field is used to map the PLL input channel to a specific frequency band. Note, all link layer channel selections are MUXed into a single PLL input. This register field will select how the input channel is mapped to a specific carrier frequency. See section Carrier Frequency Tuning for a table outlining the channel vs. frequency relationship for all supported protocols. 000b - Bluetooth Low Energy 001b - Bluetooth Low Energy in MBAN 010b - Bluetooth Low Energy overlap MBAN 011b - RESERVED 100b - IEEE 802.15.4 O-QPSK PHY in ISM band 101b - IEEE 802.15.4 O-QPSK PHY in MBAN band 110b-111b - Radio Channels 0-127 selectable
15-0 CHANNEL_NUM_OVRD	Channel Selection Override When the HOP_TBL_CFG_OVRD_EN register field is set to 0b1, the CHANNEL_NUM_OVRD will be selected for the input PLL channel. The HOP_TBL_CFG_OVRD register field is used to decode CHANNEL_NUM_OVRD to determine the final PLL locking frequency.

55.4.7.3.7.1.5 PLL Channel Mapping Extended (CHAN_MAP_EXT)

Offset

Register	Offset
CHAN_MAP_EXT	10h

Diagram



Fields

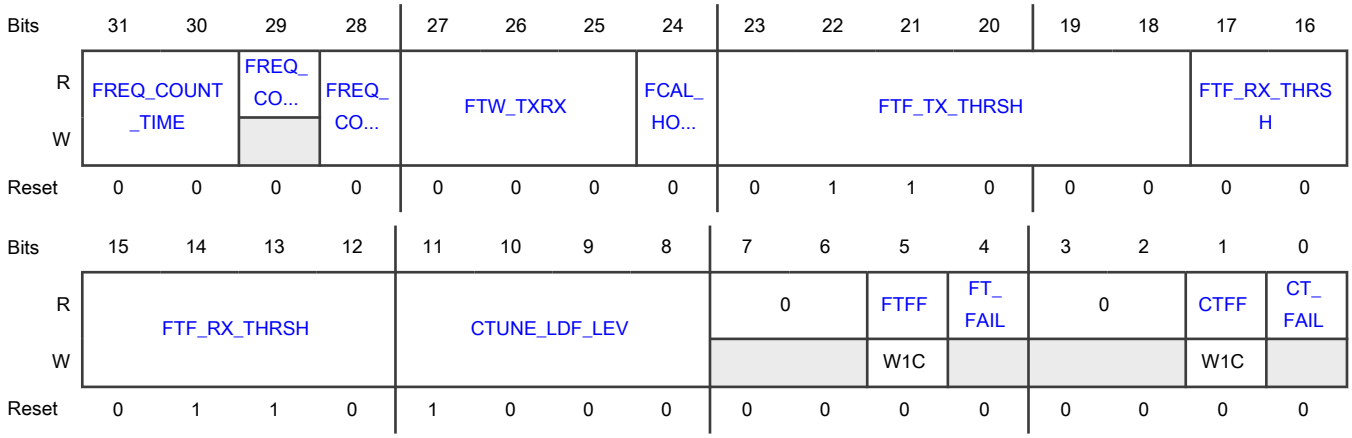
Field	Function
31 —	Reserved
30-28 CTUNE_TGT_OFFSET	Coarse Tune Target Frequency Offset In RX Mode, the PLL Coarse Tune Target Frequency will be subtracted from the CTUNE_TGT_OFFSET to form the final CTUNE Target Frequency. Note, if this register field is left as zero, then it will have no effect. A value should only written here if the NUM_OFFSET is large enough to require a CTUNE modification. This value should be written as a 3-bit two's compliment number. Register bit 30 is the CTUNE_TGT_OFFSET sign bit. The unit for CTUNE_TGT_OFFSET is MHz. This register can provide a maximum offset of +3 MHz or -4 MHz from the internally calculated CTUNE Target.
27-0 NUM_OFFSET	Numerator Offset The NUM_OFFSET register field represents the IF frequency difference between RX and TX. To achieve this offset, the NUM_OFFSET value will be subtracted from the internally derived PLL Numerator value. Note that the numerator LSB will change the PLL target frequency according to the table in section Low Port Modulation . This value represents a multiple of the pll minimum frequency step size. In RX Mode, this register holds a signed two's compliment offset which will be subtracted from the PLL Numerator value calculated using the PLL internal channel mapping logic. Warm-up in RX mode will automatically subtract this value from the PLL Numerator. If NUM_OFFSET == 0 then no frequency offset will exist between Tx to Rx. Bit 27 of this register provides the sign bit for RX_NUM_OFSET. The largest positive number supported is 0x3FF_FFFF. The largest negative number supported is 0xC00_0000. Note, manual override mode won't use this register field.

55.4.7.3.7.1.6 PLL Lock Detect Control (LOCK_DETECT)

Offset

Register	Offset
LOCK_DETECT	18h

Diagram



Fields

Field	Function
31-30 FREQ_COUNT_TIME	Frequency Meter Count Time This is the length of time that the Frequency Meter will count VCO clocks. 00b - 800 us 01b - 25 us 10b - 50 us 11b - 100 us
29 FREQ_COUNT_FINISHED	Frequency Meter has finished the Count Time This bit is set when the FREQ_COUNT_TIME value is reached, it is cleared when FREQ_COUNT_GO is cleared.
28 FREQ_COUNT_GO	Start the Frequency Meter The Frequency Meter starts when this bit is set and runs until the FREQ_COUNT_TIME value is reached. The bit should not be cleared until after the counting is finished. After the Counting Time finishes, the FREQ_COUNT_FINISHED bit will read as a one. Then the measured frequency can be calculated by reading the frequency counts in the DFT_FREQ_COUNTER register (PLL_DFT_OCR_1[DFT_FREQ_COUNTER]) and dividing by the FREQ_COUNT_TIME. Note that the counting is done at the VCO frequency which is 2X the Carrier Frequency
27-25 FTW_TXRX	TX and RX Frequency Target Window time select In Radio Receive Mode, register field is decoded to select the length of time to count for the estimation of PLL lock frequency, which is compared with the Frequency Target Window, we refer to this decoding as FTW_RX. In Radio Transmit Mode, register field is decoded to select the length of time to count for the estimation of PLL lock frequency, which is compared with the Frequency Target Window, we refer to this decoding as FTW_TX.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - FTW_TX = 4us ; FTW_RX = 4us 001b - FTW_TX = 4us ; FTW_RX = 8us 010b - FTW_TX = 8us ; FTW_RX = 4us 011b - FTW_TX = 8us ; FTW_RX = 8us 100b - FTW_TX = 16us ; FTW_RX = 16us 101b - FTW_TX = 16us ; FTW_RX = 32us 110b - FTW_TX = 32us ; FTW_RX = 16us 111b - FTW_TX = 32us ; FTW_RX = 32us
24 FCAL_HOLD_EN	Frequency Counter Hold Enable This bit controls if the pll ripple counter stays on after the PLL warm-up sequence (TX or RX). By default (FCAL_HOLD_EN==0b0), the ripple counter will be turned off after completing CTUNE in RX mode and HPM CAL in TX mode. If the ripple counter is turned off after the standard warm-up sequence, then a lock detect sequence cannot be inserted during warm-up. To support this lock detect sequence, FCAL_HOLD_EN can be asserted to ensure the PLL ripple counter is maintained in an on state after the default CTUNE or HPM CAL state. In addition, the TSM should be programmed to run the lock detect sequence at the appropriate time during warmup. 0b - The frequency counter is turned off after CTUNE (RX Mode) or HPM CAL (TX Mode) 1b - The frequency counter is held on after CTUNE (RX Mode) or HPM CAL (TX Mode) for an optional lock detect sequence.
23-18 FTF_TX_THRSH	TX Frequency Target Fail Threshold In Radio Transmit Mode, the FT_FAIL and FTFF bits will be set after the Frequency Target Count completes if the absolute value of the count difference from the frequency target count is greater than this register value.
17-12 FTF_RX_THRSH	RX Frequency Target Fail Threshold In Radio Receive Mode, the FT_FAIL and FTFF bits will be set after the Frequency Target Count completes if the absolute value of the count difference from the frequency target count is greater than this register value.
11-8 CTUNE_LDF_LEV	CTUNE Lock Detect Fail Level The CT_FAIL and CTFF bits will be set after Coarse Tune Calibration completes if the absolute value of the Coarse Tune best count difference (CTUNE_BEST_DIFF in the PLL_CTUNE_RESULTS register) is greater than this register value.
7-6 —	Reserved
5 FTFF	Frequency Target Failure Flag This bit is set when FT_FAIL is first set, and this bit is cleared by writing a 1 to it.
4	Real time status of Frequency Target Failure

Table continues on the next page...

Table continued from the previous page...

Field	Function
FT_FAIL	If the Frequency Target Count has completed and the count was out of the range selected by FTW_TX or FTW_RX, then this bit will be set.
3-2 —	Reserved
1 CTFF	CTUNE Failure Flag, held until cleared This bit is set when CT_FAIL is first set, and this bit is cleared by writing a 1 to it.
0 CT_FAIL	Real time status of Coarse Tune Fail signal If the Coarse Tune Calibration has completed and the best count difference is out of the range selected by CTUNE_LDF_LEV, then this bit will be set.

55.4.7.3.7.1.7 PLL High Port Modulator Control (HPM_CTRL)

Offset

Register	Offset
HPM_CTRL	1Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HPM_MOD...	HPM_CAL_TIME		HPM_CAL...	HPM_INT...	HPM_SCALE			HPM_DTH...	0		HPM_DTH...	RX_HPM_...	HPM_LFSR_SIZE		
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HPM_SDM...	HPM_SDM...	HPFF	HPM_CLK...	0		HPM_SDM_IN_MANUAL									
W			W1C													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 HPM_MOD_IN_INVERT	Invert High Port Modulation If this bit is set then the High Port Modulation Word is inverted before it is multiplied and split into Integer and Fractional values.
30-29	High Port Modulation Calibration Time

Table continues on the next page...

Table continued from the previous page...

Field	Function
HPM_CAL_TIME	This bit selects the length of time to count during HPM Calibration and is used by the math implemented in the HPM Calibration Factor lookup table. 00b - 25 us 01b - 50 us 10b - 100 us 11b - N/A
28 HPM_CAL_INVERT	Invert High Port Modulator Calibration If this bit is set then the order of the High Port Calibration is reversed in order to get a positive count difference between the Maximum and Minimum settings of the HPM DAC.
27 HPM_INTEGER_INVERT	Invert High Port Modulation Integer If this bit is set then the High Port Modulation Integer value, after the multiply and split into Integer and Fractional values, will be inverted before it is applied to the VCO High Port DAC Array.
26-24 HPM_SCALE	High Port Modulation Scale This register controls the scaling of the High Port Modulation word. 000b - No Scaling 001b - Divide by 2 010b - Multiply by 2 011b - Multiply by 4 100b - Divide by 4 101b - Multiply by 8 110b - Divide by 8 111b - N/A
23 HPM_DTH_EN	Dither Enable for HPM LFSR If this bit is set, the High Port Fraction will be Dithered by the High Port LFSR before it is applied to the High Port SDM.
22-21 —	Reserved
20 HPM_DTH_SCL	HPM Dither Scale If this bit is set, the LFSR dithering of the High Port Fraction will be multiplied by 2.
19 RX_HPM_CAL_EN	Receive HPM Calibration Enable If this bit is set, HPM Calibration takes place in receive mode.
18-16	HPM LFSR Length

Table continues on the next page...

Table continued from the previous page...

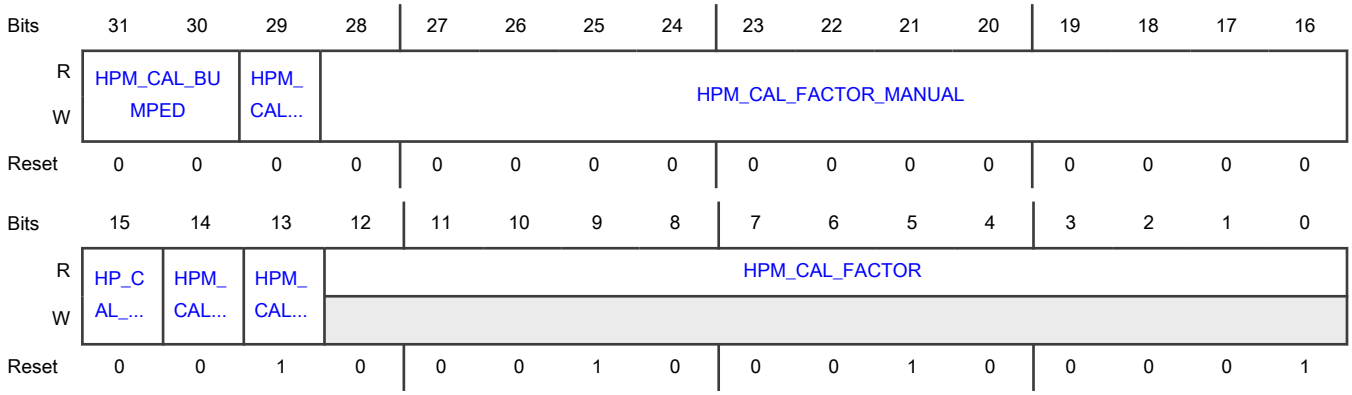
Field	Function
HPM_LFSR_SIZE	This register selects the length of the HPM LFSR and the associated LFSR Tap Mask 000b - LFSR 9, tap mask 100010000 001b - LFSR 10, tap mask 1001000000 010b - LFSR 11, tap mask 11101000000 011b - LFSR 13, tap mask 1101100000000 100b - LFSR 15, tap mask 111010000000000 101b - LFSR 17, tap mask 11110000000000000 110b - Reserved 111b - Reserved
15 HPM_SDM_IN_DISABLE	Disable HPM SDM Input If this bit is set, the Fractional portion of the High Port Modulation to the High Port SDM is disabled, and the High Port SDM input comes from the HPM_SDM_IN_MANUAL register.
14 HPM_SDM_OUT_INVERT	Invert HPM SDM Output If this bit is set the High Port SDM result will be Inverted before it is applied to the VCO High Port DAC Array.
13 HPFF	HPM SDM Invalid Flag This bit is set if the High Port Sigma Delta Modulator output is invalid due to an error in the fraction applied, and this bit is cleared by writing a 1 to it.
12 HPM_CLK_CONFIG	HPM Clock Config When this bit is set, the HPM modulation path including the HPM_SDM operate at the CLK_SDM clock (which is the 2x reference clock derived from PLL loop divider). By default they operate at the reference clock.
11-10 —	Reserved
9-0 HPM_SDM_IN_MANUAL	Manual High Port SDM Fractional value If HPM_SDM_IN_DISABLE is set, this register is the value that is applied as the Fractional value to the input of the High Port SDM.

55.4.7.3.7.1.8 PLL High Port Calibration Control (HPMCAL_CTRL)

Offset

Register	Offset
HPMCAL_CTRL	20h

Diagram



Fields

Field	Function
31-30 HPM_CAL_BUMPED	HPM_CAL_BUMPED This field adjusts the High Port Calibration Value; the default of 0 assumes that the HPM DAC range during the HPM Calibration sequence is equivalent to the range during run mode. 00b - No calibration boost 01b - x2 10b - x4 11b - x8
29 HPM_CAL_SKIP	HPM_CAL_SKIP This bit controls if the high port calibration routine is run during the warm-up sequence. When this bit is set to 1, the warm-up sequence will commence after coarse tune calibration. Note, disabling the coarse tune calibration sequence won't disable high port modulation. The user should set a manual override using either HPM_SDM_IN_DISABLE or HPM_CAL_DISABLE as needed for the application. As an example, the programmer can set HPM_CAL_SKIP = 0b1 and HPM_CAL_DISABLE = 0b1, then provide a manual override of the CAL Factor from external memory. If the CAL Factor is already known for the desired radio configuration, then this will result in a shorter warm-up time. Note that the TSM warm-up sequence will need to be modified to take into account the shorter warm-up time. In addition, if the desired modulation rate only requires single port modulation, setting HPM_CAL_DISABLE = 0b1 will allow shorter warmup time. In this case, the high port modulation path should be disabled by setting a manual override with HPM_MOD_DISABLE and HPM_MOD_MANUAL to half-rail.
28-16 HPM_CAL_FACTOR_MANUAL	Manual HPM Calibration Factor If HP_CAL_DISABLE is set, this register value is the unsigned 13 bit value used by the HPM Multiplier.
15 HP_CAL_DISABLE	Disable HPM Manual Calibration If this bit is set, the lookup table value for the HPM Calibration Factor is overridden by the HPM_CAL_FACTOR_MANUAL register.
14	HPM_CAL_COUNT_SCALE

Table continues on the next page...

Table continued from the previous page...

Field	Function
HPM_CAL_COUNT_SCALE	This bit adjusts the High Port Calibration Value; the default of 0 assumes that the VCO 2x clock (4.8 GHz) has been used by the Frequency Meter counter during the High Port Calibration sequence. If the LO clock (2.4 GHz) has been used instead, then the calibration math can be adjusted by setting this bit
13 HPM_CAL_ARRAY_SIZE	High Port Modulation Calibration Array Size This bit selects the size of the array to be used by the math implemented in the HPM Calibration Factor lookup table. 0b - 128 1b - 256
12-0 HPM_CAL_FACTOR	High Port Modulation Calibration Factor This is the value currently being used by High Port Modulation Multiplier.

55.4.7.3.7.1.9 PLL High Port Calibration Result 1 (HPM_CAL1)

Offset

Register	Offset
HPM_CAL1	24h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0		0		0		0		0	HPM_COUNT_1		
W																
Reset	0	1	0	0	0	1	0	0	0	0	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HPM_COUNT_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 —	Reserved
26-24 —	Reserved
23 —	Reserved
22-20 —	Reserved
19 —	Reserved
18-0 HPM_COUNT_1	High Port Modulation Counter Value 1 This is the Frequency Counter value used as the Maximum HPM Calibration Count.

55.4.7.3.7.1.10 PLL High Port Calibration Result 2 (HPM_CAL2)

Offset

Register	Offset
HPM_CAL2	28h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				0			0	0	HPM_COUNT_2		
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HPM_COUNT_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

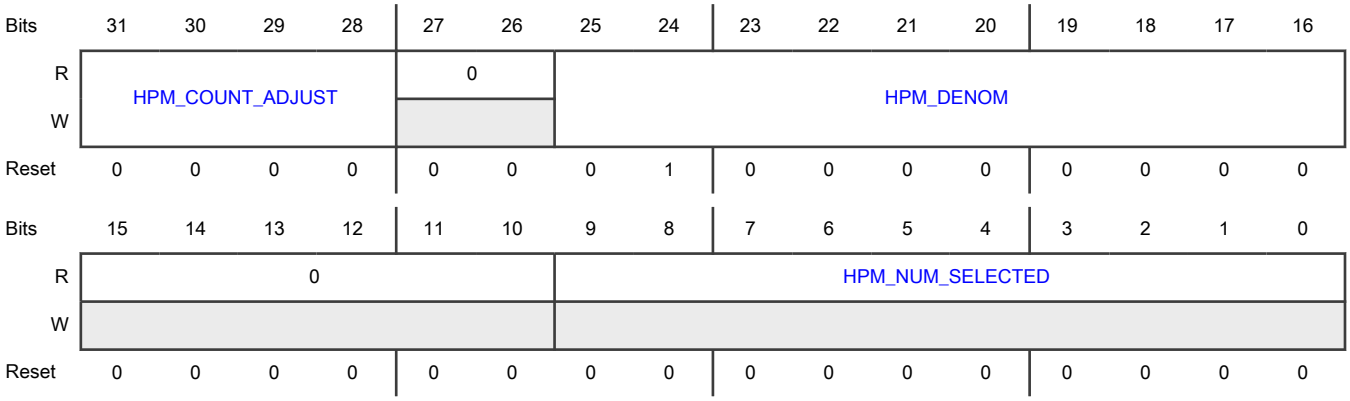
Field	Function
31-29 —	Reserved
28-24 —	Reserved
23-21 —	Reserved
20 —	Reserved
19 —	Reserved
18-0 HPM_COUNT_2	High Port Modulation Counter Value 2 This is the Frequency Counter value used as the Minimum HPM Calibration Count.

55.4.7.3.7.1.11 PLL High Port Sigma Delta Results (HPM_SDM_RES)

Offset

Register	Offset
HPM_SDM_RES	2Ch

Diagram



Fields

Field	Function
31-28 HPM_COUNT_ADJUST	HPM_COUNT_ADJUST This register represents a signed three bit value that is used to adjust the High Port Calibration Frequency Count Difference. The range of adjustment is -8 to +7, applied to the difference between Count 1 and Count 2.
27-26 —	Reserved
25-16 HPM_DENOM	High Port Modulator SDM Denominator This is the denominator that is currently being applied to the High Port Sigma Delta Modulator, for valid Sigma Delta operation the resulting fraction NUM/DENOM must be in the range -0.6 to +0.6
15-10 —	Reserved
9-0 HPM_NUM_SELECTED	High Port Modulator SDM Numerator This is the numerator that is currently being applied to the High Port Sigma Delta Modulator, for valid Sigma Delta operation the resulting fraction NUM/DENOM must be in the range -0.6 to +0.6

55.4.7.3.7.1.12 PLL Low Port Modulator Control (LPM_CTRL)

Offset

Register	Offset
LPM_CTRL	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LPM_SDM...	0			LPM_SCALE				LPM_D_O...	LPM_D_C...	0		LPM_DTH_SCL			
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPM_DIS...	LPM_SDM...	LPFF	PLL_LD_...	HPM_CAL_SCALE				0				PLL_LD_MANUAL			
W			W1C													
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LPM_SDM_USE_NEG	Use the Negedge of the Sigma Delta clock If this bit is set then the negative edge of the Sigma Delta clock is used to launch the Low Port Modulation word to the VCO Loop Divider.
30-28 —	Reserved
27-24 LPM_SCALE	LPM Scale Factor This register controls the scaling of the Baseband Frequency Word and is used to match the Modulation Frequency Deviation required to the Low Port Sigma Delta Modulator LSB size in Hz. 0000b - No Scaling 0001b - Multiply by 2 0010b - Multiply by 4 0011b - Multiply by 8 0100b - Multiply by 16 0101b - Multiply by 32 0110b - Multiply by 64 0111b - Multiply by 128 1000b - Multiply by 256 1001b - Multiply by 512 1010b - Multiply by 1024 1011b - Multiply by 2048 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
23 LPM_D_OVRD	LPM Dither Override Mode Select When this bit is set, the Scaled Baseband Frequency Word applied to the Low Port Sigma Delta Modulator will be dithered if LPM_D_CTRL is set, and not dithered if LPM_D_CTRL is cleared. If this bit is cleared, then the LPM Numerator will be dithered in Radio Receive mode, and also in Radio Transmit mode when the LPM Numerator approaches an Integer value in order to preserve the validity of the Sigma Delta Modulator output.
22 LPM_D_CTRL	LPM Dither Control in Override Mode If LPM_D_OVRD is set, this bit turns LPM Dithering on and off.
21-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-16 LPM_DTH_SCL	LPM Dither Scale This register controls the scale of the Dithering added to the Scaled Baseband Frequency Word before it is applied to the Low Port Sigma Delta Modulator as the LPM Numerator. The unit for the ranges shown below is the LP SDM LSB in Hz. 0000b - Reserved 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - -128 to 96 0110b - -256 to 192 0111b - -512 to 384 1000b - -1024 to 768 1001b - -2048 to 1536 1010b - -4096 to 3072 1011b - -8192 to 6144 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
15 LPM_DISABLE	Disable LPM SDM This bit controls the Modulation of the Low Port Sigma Delta. If this bit is set, the Low Port Sigma Delta Modulator will be active and control the PLL to maintain a steady frequency based on the current Integer, Numerator, and Denominator values that are being applied. No Modulation or Dithering will be added to the steady frequency result.
14 LPM_SDM_INV	Invert LPM SDM If this bit is set the Scaled Baseband Frequency Word, including any Dithering, will be Inverted before it is applied to the Low Port Sigma Delta Modulator.
13 LPFF	LPM SDM Invalid Flag This bit is set if the Low Port Sigma Delta Modulator output is invalid due to an error in the fraction applied, and this bit is cleared by writing a 1 to it.
12	Disable PLL Loop Divider

Table continues on the next page...

Table continued from the previous page...

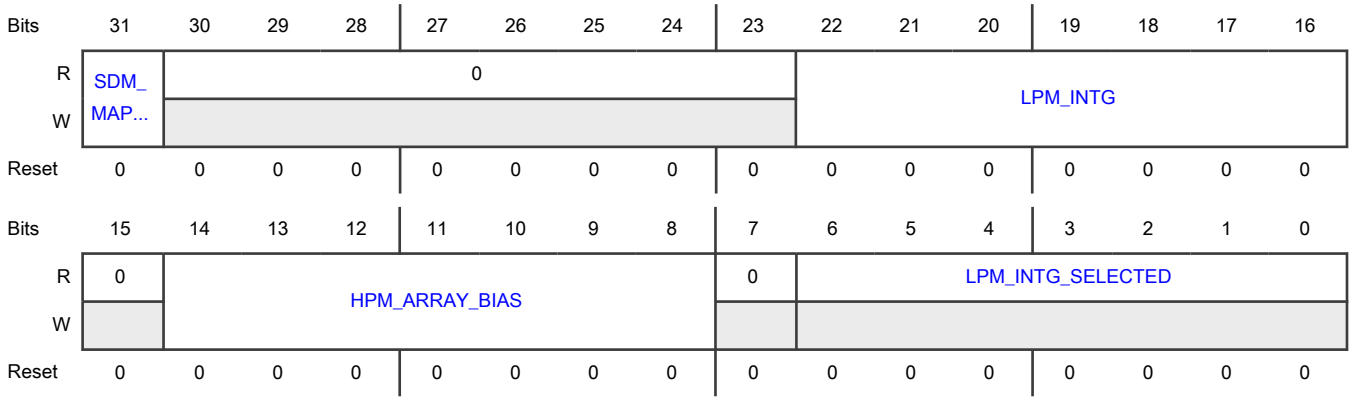
Field	Function
PLL_LD_DISABLE	If this bit is set, the Low Port Sigma Delta Modulator output is disabled, and the PLL Loop Divider value applied to the PLL comes from the PLL_LD_MANUAL register.
11-8 HPM_CAL_SCALE	High Port Calibration Word Scaling The calibration factor LPM_LSB/HPM_LSB will be scaled using the following bitfield settings. This bit field should be adjusted in conjunction with LPM_SCALE according to the modulation range. The default setting is no scaling factor applied. 0000b-0010b - No Scaling 0011b - Divide by 32 0100b - Divide by 16 0101b - Divide by 8 0110b - Divide by 4 0111b - Divide by 2 1000b - No Scaling 1001b - Multiply by 2 1010b - Multiply by 4 1011b - Multiply by 8 1011b-1111b - No Scaling
7-5 —	Reserved
4-0 PLL_LD_MANUAL	Manual PLL Loop Divider value If PLL_LD_DISABLE is set, this register is the value that is applied to the PLL Loop Divider.

55.4.7.3.7.1.13 PLL Low Port Sigma Delta Control 1 (LPM_SDM_CTRL1)

Offset

Register	Offset
LPM_SDM_CTRL1	34h

Diagram



Fields

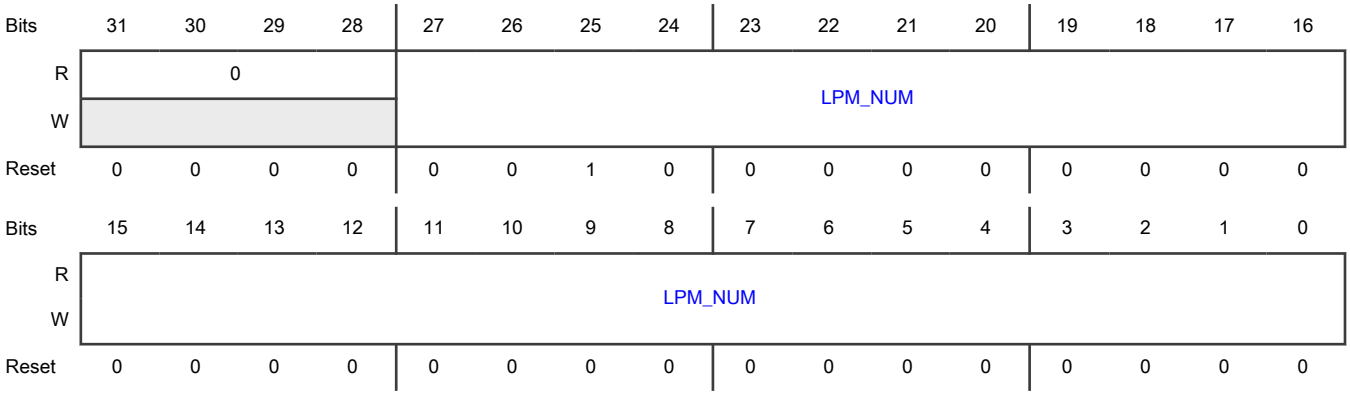
Field	Function
31 SDM_MAP_DISABLE	Disable SDM Mapping If this bit is set, the Low Port Sigma Delta Modulator internal frequency mapping based on Protocol specific channel numbers is disabled, and the Radio Channel Frequency is selected by setting the LPM_INTG, LPM_NUM, and LPM_DENOM registers to get a frequency that equals : ((Reference Clock Frequency x 2) x (LPM_INTG + (LPM_NUM / LPM_DENOM))
30-23 —	Reserved
22-16 LPM_INTG	Manual Low Port Modulation Integer Value If SDM_MAP_DISABLE is set, this register is the value that is applied to the Low Port Sigma Delta Modulator for the Integer, the nominal range is 36 to 39 in decimal.
15 —	Reserved
14-8 HPM_ARRAY_BIAS	Bias value for High Port DAC Array Midpoint The value of this register represents a signed six bit value that can be used to adjust the midpoint of the High Port Modulator DAC. The range of adjustment is -64 to +63, and the adjustment is made to the High Port Modulation Word before the multiply, and before the split into Integer and Fractional values. The range of adjustment in Hz is approximately +/- 62.5 kHz
7 —	Reserved
6-0 LPM_INTG_SELECTED	Low Port Modulation Integer Value Selected This shows the Integer value that is currently being applied to the Low Port Sigma Delta Modulator.

55.4.7.3.7.1.14 PLL Low Port Sigma Delta Control 2 (LPM_SDM_CTRL2)

Offset

Register	Offset
LPM_SDM_CTRL2	38h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0 LPM_NUM	Low Port Modulation Numerator If SDM_MAP_DISABLE is set, this register is the signed 27 bit value that is applied to the Low Port Sigma Delta Modulator for the Numerator. For valid Sigma Delta operation the resulting fraction NUM/DENOM must be in the range -0.6 to +0.6

55.4.7.3.7.1.15 PLL Low Port Sigma Delta Control 3 (LPM_SDM_CTRL3)

Offset

Register	Offset
LPM_SDM_CTRL3	3Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LPM_DENOM											
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPM_DENOM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 LPM_DENOM	Low Port Modulation Denominator If SDM_MAP_DISABLE is set, this register is the signed 27 bit value that is applied to the Low Port Sigma Delta Modulator for the Denominator. For valid Sigma Delta operation the resulting fraction NUM/DENOM must be in the range -0.6 to +0.6

55.4.7.3.7.1.16 PLL Low Port Sigma Delta Result 1 (LPM_SDM_RES1)

Offset

Register	Offset
LPM_SDM_RES1	40h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LPM_NUM_SELECTED											
W																
Reset	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPM_NUM_SELECTED															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 LPM_NUM_SELECTED	Low Port Modulation Numerator Applied This is the value that is currently being applied to the Low Port Sigma Delta Modulator, for valid Sigma Delta operation the resulting fraction NUM/DENOM must be in the range -0.6 to +0.6

55.4.7.3.7.1.17 PLL Low Port Sigma Delta Result 2 (LPM_SDM_RES2)

Offset

Register	Offset
LPM_SDM_RES2	44h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LPM_DENOM_SELECTED											
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPM_DENOM_SELECTED															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

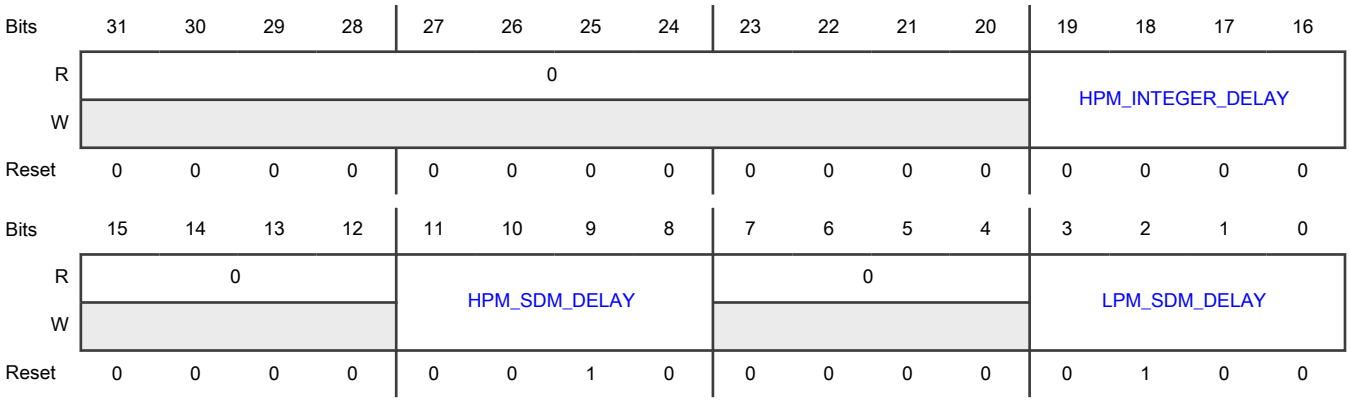
Field	Function
31-28 —	Reserved
27-0 LPM_DENOM_SELECTED	Low Port Modulation Denominator Selected This is the value that is currently being applied to the Low Port Sigma Delta Modulator, for valid Sigma Delta operation the resulting fraction NUM/DENOM must be in the range -0.6 to +0.6

55.4.7.3.7.1.18 PLL Delay Matching (DELAY_MATCH)

Offset

Register	Offset
DELAY_MATCH	48h

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 HPM_INTEGER_DELAY	High Port Integer Delay Matching This register selects the number of clock cycles of the PLL Sigma Delta Clock divided by 2 to delay the High Port Integer modulation of the VCO High Port DAC Array.
15-12 —	Reserved
11-8 HPM_SDM_DELAY	High Port SDM Delay Matching This register selects the number of clock cycles of the PLL Sigma Delta Clock divided by 2 to delay the High Port Sigma Delta modulation of the VCO High Port Fraction. Note that the High Port SDM is clocked by the PLL Sigma Delta Clock but the modulation is based on a divide by 2 version of this same clock.
7-4 —	Reserved
3-0 LPM_SDM_DELAY	Low Port SDM Delay Matching This register selects the number of clock cycles of the PLL Sigma Delta Clock to delay the Low Port Sigma Delta modulation of the PLL Loop Divider.

55.4.7.3.7.1.19 Tuning Cap Settings in Transmit Mode (TUNING_CAP_TX_CTRL)

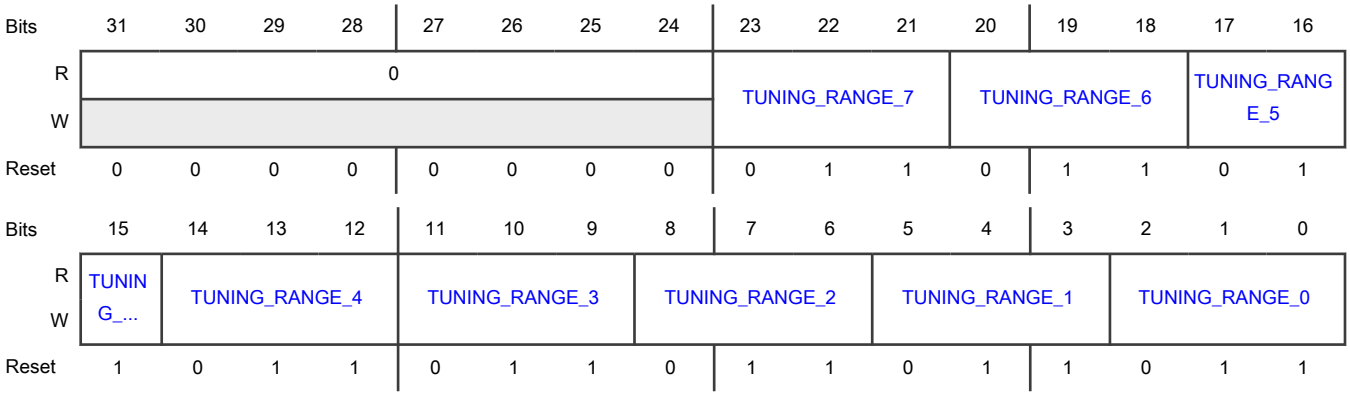
Offset

Register	Offset
TUNING_CAP_TX_CTRL	4Ch

Function

Tuning Cap Settings in Transmit Mode. This register contains the Cap Settings for Eight Tuning Ranges.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-21 TUNING_RANGE_7	Tuning Range 7 Radio channel above 2465 MHz.
20-18 TUNING_RANGE_6	Tuning Range 6 Radio channel between 2450 and 2465 MHz.
17-15 TUNING_RANGE_5	Tuning Range 5 Radio channel between 2435 and 2450 MHz.
14-12 TUNING_RANGE_4	Tuning Range 4 Radio channel between 2420 and 2435 MHz.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-9 TUNING_RANGE_3	Tuning Range 3 Radio channel between 2405 and 2420 MHz.
8-6 TUNING_RANGE_2	Tuning Range 2 Radio channel between 2390 and 2405 MHz.
5-3 TUNING_RANGE_1	Tuning Range 1 Radio channel between 2375 and 2390 MHz.
2-0 TUNING_RANGE_0	Tuning Range 0 Radio channel less than or equal to 2375 MHz.

55.4.7.3.7.1.20 Tuning Cap Settings in Receive Mode (TUNING_CAP_RX_CTRL)

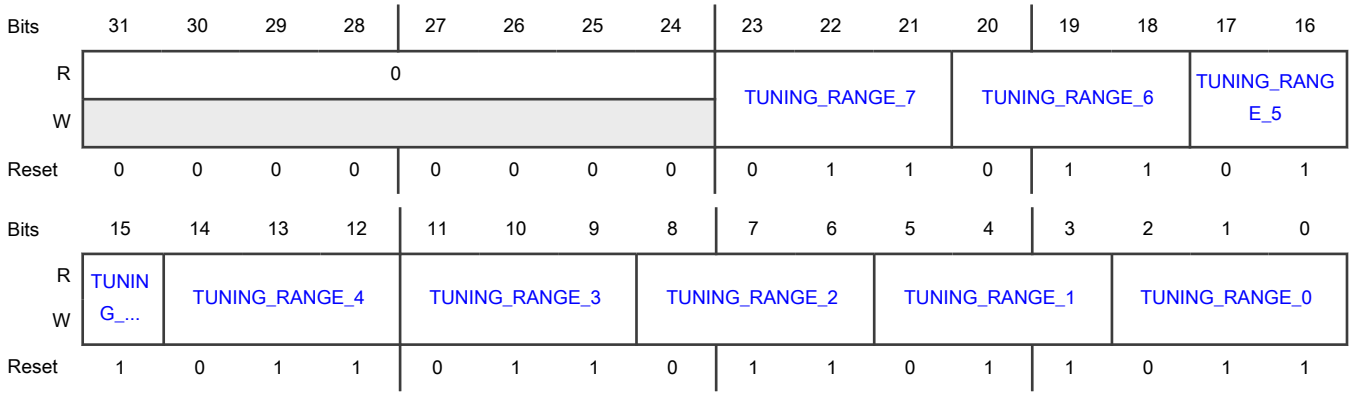
Offset

Register	Offset
TUNING_CAP_RX_CTRL	50h

Function

Tuning Cap Settings in Receive Mode. This register contains the Cap Settings for Eight Tuning Ranges.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-21 TUNING_RANGE_7	Tuning Range 7 Radio channel above 2465 MHz.
20-18 TUNING_RANGE_6	Tuning Range 6 Radio channel between 2450 and 2465 MHz.
17-15 TUNING_RANGE_5	Tuning Range 5 Radio channel between 2435 and 2450 MHz.
14-12 TUNING_RANGE_4	Tuning Range 4 Radio channel between 2420 and 2435 MHz.
11-9 TUNING_RANGE_3	Tuning Range 3 Radio channel between 2405 and 2420 MHz.
8-6 TUNING_RANGE_2	Tuning Range 2 Radio channel between 2390 and 2405 MHz.
5-3 TUNING_RANGE_1	Tuning Range 1 Radio channel between 2375 and 2390 MHz.
2-0 TUNING_RANGE_0	Tuning Range 0 Radio channel less than or equal to 2375 MHz.

55.4.7.3.7.1.21 Max Transmit Frequency For TX Configuration 1 (MAX_TX_CFG1_FREQ)

Offset

Register	Offset
MAX_TX_CFG1_FREQ	58h

Function

Due to regulatory compliance requirements, more than one set of TX configuration settings may be required to meet out-of-band spectral emission requirements for TX frequencies near either the upper or lower limit of a frequency band. The PLL has a built in capability to detect when the TX carrier frequency is below or above a carrier frequency in MHz.

The MAX_TX_CFG1_FREQ register field should be programmed with a frequency threshold to detect when an alternate TX configuration must be utilized. When the PLL center frequency is selected to be greater than this threshold, a signal will be sent to the TX Digital logic indicating the need to use an alternate configuration. Note, to disable this feature, program the MAX_TX_CFG1_FREQ to 12'hFFF. This will ensure the alternatite TX configuration is not selected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MAX_TX_CFG1_FREQ											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Fields

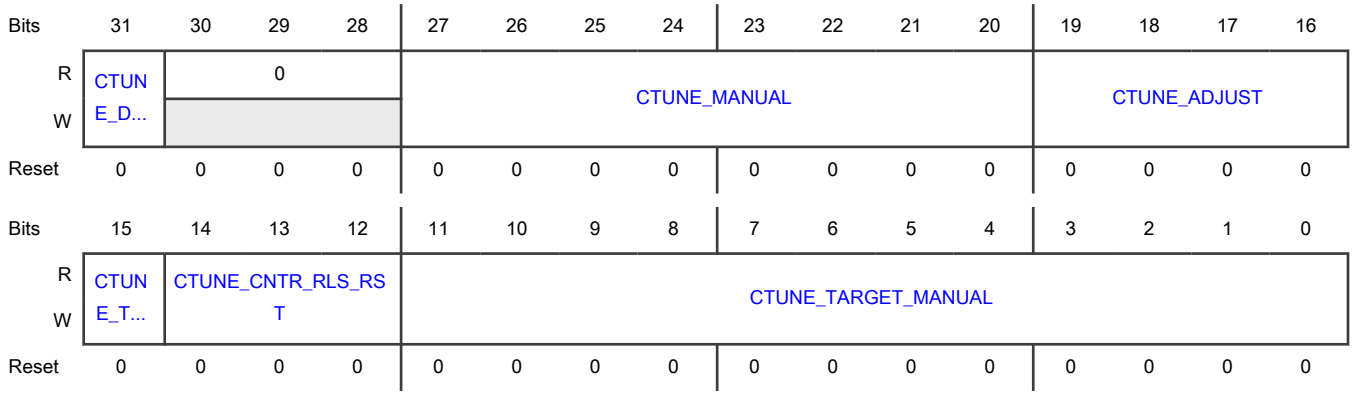
Field	Function
31-12 —	Reserved
11-0 MAX_TX_CFG1_FREQ	Maximum Transmit Frequency for Standard TX Settings This register should be programmed with the frequency threshold (MHz) to be used by PLL Digital to determine when to switch to alternate TX settings. If the PLL center frequency is greater than this programmed value, the TX digital will select an alternate configuration for modulation.

55.4.7.3.7.1.22 PLL Coarse Tune Control (CTUNE_CTRL)

Offset

Register	Offset
CTUNE_CTRL	5Ch

Diagram



Fields

Field	Function
31 CTUNE_DISABLE	Coarse Tune Disable If this bit is set, the Coarse Tune Setting applied to the VCO comes from the CTUNE_MANUAL register.
30-28 —	Reserved
27-20 CTUNE_MANUAL	Manual Coarse Tune Setting If CTUNE_DISABLE is set, this register is the value that is applied to the VCO as the Coarse Tune Setting.
19-16 CTUNE_ADJUST	Coarse Tune Count Adjustment This register is a signed three bit value that adjusts the PLL Frequency Meter count used in the Coarse Tune Calibration. The range of adjustment is -8 to +7, and the adjustment is only made to the PLL Frequency count used by the Coarse Tune Calibration sequence.
15 CTUNE_TARGET_DISABLE	Disable Coarse Tune Target If this bit is set, the Frequency Target presented to the Coarse Tune Calibrator comes from the CTUNE_TARGET_MANUAL register.
14-12 CTUNE_CNTR_RLS_RST	Coarse Tune Counter Release Reset This field adds up to 7 ref_clk cycles to ctune_counter_release_rst. Right now it is hard coded to 2, so the maximum value could be 9.
11-0 CTUNE_TARGET_MANUAL	Manual Coarse Tune Target If CTUNE_TARGET_DISABLE or LPM_SDM_CTRL1[SDM_MAP_DISABLE] is set, this register is the value that is presented to the Coarse Tune Calibrator as the Frequency Target in MHz.

55.4.7.3.7.1.23 PLL Data Rate Override Control (DATA_RATE_OVRD_CTRL1)

Offset

Register	Offset
DATA_RATE_OVRD_CTRL1	60h

Function

The Bluetooth Low Energy LL defines test cases with asymmetric data rates in multiple connection modes. The PLL configuration must be switched by hardware when a new configuration is requested. The DATA_RATE_SWITCH register is intended to provide an override for various control registers in the PLL. The following register fields will be overridden when the Bluetooth Low Energy LL requests the updated data rate:

```
PLL_LPM_CTRL[LPM_SCALE]
             PLL_LPM_CTRL[HPM_CAL_SCALE]
             HPM_BUMP[HPM_FDB_RES_CAL]
             HPM_BUMP[HPM_FDB_RES_TX]
```

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				HPM_FDB_RES_TX_...		HPM_FDB_RES_CAL...		LPM_SCALE_CFG1				HPM_CAL_SCALE_CFG1			
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-10 HPM_FDB_RES_TX_CFG1	HPM FDB RES Transmit Configuration1 This register will be used to override HPM_BUMP[HPM_FDB_RES_TX] when the Bluetooth Low Energy LL selects a data rate configuration switch.
9-8 HPM_FDB_RES_CAL_CFG1	HPM FDB RES Calibration Configuration1 This register will be used to override HPM_BUMP[HPM_FDB_RES_CAL] when the Bluetooth Low Energy LL selects a data rate configuration switch.
7-4	LPM Scale Configuration1

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPM_SCALE_CFG1	This register will be used to override PLL_LPM_CTRL[LPM_SCALE] when the Bluetooth Low Energy LL selects a data rate configuration switch.
3-0 HPM_CAL_SCALE_CFG1	HPM Scale Configuration1 This register will be used to override PLL_LPM_CTRL[HPM_CAL_SCALE] when the Bluetooth Low Energy LL selects a data rate configuration switch.

55.4.7.3.7.1.24 PLL Data Rate Override Control (DATA_RATE_OVRD_CTRL2)

Offset

Register	Offset
DATA_RATE_OVRD_CTRL2	64h

Function

The Bluetooth Low Energy LL defines test cases with asymmetric data rates in multiple connection modes. The PLL configuration must be switched by hardware when a new configuration is requested. The DATA_RATE_SWITCH register is intended to provide an override for various control registers in the PLL. The following register fields will be overridden when the Bluetooth Low Energy LL requests the updated data rate:

`CHAN_MAP_EXT[NUM_OFFSET]`

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				NUM_OFFSET_CFG1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM_OFFSET_CFG1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-0 NUM_OFFSET_CFG1	Numerator Offset Configuration1 This register will be used to override CHAN_MAP_EXT[NUM_OFFSET] when the Bluetooth Low Energy LL selects a data rate configuration switch.

55.4.7.3.7.1.25 PLL Coarse Tune Results (CTUNE_RES)

Offset

Register	Offset
CTUNE_RES	84h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		CTUNE_FREQ_SELECTED												CTUNE_BEST_DIFF	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CTUNE_BEST_DIFF						0		CTUNE_SELECTED							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-18 CTUNE_FREQ_SELECTED	Coarse Tune Frequency Selected This is the Frequency Target in MHz that is currently being presented to the Coarse Tune Calibrator.
17-10 CTUNE_BEST_DIFF	Coarse Tune Absolute Best Difference This is the absolute value of the best difference found during Coarse Tune between the targeted frequency count and the actual frequency count.
9-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-0 CTUNE_SELECTED	Coarse Tune Setting to VCO This is the current VCO Coarse Tune setting, it is the result of the Coarse Tune Calibration, unless overridden using CTUNE_DISABLE.

55.4.7.3.7.1.26 PLL HPM Calibration Timing Attributes (HPM_CAL_TIMING)

Offset

Register	Offset
HPM_CAL_TIMING	A0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HPM_VCO_MOD_DELAY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				HPM_CAL2_SETTLE_TIME				HPM_CAL1_SETTLE_TIME				HPM_CTUNE_SETTLE_TIME			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 HPM_VCO_MOD_DELAY	HPM VCO Modification Output Delay This 16-bit value specifies the delay time from a calibration state change to a new VCO DAC code. This delay time is equal to (HPM_VCO_MOD_DELAY + 2) x 62.5ns.
15-12 —	Reserved
11-8 HPM_CAL2_SETTLE_TIME	HPM Calibration2 Settling Time The HPM calibration2 setting time specifies addition time for the analog circuits to settle when transitioning out of the HPM calibration2 tuning stage. Settling time is (HPM_CAL2_SETTLE_TIME + 1)us.
7-4	HPM Calibration1 Settling Time

Table continues on the next page...

Table continued from the previous page...

Field	Function
HPM_CAL1_SETTLE_TIME	The HPM calibration1 setting time specifies addition time for the analog circuits to settle when transitioning out of the HPM calibration1 tuning stage. Settling time is (HPM_CAL1_SETTLE_TIME + 1)us.
3-0 HPM_CTUNE_SETTLE_TIME	CTUNE Settling Time The CTUNE setting time specifies addition time for the analog circuits to settle when transitioning out of the course tuning stage. Settling time is (HPM_CTUNE_SETTLE_TIME + 1)us.

55.4.7.3.7.1.27 PLL Offset Control (PLL_OFFSET_CTRL)

Offset

Register	Offset
PLL_OFFSET_CTRL	A4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				PLL_NUMERATOR_OFFSET											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PLL_NUMERATOR_OFFSET															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 PLL_NUMERATOR_OFFSET	PLL Numerator Offset Signed value that allows for a carrier frequency offset of up to +/-100ppm be added to the PLL numerator to allow for a SW controlled static shift of PLL frequency.

55.4.7.3.7.1.28 PLL Data Rate Switch Control (PLL_DATARATE_CTRL)

Offset

Register	Offset
PLL_DATARATE_CTRL	A8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				HPM_INTEGER_DELAY_DRS				HPM_SDM_DELAY_DRS				LPM_SDM_DELAY_DRS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	PLL_VCO_TRIM_KVM_CAL_DRS			0	PLL_VCO_TRIM_KVM_TX_DRS			0	HPM_VCM_CAL_DRS			0	HPM_VCM_TX_DRS		
W																
Reset	0	1	1	0	0	1	1	0	0	0	0	1	0	0	1	0

Fields

Field	Function
31-28 —	Reserved
27-24 HPM_INTEGER_DELAY_DRS	DRS HPM_SDM_DELAY This field selects the number of clock cycles of the PLL Sigma Delta Clock divided by 2 to delay the High Port Integer modulation of the VCO High Port DAC Array during Data Rate Switch.
23-20 HPM_SDM_DELAY_DRS	DRS HPM_SDM_DELAY This field selects the number of clock cycles of the PLL Sigma Delta Clock divided by 2 to delay the High Port Sigma Delta modulation of the VCO High Port Fraction during Data Rate Switch. Note that the High Port SDM is clocked by the PLL Sigma Delta Clock but the modulation is based on a divide by 2 version of this same clock.
19-16 LPM_SDM_DELAY_DRS	DRS LPM_SDM_DELAY This field selects the number of clock cycles of the PLL Sigma Delta Clock to delay the Low Port Sigma Delta modulation of the PLL Loop Divider during Data Rate Switch.
15 —	Reserved
14-12	Data Rate Switch for pll_vco_trim_kvm_cal

Table continues on the next page...

Table continued from the previous page...

Field	Function
PLL_VCO_TRIM_KVM_CAL_DRS	000b - 10MHz/V 100b - 20MHz/V 110b - 30MHz/V 111b - 40MHz/V
11 —	Reserved
10-8 PLL_VCO_TRIM_KVM_TX_DRS	Data Rate Switch for pll_vco_trim_kvm_tx. 000b - 10MHz/V 100b - 20MHz/V 110b - 30MHz/V 111b - 40MHz/V
7 —	Reserved
6-4 HPM_VCM_CAL_DRS	Data Rate Switch for hpm_vcm_cal 000b - 432 mV 001b - 328 mV 010b - 456 mV 011b - 473 mV 100b - 488 mV 101b - 408 mV 110b - 392 mV 111b - 376 mV
3 —	Reserved
2-0 HPM_VCM_TX_DRS	Data Rate Switch for hpm_vcm_tx 000b - 432 mV 001b - 328 mV 010b - 456 mV 011b - 473 mV 100b - 488 mV 101b - 408 mV

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 392 mV 111b - 376 mV

55.4.7.4 Receiver Digital Module

55.4.7.4.1 Introduction

The 2.4GHz Receive Digital Module(RX_DIG) implements the receive digital signal processing functions common to Bluetooth LE, 802.15.4, and Generic Link Layer modes. Generally this includes conditioning the input samples from the ADC and produces a data output for the PHY and an RSSI/LQI for the Link Layers.

55.4.7.4.1.1 Features

The 2.4GHz RX_DIG module includes the following features:

- Automatic Gain Control (AGC)
- CIC Decimation Filtering
- Digital IF mixer
- CFO correction
- IQ Mismatch Correction
- Fractional Correction
- Acquisition Channel Filter(16, 24-tap configurable)
- Demod Channel Filter
- Sample Rate Converter
- DC Offset Calibration and Correction
- DC Residual Correction
- Wide-Band and Narrow-Band RSSI estimator module
- Two instances of Normalizer(Wide-Band, Narrow-Band)
- RC Calibration
- Tone Analyzer
- Multiple Auxiliary Output

55.4.7.4.1.2 Block Diagram

55.4.7.4.1.2.1 Data Path

RX_DIG data path is shown below.

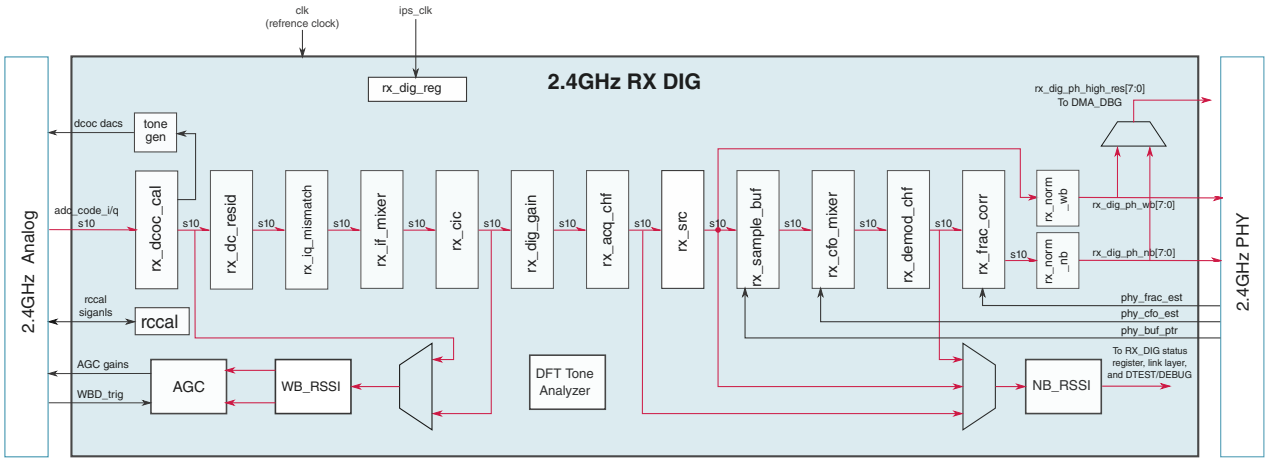


Figure 293. RX_DIG Block diagram

55.4.7.4.1.2.2 Auxiliary Output

RX_DIG has different kinds of auxiliary output for localization applications, test, or debug purposes. The following figure illustrates the structure of RX_DIG auxiliary outputs. The IQ, phase, and magnitude output support window average function, which controlled by the DFT_RX_IQ_OUT_SEL field, and the average window size can be chosen from 1(disabled), 4, 8, 16, 32, 64, 128, and 256. The rx_iq_8b_out is sent to NBU, while other signals mapped to DTEST and/or DMA_DEBUG pages.

NOTE

Check RSSI Estimator section to get the detail information of RSSI and magnitude signals

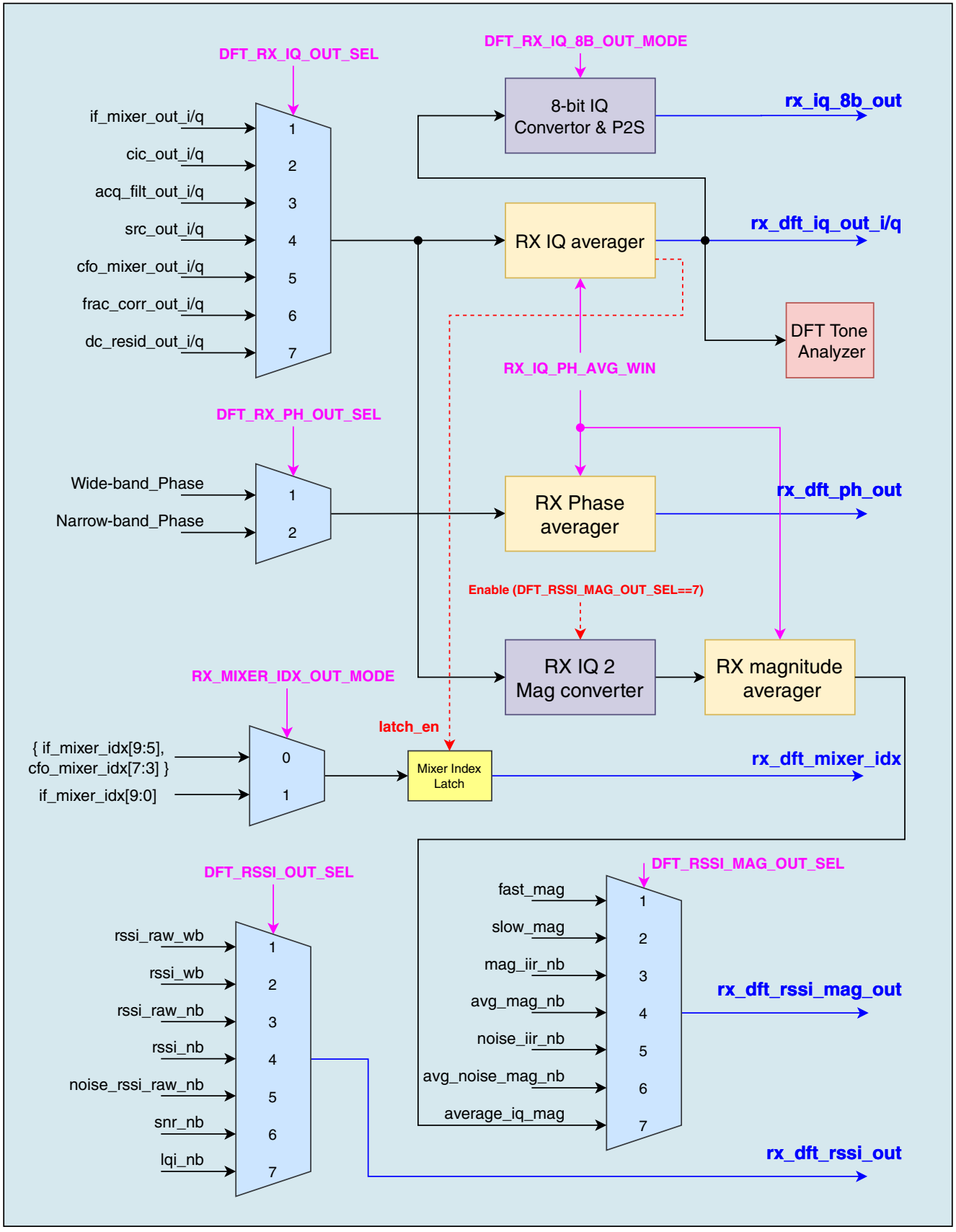


Figure 294. RX_DIG outputs selection diagram

55.4.7.4.1.3 Data Format

The 2.4GHz Radio contains fixed-point data path. In this document, there're two kinds of format to represent the fixed-point data, one is Simulink format [u/s]fix[Nb]_En[Nx], e.g. sfix8_En5, ufix8. And another is the customary format [u/s][N].[M], e.g. s2.5, u8(or u8.0). The below tables show how to parse the fixed point values to real.

Simulink format	
ufix or sfix	Denotes the number is unsigned or singed
Nb	Denotes the total number of bits, including the sign bit if signed
Nx	Denotes the number of bits for fractional

Customary format	
u or s	Denotes the number is unsigned or singed
N	Denotes the number of bits for integer
M	Denotes the number of bits for fractional

Examples	Note: signed values stored as 2's complement.			
Bin	Simulink	Customary	Total bits	Real value
00001111	ufix8	u8 or u8.0	8	15
10000011	ufix8	u8 or u8.0	8	131
0_0011001	sfix8	s7 or s7.0	8	25
1_1111110	sfix8	s7 or s7.0	8	-2
01_011000	ufix8_En6	u2.6	8	1.375
11_000001	ufix8_En6	u2.6	8	3.015625
0_10_11000	sfix8_En5	s2.5	8	2.75
1_11_11100	sfix8_En5	s2.5	8	-0.125

55.4.7.4.2 Functional Description

55.4.7.4.2.1 Reset

All RX_DIG registers that require a reset are reset by a global asynchronous reset. The rx_init signal also synchronously resets the state of various rxdig sub-modules. In addition, a software reset from 2P4Ghz XCVR can reset all RX_DIG modules except RX_DIG control register module.

55.4.7.4.2.2 Clocks

The RX_DIG uses two clocks, the 26 or 32MHz RF reference clock (clk), and the IPS gated clock (ips_clk) that has been synchronized to the reference clock. All programming model registers are clocked by the IPS clock. All other registers are clocked by the RF reference clock.

55.4.7.4.2.3 Datarate Selection

The Gen4 Radio has a hardware feature, called datarate selection, to change TX and RX datarate automatically. When RX_DIG.datarate_config_sel = 0, RX_DIG will use primary configuration values. When RX_DIG.datarate_config_sel = 1, RX_DIG will switch to use another set of configuration values for the secondary datarate. All config registers or register fields with "_DRS" postfix indicate which are using for the secondary datarate.

55.4.7.4.2.4 DC Offset Calibration and Correction

The DCOC block supports the DC offset calibration during RX warmup, and DC correction after RX warmup.

55.4.7.4.2.4.1 DC Calibration

DC calibration is expected to be performed during each RX warmup. A figure illustrating DCOC during DC calibration is shown below.

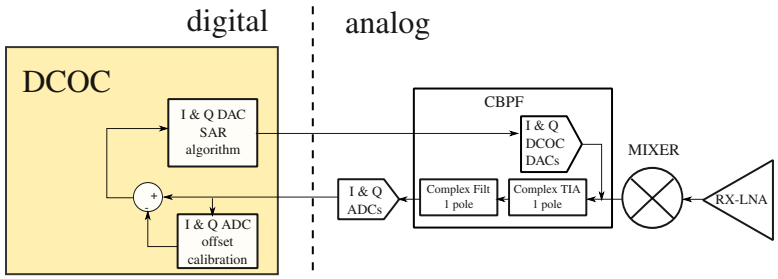


Figure 295. DC Calibration

During DC calibration the following steps are performed

1. The ADC is isolated by putting the CBPF output in a HiZ state, and the ADC inputs are shorted
2. The I and Q ADC outputs are averaged over 4 samples to estimate the DC offset related to the ADCs. These DC offsets are then subtracted from the ADC outputs which are used in steps #4-X below
3. The LNA output is put in HiZ, and the CBPF is put in high gain configuration
4. A SAR algorithm is used to determine the best values for the I DAC. If DCOC_CTRL[DCOC_DAC_ORDER]=1, the Q DAC is used instead in this step.
5. A SAR algorithm is used to determine the best value for the Q DAC. If DCOC_CTRL[DCOC_DAC_ORDER]=1, the I DAC is used instead in this step.
6. Measure the digital correction value on I and Q channel and send the result to the RO register DCOC_DIG_CORR_RESULT. If DCOC_CTRL0.DCOC_DIG_CORR_EN=1(default value), during RX_DIG enabled, the DCOC will compensate the digital correction values on both I and Q channel.

During the SAR for the I DAC, the SAR algorithm evalutates a 4-sample average of the following expression:

SAR_IN = SFIQ * QADC2 + (SFIIP)*(SFII/16) *IADC2

where SFIQ is programmable to be either 0 or 1, SFIIP is programmable to be +1 or -1, SFII is programmable from 0 to 15, IADC2 and QADC2 are the ADC samples corrected by the offsets found in step #2 above.

During the SAR for the I DAC, the SAR algorithm evalutates a 4-sample average of the following expression:

SAR_IN = SFQI * IADC2 + (SFQQP)*(SFQQ/16) *QADC2

where SFQI is programmable to be either 0 or 1, SFQQP is programmable to be +1 or -1, SFQQ is programmable from 0 to 15, and IADC2 and QADC2 are the ADC samples corrected by the offsets found in step #2 above.

After calibration, the ADC offsets and DAC codes determined during calibration can be read in the DCOC_STAT register.

The DCOC has the following additional programmable settings which affect the calibration:

- DCOC_I_CAL_POL. Polarity of the DCOC I DAC
- DCOC_Q_CAL_POL. Polarity of the DCOC Q DAC
- DCOC_CBPF_STL_TIME[3:0]. Settle time in microseconds after estimating the ADC offsets but prior to running the SAR algorithm. This is for the CBPF to be switched from HiZ to high gain.
- DCOC_SAR_STL_TIME[3:0]. Settle time in microseconds during the SAR algorithm, after changing the DAC value
- PULSE_CAPCODE. If this bit is set, then as part of the DCOC calibration sequence, the DCOC calibration logic will direct the RC calibration logic to set its seq_cbpf_ccode output to the maximum value for 3us in order to precharge cap array HiZ nodes.

The programmable values described above are determined through device characterization for the four different CBPF use cases. Preliminary guidance on settings is shown in the table below.

Table 429. DCOC Calibration Settings for 2.4GHz

Programmable Variable	
SFIQ	1
SFQI	1
SFIIP	1
SFQQP	0
SFII	13
SFQQ	9
I_CAL_POL	0
Q_CAL_POL	1
DAC_ORDER	0
CBPF_STL_TIME	3
SAR_STL_TIME	3
PULSE_CAPCODE	0

The total DC calibration duration is a function of the ref clock frequency and some DCOC configs. The referecne clock cycle count during DCOC calibration can be calculated as follows:

dcoc_cycle_count = 34 + (CBPF_STL_TIME + SAR_STL_TIME * 12) * count_in_1us + 14 * AVG_WIN_SIZE

Where, count_in_1us = 32 for 32MHz ref clock, and 26 for 26MHz ref clk. CBPF_STL_TIME and SAR_STL_TIME comes from DCOC_CTRL0. AVG_WIN_SIZE = DCOC_CTRL0.DCOC_AVG_WIN ? 8 : 4.

By default config(CBPF_STL_TIME=SAR_STL_TIME=3, AVG_WIN_SIZE=4), dcoc_cycle_count = (ref clock == 32MHz) ? 1338 : 1104. Thus the DC calibration druation is about 42us for 32MHz ref clock and 43us for 26MHz ref clock.

NOTE: DCOC TSM related timing should be updated if the DCOC calibration configs use non-default value.

For a fast RX warmup, the DC calibration is either mostly or entirely skipped, depending on PULSE_CAPCODE. If PULSE_CAPCODE bit is clear, then the TSM fast RX warmup can be programmed such that the TSM enable for the DCOC calibration remains de-asserted during the RX warmup. If PULSE_CAPCODE bit is set, then the TSM fast RX warmup can be programmed such that the TSM enable for the DCOC calibration should assert for only 3us (precharge time)+ CBPF_STL_TIME (settle time). In either case, for a fast RX warmup none of the DCOC offsets will be updated from those calculated in the previous DCOC calibration.

55.4.7.4.2.4.2 DC Correction

After the RX warmup is complete, the DCOC block handles DC correction. A figure illustrating DCOC operation after RX warmup is shown below.

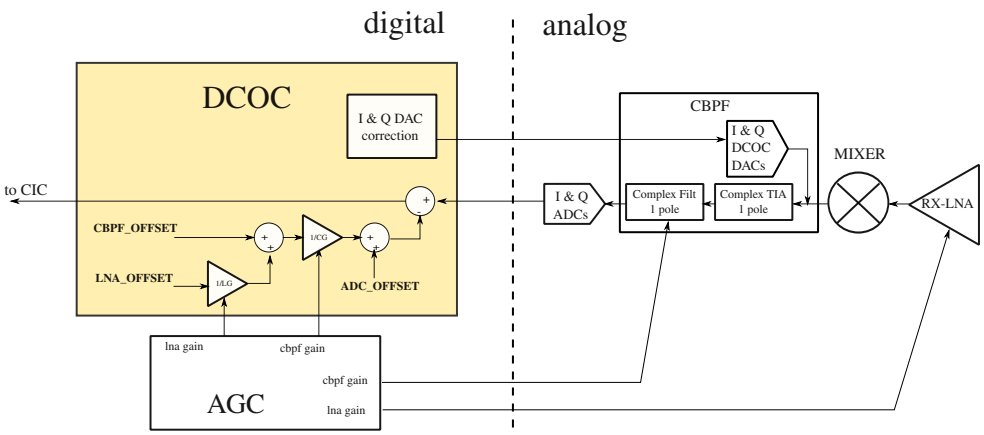


Figure 296. DC Correction

The DAC values and ADC_OFFSETs found during calibration are applied to correct the DC in the receiver. Alternately, the user can override the DAC values by programming DCOC_CTRL2[DCOC_DAC_OVRD_I] and DCOC_CTRL2[DCOC_DAC_OVRD_Q] and setting the DCOC_CTRL0[DCOC_DAC_OVRD_EN] bit. The user can also override the ADC offsets by programming the DCOC_CTRL2[DCOC_ADC_OFFSET_OVRD_I] and DCOC_CTRL2[DCOC_ADC_OFFSET_OVRD_Q] and setting the DCOC_CTRL0[DCOC_ADC_OFFSET_OVRD_EN] bit.

Additionally, the DCOC is capable of correcting for DC offsets which are depednent on the LNA and CBPF gains. These offsets are not determined during the RX warmup calibration. They have to be programmed by the user in the DCOC_CTRL1 register bitfields DCOC_ILNA_OFFSET, DCOC_QLNA_OFFSET, DCOC_ICBPF_OFFSET, and DCOC_QCBPF_OFFSET. These offsets are then scaled based on the AGC gains. In the case of the AGC's LNA gain, the scale factor 1/LG is as follows:

- highest lna gain: 1/LG = 1
- next highest lna gain: 1/LG = 1/4
- next highest lna gain: 1/LG = 1/16
- all other lna gains: 1/LG = 0

In the case of the AGC's CBPF gain, the scale factor 1/CG is as follows:

- high cbpf gain: 1/CG = 1
- low cpbf gain: 1/CG = 1/2

Note that the I & Q DAC correction value is not dependent on the CBPF gain.

55.4.7.4.2.5 DC Residual Correction

This block is used to compensate for the DC offset which is still present after AGC and DCOC calibration/tracking has completed. It estimates the DC offset by measuring the minimum and maximum value and then applies an additive correction factor to the I and Q channels. This block does not change the value of the DCOC DACs.

The DC offset is compensated independently on I and Q channels. The correction loop determines the minimum and maximum values over a specified window of samples (*Nwin*) which are then averaged and used (with an update parameter *Alpha*) to create the DC estimate. This DC estimate is then subtracted from the input samples. The applied DC estimate continues to be updated every *NWIN* samples until *IT_FREEZE* of these windows have elapsed, at which time the DC estimate is frozen.

Some addition capabilities as of Radio 3.6 are not reflected in figure above:

- the 2^{-Alpha} term can assume values of 1, 1/2, 1/4+1/8, 1/4, 1/8+1/16, 1/8, 1/16+1/32, and 1/16
- the 2^{-Alpha} term can be gearshifted using the DC_RESID_GS_EN and DC_RESID_GEARSHIFT bits.
- Further capabilities related to Long Range support. These are described at the end of this DC Residual Correction section.

Multiple modes of operations can be identified for residual DC offset compensation block. They are presented in table below.

Table 430. Residual DC offset compensation operational modes

Mode	Description	DC_RESID_EN	EXT_DC_EN	IT_FREEZE	DLY	NWIN, ALPHA
Bypass	The DC residual is disabled. The input I and Q samples to the DC Residual block are passed directly to the DC Redisdual output without any modifications. .	0	-	-	-	-
Mode 1	In this mode, DC residual has the role of calculating and applying a DC offset without use of an initial estimate from the DCOC. The DC residual processing is restarted at every AGC gain change This mode is suitable for use when an AGC gain change occurs which is not followed by DCOC tracking.	1	0	>1	>0	program as needed
Mode 2	In this mode, DC residual has the role of refining the DC offset estimate provided by the DCOC at the end of the DCOC tracking after the last DCOC DAC update. The DC residual processing is restarted whenever DCOC's tracking hold timer expires. This mode is suitable after AGC gain change followed by DCOC tracking.	1	1	>1	-	program as needed
Mode 3	In this mode, DC residual has the role of applying a correction based on the DC offset estimate provided by the DCOC at the end of the DCOC tracking after the last DCOC DAC update.	1	1	1	-	-

Table continues on the next page...

Table 430. Residual DC offset compensation operational modes (continued)

	The DC residual processing is restarted whenever DCOC's tracking hold timer expires. This mode is suitable after AGC gain change and DCOC tracking when no further refining is desired.					

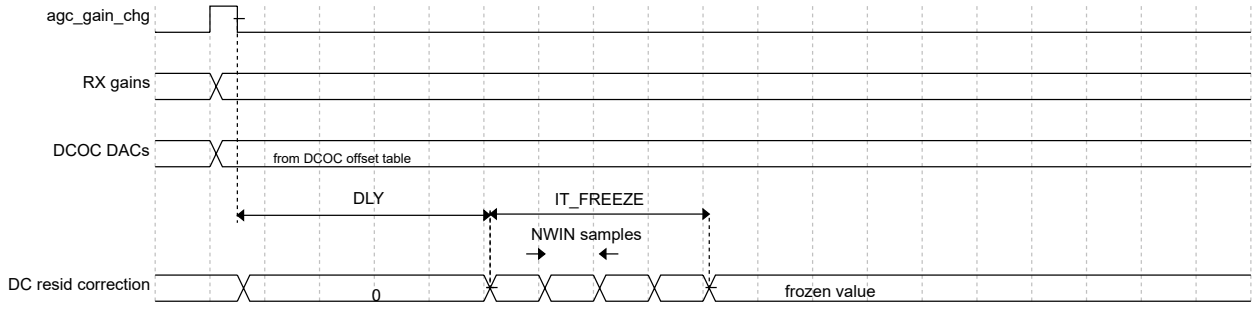


Figure 297. DC residual mode 1 timing

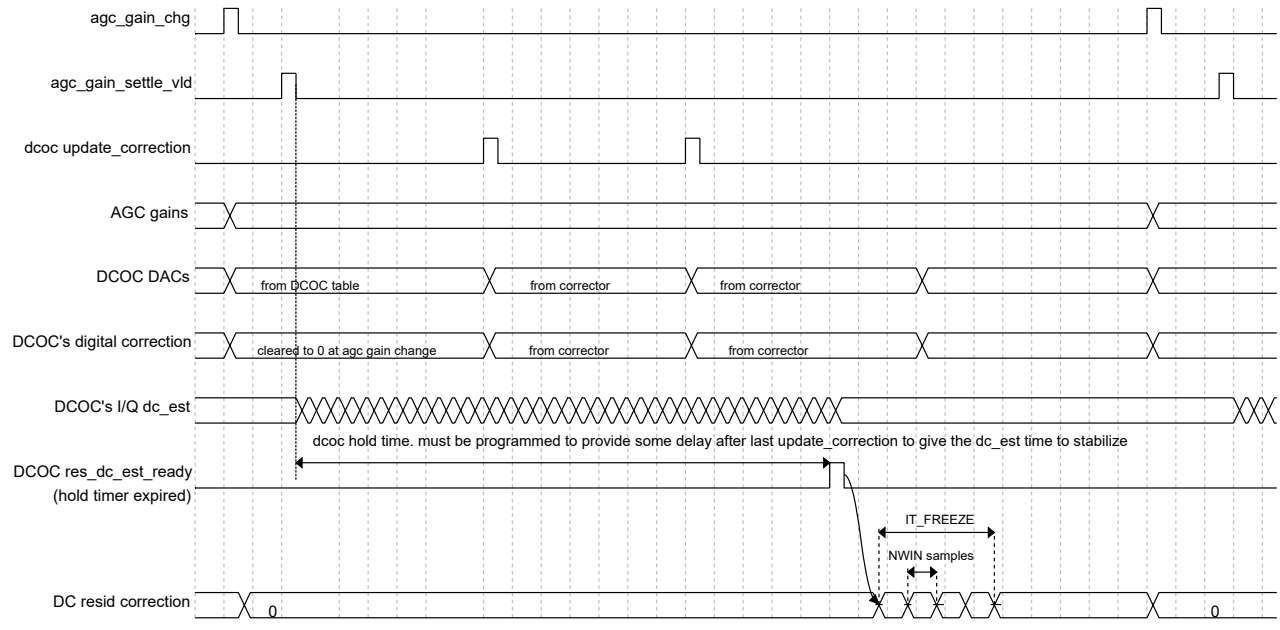


Figure 298. DC residual mode 2 timing

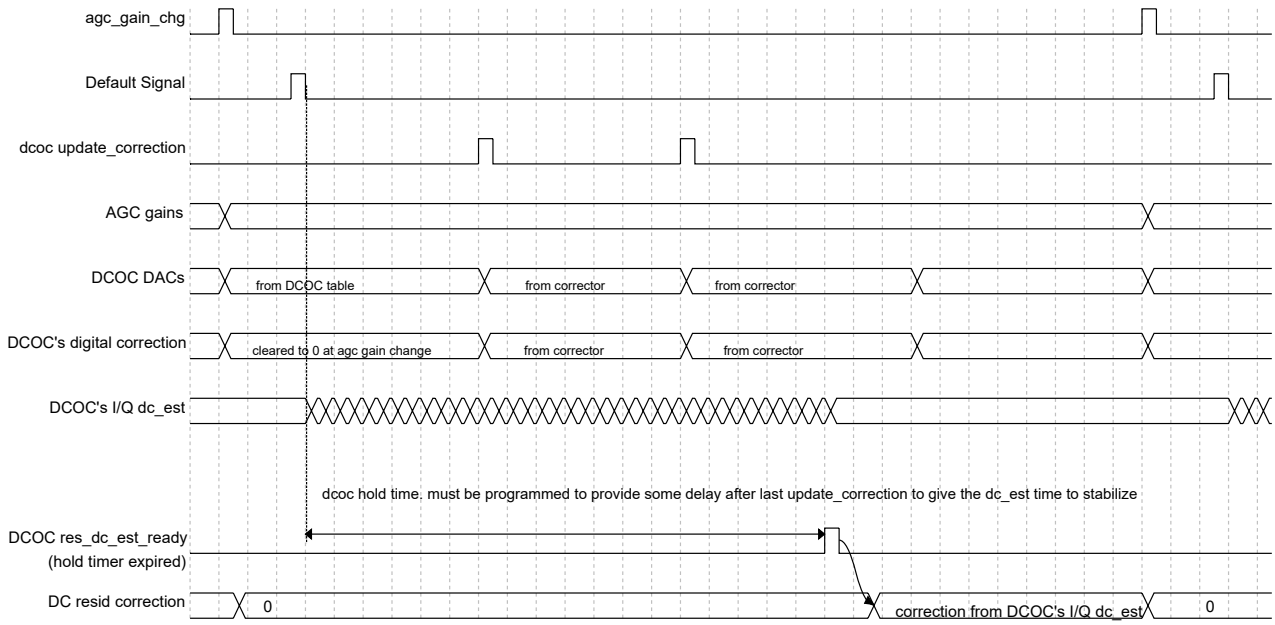


Figure 299. DC residual mode 3 timing

The behavior of the Residual DC block is also dependent on the AGC index and the values programmed for DC_RESID's MIN_AGC_IDX and the DCOC's DCOC_CTRL_1[DCOC_TRK_MIN_AGC_IDX]. This behavior is defined in the table below, which assumes DC_RESID_EN=1.

Table 431. Residual DC offset compensation operation dependency on AGC table index

EXT_DC_EN	MIN_AGC_IDX	DCOC_TRK_MIN_AGC_IDX	Behavior
-	AGC_IDX < MIN_AGC_IDX	-	DC residual is disabled
0	AGC_IDX >= MIN_AGC_IDX	-	DC residual operates in Mode 1 (it does not use an initial estimate from the DCOC)
1		AGC_IDX >= DCOC_TRK_MIN_AGC_IDX	DC residual operates in Mode 2 or 3 (It uses the estimate from the DCOC)
		AGC_IDX < DCOC_TRK_MIN_AGC_IDX	DC residual operates in Mode 1. it does not use the estimate from the DCOC, since the DCOC's tracker is disabled for this AGC gain index

For Long Range, the normal operation of the Residual DC block (as described above) may not reduce the DC level enough to support operation at very low signal levels. The Residual DC block includes an additional feature to handled such cases. If DC_RESID_SECOND_RUN_EN bit is set, then when the AGC index is greater than or equal to DC_RESID_MIN_AGC_IDX2 and the Residual DC block's normal operation has completed, additional DC compensation within the Residual DC block will activate. This "Second Run" DC compensation, when engaged, uses its own set of configuration parameters: DC_RESID_NWIN2, DC_RESID_ALPHA2, DC_RESID_ITER_FREEZE2, DC_RESID_GS2_EN, DC_RESID_GEARSHIFT2. The operation is otherwise similar to the "Mode 2" described above except that the initial DC estimate is the DC offset found at end of Residual DC block's normal operation instead of the estimate from the DCOC module. The following additional features are also supported for "Second Run"

- Slew Rate Control to limit the DC corrections. refer to DC_RESID_SR2_EN and DC_RESID_SLEWRATE2 register descriptions
- Continuous Correction. If DC_RESID_CC_EN bit is set, the DC_RESID_ITER_FREEZ2 is ignored and corrections are applied continuously until the next gain change pulse or initialization
- Operation can be stopped by the PHY. If DC_RESID_PHY_STOP_EN=1, then when the PHY signals that AA found for uncoded has been found, the "Second Run" DC compensation operation is halted (regardless of DC_RESID_ITER_FREEZE2 or DC_RESID_CC_EN bit settings) until the next gain change pulse or initialization.

55.4.7.4.2.6 I/Q Mismatch Correction

The I/Q mismatch block corrects for I/Q gain and phase mismatch. It can correct up to +/- 9 degrees of phase mismatch and +/- 4 dB of gain mismatch. After correction, phase mismatch is reduced to less than 0.5 degree and gain mismatch less than 0.3 dB.

During factory calibration, the I/Q phase adjustment coefficient (iqmc_phase_adj) and the I/Q gain adjustment coefficient (iqmc_gain_adj) are determined. Calibration is enabled by setting the CTRL0[RX_IQMC_EN] and IQMC_CTRL0[IQMC_CAL_EN] bit. The following value should be used for calibration: IQMC_CTRL0[IQMC_NUM_ITER]=0x80. The IQMC_CTRL1 register, which also stores the calibration result, should be set to its default value 0x0000_0400. A CW input with a 250 kHz (+/-75 kHz) offset and -2 dBm (+/- 6 dBm) level at the ADC I channel input is required for calibration, though the calibration should also work with lower signal levels down to -18 dBm.

NOTE

DCOC calibration should be done before doing I/Q mismatch calibration. Also, during the calibration, the AGC should be disabled.

The IQMC_CTRL0[IQMC_CAL_FREQ_SEL] controls the sample clock frequency of the IQMC module. By default IQMC uses 1/2 reference clock(16MHz or 13MHz). Setting the IQMC_CTRL0[IQMC_CAL_FREQ_SEL] bit let IQMC use 1/4 reference clock(8MHz or 6.5MHz). With the standard number of iterations (IQMC_NUM_ITER=0x80), calibration needs 128 * 64(fixed value for each iteration) samples. Thus takes 512 us or 1024 us, based on the IQMC_CAL_FREQ_SEL setting, when using 32MHz ref clock.

After calibration is completed the IQMC_CTRL1 register is updated and the IQMC_CAL_EN bit is cleared automatically. The gain calibration and phase calibration loops are shown in the figures below.

NOTE

For each iteration, the 64 samples(fixed value) should cover at least one full cycle of the CW. That means, suppose using 32MHz ref clock, the MINIMUM CW input freq should be 250KHz(period = 4 us) when IQMC_CAL_FREQ_SEL=0, or 125KHz(period = 8 us) when IQMC_CAL_FREQ_SEL=1.

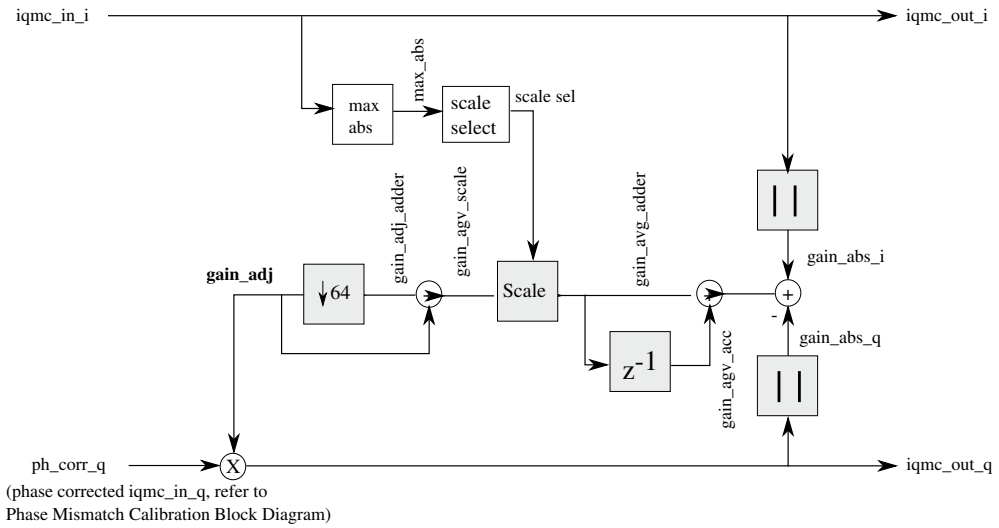


Figure 300. I/Q Gain Mismatch Calibration Block Diagram

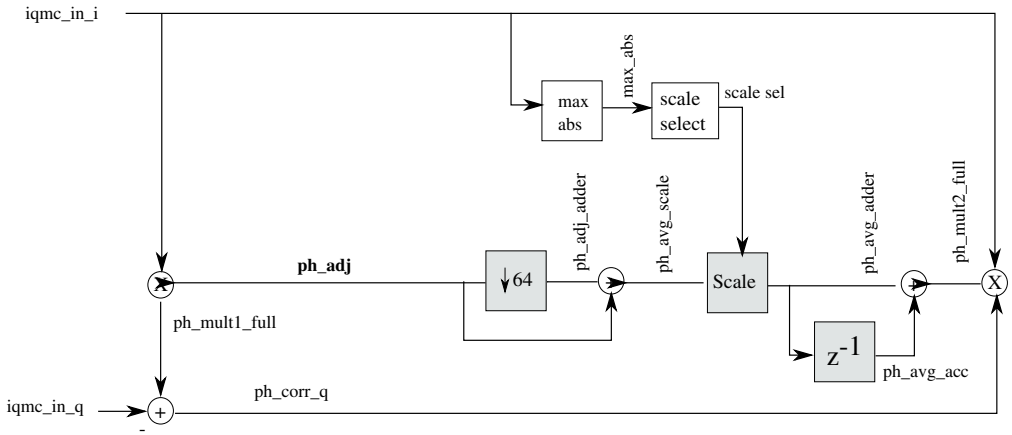


Figure 301. I/Q Phase Mismatch Calibration Block Diagram

Here is the detailed procedures to do I/Q mismatch calibration:

- Do DCOC calibration and apply the result to the input CW signal later
- Disable AGC by clear the CTRL0[AGC_EN]
- Set 0x00000400 for IQMC_CTRL1 register(Default value)
- Set proper values for IQMC_NUM_ITER and IQMC_CAL_FREQ_SEL
- Enter into RX mode and drive the CW tone to the antenna
- Set CTRL0[RX_IQMC_EN]
- Set IQMC_CTRL0[IQMC_CAL_EN]
- Poll IQMC_CTRL0[IQMC_CAL_EN] until it cleared
- Read the calibrited result from IQMC_CTRL2 register

When the receiver is enabled for packet reception, the calibration loops are disabled and the corrections are static. For gain correction, the Q channel is scaled by the gain adjustment coefficient. For phase correction, the I channel is scaled by the phase

adjustment coefficient and subtracted from the Q channel. The gain and phase correction values are stored in programming model registers and may be stored in flash to eliminate the need to run I/Q mismatch calibration after each reset.

The gain and phase corrections are illustrated in the figure below.

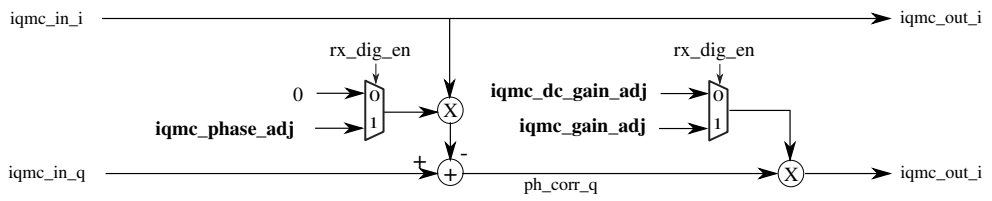


Figure 302. I/Q Mismatch Correction Block Diagram

55.4.7.4.2.7 Digital IF Mixer

The digital IF mixer performs complex mixing from low IF to DC. The mixer operates on samples at the output of the I/Q mismatch correction module. The mixer is implemented with an effective look-up table size of 1024 entries. The programmable mixer frequency is computed as follows:

DIG_MIXER_FREQ = round[1024*desired_mixer_freq/mixer_sample_frequency]

Where supported ratios of desired_mixer_freq/mixer_sample_frequency are within (-0.25, +0.25).

The table below shows the typical DIG_MIXER_FREQ configuration

Table 432. DIG_MIXER_FREQ confugration based on IF and reference clock frequency

	26MHz	32MHz
IF = 1MHz	39	32
IF = 1.5MHz	59	48

55.4.7.4.2.8 CIC Decimation Filter

RX_DIG has an Cascaded Integrator Comb (CIC) decimation filters. The CIC operates on output samples from the Digital IF Mixer at the RF reference clock rate. The order of CIC is 3 or 4 configurable and also the decimation rate can be programmed as needed by the application.

A block diagram of the CIC filter is shown below.

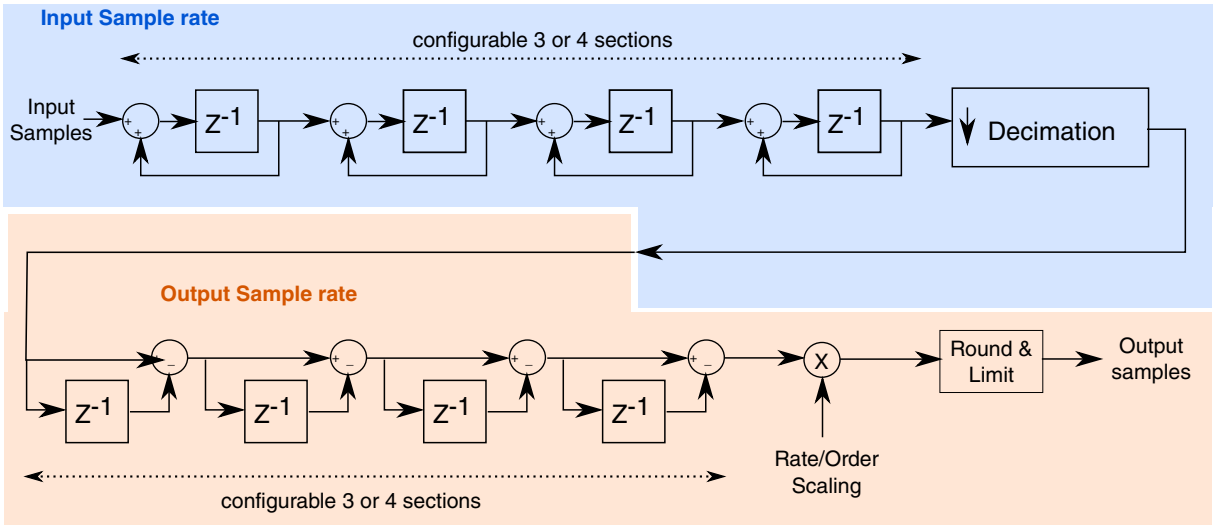


Figure 303. CIC Filter Block Diagram

The table below shows the typical CIC rate configuration.

Table 433. CIC rate confugration base on bitrate and reference clock frequency

	26MHz	32MHz
250Kbps	4	5(Only support 3-order)
500Kbps	3	4
1Mbps	2	3
2Mbps	1	2

55.4.7.4.2.9 Digital Gain

This block is a simple configurable multiplier applied at the output of CIC. The output of this module is computed as follows:

dig_gain_out_i/q = cic_out_i/q * (1 + RX_DIG_GAIN * 1 / 8)

Here the "RX_DIG_GAIN" can be choose from 0 ~ 7, therefore the actual gain value can be 1, 1.125,1.25, 1.375, 1.5, 1.625, 1.75, 1.875.

55.4.7.4.2.10 Acquisition Channel Filter

The acquisition channel filter is a 16/24-tap configurable symmetric FIR filter with real 6-bit to 10-bit coefficients. It operates at the CIC output rate on the output from Digital Gain module.

For 24-tap configuration:

$H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + ... + h_{11}z^{-11} + h_{11}z^{-12} + ... + h_1z^{-22} + h_0z^{-23}$

For 16-tap configuration:

$H(z) = h_3 + h_4z^{-1} + h_5z^{-2} + ... + h_{11}z^{-7} + h_{11}z^{-8} + ... + h_4z^{-14} + h_3z^{-15}$

The filter output is scaled by 1/512, therefore the sum of the 12 or 8(depends on the CTRL0[RX_ACQ_FILT_LEN]) coefficients programmed in register ACQ_FILT_0_3 ~ ACQ_FILT_10_11 is expected to be <=256 in order to avoid saturation.

55.4.7.4.2.11 Sample Rate Conversion (SRC) Filter

The sample rate conversion block is used for a fractional clock rate conversion of the sampled decimated signal, when 26MHz reference clock is used. For the 2.4GHz targeted modulations, the PHY data rate is always 4x the symbol rate (of 2Mbps, 1Mbps, 500kbps or 250kbps) for (G)FSK modes and 4x the 2Mchips/sec rate for IEEE 802.15.4. These desired frequencies are not related to the ADC clock rate by an integer factor, which necessitates the need for a fractional sample rate conversion that is performed by SRC. For a complete list of the fractional conversion modes refer to . The SRC is enabled by setting RX_DIG_CTRL[RX_SRC_EN] to 1, and the conversion factor is selected by RX_DIG_CTRL[RX_SRC_RATE] as described below.

55.4.7.4.2.12 IQ Sample Buffer

This block is a 1~32 depth buffer for storing IQ values output from RX SRC. The sample buffer depth is controlled by Gen4 PHY, but can be bypassed by setting RX_SAMPLE_BUF_BYPASS in CTRL1 register.

NOTE:

1. If RX CFO MIXER bypassed, shuold plus 1 to the original IQ sample buffer pointer config value in PHY(PREPHY_MISC_BUFF_PTR_LR and PREPHY_MISC_BUFF_PTR_GFSK)

2. If RX Fractional Correction bypassed, shuold plus 1 to the original IQ sample buffer pointer config value in PHY(PREPHY_MISC_BUFF_PTR_LR and PREPHY_MISC_BUFF_PTR_GFSK)

3. If both RX CFO MIXER and RX Fractional Correction bypassed, shuold plus 2 to the original IQ sample buffer pointer config value in PHY(PREPHY_MISC_BUFF_PTR_LR and PREPHY_MISC_BUFF_PTR_GFSK)

55.4.7.4.2.13 CFO Mixer

The CFO mixer performs complex mixing to eliminate carrier frequency offset which operates at SRC out clock domain. The mixer is implemented with an effective look-up table size of 256 entries. Tpically the CFO correction coefficient(cfo_est[9:0]) comes from PHY, but can be overrided by RX_CFO_EST_OVRD_EN and RX_CFO_EST_OVRD fields in CTRL1 register. The target CFO mixer frequency computed as follows:

cfo_frequency = round[cic_out_freq*cfo_est/2048]

The supported CFO correction range is +/-0.25 datarate. e.g. for 1Mbps, +/-250KHz CFO can be elimiated

55.4.7.4.2.14 Demodulation Channel Filter

The demodulation channel filter is a 9-tap symmetric FIR filter with real 9-bit to 10-bit coefficients. It operates at the SRC output rate on the output from CFO Mixer module.

$H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_3z^{-5} + h_2z^{-6} + h_1z^{-7} + h_0z^{-8}$

The filter output is scaled by 1/512, therefore the 5 coefficients programmed in register DEMOD_FILT_0_1 and DEMOD_FILT_2_4 should be meet below formula:

$(h_0 + h_1 + h_2 + h_3) * 2 + h_4 <= 512$

55.4.7.4.2.15 Fractional Correction

This block is responsible with the timing error compensation. The correction factor comes from Gen4 PHY, but can be override by RX_DIG register.

The timing error compensation is realized via linear interpolation using past, current and future samples:

IQOut[k] = IQIn[k] + (IQIn[k+1]-IQIn[k])*abs(FracDly[k]), if FracDly[k]>=0

IQOut[k] = IQIn[k] + (IQIn[k-1]-IQIn[k])*abs(FracDly[k]), if FracDly[k]<0

55.4.7.4.2.16 Normalizer

The normalizer block converts the 11-bit I/Q input signal into phase output as shown in the block diagrams below.The RX_DIG has two instances of normalizer, wide-band and narrow-band, both of them have the same structure. The difference is the wide-band

normalizer connects to the output of RX_SRC, while the narrow-band normalizer uses RX_FRAC_CORR output. In addition, a rounded version, PH_ZB[4:0], of PH_WB[7:0] is sent to ZB_DEMOD.

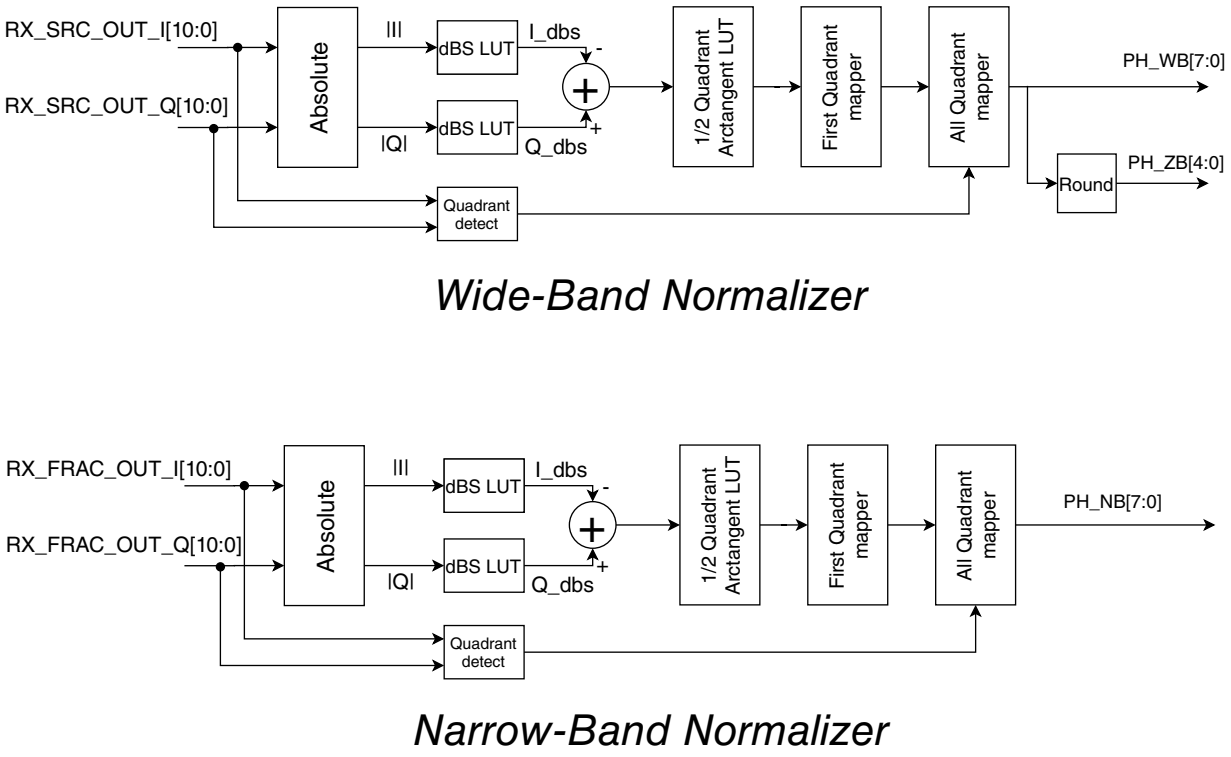


Figure 304. RX_DIG Normalizer Structure

55.4.7.4.2.17 RSSI Estimator

The role of the RSSI (Received Signal Strength Indication) estimator is to measure the energy of an incoming signal. RX_DIG has two RSSI estimator instances called Wide-Band RSSI estimator(WB_RSSI) and Narrow-Band RSSI estimator(NB_RSSI).

WB_RSSI measures the magnitude of ADC(defaule) or CIC output and provids the results to feed AGC. The input of NB_RSSI can be chosen from SRC(default), Acquisition Channel Filter, Demodulation Channel Filter, or Fractional Correction output and the results used by link layer controller or software.

55.4.7.4.2.17.1 Wide-band RSSI

Wide-Band RSSI(WB-RSSI) mainly for generatging fast_mag and slow_mag for AGC. When AGC in magnitude mode, the AGC gain index is determined by fast_mag and slow_mag.

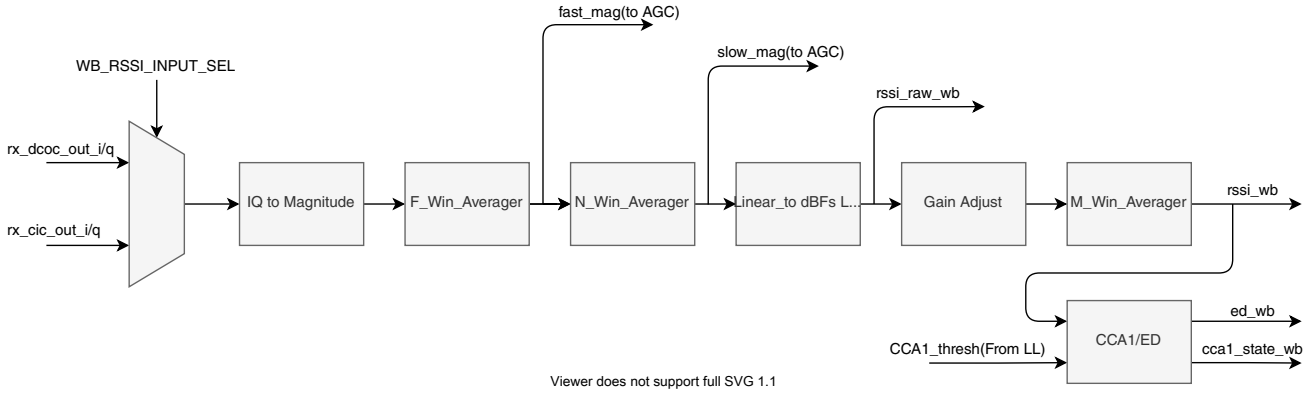


Figure 305. WB-RSSI block diagram

RSSI_WB Calcuation:

rssi_wb = rssi_raw_wb - agc_log_gain + rssi_adj_wb + 13

Where:

1. rssi_raw_wb represent the dBFs result of I/Q signal magnitude. rssi_raw_wb = 20*log10((adc_i^2 + adc_q^2)^0.5/1024) ≈ max(adc_i, adc_q) + 0.375*min(adc_i, adc_q)

2. agc_log_gain indicate current analog gain values, which store in AGC_IDX*_GAIN_VAL[LOG_GAIN_*] and selected by AGC based on current AGC gain index

3. rssi_adj_wb is set by WB_RSSI_CTRL[RSSI_ADJ_WB]

55.4.7.4.2.17.1.1 WB-RSSI Freerun Mode

By default, WB-RSSI working in continuous mode(freerun), until AGC enter into FREEZE state. For debug purpose, by setting WB_CONT_MEAS_OVRD_EN=1 and WB_CONT_MEAS_OVRD=0, the WB-RSSI will stop working once the first rssi_wb calculation done. See RSSI_RDY_WB bit description to get the information about how to get WB-RSSI results continuously.

55.4.7.4.2.17.2 Narrow-Band RSSI

The role of the Narrow-Band RSSI(NB-RSSI) is to estimate and report: narrow band received signal strength, SNR and LQI result to link layer contoller. It also can be chosen for CCA1/ED calculation.

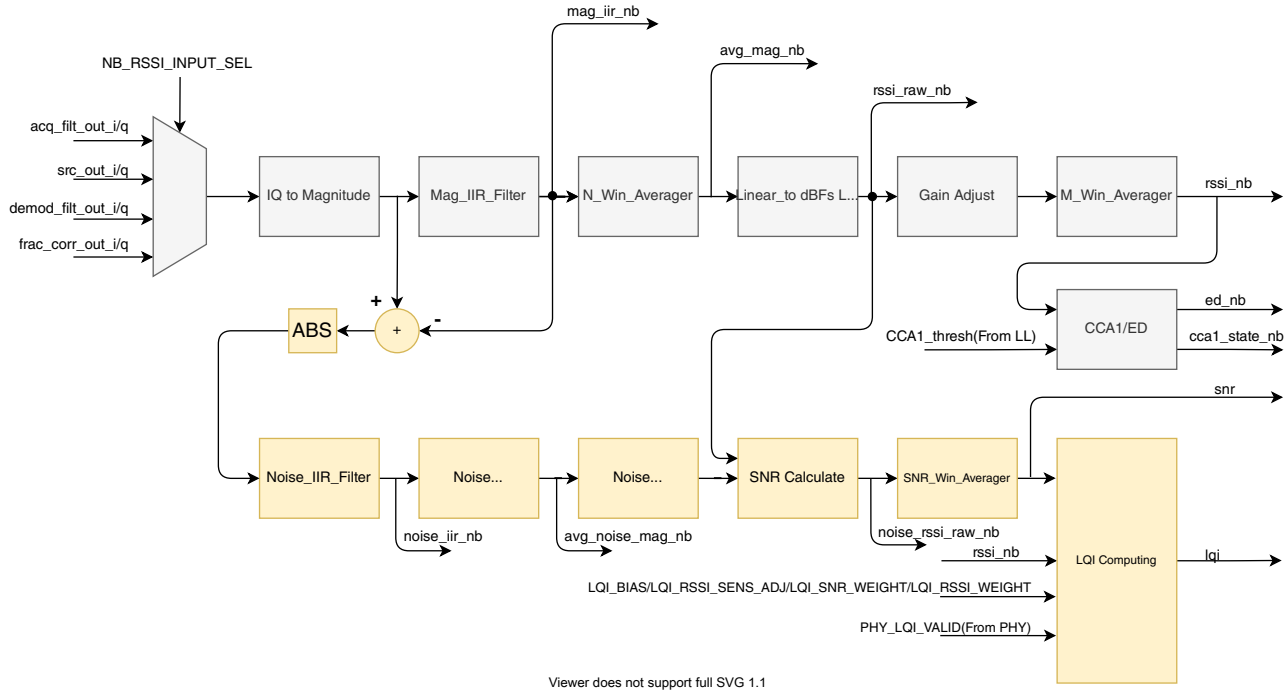


Figure 306. NB-RSSI block diagram

RSSI_NB Calcuation:

rssi_nb = rssi_raw_nb - agc_log_gain + rssi_adj_nb + 13

Where:

1. rssi_raw_nb represent the dBFs result of I/Q signal magnitude. rssi_raw_nb = 20*log10((i^2 + q^2)^0.5/1024) ≈ max(i, q) + 0.375*min(i, q)

2. agc_log_gain indicate current analog gain values, which store in AGC_IDX*_GAIN_VAL[LOG_GAIN_*] and selected by AGC based on current AGC gain index

3. rssi_adj_nb is set by NB_RSSI_CTRL[RSSI_ADJ_NB]. **NOTE**: If CTRL0[RX_DIG_GAIN] set a non-zero value. Should add a negative offset value to NB_RSSI_CTRL[RSSI_ADJ_NB]

55.4.7.4.2.17.2.1 NB-RSSI Freerun Mode

By default, NB-RSSI working in one-time mode. By setting NB_CONT_MEAS_OVRD_EN=1 and NB_CONT_MEAS_OVRD=1 bit, the NB-RSSI will calculate RSSI/LQI/SNR continuously. See RSSI_RDY_NB bit description to get the information about how to get NB-RSSI results continuously.

55.4.7.4.2.17.2.2 NB-RSSI Manually Enable Mode

By default, NB-RSSI starts when PHY AA found. To manually trigger the NB-RSSI(e.g. for software CCA), follow the below steps:

- Make sure XCVR in idle mode
- Clear RSSI_GLOBAL_CTRL[NB_RSSI_AA_MATCH_OVRD]
- Set RSSI_GLOBAL_CTRL[NB_RSSI_AA_MATCH_OVRD_EN]
- Set RSSI_GLOBAL_CTRL[NB_CONT_MEAS_OVRD_EN] and RSSI_GLOBAL_CTRL[NB_CONT_MEAS_OVRD]
- Start RX sequence
- Wait RX warmup done
- Set RSSI_GLOBAL_CTRL[NB_RSSI_AA_MATCH_OVRD], when software get ready

- Poll NB_RSSI_RES0[RSSI_RDY_NB] until 1, which means the NB-RSSI results(

  RSSI_NB/SNR_NB/LQI_NB/ED_NB

) are ready. Check RSSI_RDY_NB bit description for more information
- When measurement done, clear RSSI_GLOBAL_CTRL[NB_RSSI_AA_MATCH_OVRD], GLOBAL_CTRL[NB_RSSI_AA_MATCH_OVRD_EN], RSSI_GLOBAL_CTRL[NB_CONT_MEAS_OVRD_EN], RSSI_GLOBAL_CTRL[NB_CONT_MEAS_OVRD]
- Write 1 to RSSI_RDY_NB to clear this status bit
- Exit RX mode

55.4.7.4.2.17.3 CCA1 and ED Measurement

Both WB and NB RSSI can do CCA1 and ED measurement. The following steps show how to enable the CCA1 and ED function.

- Config CCA1_ED_FROM_NB bit to select use which RSSI instance to do CCA1 and ED measurement
- Set either WB_CCA1_ED_EN or NB_CCA1_ED_EN bit baseed on the value of CCA1_ED_FROM_NB
- Set CCA1 threshold. Note, this config lacated in GEN-LL or 15p4-LL
- Launch CCA1 sequence from GEN-LL or 15p4-LL

NOTE

If 15p4-LL is active, set 15p4_PHY->CCA_LQI_SRC[CCA1_FROM_RX_DIG], or the CCA1 will be measured by 15p4_PHY

55.4.7.4.2.18 Automatic Gain Control (AGC)

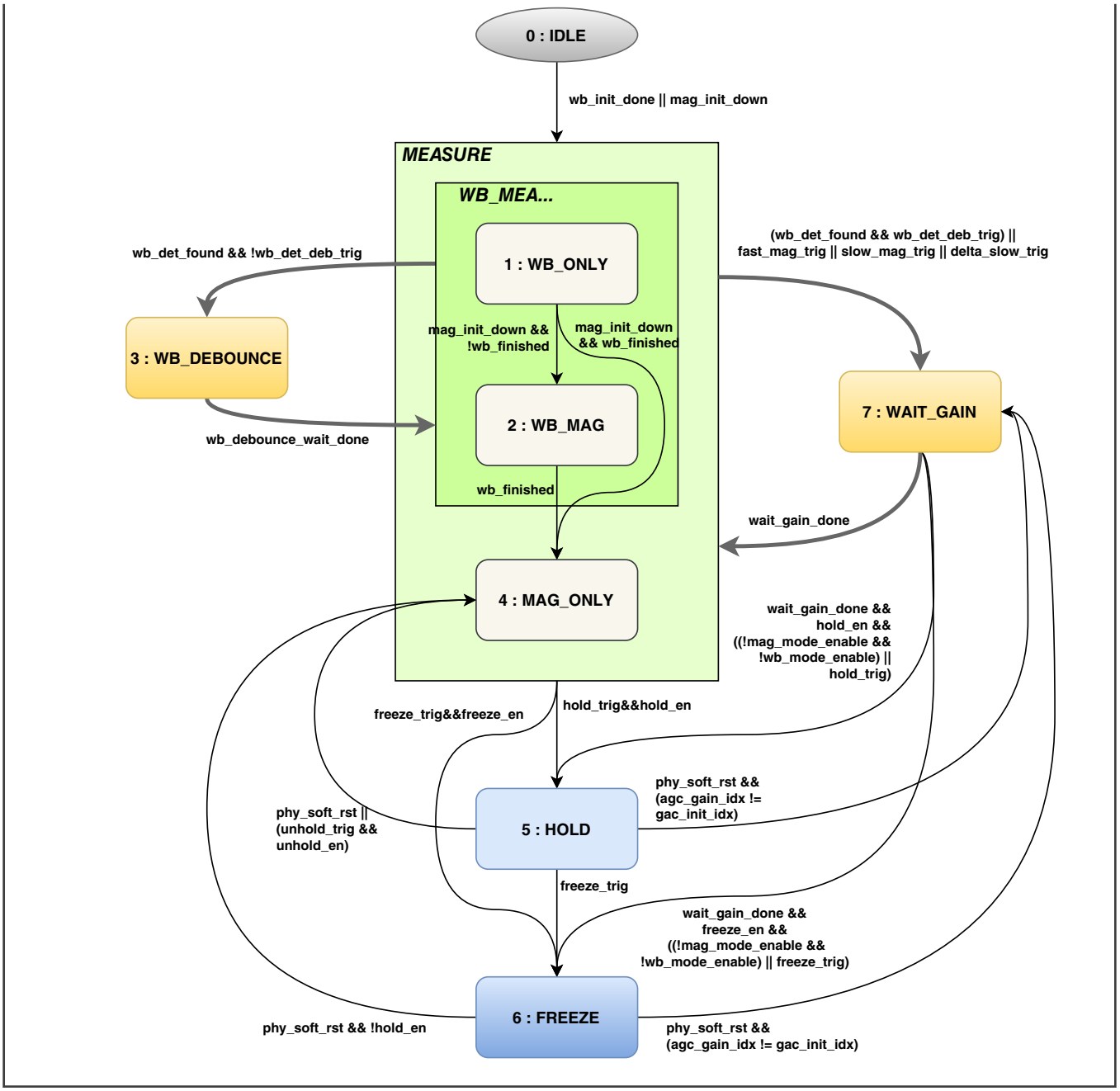
The Automatic Gain Control (AGC) system is designed to ensure that the received signal does not become distorted because of clipping in the analog receive chain, while attempting to maintain an optimal signal level for demodulation. There are two locations where gain may be adjusted: analog LNA, and analog CBPF. The AGC system manipulates these gains to maintain a targeted ADC input signal level. The targeted ADC signal level is chosen as a compromise between receiver linearity, blocker headroom requirements and the goal to maximize the digital signal SNR that is provided to the protocol-specific PHY demodulators.

The AGC sytem operated with a state machine that is depicted in below figure and has the following significant states:

- **IDLE(000b)**: No actions are taken. The system waits for RX_DIG enable and than AGC initial wait done
- **WB_ONLY(001b)**: AGC is working with Wide-Band detector enabled
- **WB_MAG(010b)**: AGC is working with Wide-Band detector and magnitude mode enabled
- **WB_DEBOUNCE(011b)**: AGC is in WB trigger debounce waiting state
- **MAG_ONLY(100b)**: AGC is working with magnitude mode enabled
- **HOLD(101b)**: AGC hold mode
- **FREEZE(110b)**: AGC is freeze
- **WAIT_GAIN(111b)**: Wait analog gain settle state.

55.4.7.4.2.18.1 AGC FSM

Figure 307. AGC FSM state diagram



55.4.7.4.2.18.2 AGC Gain Step Triggers

Gen4 AGC has two kinds of gain step trigger sources. One is from the analog comparator output called wideband detector(WBD) or peak detector, which can only make AGC gain step down. The other is the magnitude output from WB-RSSI, which can make AGC gain step down or up. When WBD enabled, the AGC works in WB_ONLY mode. When magnitude mode enabled, AGC works in MAG_MODE mode. In addition, if both WBD and magnitude mode enabled, AGC works in WB_MAG mode.

In one RX sequence, WBD can trigger once or twice(called WBD step1 and step2), based on the settings of AGC_CTRL[AGC_WBD_EN]. Each step has its own threshold and gain index step down config. In addition, to avoid the WBD triggered by noise or glitch on the antenna, users can enable the WBD dual clip function. When this function enabled, only continually twice trigger occurs can be considered as one WBD trigger. The cooldown(or debounce wait) time can be set by AGC_WBD_STEP1_DUAL_CLIP_WAIT and AGC_WBD_STEP2_DUAL_CLIP_WAIT.

For magnitude mode, AGC compares the FAST_MAG and/or SLOW_MAG value from WB_RSSI with the AGC_THR_FAST and/or AGC_IDX*_THR values to determine gain index step up or step down.

There's a special magnitude trigger called delta slow trigger. To enable delta slow detection, slow magnitude mode also need to be enabled. When the previous Nth(Defined by DELTA_SLOW_WAIT) slow mag value minus the latest slow mag value exceed the value set in DELTA_SLOW_THR, the AGC will issue a big gain index step up, defined by AGC_DELTA_SLOW_STEP.

55.4.7.4.2.18.3 AGC Gains

Gen4 AGC supports up to 12-gear of gain index and enables 11-gear(10~0) by default. Each gain index has its own analog gain control settings and stored in AGC_IDX*_GAIN_CFG register.

The AGC_IDX*_GAIN_VAL registers store the analog gain values for each gain index. The LOG_GAIN is referenced by WB-RSSI and NB-RSSI module to estimate the input power level of RF signal, while LINEAR_GAIN used by Gen4 PHY to determine whether there is RF signal on the antenna.

55.4.7.4.2.18.4 AGC State

55.4.7.4.2.18.4.1 IDLE

This is the default state of AGC. AGC will keep in this state until initialize done. The AGC internal counter will start counting(use reference clock) when RX_DIG enabled. When the AGC counter matches the value of AGC_WBD_INIT_WAIT or AGC_MAG_INIT_WAIT, the AGC will trans to measurement state(WB_ONLY, WB_MAG or MAG_ONLY). Typically, if both WBD and magnitude mode enabled, the programed AGC_WBD_INIT_WAIT should no less than AGC_MAG_INIT_WAIT.

55.4.7.4.2.18.4.2 WB_ONLY

After the AGC initialization done, if AGC_WBD_EN!=0 and AGC_WBD_INIT_WAIT<AGC_MAG_INIT_WAIT, AGC will trans to this mode. In WB_ONLY mode, the AGC only accepts WBD trigger.

If AGC_WBD_AUTO_DIS_CFG != 00, AGC will disable WBD automatically when condition matches. See description of AGC_WBD_AUTO_DIS_CFG for more info. After WBD disabled, AGC will trans to next mode, typically the MAG_ONLY.

If AGC_WBD_STEP*_DUAL_CLI P_EN set, two consecutive WBD triggers treat as one effective AGC gain change trigger. That means the first WBD trigger will make AGC trans to WB_DEBOUNCE mode and wait a moment. After AGC back to WB_ONLY mode, if the WBD trigged again in time, typically less than 5 reference clock cycles, AGC will step down the gain index.

AGC supports WBD gain limit function. If agc_wbd1_gain_limit_en set, after wbd_step1 trig occurs, the AGC max gain will be set as "agc_init_gain – wbd_step1", that means even if mag value is too small, AGC will not increase gain_idx larger than "agc_max_gain – wb_step1"(NOTE: Typically, agc_max_gain = agc_init_gain). Similar function for agc_wb2_gain_limit_en.

55.4.7.4.2.18.4.3 MAG_ONLY

In this state, AGC will compare the magnitude from WB-RSSI with the threshold to determine the AGC gain index. There are two kinks of magnitude called fast_mag and slow_mag and which are using for driven the fast mag trigger(called fast mag mode) and slow mag trigger(called slow mag mode). The difference between fast mode and slow mode is response time and adjustment accuracy. Both fast and slow mag mode can be enabled individually, however, delta slow mode can only work with slow mag mode enabled.

55.4.7.4.2.18.4.4 WB_MAG

This state can be considered as a combination of WB_ONLY and MAG_ONLY. In this state, AGC accepts both WBD and magnitude triggers.

55.4.7.4.2.18.4.5 WAIT_GAIN

After AGC gain index changed, AGC will enter into WAIT_GAIN state and keep a while. At the same time, reset the WBD and WB-RSSI.

55.4.7.4.2.18.4.6 WB_DEBOUNCE

If WBD dual clip(debounce) function enabled, AGC will enter into this state when the first WBD trigger occurs without gain index change and reset the WBD at the same time. After a while(defined by AGC_WBD_DUAL_CLIP_TIMEOUT), AGC will back to WB_ONLY or WB_MAG state.

55.4.7.4.2.18.4.7 HOLD

There are two ways to enter into HOLD state, hold mode timeout(defined by AGC_HOLD_TIMEOUT) matches or hold mode trigger(selected by AGC_PHY_HOLD_TRIG_SEL) occurs.

Note, the hold mode timeout counter will start work when AGC enter into WB_ONLY, WB_MAG or MAG_ONLY and reset when AGC gain index changed.

In HOLD state, AGC will disable WBD and fast mag comparision. However, if AGC_UNHOLD_FEAT_EN[0] and AGC_SLOW_EN set, AGC will be continuing to monitor the slow magnitude values to determine whether exit HOLD and back to MAG_ONLY state.

When AGC_UNHOLD_FEAT_EN[1] set, AGC will exit HOLD mode after N(defined by AGC_UNHOLD _FEAT_TIMEOUT) micro second and back to MAG_ONLY mode.

55.4.7.4.2.18.4.8 FREEZE

In this state, AGC will disable both WBD and mag comparision. Freeze mode timeout(defined by AGC_FREEZE_TIMEOUT) matches or freeze mode trigger(selected by AGC_PHY_FREEZE_TRIG_SEL) occurs can make AGC enter into FREEZE state.

If both HOLD and FREEZE state triggered by timeout, the AGC will enter into HOLD state first, then wait for freeze timeout and finally enter into FREEZE state. However, AGC can trans to FREEZE state immediately when PHY freeze trigger occurs.

When AGC_UNFREEZE_FEAT_EN set, if RX sequence not be finished in N(defined by AGC_UNFREEZE_FEAT_TIMEOUT) micro second, AGC will exit FREEZE mode and back to MAG_ONLY mode.

55.4.7.4.2.18.5 AGC PHY Magnitude Threshold

The 2.4GHz PHY acquisition module needs 2 threshold values, mag_thr and mag_thr_hi, to determine if the current input signal is valid.

When the magnitude of the input signal exceeds the mag_thr(pt_sig_pres=01b), the 2.4GHz PHY acquisition module starts working and use GEN4_PHY->FSK_CFG0[HAMMING_AA_LOW_PWR] to find access address. In addition, if the magnitude exceeds both mag_thr and mag_thr_hi(pt_sig_pres=11b), the acquisition module will use another hamming value (GEN_LL->NTW_ADR_CTRL[NTW_ADR_THR] or GEN4_PHY->FSK_CFG0[BLE_NTW_ADR_THR], depends on which LL is active)to do access address searching.

The mag_thr and mag_thr_hi are AGC gain index dependent, which means for each AGC gain index, there's one pair of mag_thr and mag_thr_hi value. These values are configured by AGC_IDX*_GAIN_VAL[MAG_THR_*] and AGC_IDX*_GAIN_VAL[MAG_THR_HI_*] fields.

Furthermore, when datarate_config_sel = 1, a pair of **signed** offset(AGC_IDX*_GAIN_CFG[MAG_THR_*_DRS_OFS] and AGC_IDX*_GAIN_CFG[MAG_THR_HI_*_DRS_OFS]) values will be added to the AGC_IDX*_GAIN_VAL[MAG_THR_(HI_)*] to calculate the final threshold value.

The below example shows how to determine the mag_thr and mag_thr_hi value when the AGC gain index is 11.

When datarate_config_sel = 0 :

mag_thr = AGC_IDX11_GAIN_VAL[MAG_THR_11]

mag_thr_hi = AGC_IDX11_GAIN_VAL[MAG_THR_HI_11]

When datarate_config_sel = 1 :

mag_thr = AGC_IDX11_GAIN_VAL[MAG_THR_11] + AGC_IDX11_GAIN_CFG[MAG_THR_11_DRS_OFS]

mag_thr_hi = AGC_IDX11_GAIN_VAL[MAG_THR_HI_11] + AGC_IDX11_GAIN_CFG[MAG_THR_11_HI_DRS_OFS]

NOTE: for each AGC gain index, the programmed AGC_IDX*_GAIN_VAL[MAG_THR_*] , AGC_IDX*_GAIN_VAL[MAG_THR_HI*],

AGC_IDX*_GAIN_CFG[MAG_THR_*_DRS_OFS], and AGC_IDX*_GAIN_CFG[MAG_THR_HI*_DRS_OFS] values should let mag_thr and mag_thr_hi meet the following equation:

0 <= mag_thr <= mag_thr_hi <= 2047

55.4.7.4.2.19 RC Calibration

The RX RC Calibration module uses a binary search to find the best 5bit calibration code (rccal_code[4:0]) for the RC calibration circuit in the analog. It is expected that the RC calibration will be performed on each RX warmup. The entire RCCAL sequence requires no more than 8us for all XO reference frequencies supported.

The rccal_code code is then used to calculate a 7bit code for the CBPF circuit, which is dependent on a programmable bandwidth setting (bw_code)and IF setting (sc_code). Also, programmable offsets are provided to allow additional adjustments to the rccal_code or to the computed CBPF code:

cbpf_ccode = round[9 + 2*bw_code)*(2 + sc_code)*(48 + rccal_code + rccal_code_offset)/32] + cpbf_ccode_offset

The figure below illustrates the RC calibration timing. The calibration sequence is initiated by a TSM control signal (seq_rccal_pup). After the seq_rccal_pup asserts, there is a 1.0us settling time followed by a few clock cycles where rccal_code[4:0] = 5'h1_F. Then, 5 calibration stages are performed, each of which requires 1us + 10clks.

Near the end of each calibration stage, the rccal_cmpout signal from the analog is used to determine whether the appropriate bit of rccal_code[4:0] will be set or cleared. By default, the rccal_cmpout is sampled 2 clk cycles after rccal_sample de-asserts, but there are also programmable options for 1clk and 3clk cycles. Also, there is a programmable option to invert the rccal_cmpout signal if needed.

The total calibration time is 1.0us + 5 clks + 5*(1us + 10clks), which is less than 9us for 26MHz and less than 8us for all other XO reference frequencies supported.

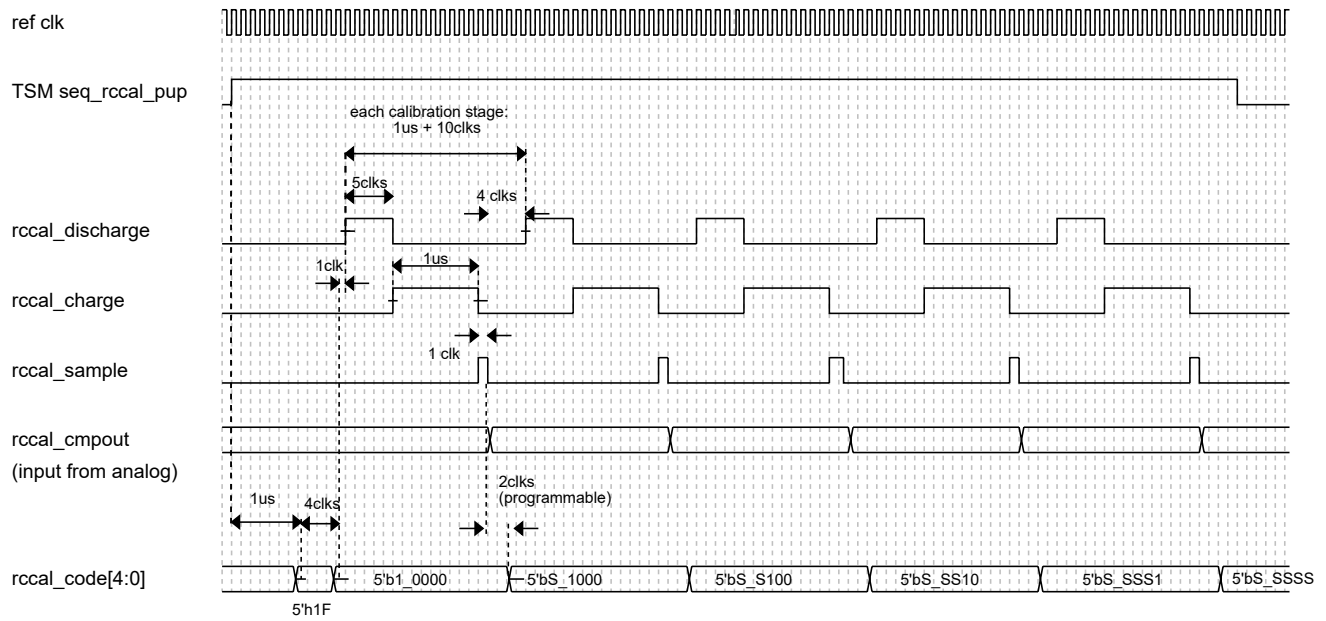


Figure 308. RCCAL timing

55.4.7.4.3 Memory Map and register definition

The 2p4GHz RX_DIG memory map and register description is included in the following section.

55.4.7.4.3.1 2P4GHZ_RX_DIG register descriptions

55.4.7.4.3.1.1 RX_DIG_REG memory map

XCVR_RX_DIG base address: 48A0_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RXDIG Control 0 (CTRL0)	32	RW	2000_3000h
4h	RXDIG Control 0 DRS (CTRL0_DRS)	32	RW	0000_20C0h
8h	RXDIG Control 1 (CTRL1)	32	RW	0000_0180h
Ch	RXDIG DFT Control (DFT_CTRL)	32	RW	0000_0000h
10h	RCCAL Control 0 (RCCAL_CTRL0)	32	RW	0000_002Ah
14h	RCCAL Control 1 (RCCAL_CTRL1)	32	RW	0000_0000h
18h	RCCAL Result (RCCAL_RES)	32	R	0000_4E10h
1Ch	DCOC Control 0 (DCOC_CTRL0)	32	RW	0033_2F56h
20h	DCOC Control 0 DRS (DCOC_CTRL0_DRS)	32	RW	0000_0356h
24h	DCOC CONTROL 1 (DCOC_CTRL1)	32	RW	0000_0000h
28h	DCOC CONTROL 2 (DCOC_CTRL2)	32	RW	0000_0000h
2Ch	DCOC Status (DCOC_STAT)	32	R	0000_2020h
30h	IQ Mismatch Control 0 (IQMC_CTRL0)	32	RW	0400_8000h
34h	IQ Mismatch Control 1 (IQMC_CTRL1)	32	RW	0000_0400h
38h	Acquisition Filter Coeffs 0~3 (ACQ_FILT_0_3)	32	RW	0000_0000h
3Ch	Acquisition Filter Coeffs 4~7 (ACQ_FILT_4_7)	32	RW	0000_0000h
40h	Acquisition Filter Coeffs 8~9 (ACQ_FILT_8_9)	32	RW	0000_0000h
44h	Acquisition Filter Coeffs 10~11 (ACQ_FILT_10_11)	32	RW	0000_0000h
48h	Demod Filter Coeffs 0~1 (DEMOD_FILT_0_1)	32	RW	0000_0000h
4Ch	Demod Filter Coeffs 2~4 (DEMOD_FILT_2_4)	32	RW	0000_0000h
50h	Acquisition Filter Coeffs 0~3 DRS (ACQ_FILT_0_3_DRS)	32	RW	0000_0000h
54h	Acquisition Filter Coeffs 4~7 DRS (ACQ_FILT_4_7_DRS)	32	RW	0000_0000h
58h	Acquisition Filter Coeffs 8~9 DRS (ACQ_FILT_8_9_DRS)	32	RW	0000_0000h
5Ch	Acquisition Filter Coeffs 10~11 DRS (ACQ_FILT_10_11_DRS)	32	RW	0000_0000h
60h	Demod Filter Coeffs 0~1 DRS (DEMOD_FILT_0_1_DRS)	32	RW	0000_0000h
64h	Demod Filter Coeffs 2~4 DRS (DEMOD_FILT_2_4_DRS)	32	RW	0000_0000h
68h	RSSI Global Control (RSSI_GLOBAL_CTRL)	32	RW	0000_0310h
6Ch	Wide-Band RSSI Control (WB_RSSI_CTRL)	32	RW	0022_3222h
70h	Wide-Band RSSI Result 0 (WB_RSSI_RES0)	32	RW	0080_0100h
74h	Wide-Band RSSI Result 1 (WB_RSSI_RES1)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
78h	Wide-Band RSSI DFT Result (WB_RSSI_DFT)	32	R	0000_0000h
7Ch	Narrow-Band RSSI Control 0 (NB_RSSI_CTRL0)	32	RW	0040_0022h
80h	Narrow-Band RSSI Control 1 (NB_RSSI_CTRL1)	32	RW	0000_0000h
84h	Narrow-Band RSSI Result 0 (NB_RSSI_RES0)	32	RW	8080_0100h
88h	Narrow-Band RSSI Result 1 (NB_RSSI_RES1)	32	R	0000_0000h
8Ch	Narrow-Band RSSI DFT Result (NB_RSSI_DFT)	32	R	0000_0000h
90h	AGC Control (AGC_CTRL)	32	RW	805C_CD10h
94h	AGC Control Status (AGC_CTRL_STAT)	32	RW	017E_802Ch
98h	AGC Timing Control 0 (AGC_TIMING0)	32	RW	0810_4398h
9Ch	AGC Timing Control 1 (AGC_TIMING1)	32	RW	1902_440Ah
A0h	AGC Timing Control 2 (AGC_TIMING2)	32	RW	C000_0000h
A4h	AGC Timing Control 0 DRS (AGC_TIMING0_DRS)	32	RW	0000_4000h
A8h	AGC Timing Control 1 DRS (AGC_TIMING1_DRS)	32	RW	0000_040Ah
ACh	AGC Timing Control 2 DRS (AGC_TIMING2_DRS)	32	RW	0000_0000h
B0h	AGC IDX11 Gain Config (AGC_IDX11_GAIN_CFG)	32	RW	0000_7E07h
B4h	AGC IDX10 Gain Config (AGC_IDX10_GAIN_CFG)	32	RW	0000_2E07h
B8h	AGC IDX9 Gain Config (AGC_IDX9_GAIN_CFG)	32	RW	0000_1207h
BCh	AGC IDX8 Gain Config (AGC_IDX8_GAIN_CFG)	32	RW	0000_0607h
C0h	AGC IDX7 Gain Config (AGC_IDX7_GAIN_CFG)	32	RW	0000_0207h
C4h	AGC IDX6 Gain Config (AGC_IDX6_GAIN_CFG)	32	RW	0000_0187h
C8h	AGC IDX5 Gain Config (AGC_IDX5_GAIN_CFG)	32	RW	0000_0107h
CCh	AGC IDX4 Gain Config (AGC_IDX4_GAIN_CFG)	32	RW	0000_01A7h
D0h	AGC IDX3 Gain Config (AGC_IDX3_GAIN_CFG)	32	RW	0000_0127h
D4h	AGC IDX2 Gain Config (AGC_IDX2_GAIN_CFG)	32	RW	0000_00B7h
D8h	AGC IDX1 Gain Config (AGC_IDX1_GAIN_CFG)	32	RW	0000_00B6h
DCh	AGC IDX0 Gain Config (AGC_IDX0_GAIN_CFG)	32	RW	0000_80F6h
E0h	AGC Miscellaneous Gain Config (AGC_MIS_GAIN_CFG)	32	RW	0000_0043h
E4h	AGC IDX11 Gain Value (AGC_IDX11_GAIN_VAL)	32	RW	0000_010Bh
E8h	AGC_IDX10_GAIN_VAL (AGC_IDX10_GAIN_VAL)	32	RW	0000_00EBh
ECh	AGC_IDX9_GAIN_VAL (AGC_IDX9_GAIN_VAL)	32	RW	0000_00CDh
F0h	AGC_IDX8_GAIN_VAL (AGC_IDX8_GAIN_VAL)	32	RW	0000_00AEh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F4h	AGC_IDX7_GAIN_VAL (AGC_IDX7_GAIN_VAL)	32	RW	0000_0096h
F8h	AGC_IDX6_GAIN_VAL (AGC_IDX6_GAIN_VAL)	32	RW	0000_0077h
FCh	AGC_IDX5_GAIN_VAL (AGC_IDX5_GAIN_VAL)	32	RW	0000_005Fh
100h	AGC_IDX4_GAIN_VAL (AGC_IDX4_GAIN_VAL)	32	RW	0000_0049h
104h	AGC_IDX3_GAIN_VAL (AGC_IDX3_GAIN_VAL)	32	RW	0000_0031h
108h	AGC_IDX2_GAIN_VAL (AGC_IDX2_GAIN_VAL)	32	RW	0000_001Ah
10Ch	AGC_IDX1_GAIN_VAL (AGC_IDX1_GAIN_VAL)	32	RW	0000_0002h
110h	AGC_IDX0_GAIN_VAL (AGC_IDX0_GAIN_VAL)	32	RW	0000_03E4h
114h	AGC Fast Mode Threshold (AGC_THR_FAST)	32	RW	01C2_001Eh
118h	AGC Fast Mode Threshold DRS (AGC_THR_FAST_DRS)	32	RW	01C2_001Eh
11Ch	AGC IDX11 Slow Mode Threshold (AGC_IDX11_THR)	32	RW	0190_0000h
120h	AGC IDX10 Slow Mode Threshold (AGC_IDX10_THR)	32	RW	0190_0032h
124h	AGC IDX9 Slow Mode Threshold (AGC_IDX9_THR)	32	RW	0190_0032h
128h	AGC IDX8 Slow Mode Threshold (AGC_IDX8_THR)	32	RW	0190_0032h
12Ch	AGC IDX7 Slow Mode Threshold (AGC_IDX7_THR)	32	RW	0190_0032h
130h	AGC IDX6 Slow Mode Threshold (AGC_IDX6_THR)	32	RW	0190_0032h
134h	AGC IDX5 Slow Mode Threshold (AGC_IDX5_THR)	32	RW	0190_0032h
138h	AGC IDX4 Slow Mode Threshold (AGC_IDX4_THR)	32	RW	0190_0032h
13Ch	AGC IDX3 Slow Mode Threshold (AGC_IDX3_THR)	32	RW	0190_0032h
140h	AGC IDX2 Slow Mode Threshold (AGC_IDX2_THR)	32	RW	0190_0032h
144h	AGC IDX1 Slow Mode Threshold (AGC_IDX1_THR)	32	RW	0190_0032h
148h	AGC IDX0 Slow Mode Threshold (AGC_IDX0_THR)	32	RW	0000_0032h
14Ch	AGC Miscellaneous Thresholds (AGC_THR_MIS)	32	RW	0014_0028h
150h	AGC Override Control (AGC_OVRD)	32	RW	0000_0000h
154h	DC Residual Control (DC_RESID_CTRL)	32	RW	0000_0000h
158h	DC Residual Control2 (DC_RESID_CTRL2)	32	RW	0000_0000h
15Ch	DC Residual Control DataRate1 (DC_RESID_CTRL_DRS)	32	RW	0000_0000h
160h	DC Residual Estimate (DC_RESID_EST)	32	R	0000_0000h
164h	DfT tone analyzer (DFT_TONE_ANALYZER0)	32	RW	0000_A000h
168h	DfT tone analyzer (DFT_TONE_ANALYZER1)	32	RW	0000_0000h
16Ch	DfT tone analyzer (DFT_TONE_ANALYZER2)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
170h	DfT tone analyzer (DFT_TONE_ANALYZER3)	32	RW	0000_0000h
174h	DCOC Digital Correction Result (DCOC_DIG_CORR_RESULT)	32	R	0000_0000h

55.4.7.4.3.1.2 RXDIG Control 0 (CTRL0)

Offset

Register	Offset
CTRL0	0h

Function

RXDIG control register 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DR_OVRD...	RX_AGC_...	CIC_CNT...	0	RX_FSK_...	0	RX_IQ_8B_OUT_MODE			RX_SRC_...	RX_ACQ_...	RX_ACQ_...	0	RX_DIG_GAIN		
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CIC_RATE			CIC_ORD...	DIG_MIXER_FREQ									RX_IQMC...	ADC_CLI...
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DR_OVRD_IN_CTE	DATARATE_CONFIG_SEL Override In CTE For debugging purpose. Keep 0 for the normal case.
30 RX_AGC_EN	AGC Enable 0b - AGC is disabled 1b - AGC is enabled
29	CIC Dec Counter Free Run Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CIC_CNTR_FREE_RUN_EN	This bit is for RSM usecase. If this bit set, during rsm_active=1, the CIC decimation counter will keep running even rx_dig_en=0. In addition, the rx_init will not reset the CIC decimation counter.
28 —	Reserved
27 RX_FSK_ZB_SEL	PHY/Demodulator selection 0b - 2.4GHz PHY is selected 1b - 15.4 PHY is selected
26 —	Reserved
25-23 RX_IQ_8B_OUT_MODE	RX 8-bit IQ Output Mode The 8-bit I/Q source is selected by DFT_CTRL[DFT_RX_IQ_OUT_SEL] 000b - Disable 8-bit IQ output 001b - {I[10],I[9:3]}, {Q[10],Q[9:3]} 010b - {I[10],I[8:2]}, {Q[10],Q[8:2]} 011b - {I[10],I[7:1]}, {Q[10],Q[7:1]} 100b - Dynamic scaling
22 RX_SRC_EN	RX Sample Rate Converter Enable When use 26MHz reference clock, SRC should be enabled. 0b - SRC is disabled. 1b - SRC is enabled.
21 RX_ACQ_FILT_BYPASS	Acquisition Filter Bypass This bit is for test purpose and should not be set in normal mode. 0b - Acquisition filter is enabled 1b - Acquisition filter is bypassed
20 RX_ACQ_FILT_LEN	Acquisition Filter Length When set 1, only ACQ_FILT_H4 - ACQ_FILT_H11 are used. 0b - Acquisition filter length is 24. 1b - Acquisition filter length is 16.
19 —	Reserved
18-16	RX Digital Gain Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
RX_DIG_GAIN	000b - Digital gain value is 1.000. 001b - Digital gain value is 1.125. 010b - Digital gain value is 1.250. 011b - Digital gain value is 1.375. 100b - Digital gain value is 1.500. 101b - Digital gain value is 1.625. 110b - Digital gain value is 1.750. 111b - Digital gain value is 1.875.
15 —	Reserved
14-12 CIC_RATE	CIC Decimation Rate The actual CIC decimation Rate is 2^CIC_RATE. Note: check CIC_ORDER description for more information. 000b - Decimation Rate is 1. 001b - Decimation Rate is 2. 010b - Decimation Rate is 4. 011b - Decimation Rate is 8. 100b - Decimation Rate is 16. 101b - Decimation Rate is 32. 110b - Reserved 111b - Reserved
11 CIC_ORDER	CIC Order(Stage) Selection The programmed value of CIC_ORDER and the CIC decimation rate(dec_rate) should conform to the formula: (4-CIC_ORDER)*log2(dec_rate) <= 16 0b - 4-stage CIC 1b - 3-stage CIC
10-2 DIG_MIXER_FREQ	Digital Mixer Frequency Represents the ratio of the digital mixer frequency to the sample rate frequency at the digital mixer (which is at the CIC output rate). Supported rations are in the range (-0.25, 0.25). Computed as nearest integer to 1024*desired_mixer_freq/mixer_sample_rate_frequency. Check Digital IF Mixer section for more information.
1 RX_IQMC_EN	IQ Mismatch Compensation Enable NOTE: When IQMC_CTRL0[IQMC_CAL_EN]=1, RX_IQMC_EN should also be set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - IQ mismatch compensation is disabled. 1b - IQ mismatch compensation is enabled.
0 ADC_CLIP_EN	ADC Output Clip Enable When enabled, RXDIG input adc_i and adc_q value will be saturated to +-512. 0b - ADC clip is disabled. 1b - ADC clip is enabled.

55.4.7.4.3.1.3 RXDIG Control 0 DRS (CTRL0_DRS)

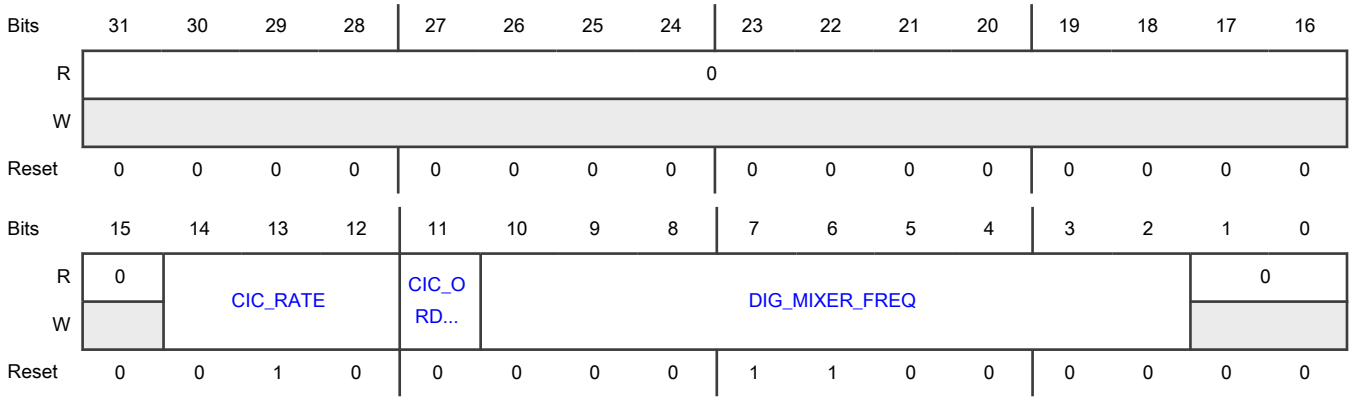
Offset

Register	Offset
CTRL0_DRS	4h

Function

When datarate_config_sel=1, the RX_DIG will use these config values instead of CTRL0.

Diagram



Fields

Field	Function
31-15 —	Reserved
14-12 CIC_RATE	CIC Decimation Rate

Table continues on the next page...

Table continued from the previous page...

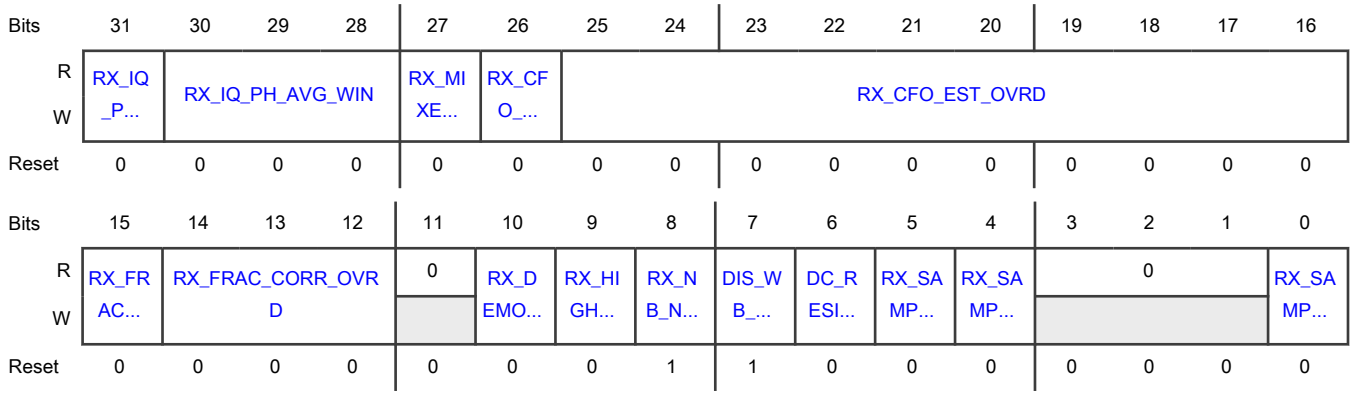
Field	Function
	The actual CIC decimation Rate is 2^CIC_RATE. Note: check CIC_ORDER description to get more information. 000b - Decimation Rate is 1. 001b - Decimation Rate is 2. 010b - Decimation Rate is 4. 011b - Decimation Rate is 8. 100b - Decimation Rate is 16. 101b - Decimation Rate is 32. 110b - Reserved 111b - Reserved
11 CIC_ORDER	CIC Order(Stage) Selection The programmed value of CIC_ORDER and the CIC decimation rate(dec_rate) should conform to the following formula: (4-CIC_ORDER)*log2(dec_rate) <= 16 0b - 4-stage CIC 1b - 3-stage CIC
10-2 DIG_MIXER_FREQ	Digital Mixer Frequency Represents the ratio of the digital mixer frequency to the sample rate frequency at the digital mixer (which is at the CIC output rate). Supported rations are in the range (-0.25, 0.25). Computed as nearest integer to 1024*desired_mixer_freq/mixer_sample_rate_frequency. Check Digital IF Mixer section for more information.
1-0 —	Reserved

55.4.7.4.3.1.4 RXDIG Control 1 (CTRL1)

Offset

Register	Offset
CTRL1	8h

Diagram



Fields

Field	Function
31 RX_IQ_PH_OUTPUT_COND	RX IQ or Phase Output Conditioning This field only has effect when DFT_CTRL[DFT_RX_IQ_OUT_SEL], DFT_CTRL[DFT_RX_PH_OUT_SEL] set a non-zero value or DFT_CTRL[DFT_RSSI_MAG_OUT_SEL]=7. When this bit cleared, the RX_DIG will output the selected IQ and/or Phase, Mag signal to DTEST and/or DMA_DEBUG as soon as RX_DIG starts work. For localization application, set this bit can let RX_DIG send the IQ and/or Phase signal to DTEST and/or DMA_DEBUG only during localization(AoA/AoD/HADM/PDE) sample slot. It will reduce the packet buffer usage and software loading. NOTE: When this bit set, should keep DMA_SIGNAL_VALID_MASK_EN and DBG_SIGNAL_VALID_MASK_EN cleared to avoid data capture missing. 0b - Output IQ and/or Phase all-time 1b - Only output IQ and/or Phase during localization sample slot
30-28 RX_IQ_PH_AVG_WIN	RX IQ Phase Output Average Window Config Programing a non-zero value to this field will apply a window averager on RX_DIG output IQ, Phase signal, or Mag. The actual average window size is 2^(RX_IQ_PH_AVG_WIN+1). NOTE: Programing a non-zero value to this field will reduce the RX IQ and/or Phase output rate to 1/2^(RX_IQ_PH_AVG_WIN+1). 000b - Disable RX IQ and/or Phase output average function 001b - Average window size = 4 010b - Average window size = 8 011b - Average window size = 16 100b - Average window size = 32 101b - Average window size = 64 110b - Average window size = 128 111b - Average window size = 256
27	RX_DIG Mixer Index Output Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
RX_MIXER_IDX_OUT_MODE	RX_DIG has 10-bit mixer(s) index output to DMA_DEBUG. By default, the 10-bit output is composed by rx_if_mixer_idx[9:5] and rx_cfo_mixer[7:3]. By setting RX_MIXER_IDX_OUT_MODE, the RX_DIG will send rx_if_mixer_idx[9:0] to DMA_DEBUG.
26 RX_CFO_EST_OVRD_EN	CFO Estimation Override Enable When this bit set, the CFO mixer will use RX_CFO_EST_OVRD value to do CFO compensation. 0b - CFO override is enabled 1b - CFO override is disabled
25-16 RX_CFO_EST_OVRD	CFO Estimation Override Value This field only valid when RX_CFO_EST_OVRD_EN=1
15 RX_FRAC_CORR_OVRD_EN	Fractional Correction Coefficient Override Enable When this bit set, the Fractional correction module will use RX_FRAC_CORR_OVRD value to do correction.
14-12 RX_FRAC_CORR_OVRD	Fractional Correction Coefficient Override Value This field only valid when RX_FRAC_CORR_OVRD_EN=1
11 —	Reserved
10 RX_DEMOD_FILT_BYPASS	Demod Channel Filter Bypass This bit is for test purpose and should not be set in normal mode. 0b - Demod channel filter is enabled. 1b - Demod channel filter is bypassed.
9 RX_HIGH_RES_NORM_SEL	High Resolution Phase Source Select 0b - From RX_NORM_NB 1b - From RX_NORM_WB
8 RX_NB_NORM_EN	Narrow-Band Normalizer Enable 0b - Narrow-Band normalizer is disabled. 1b - Narrow-Band normalizer is enabled.
7 DIS_WB_NORM_AA_FOUND	Disable WB-NORM when AA found When this bit set, RX_DIG will disable the WB-NORM as soon as AA(access address) found by 2.4GHz PHY to save power. NOTE: when using 15p4 PHY, this bit should be cleared.
6 DC_RESID_EN	DC_RESID Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 RX_SAMPLE_BUF_AUTO_GATE	Sample Buffer Automatically Gate Off When enabled, after PHY_AA_MATCH asserted, unused taps of sample buffer will be gated off to save power. Here, unused taps is controlled by 2.4GHz PHY.
4 RX_SAMPLE_BUF_BYPASS_IN_CTE	Bypass Sample Buffer During CTE If enabled, during CTE field receiving, sample buffer will be bypassed to save power and reduce latency.
3-1 —	Reserved
0 RX_SAMPLE_BUF_BYPASS	Bypass Sample Buffer This bit is for test or debug purpose, should not be set in normal mode.

55.4.7.4.3.1.5 RXDIG DFT Control (DFT_CTRL)

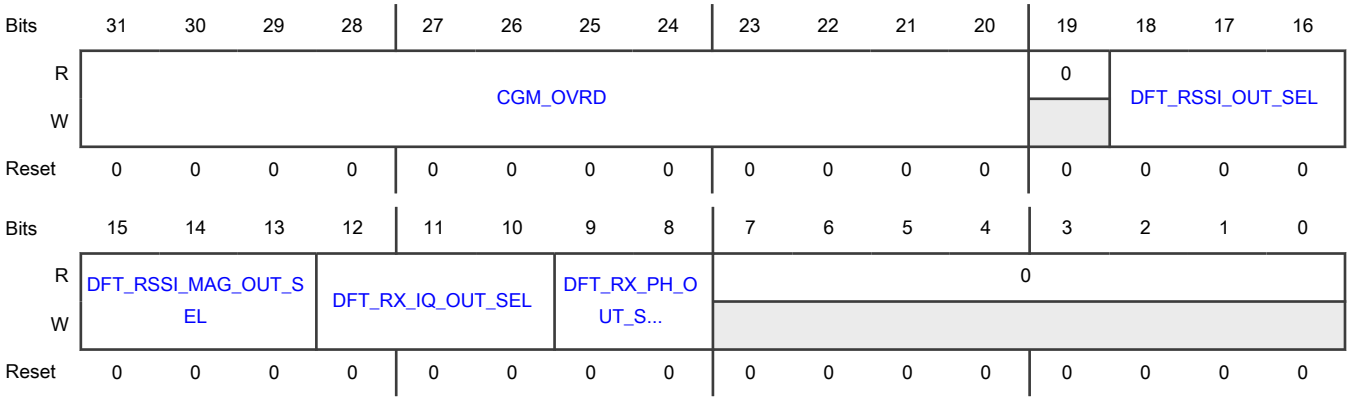
Offset

Register	Offset
DFT_CTRL	Ch

Function

RXDIG test control register.

Diagram



Fields

Field	Function
31-20 CGM_OVRD	CGM Override RXDIG sub-modules clock gating control overrides. If a bit is cleared, the clock is enabled only for the clock cycles when it is required during RX; If the bit is set, the clock is continuously enabled during rx_dig_en=1. 0000_0000_0001b - RCCAL 0000_0000_0010b - DCOC 0000_0000_0100b - IF_MIXER 0000_0000_1000b - CIC 0000_0001_0000b - ACQ_CHF 0000_0010_0000b - SRC 0000_0100_0000b - SAMPLE_BUF and CFO_MIXER 0000_1000_0000b - DEMOD_CHF and FRAC_CORR 0001_0000_0000b - NB_NORM and HIGH_RES_NORM 0010_0000_0000b - AGC 0100_0000_0000b - IQ_MISMATCH 1000_0000_0000b - DIG_GAIN
19 —	Reserved
18-16 DFT_RSSI_OUT_SEL	DFT RSSI Result Output Selection WB-RSSI and NB-RSSI results output to DTEST/DMA_DEBUG selection. 000b - Disable RSSI output 001b - Wide-band RSSI_RAW output 010b - Wide-band RSSI output 011b - Narrow-band RSSI_RAW output 100b - Narrow-band RSSI output 101b - Narrow-band NOISE_RAW output 110b - Narrow-band SNR output 111b - Narrow-band LQI output
15-13 DFT_RSSI_MAG_OUT_SEL	DFT RSSI Magnitude Output Selection WB-RSSI and NB-RSSI magnitude output to DTEST/DMA_DEBUG selection. NOTE: When DFT_RSSI_MAG_OUT_SEL=7, the Mag is calucated based on the IQ signal selected by DFT_RX_IQ_OUT_SEL and send to Mag averager. The typical usecase is select SRC out IQ with WB-Phase or Frac out IQ with NB-Phase to make sure the mag and phase is aligned in the MAG-PHASE page of DMA_DEBUG.

Table continues on the next page...

Table continued from the previous page...

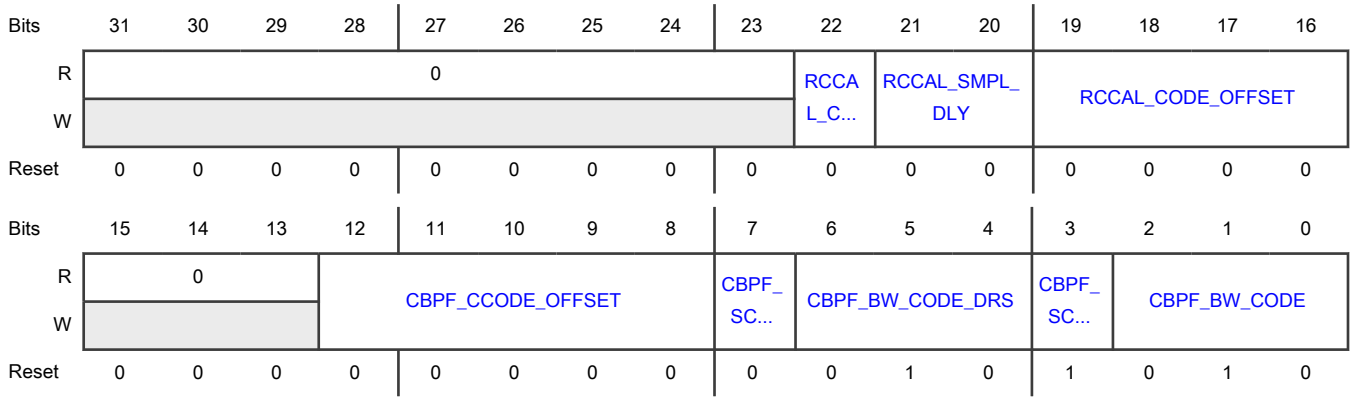
Field	Function
	000b - Disabled 001b - WB-RSSI fast magnitude 010b - WB-RSSI slow magnitude 011b - NB-RSSI mag IIR 100b - NB-RSSI mag avg 101b - NB-RSSI noise mag IIR 110b - NB-RSSI noise mag avg 111b - DFT_RX_IQ_OUT mag
12-10 DFT_RX_IQ_OUT_SEL	DFT I/Q Output Selection RXDIG sub-module's I/Q output to DTEST/DMA_DEBUG selection. NOTE: The selected DFT_RX_IQ signal is also used as the input of DFT_TONE_ANALYZER 000b - Disabled 001b - Select IF_MIXER 010b - Select CIC 011b - Select ACQ channel filter 100b - Select SRC 101b - Select CFO_MIXER 110b - Select FRAC_CORR 111b - Select DC_RESID
9-8 DFT_RX_PH_OUT_SEL	DFT RXDIG Phase Output Selection RXDIG normalizers output to DTEST/DMA_DEBUG selection 00b - Disable DFT phase output 01b - Sel wide-band phase output 10b - Sel narrow-band phase output 11b - Disable DFT phase output
7-0 —	Reserved

55.4.7.4.3.1.6 RCCAL Control 0 (RCCAL_CTRL0)

Offset

Register	Offset
RCCAL_CTRL0	10h

Diagram



Fields

Field	Function
31-23 —	Reserved
22 RCCAL_CMPOUT_INV	RCCAL Comparator Output Invert If set, the RCCAL comparator output from analog is inverted before it is used to determine the RCCAL code value
21-20 RCCAL_SMPL_DLY	RCCAL Sample Delay Indicates the number of clk cycles after sample vld is deasserted when the rccal comparator output is captured 00b - 2 cycles (default) 01b - 1 cycle 10b - 2 cycles 11b - 3 cycles
19-16 RCCAL_CODE_OFFSET	RCCAL_CODE Offset This signed value is used as an offset from the calculated RCCAL_CODE value in the computation of the CBPF_CCODE value
15-13 —	Reserved
12-8 CBPF_CCODE_OFFSET	CBPF_CCODE Offset This signed value is used as an offset to CBPF_CCODE from the calculated RCCAL_CODE value
7 CBPF_SC_CODE_DRS	When datarate_config_sel=1, will choose this value instead of CBPF_SC_CODE

Table continues on the next page...

Table continued from the previous page...

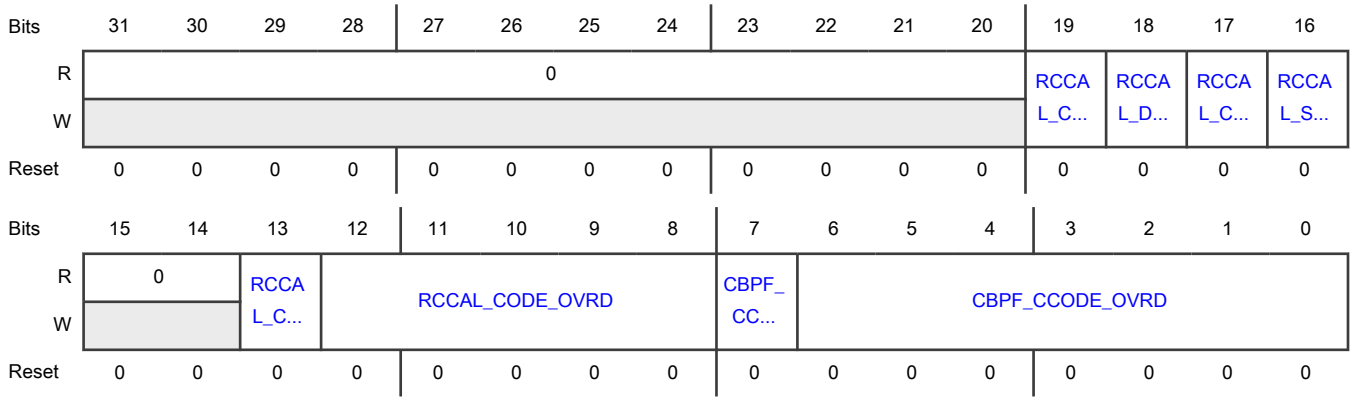
Field	Function
6-4 CBPF_BW_CODE_DRS	When datarate_config_sel=1, will choose this value instead of CBPF_BW_CODE
3 CBPF_SC_CODE	See detail in CBPF_BW_CODE description
2-0 CBPF_BW_CODE	CBPF BW_CODE BW_CODE and SC_CODE used in calculation of the CBPF_CCODE and determine the bandwidth of CBPF. When CBPF_SC_CODE=1 --> 0 (1007.2kHz), 1 (824kHz), 2 (697.6kHz), 3 (604.8kHz),4 (533.6kHz). When CBPF_SC_CODE=0 --> 0 (1511.2kHz), 1 (1236kHz), 2 (1046.4kHz),3 (906.4kHz),4 (800kHz).

55.4.7.4.3.1.7 RCCAL Control 1 (RCCAL_CTRL1)

Offset

Register	Offset
RCCAL_CTRL1	14h

Diagram



Fields

Field	Function
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

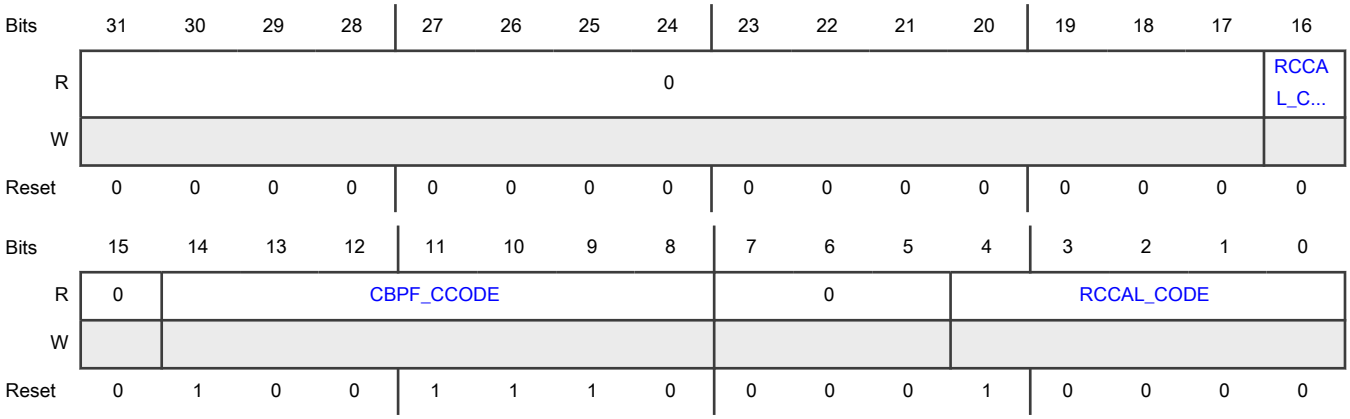
Field	Function
19 RCCAL_CTRL_OVRD_EN	RCCAL Control Signals Override Enable If set, the RCCAL_CHARGE_OVRD, RCCAL_DISCHARGE_OVRD, and RCCAL_SAMPLE_OVRD are used instead of being controlled by RCCAL logic
18 RCCAL_DISCHARGE_OVRD	RCCAL_DISCHARGE Override Value If RCCAL_CTRL_OVR_EN=1, this value is used for RCCAL_DISCHARGE signal instead of being controlled by the RCCAL logic
17 RCCAL_CHARGE_OVRD	RCCAL_CHARGE Override Value If RCCAL_CTRL_OVR_EN=1, this value is used for RCCAL_CHARGE signal instead of being controlled by the RCCAL logic
16 RCCAL_SAMPLE_OVRD	RCCAL_SAMPLE Override Value If RCCAL_CTRL_OVR_EN=1, this value is used for RCCAL_SAMPLE signal instead of being controlled by the RCCAL logic
15-14 —	Reserved
13 RCCAL_CODE_OVRD_EN	RCCAL_CODE Override Enable If set, the RCCAL_CODE_OVRD value is used for RCCAL_CODE signal instead of the value calculated by RCCAL logic
12-8 RCCAL_CODE_OVRD	RCCAL_CODE Override Value If RCCAL_CODE_OVR_EN=1, this value is used for RCCAL_CODE instead of the value calculated by RCCAL logic
7 CBPF_CCODE_OVRD_EN	CBPF_CCODE Override Enable If set, the CBPF_CCODE_OVRD value is used for CBPF_CCODE instead of the value calculated by RCCAL logic
6-0 CBPF_CCODE_OVRD	CBPF_CCODE Override Value If CBPF_CCODE_OVR_EN=1, this value is used for CBPF_CCODE instead of the value calculated by RCCAL logic

55.4.7.4.3.1.8 RCCAL Result (RCCAL_RES)

Offset

Register	Offset
RCCAL_RES	18h

Diagram



Fields

Field	Function
31-17 —	Reserved
16 RCCAL_CMPOUT	RCCAL CMPOUT Instantaneous value of the RCCAL_CMPOUT signal
15 —	Reserved
14-8 CBPF_CCODE	CBPF_CCODE The CBPF_CCODE value determined by the RCCAL logic. (When CBPF_CCODE_OVRD_EN=1, this value reflects CBPF_CCODE_OVRD.). This result will be kept until the next RX sequence started.
7-5 —	Reserved
4-0 RCCAL_CODE	RCCAL_CODE The RCCAL_CODE determined by the RCCAL logic. This result will be kept until the next RX sequence started.

55.4.7.4.3.1.9 DCOC Control 0 (DCOC_CTRL0)

Offset

Register	Offset
DCOC_CTRL0	1Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DCOC_CB...	DCOC_CB...	DCOC_CB...	DCOC_AD...	DCOC_DA...	DCOC_DI...	DCOC_AV...	DCOC_CA...	DCOC_SAR_STL_TIME				DCOC_CBPF_STL_TIME			
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DCOC_PU...	DCOC_DA...	DCOC_Q_...	DCOC_I_...	DCOC_SF...	DCOC_SF...	DCOC_SF...	DCOC_SF...	DCOC_SFQQ				DCOC_SFII			
Reset	0	0	1	0	1	1	1	1	0	1	0	1	0	1	1	0

Fields

Field	Function
31 DCOC_CBPF_HIZ_SHORT_OVRD_EN	DCOC CBPF HIZ SHORT Override Enable If this bit is set, then the CBPF_HIZ and CBPF_SHORT signals are controlled by the DCOC_CBPF_HIZ_OVRD and DCOC_CBPF_SHORT_OVRD bits, respectively. Otherwise, the DCOC controls these signals which are used during DCOC calibration.
30 DCOC_CBPF_HIZ_OVRD	DCOC CBPF_HIZ Override Value
29 DCOC_CBPF_SHORT_OVRD	DCOC CBPF_SHORT Override Value
28 DCOC_ADC_OFFSET_OVRD_EN	DCOC_ADC_OFFSET_OVRD_EN If set, the DCOC_ADC_OFFSET_OVRD_I and DCOC_ADC_OFFSET_OVRD_Q values in the DCOC_CTRL2 register are used instead of the values found during DCOC calibration
27 DCOC_DAC_OVRD_EN	DCOC_DAC_OVRD_EN If set, the DCOC_DAC_OVRD_I and DCOC_DAC_OVRD_Q values in the DCOC_CTRL2 register are used instead of the values found during DCOC calibration
26 DCOC_DIG_CORR_EN	DCOC Digital Correction Enable Compensate the residual DC after CBPF DAC calibration done. NOTE: 1. The measurement of digital correction value is always enabled and the result can be read from DCOC_DIG_CORR_RESULT register. 2. When fast RX warmup enabled, the digital measurement stage will be skipped.
25 DCOC_AVG_WIN	DCOC Average Window Select Sample average window size for ADC, DAC canlibration, and digital residual DC measurement. NOTE: If set this bit, the TSM timing of RX warmup and DCOC cal duration need to extend by 2us(32MHz ref clock) or 3us(26MHz ref clock)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - 4-sample 1b - 8-sample
24 DCOC_CAL_USE_OFFSET	Apply dcoc_i/qcbpf_offset during DCOC calibration. 0b - Do not apply dcoc_i/qcbpf_offset during DCOC calibration 1b - Apply dcoc_i/qcbpf_offset during DCOC calibration
23-20 DCOC_SAR_STL_TIME	DCOC CBPF Settle Time DCOC SAR Settle time in us
19-16 DCOC_CBPF_STL_TIME	DCOC CBPF Settle Time DCOC CBPF Settle time in us
15 DCOC_PULSE_CAPCODE	
14 DCOC_DAC_ORDER	DCOC_DAC_ORDER 0b - DCOC I DAC is calibrated first 1b - DCOC Q DAC is calibrated first
13 DCOC_Q_CAL_POL	DCOC_Q_CAL_POL Polarity of the DCOC Q DAC
12 DCOC_I_CAL_POL	DCOC_I_CAL_POL Polarity of the DCOC I DAC
11 DCOC_SFQI	DCOC_SFQI The SFQI parameter used in DCOC calibration.
10 DCOC_SFIQ	DCOC_SFIQ The SFIQ parameter used in DCOC calibration.
9 DCOC_SFQQP	DCOC_SFQQP If set, the SFQQ is negated
8 DCOC_SFIIP	DCOC_SFIIP If set, the SFII is negated

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 DCOC_SFQQ	DCOC_SFQQ The SFQQ parameter used in DCOC calibration. Unsigned
3-0 DCOC_SFII	DCOC_SFII The SFII parameter used in DCOC calibration. Unsigned

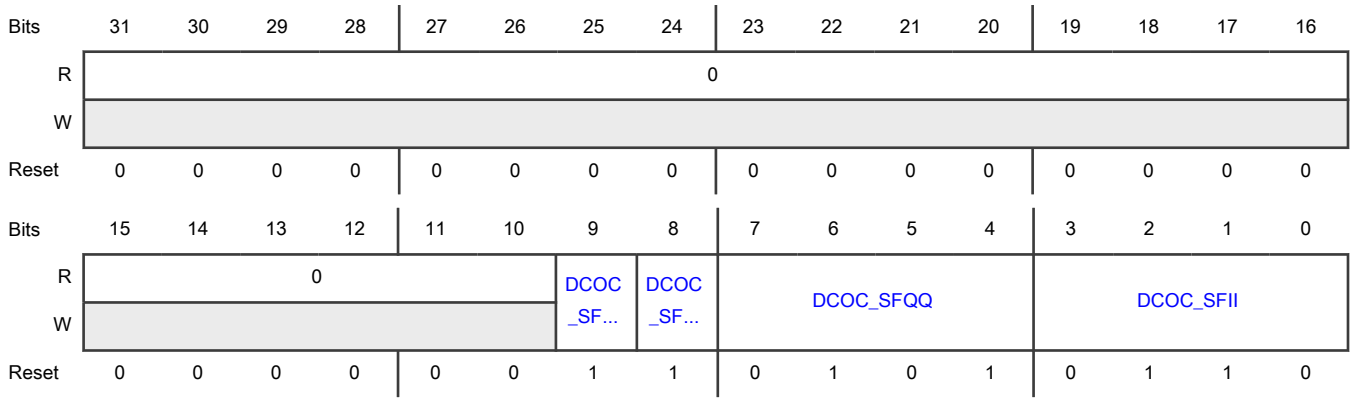
55.4.7.4.3.1.10 DCOC Control 0 DRS (DCOC_CTRL0_DRS)

Offset

Register	Offset
DCOC_CTRL0_DRS	20h

Function
When datarate_config_sel=1, will use this register instead of DCOC_CTRL0

Diagram



Fields

Field	Function
31-10 —	Reserved
9 DCOC_SFQQP	DCOC_SFQQP If set, the SFQQ is negated
8 DCOC_SFIIP	DCOC_SFIIP If set, the SFII is negated

Table continues on the next page...

Table continued from the previous page...

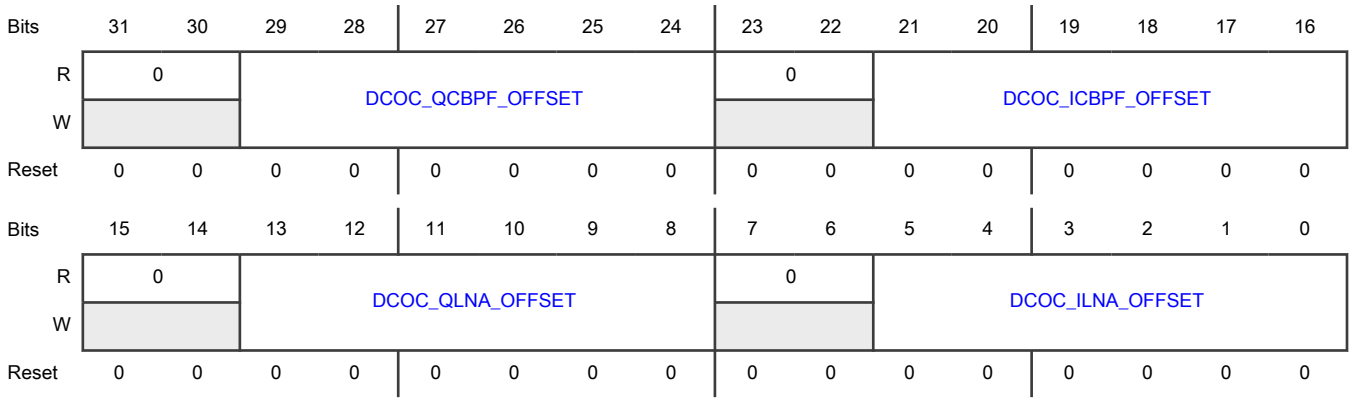
Field	Function
7-4 DCOC_SFQQ	DCOC_SFQQ The SFQQ parameter used in DCOC calibration. Unsigned
3-0 DCOC_SFII	DCOC_SFII The SFII parameter used in DCOC calibration.Unsigned

55.4.7.4.3.1.11 DCOC CONTROL 1 (DCOC_CTRL1)

Offset

Register	Offset
DCOC_CTRL1	24h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 DCOC_QCBPF_OFFSET	DCOC_QCBPF_OFFSET This signed value is scaled by the BPF gain and used as part of the ADC Q channel offset
23-22 —	Reserved
21-16	DCOC_ICBPF_OFFSET

Table continues on the next page...

Table continued from the previous page...

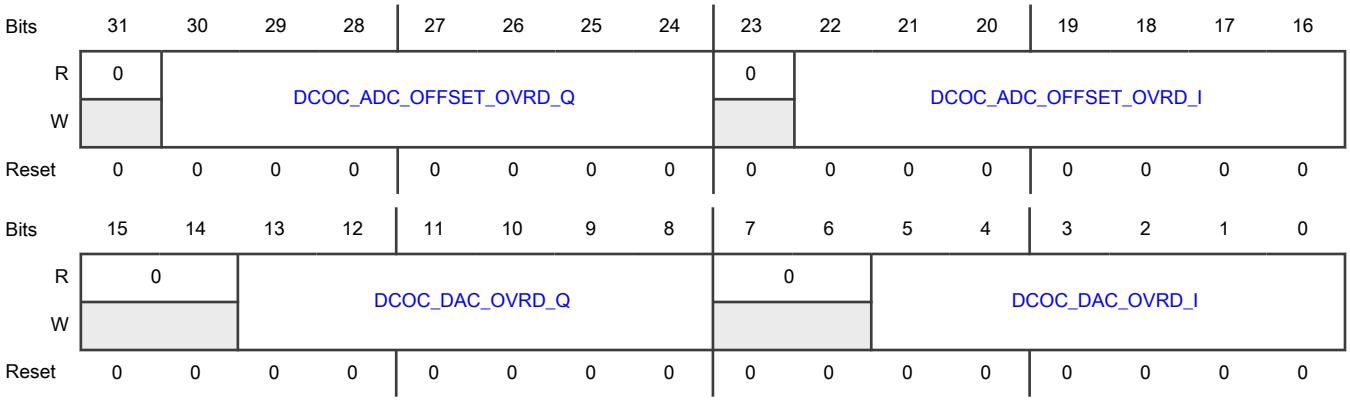
Field	Function
DCOC_ICBPF_OFFSET	This signed value is scaled by the BPF gain and used as part of the ADC I channel offset
15-14 —	Reserved
13-8 DCOC_QLNA_OFFSET	DCOC_QLNA_OFFSET This signed value is scaled by the LNA gain and used as part of the ADC Q channel offset
7-6 —	Reserved
5-0 DCOC_ILNA_OFFSET	DCOC_ILNA_OFFSET This signed value is scaled by the LNA gain and used as part of the ADC I channel offset

55.4.7.4.3.1.12 DCOC CONTROL 2 (DCOC_CTRL2)

Offset

Register	Offset
DCOC_CTRL2	28h

Diagram



Fields

Field	Function
31	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30-24 DCOC_ADC_OFFSET_OVRD_Q	DCOC_ADC_OFFSET_OVRD_Q If DCOC_ADC_OFFSET_OVRD_EN is set (see DCOC_CTRL0 register), this value is used for the ADC Q channel offset instead of the value found during calibration
23 —	Reserved
22-16 DCOC_ADC_OFFSET_OVRD_I	DCOC_ADC_OFFSET_OVRD_I If DCOC_ADC_OFFSET_OVRD_EN is set (see DCOC_CTRL0 register), this value is used for the ADC I channel offset instead of the value found during calibration
15-14 —	Reserved
13-8 DCOC_DAC_OVRD_Q	DCOC_DAC_OVRD_Q If DCOC_DAC_OVRD_EN is set (see DCOC_CTRL0 register), this value is used for the DCOC Q channel DAC instead of the value found during calibration
7-6 —	Reserved
5-0 DCOC_DAC_OVRD_I	DCOC_DAC_OVRD_I If DCOC_DAC_OVRD_EN is set (see DCOC_CTRL0 register), this value is used for the DCOC I channel DAC instead of the value found during calibration

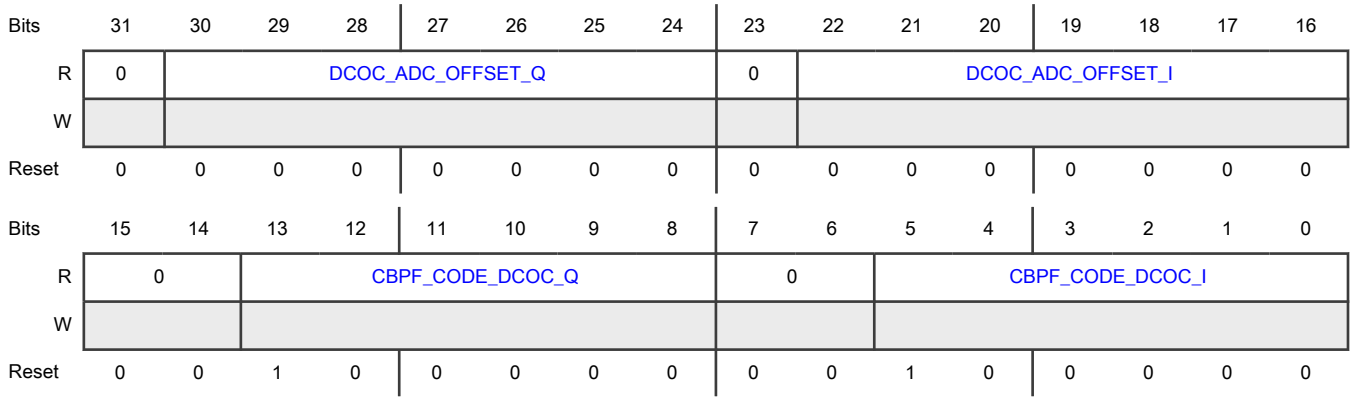
55.4.7.4.3.1.13 DCOC Status (DCOC_STAT)

Offset

Register	Offset
DCOC_STAT	2Ch

Function
DCOC_STAT

Diagram



Fields

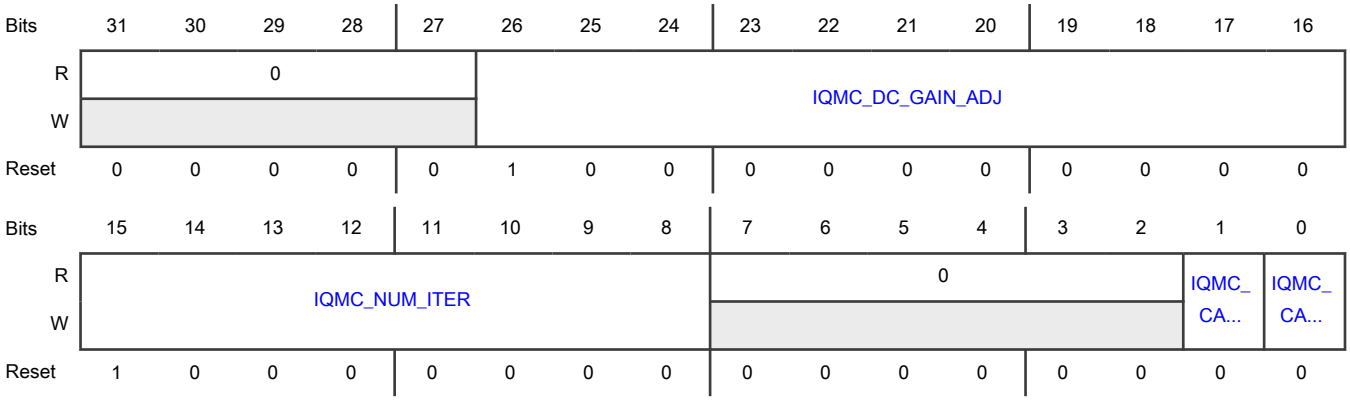
Field	Function
31 —	Reserved
30-24 DCOC_ADC_OFFSET_Q	DCOC_ADC_OFFSET_Q The value of the ADC Q channel offset found during DCOC calibration. This result will be kept until the next DCOC calibration.
23 —	Reserved
22-16 DCOC_ADC_OFFSET_I	DCOC_ADC_OFFSET_I The value of the ADC I channel offset found during DCOC calibration. This result will be kept until the next DCOC calibration.
15-14 —	Reserved
13-8 CBPF_CODE_DCOC_Q	CBPF_CODE_DCOC_Q The current value of the DCOC Q channel DAC. If DCOC_DAC_OVRD_EN=0, this is the value found during DCOC calibration, and this result will be kept until the next DCOC calibration. If DCOC_DAC_OVRD_EN=1, this reflects the DCOC_DAC_OVRD_Q override values.
7-6 —	Reserved
5-0 CBPF_CODE_DCOC_I	CBPF_CODE_DCOC_I The current value of the DCOC I channel DAC. If DCOC_DAC_OVRD_EN=0, this is the value found during DCOC calibration, and this result will be kept until the next DCOC calibration. If DCOC_DAC_OVRD_EN=1, this reflects the DCOC_DAC_OVRD_I override values.

55.4.7.4.3.1.14 IQ Mismatch Control 0 (IQMC_CTRL0)

Offset

Register	Offset
IQMC_CTRL0	30h

Diagram



Fields

Field	Function
31-27 —	Reserved
26-16 IQMC_DC_GAIN_ADJ	Not currently used.
15-8 IQMC_NUM_ITER	IQ Mismatch Cal Num Iter
7-2 —	Reserved
1 IQMC_CAL_FREQ_SEL	IQMC_CAL_FREQ_SEL Clock frequency selection for IQ mismatch calibration. Note, the IQMC module will work at reference clock during correction. 0b - Reference clk divided by 2 1b - Reference clk divided by 4
0	IQ Mismatch Cal Enable

Table continues on the next page...

Table continued from the previous page...

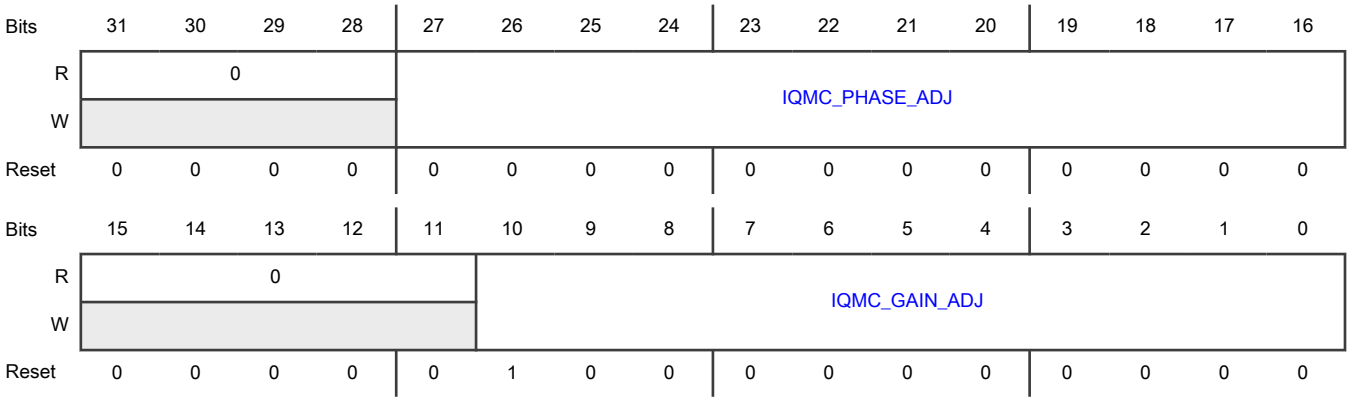
Field	Function
IQMC_CAL_EN	Enables IQ mismatch calibration. This bit is self-clearing; it will clear automatically at the end of the calibration sequence. NOTE: To start IQ mismatch calibration, should set IQMC_CAL_EN and CTRL0[RX_IQMC_EN] together.

55.4.7.4.3.1.15 IQ Mismatch Control 1 (IQMC_CTRL1)

Offset

Register	Offset
IQMC_CTRL1	34h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 IQMC_PHASE_ADJ	IQ Mismatch Correction Phase Coeff I/Q mismatch correction phase coefficient. This value is updated automatically at the end of the IQMC calibration sequence. It can also be written by software. The format is signed, with the maximum positive value 0x7ff corresponding to a phase coefficient of 0.25, and the maximum negative value 0x800 corresponding to -0.25. The reset value of 0x000 corresponds to a phase coefficient of 0.
15-11 —	Reserved
10-0 IQMC_GAIN_ADJ	IQ Mismatch Correction Gain Coeff I/Q mismatch correction gain coefficient. This value is updated automatically at the end of the IQMC calibration sequence. It can also be written by software. The format is u1.10. The reset value of 0x400 corresponds to a gain coefficient of 1.0.

55.4.7.4.3.1.16 Acquisition Filter Coeffs 0~3 (ACQ_FILT_0_3)

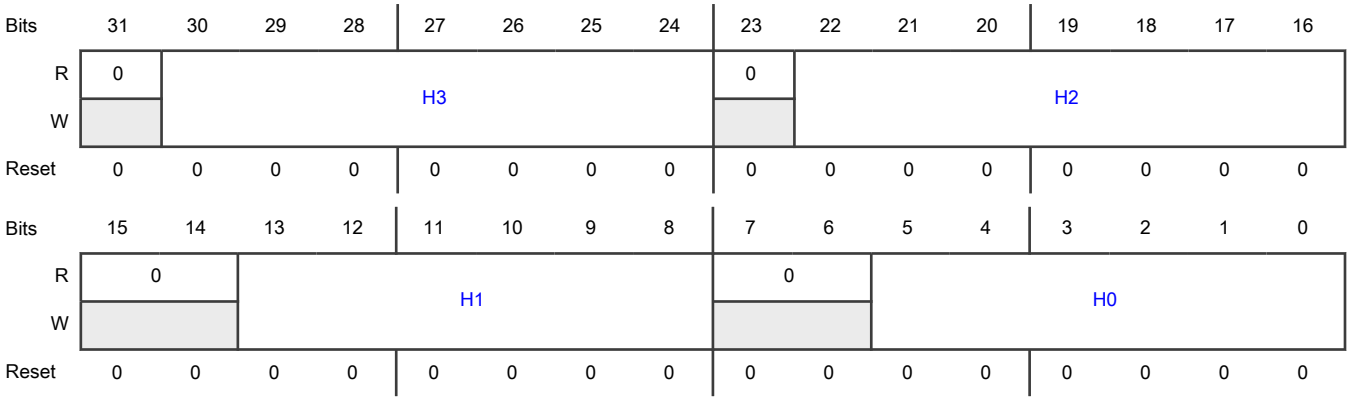
Offset

Register	Offset
ACQ_FILT_0_3	38h

Function

Acquisition filter coefficient 0~3

Diagram



Fields

Field	Function
31 —	Reserved
30-24 H3	Acquisition Filter Coefficient 3 Acquisition filter coefficient 3, 7-bit signed fractional
23 —	Reserved
22-16 H2	Acquisition Filter Coefficient 2 Acquisition filter coefficient 2, 7-bit signed fractional
15-14 —	Reserved
13-8 H1	Acquisition Filter Coefficient 1 Acquisition filter coefficient 1, 6-bit signed fractional
7-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 H0	Acquisition Filter Coefficient 0 Acquisition filter coefficient 0, 6-bit signed fractional

55.4.7.4.3.1.17 Acquisition Filter Coeffs 4~7 (ACQ_FILT_4_7)

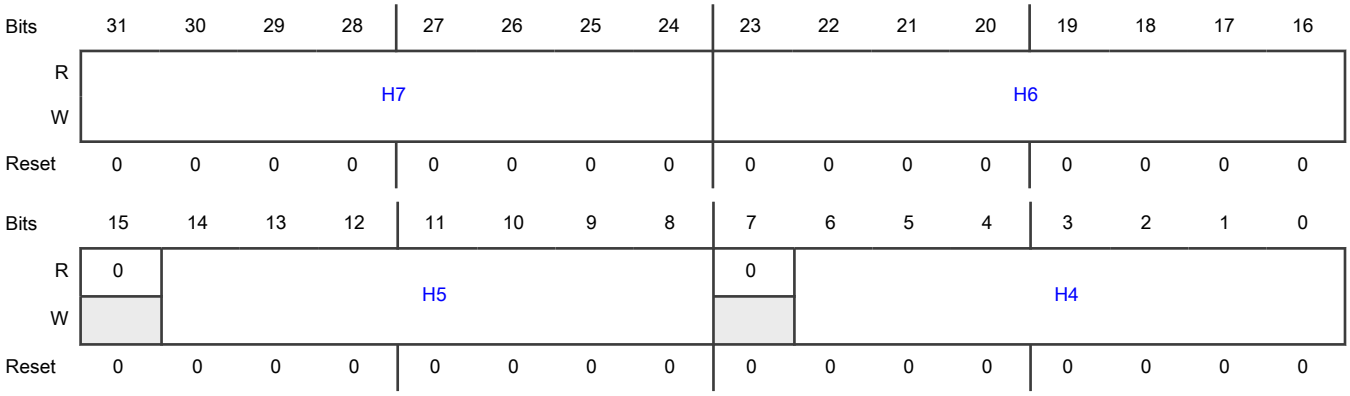
Offset

Register	Offset
ACQ_FILT_4_7	3Ch

Function

Acquisition filter coefficient 4~7

Diagram



Fields

Field	Function
31-24 H7	Acquisition Filter Coefficient 7 Acquisition filter coefficient 7, 8-bit signed fractional
23-16 H6	Acquisition Filter Coefficient 6 Acquisition filter coefficient 6, 8-bit signed fractional
15 —	Reserved
14-8	Acquisition Filter Coefficient 5

Table continues on the next page...

Table continued from the previous page...

Field	Function
H5	Acquisition filter coefficient 5, 7-bit signed fractional
7 —	Reserved
6-0 H4	Acquisition Filter Coefficient 4 Acquisition filter coefficient 4, 7-bit signed fractional

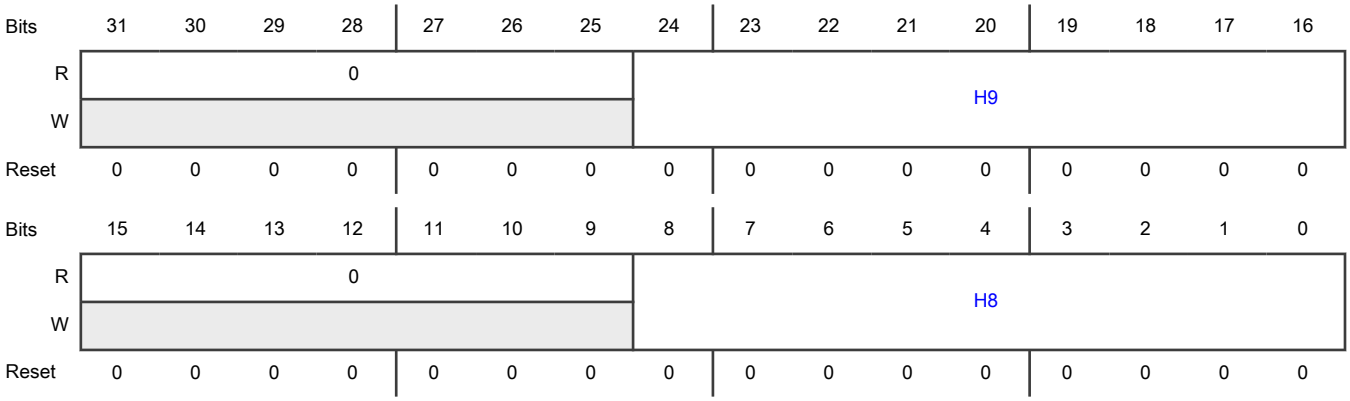
55.4.7.4.3.1.18 Acquisition Filter Coeffs 8~9 (ACQ_FILT_8_9)

Offset

Register	Offset
ACQ_FILT_8_9	40h

Function
Acquisition filter coefficient 8~9

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 H9	Acquisition Filter Coefficient 9 Acquisition filter coefficient 9, 9-bit signed fractional

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-9 —	Reserved
8-0 H8	Acquisition Filter Coefficient 8 Acquisition filter coefficient 8, 9-bit signed fractional

55.4.7.4.3.1.19 Acquisition Filter Coeffs 10~11 (ACQ_FILT_10_11)

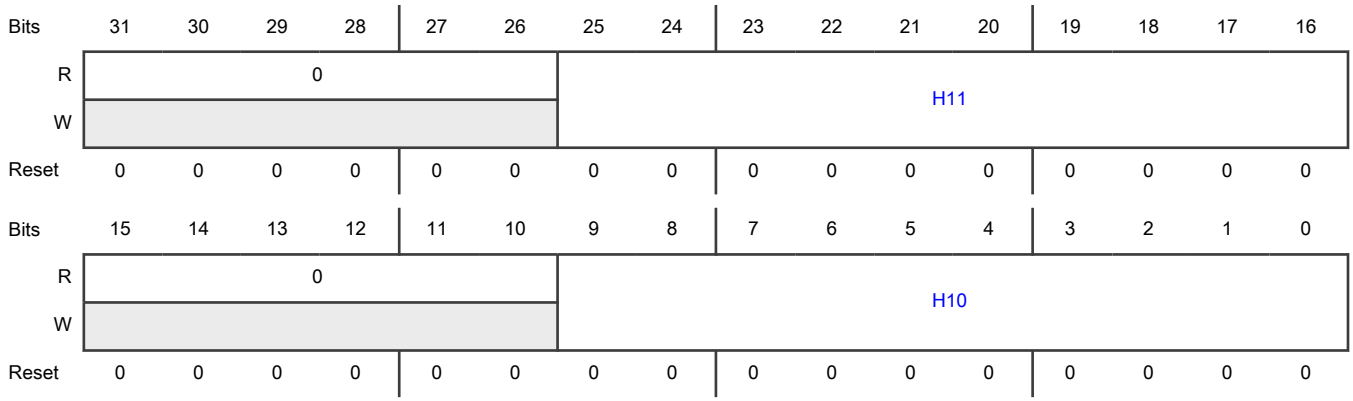
Offset

Register	Offset
ACQ_FILT_10_11	44h

Function

Acquisition filter coefficient 10~11

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 H11	Acquisition Filter Coefficient 11 Acquisition filter coefficient 11, 10-bit signed fractional
15-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 H10	Acquisition Filter Coefficient 10 Acquisition filter coefficient 10, 10-bit signed fractional

55.4.7.4.3.1.20 Demod Filter Coeffs 0~1 (DEMOD_FILT_0_1)

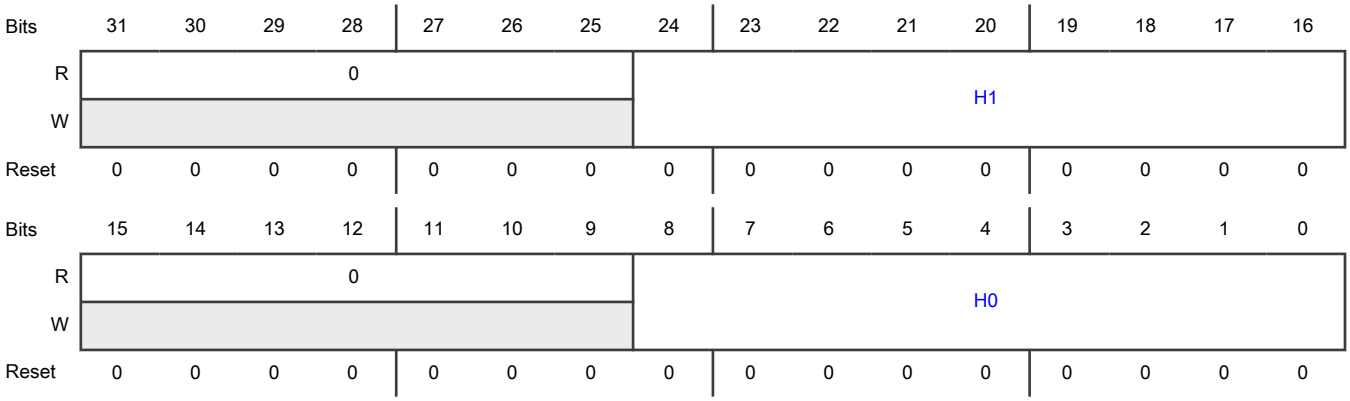
Offset

Register	Offset
DEMOD_FILT_0_1	48h

Function

Demod channel filter coefficient 0~1

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 H1	Demod Channel Filter Coefficient 1 Demod channel filter coefficient 1, 9-bit signed fractional
15-9 —	Reserved
8-0 H0	Demod Channel Filter Coefficient 0 Demod channel filter coefficient 0, 9-bit signed fractional

55.4.7.4.3.1.21 Demod Filter Coeffs 2~4 (DEMOD_FILT_2_4)

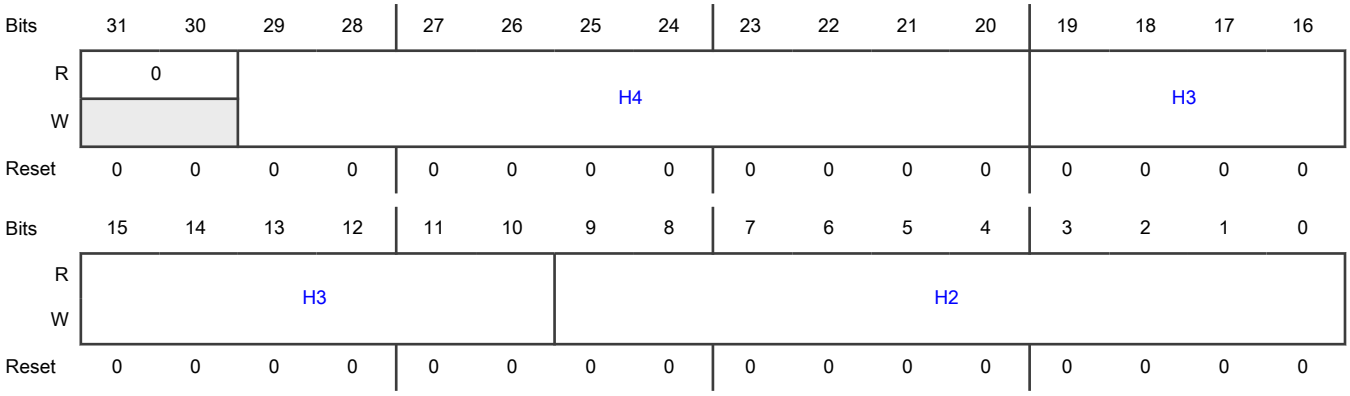
Offset

Register	Offset
DEMOD_FILT_2_4	4Ch

Function

Demod channel filter coefficient 2~4

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 H4	Demod Channel Filter Coefficient 4 Demod channel filter coefficient 4, 10-bit signed fractional
19-10 H3	Demod Channel Filter Coefficient 3 Demod channel filter coefficient 3, 10-bit signed fractional
9-0 H2	Demod Channel Filter Coefficient 2 Demod channel filter coefficient 2, 10-bit signed fractional

55.4.7.4.3.1.22 Acquisition Filter Coeffs 0~3 DRS (ACQ_FILT_0_3_DRS)

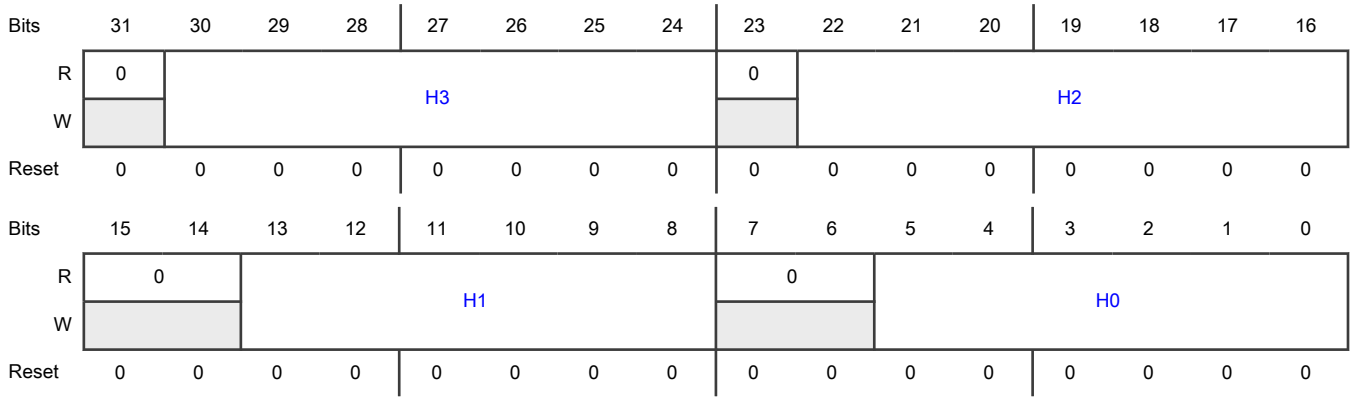
Offset

Register	Offset
ACQ_FILT_0_3_DRS	50h

Function

Acquisition filter coefficient 0~3 for datarate_config_sel=1

Diagram



Fields

Field	Function
31 —	Reserved
30-24 H3	Acquisition Filter Coefficient 3 Acquisition filter coefficient 3, 7-bit signed fractional
23 —	Reserved
22-16 H2	Acquisition Filter Coefficient 2 Acquisition filter coefficient 2, 7-bit signed fractional
15-14 —	Reserved
13-8 H1	Acquisition Filter Coefficient 1 Acquisition filter coefficient 1, 6-bit signed fractional
7-6 —	Reserved
5-0 H0	Acquisition Filter Coefficient 0 Acquisition filter coefficient 0, 6-bit signed fractional

55.4.7.4.3.1.23 Acquisition Filter Coeffs 4~7 DRS (ACQ_FILT_4_7_DRS)

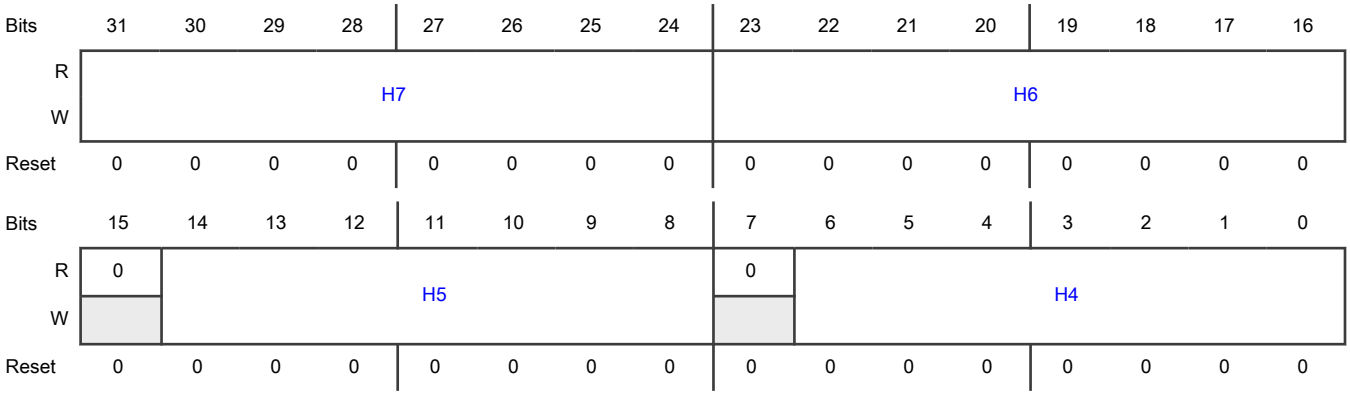
Offset

Register	Offset
ACQ_FILT_4_7_DRS	54h

Function

Acquisition filter coefficient 4~7 for datarate_config_sel=1

Diagram



Fields

Field	Function
31-24 H7	Acquisition Filter Coefficient 7 Acquisition filter coefficient 7, 8-bit signed fractional
23-16 H6	Acquisition Filter Coefficient 6 Acquisition filter coefficient 6, 8-bit signed fractional
15 —	Reserved
14-8 H5	Acquisition Filter Coefficient 5 Acquisition filter coefficient 5, 7-bit signed fractional
7 —	Reserved
6-0 H4	Acquisition Filter Coefficient 4 Acquisition filter coefficient 4, 7-bit signed fractional

55.4.7.4.3.1.24 Acquisition Filter Coeffs 8~9 DRS (ACQ_FILT_8_9_DRS)

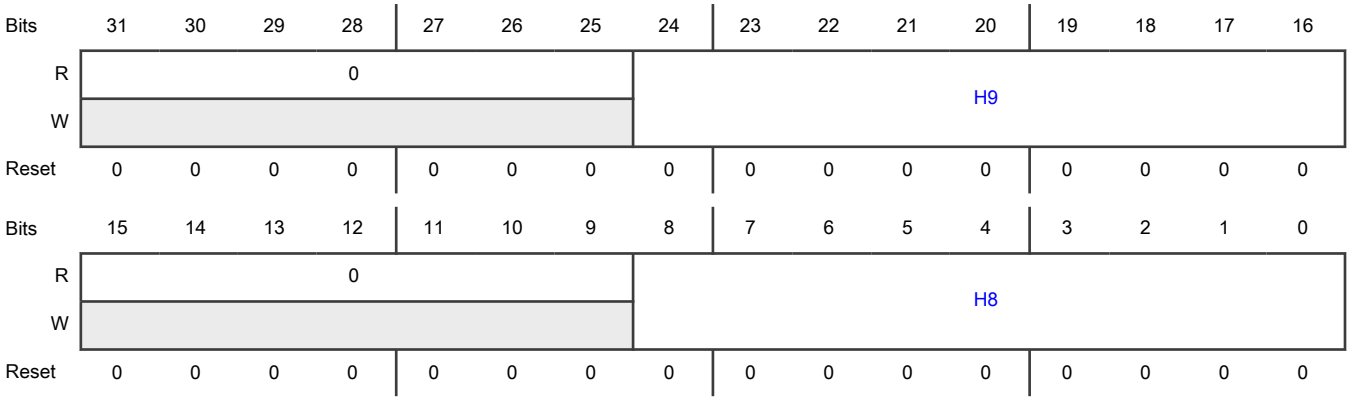
Offset

Register	Offset
ACQ_FILT_8_9_DRS	58h

Function

Acquisition filter coefficient 8~9 for datarate_config_sel=1

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 H9	Acquisition Filter Coefficient 9 Acquisition filter coefficient 9, 9-bit signed fractional
15-9 —	Reserved
8-0 H8	Acquisition Filter Coefficient 8 Acquisition filter coefficient 8, 9-bit signed fractional

55.4.7.4.3.1.25 Acquisition Filter Coeffs 10~11 DRS (ACQ_FILT_10_11_DRS)

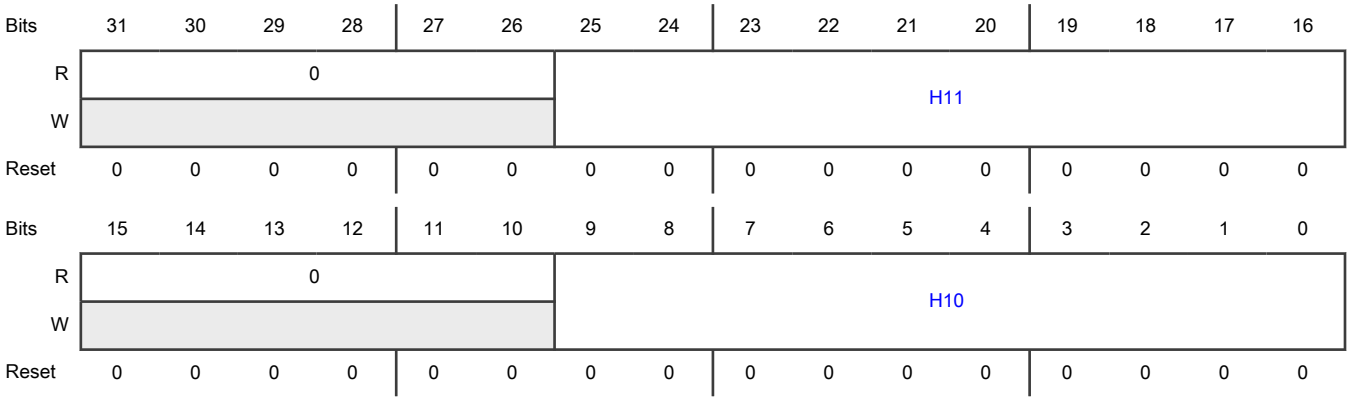
Offset

Register	Offset
ACQ_FILT_10_11_DRS	5Ch

Function

Acquisition filter coefficient 10~11 for datarate_config_sel=1

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 H11	Acquisition Filter Coefficient 11 Acquisition filter coefficient 11, 10-bit signed fractional
15-10 —	Reserved
9-0 H10	Acquisition Filter Coefficient 10 Acquisition filter coefficient 10, 10-bit signed fractional

55.4.7.4.3.1.26 Demod Filter Coeffs 0~1 DRS (DEMOD_FILT_0_1_DRS)

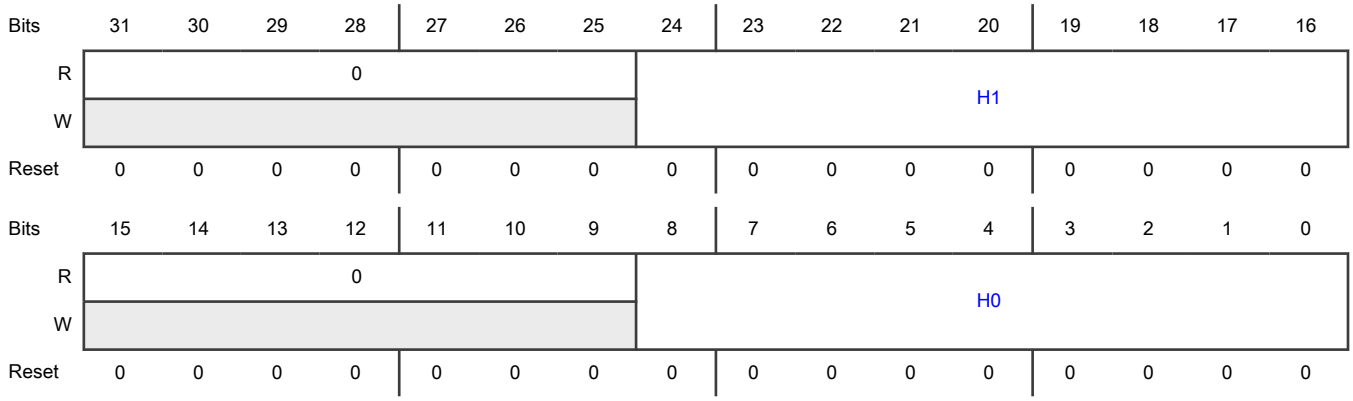
Offset

Register	Offset
DEMOD_FILT_0_1_DRS	60h

Function

Demod channel filter coefficient 0~1 for datarate_config_sel=1

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 H1	Demod Channel Filter Coefficient 1 Demod channel filter coefficient 1, 9-bit signed fractional
15-9 —	Reserved
8-0 H0	Demod Channel Filter Coefficient 0 Demod channel filter coefficient 0, 9-bit signed fractional

55.4.7.4.3.1.27 Demod Filter Coeffs 2~4 DRS (DEMOD_FILT_2_4_DRS)

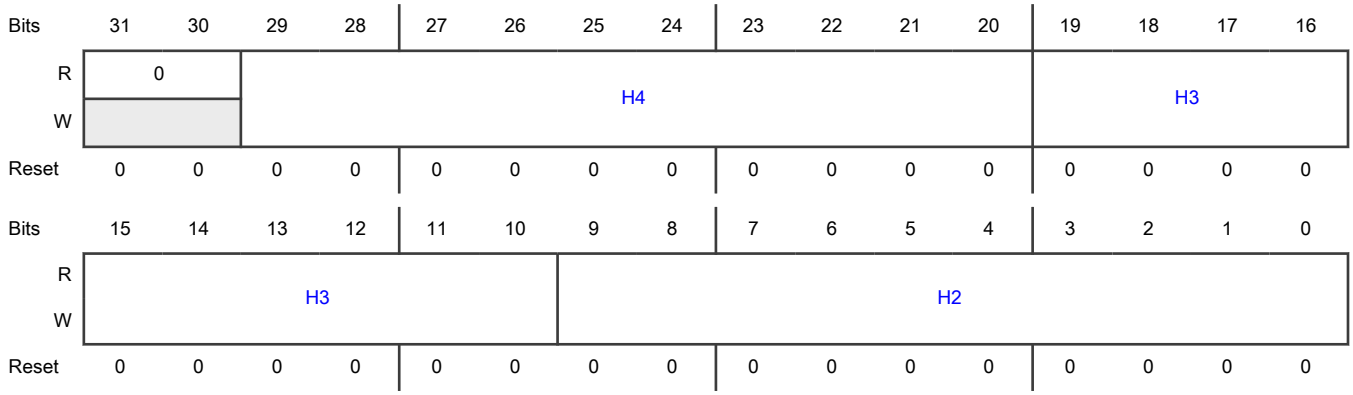
Offset

Register	Offset
DEMOD_FILT_2_4_DRS	64h

Function

Demod channel filter coefficient 2~4 for datarate_config_sel=1

Diagram



Fields

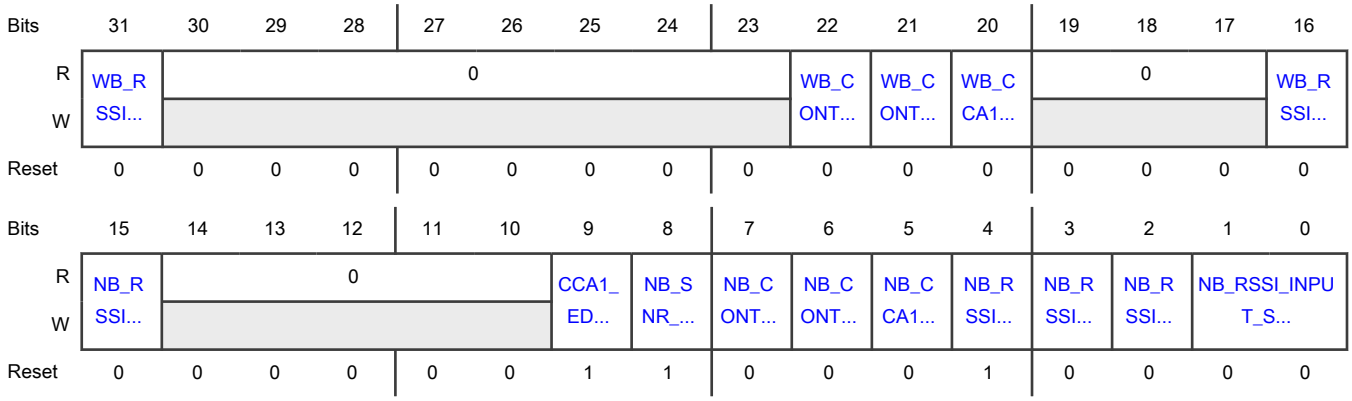
Field	Function
31-30 —	Reserved
29-20 H4	Demod Channel Filter Coefficient 4 Demod channel filter coefficient 4, 10-bit signed fractional
19-10 H3	Demod Channel Filter Coefficient 3 Demod channel filter coefficient 3, 10-bit signed fractional
9-0 H2	Demod Channel Filter Coefficient 2 Demod channel filter coefficient 2, 10-bit signed fractional

55.4.7.4.3.1.28 RSSI Global Control (RSSI_GLOBAL_CTRL)

Offset

Register	Offset
RSSI_GLOBAL_CTRL	68h

Diagram



Fields

Field	Function
31 WB_RSSI_EN	WB RSSI Enable 0b - WB-RSSI is disabled 1b - WB-RSSI is enabled
30-23 —	Reserved
22 WB_CONT_MEAS_OVRD_EN	WB RSSI Continuous Measurment Override Enable By default, WB-RSSI working in continuous mode. By setting WB_CONT_MEAS_OVRD_EN=1 and WB_CONT_MEAS_OVRD=0, the WB-RSSI will stop working once the first rssi_wb calculation done.
21 WB_CONT_MEAS_OVRD	WB RSSI Continuous Measurment Override Value
20 WB_CCA1_ED_EN	WB RSSI CCA1 ED Enable 0b - WB-RSSI CCA1/ED disabled 1b - WB-RSSI CCA1/ED enabled
19-17 —	Reserved
16 WB_RSSI_INPUT_SEL	WB RSSI Input Select 0b - DCOC output I/Q 1b - CIC output I/Q
15 NB_RSSI_EN	NB RSSI Enable 0b - NB-RSSI is disabled 1b - NB-RSSI is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-10 —	Reserved
9 CCA1_ED_FROM_NB	CCA1/ED Result Selection 0b - WB-RSSI's CCA1/ED result is selected 1b - NB-RSSI's CCA1/ED result is selected
8 NB_SNR_LQI_ENABLE	NB RSSI SNR LQI Enable 0b - NB-RSSI SNR/LQI calculation is disabled 1b - NB-RSSI SNR/LQI calculation is enabled
7 NB_CONT_MEAS_OVRD_EN	NB RSSI One-time Measure Override Enable By default, NB-RSSI working in one-time mode. By setting NB_CONT_MEAS_OVRD_EN=1 and NB_CONT_MEAS_OVRD=1 bit, the NB-RSSI will calculate RSSI/LQI/SNR continuously.
6 NB_CONT_MEAS_OVRD	NB RSSI Onetime Measure Override
5 NB_CCA1_ED_EN	NB RSSI CCA1 ED Enable 0b - NB-RSSI CCA1/ED is disabled 1b - NB-RSSI CCA1/ED is enabled
4 NB_RSSI_PA_AA_MATCH_SEL	NB RSSI PHY Trigger Select 0b - NB-RSSI starts work when PHY_PD_FOUND asserted 1b - NB-RSSI starts work when PHY_AA_MATCH asserted
3 NB_RSSI_AA_MATCH_OVRD_EN	NB RSSI PHY Trigger Override Enable By setting the NB_RSSI_AA_MATCH_OVRD_EN=1, the software can determine when will NB-RSSI start working.
2 NB_RSSI_AA_MATCH_OVRD	NB RSSI PHY Trigger Override
1-0 NB_RSSI_INPUT_SEL	NB RSSI Input Select 00b - ACQ_CHF output I/Q 01b - SRC output I/Q 10b - DEMOD_CHF output I/Q 11b - Reserved

55.4.7.4.3.1.29 Wide-Band RSSI Control (WB_RSSI_CTRL)

Offset

Register	Offset
WB_RSSI_CTRL	6Ch

Function
Wide-Band RSSI control

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSSI_ADJ_WB								0	RSSI_F_WINDOW_WB_DRS			0	RSSI_N_WINDOW_WB_DRS		
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		KEEP_RS...	RSSI_DB...	0	RSSI_F_WINDOW_WB			0	RSSI_M_WINDOW_WB			0	RSSI_N_WINDOW_WB		
W																
Reset	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1	0

Fields

Field	Function
31-24 RSSI_ADJ_WB	WB RSSI Adjust Value WB_RSSI adjust value. Store in s5.2 format.
23 —	Reserved
22-20 RSSI_F_WINDOW_WB_DRS	
19 —	Reserved
18-16 RSSI_N_WINDOW_WB_DRS	
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 KEEP_RSSI_RESULT_WB	When enabled, the WB-RSSI results will keep until next update/refresh, or the WB-RSSI results will be cleared when rssi_init asserts.
12 RSSI_DB_EN_WB	WB RSSI dB Calculate Enable If only need to use WB-RSSI to calculate fast_mag and slow_mag to feed AGC, this bit can be cleared to disable WB-RSSI's window-m-averager to save power. However, if need RSSI/CCA1/ED results from WB-RSSI, this bit should be set.
11 —	Reserved
10-8 RSSI_F_WINDOW_WB	WB RSSI F Window Averager Factor Actual F window averager width is 2^(RSSI_F_WINDOW_WB). Max value for this field is 6.
7 —	Reserved
6-4 RSSI_M_WINDOW_WB	WB RSSI M Window Averager Factor Actual M window averager width is 2^(RSSI_M_WINDOW_WB). Max value for this field is 4.
3 —	Reserved
2-0 RSSI_N_WINDOW_WB	WB RSSI N Window Averager Factor Actual N window averager width is 2^(RSSI_N_WINDOW_WB). Max value for this field is 7.

55.4.7.4.3.1.30 Wide-Band RSSI Result 0 (WB_RSSI_RES0)

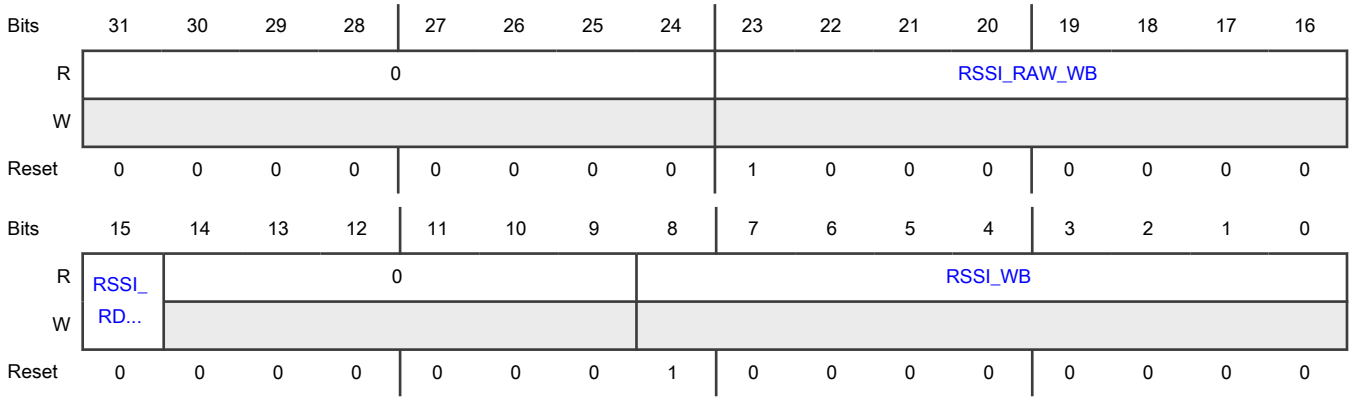
Offset

Register	Offset
WB_RSSI_RES0	70h

Function

Check WB-RSSI block diagram in "RSSI Estimator" section to find where these results come from.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RSSI_RAW_WB	WB Raw RSSI Result Store in s7.0 format. This field is for debug purpose and the refresh rate depends on ref CLK frequency, WB_RSSI_INPUT_SEL, RSSI_F_WINDOW_WB and RSSI_N_WINDOW_WB. In some cases, the software may not capture all values. To caputre all RSSI_RAW_WB values accurately, it is recommended to use DTEST function.
15 RSSI_RDY_WB	This bit set when RSSI_WB value ready to read(or updated) and cleared by rx_init or write "1" to this field(W1C). In RX mode, the software can poll this filed until 1, then read the RSSI_WB value and finally write 1 to clear it. Do this loop to get RSSI_WB values continuously.
14-9 —	Reserved
8-0 RSSI_WB	WB RSSI Result Store in s8.0 format. After this field updated, the result will keep until: KEEP_RSSI_RESULT_WB=0 : Next result update(In same RX sequence) or RX initial(Next RX sequence started) KEEP_RSSI_RESULT_WB=1 : Next result update, no matter in same or different RX sequency

55.4.7.4.3.1.31 Wide-Band RSSI Result 1 (WB_RSSI_RES1)

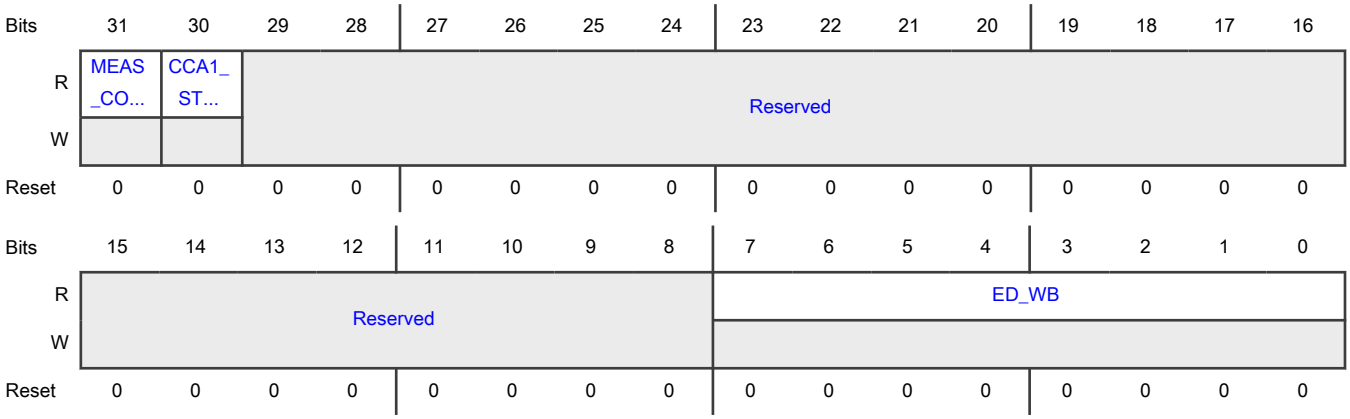
Offset

Register	Offset
WB_RSSI_RES1	74h

Function

Check WB-RSSI block diagram in "RSSI Estimator" section to find where these results come from.

Diagram



Fields

Field	Function
31 MEAS_COMPLETE_WB	WB RSSI Measure Complete Measure complete status bit. Indicate that RSSI_WB, ED_WB, CCA1_STATE_WB updated. However, in continuous measurement mode(check WB_CONT_MEAS_OVRD_EN for more info), this bit is an instant pulse, the software may not capture it. While in non-continuous mode, this bit will keep high until next RX initial or AGC gain change.
30 CCA1_STATE_WB	WB RSSI CCA1 State CCA1/ED status bit. After this field updated, the result will keep until next result update(If continues mode enabled) or RX initial(Next RX sequence started).
29-8 —	Reserved
7-0 ED_WB	WB RSSI ED Result Store in s7.0 format. After this field updated, the result will keep until next result update(If continues mode enabled) or RX initial(Next RX sequence started)

55.4.7.4.3.1.32 Wide-Band RSSI DFT Result (WB_RSSI_DFT)

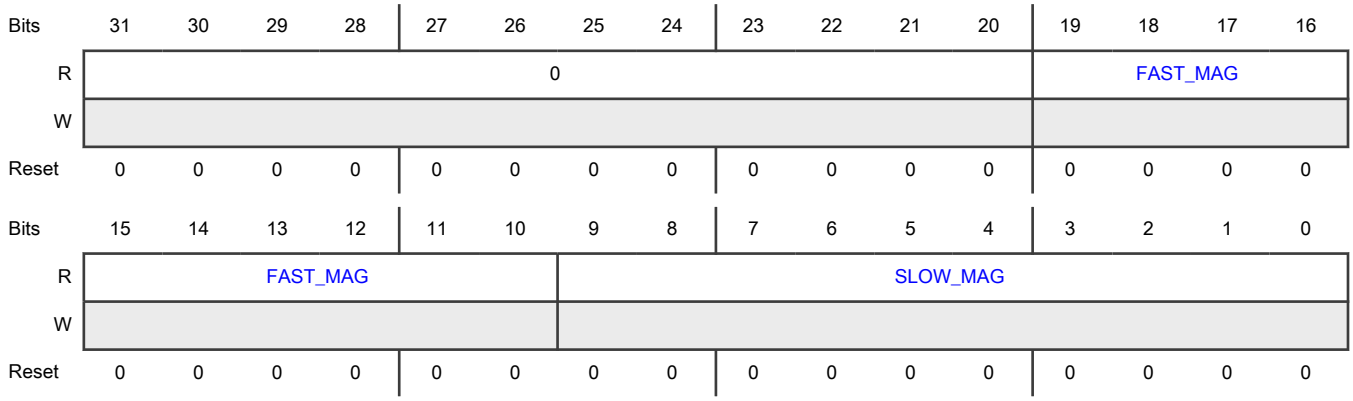
Offset

Register	Offset
WB_RSSI_DFT	78h

Function

WB_RSSI_DFT

Diagram



Fields

Field	Function
31-20 —	Reserved
19-10 FAST_MAG	WB RSSI Fast Magnitude Value Due to this field is updated at high frequency, software may not read every value. Should use DTEST or DMA_DEBUG to capture all fast_mag result.
9-0 SLOW_MAG	WB RSSI Slow Magnitude Value Due to this field is updated at high frequency, software may not read every value. Should use DTEST or DMA_DEBUG to capture all slow_mag result.

55.4.7.4.3.1.33 Narrow-Band RSSI Control 0 (NB_RSSI_CTRL0)

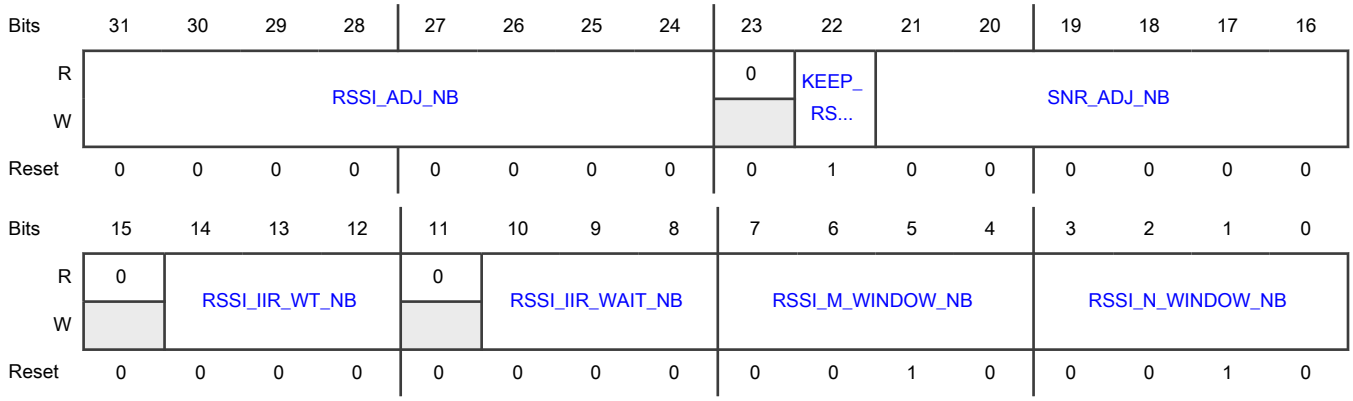
Offset

Register	Offset
NB_RSSI_CTRL0	7Ch

Function

NB_RSSI_CTRL0

Diagram



Fields

Field	Function
31-24 RSSI_ADJ_NB	NB RSSI Adjust Value NB_RSSI adjust value. Store in s5.2 format.
23 —	Reserved
22 KEEP_RSSI_RESULT_NB	When enabled, the NB-RSSI results will keep until next update/refresh, or the NB-RSSI results will be cleared when rx_init or rssi_init asserts.
21-16 SNR_ADJ_NB	NB RSSI SNR Adjust Value Store in s5.0 format.
15 —	Reserved
14-12 RSSI_IIR_WT_NB	NB RSSI IIR Filter Factor Factor for NB-RSSI mag and noise IIR filter. The actual factor is 1/(2^RSSI_IIR_WT_NB). The max valid value for this field is 5.
11 —	Reserved
10-8 RSSI_IIR_WAIT_NB	NB RSSI IIR Filter Initial Wait Time NB-RSSI IIR filter warmup time, count by input I/Q signal frequency. The actual warmup cycle number is 2^RSSI_IIR_WAIT_NB.
7-4 RSSI_M_WINDOW_NB	NB RSSI M Window Averager Factor Actual M window averager width is 2^(RSSI_M_WINDOW_NB). Max value for this field is 10.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 RSSI_N_WINDOW_NB	NB RSSI N Window Averager Factor Actual N window averager width is 2^(RSSI_N_WINDOW_NB). Max value for this field is 8.

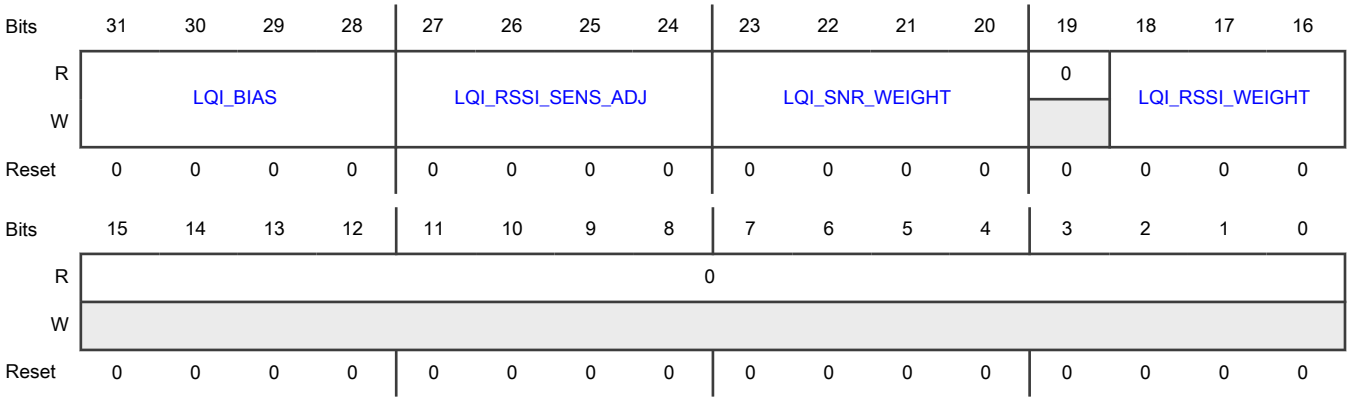
55.4.7.4.3.1.34 Narrow-Band RSSI Control 1 (NB_RSSI_CTRL1)

Offset

Register	Offset
NB_RSSI_CTRL1	80h

Function
NB_RSSI_CTRL1

Diagram



Fields

Field	Function
31-28 LQI_BIAS	LQI Bias Value Actual LQI bias value = -36 + 2*ipr_lqi_bias
27-24 LQI_RSSI_SENS_ADJ	LQI Sensitivity Adjust Value Actual LQI sensitivity value is: -103 + LQI_RSSI_SENS_ADJ
23-20 LQI_SNR_WEIGHT	SNR Weight For LQI Calulation SNR result weight in LQI calculation. When LQI_SNR_WEIGHT!=0, the actual weight value is: 1 + 0.125 * LQI_SNR_WEIGHT. When LQI_SNR_WEIGHT=0, the actual weight value is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 —	Reserved
18-16 LQI_RSSI_WEIGHT	RSSI Weight For LQI Calulation RSSI result weight in LQI calculation. Actual weight is: 2 + 0.125 * LQI_SNR_WEIGHT
15-0 —	Reserved

55.4.7.4.3.1.35 Narrow-Band RSSI Result 0 (NB_RSSI_RES0)

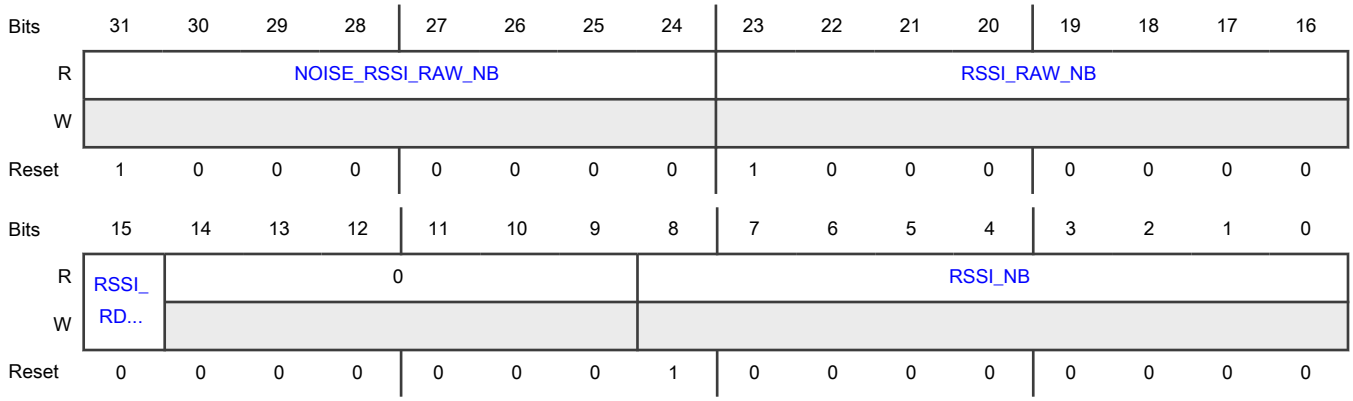
Offset

Register	Offset
NB_RSSI_RES0	84h

Function

Check NB-RSSI block diagram in "RSSI Estimator" section to find where these results come from.

Diagram



Fields

Field	Function
31-24 NOISE_RSSI_RAW_NB	Raw Noise Result Store in s7.0 format

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is for debug purpose and the refresh rate depends on ref CLK frequency, NB_RSSI_INPUT_SEL, RSSI_N_WINDOW_NB. In some cases, the software may not capture all values. To caputre all NOISE_RSSI_RAW_NB values accurately, it is recommended to use DTEST function.
23-16 RSSI_RAW_NB	Raw NB RSSI Result Store in s7.0 format This field is for debug purpose and the refresh rate depends on ref CLK frequency, NB_RSSI_INPUT_SEL, RSSI_N_WINDOW_NB. In some cases, the software may not capture all values. To caputre all RSSI_RAW_NB values accurately, it is recommended to use DTEST function.
15 RSSI_RDY_NB	This bit set when RSSI_NB/SNR_NB/LQI_NB/ED_NB value ready to read(or updated) and cleared by rx_init or write "1" to this field(W1C). In RX mode, the software can poll this filed until 1, then read the RSSI_NB/SNR_NB/LQI_NB/ED_NB value and finally write 1 to clear it. Do this loop to get values continuously if NB_RSSI is working in continuous mode.
14-9 —	Reserved
8-0 RSSI_NB	NB RSSI Result Store in s8.0 format. After this field updated, the result will keep until: KEEP_RSSI_RESULT_NB=0 : Next result update(In same RX sequence) or RX initial(Next RX sequence started) KEEP_RSSI_RESULT_NB=1 : Next result update, no matter in same or different RX sequency

55.4.7.4.3.1.36 Narrow-Band RSSI Result 1 (NB_RSSI_RES1)

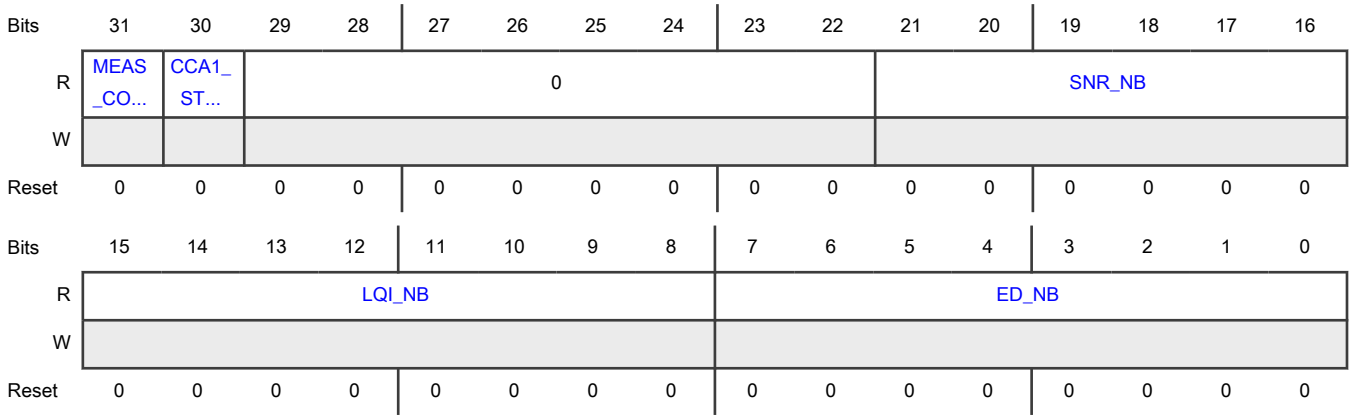
Offset

Register	Offset
NB_RSSI_RES1	88h

Function

Check NB-RSSI block diagram in "RSSI Estimator" section to find where these results come from.

Diagram



Fields

Field	Function
31 MEAS_COMPLETE_NB	NB RSSI Measure Complete Measure complete status bit. Indicate that RSSI_NB, ED_NB, CCA1_STATE_NB, SNR_NB, LQI_NB updated. However, in continuous measurement mode(check NB_CONT_MEAS_OVRD_EN for more info), this bit is an instant pulse, the software may not capture it. While in non-continuous mode, this bit will keep high until next RX initial or AGC gain change.
30 CCA1_STATE_NB	NB RSSI CCA1 State CCA1/ED status bit. After this field updated, the result will keep until next result update(If continues mode enabled) or RX initial(Next RX sequence started)
29-22 —	Reserved
21-16 SNR_NB	NB RSSI SNR Result After this field updated, the result will keep until: KEEP_RSSI_RESULT_NB=0 : Next result update(In same RX sequence) or RX initial(Next RX sequence started) KEEP_RSSI_RESULT_NB=1 : Next result update, no matter in same or different RX sequency
15-8 LQI_NB	NB RSSI LQI Result After this field updated, the result will keep until: KEEP_RSSI_RESULT_NB=0 : Next result update(In same RX sequence) or RX initial(Next RX sequence started) KEEP_RSSI_RESULT_NB=1 : Next result update, no matter in same or different RX sequency
7-0 ED_NB	NB RSSI ED Result Store in s7.0 format. After this field updated, the result will keep until next result update(If continues mode enabled) or RX initial(Next RX sequence started)

55.4.7.4.3.1.37 Narrow-Band RSSI DFT Result (NB_RSSI_DFT)

Offset

Register	Offset
NB_RSSI_DFT	8Ch

Function

Check NB-RSSI block diagram in "RSSI Estimator" section to find where these results come from.

NOTE: The result values in this register only refresh and valid during NB_RSSI is working.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				AVG_MAG_NB											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				AVG_NOISE_MAG_NB											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-16 AVG_MAG_NB	NB RSSI Averaged Magnitude Value Due to this field is updated at high frequency, software may not read every value. Should use DTEST or DMA_DEBUG to capture all fast_mag result.
15-12 —	Reserved
11-0 AVG_NOISE_MAG_NB	NB RSSI Averaged Noise Magnitude Value Due to this field is updated at high frequency, software may not read every value. Should use DTEST or DMA_DEBUG to capture all fast_mag result.

55.4.7.4.3.1.38 AGC Control (AGC_CTRL)

Offset

Register	Offset
AGC_CTRL	90h

Function

AGC control register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	AGC_WBD_EN		AGC_WBD_AUTO_DI...		AGC_WBD...	AGC_WBD...	AGC_WBD...	AGC_WBD_THR1					AGC_WBD_THR2			AGC_WBD...
Reset	1	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AGC_WBD_STEP1_SZ		AGC_WBD_STEP2_SZ			AGC_FAS...	AGC_FAS...	AGC_SLO...	AGC_DEL...	AGC_DELTA_SLOW_STEP			AGC_HOLD_EN		AGC_UNHOLD_FEAT...	
Reset	1	1	0	0	1	1	0	1	0	0	0	1	0	0	0	0

Fields

Field	Function
31-30 AGC_WBD_EN	AGC WBD Enable AGC WBD enable 00b - AGC WBD is disabled 01b - AGC WBD step1 is enabled 10b - AGC WBD step1 and step2 is enabled 11b - Reserved
29-28 AGC_WBD_AUTO_DIS_CFG	AGC WBD Auto Disable If AGC_WBD_AUTO_DIS[0] set and fast or slow magnitude triggers occurs with no WB trigged, WB detect function will be disabled automatically. If AGC_WBD_AUTO_DIS[1] set and WB trigger not occurs in period, specified by AGC_WBD_TIMEOUT, WB detect function will be disabled automatically. NOTE: AGC_WBD_AUTO_DIS = 2'b11 is valid.
27 AGC_WBD_GAIN_LIMIT_EN	AGC WBD Gain Limit When this bit set:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	-- If WBD takes the first step, then the AGC_max_index is set equal to the gain index after the AGC_WBD_det_step -- If WBD takes the 2nd step, then the AGC_max_index is set equal to the gain index after the AGC_WBD_det_step2
26 AGC_WBD_STEP1_DUAL_CLIP_EN	AGC WBD Step1 Dual Clip Enable When this bit set. AGC need two consecutive WB trigger occurs then issue WB stpe 1 gain step down. The cooldown duration is specified by AGC_WBD_STEP1_DUAL_CLIP_WAIT
25 AGC_WBD_STEP2_DUAL_CLIP_EN	AGC WBD Step2 Dual Clip Enable When this bit set. AGC need two consecutive WB trigger occurs then issue WB stpe 2 gain step down. The cooldown duration is specified by AGC_WBD_STEP2_DUAL_CLIP_WAIT
24-21 AGC_WBD_THR1	AGC WBD Step1 threshold Threshold value(unit: mV) for AGC WBD step1. 0000b - 49.31 0001b - 67.56 0010b - 90.98 0011b - 117.42 0100b - 150.66 0101b - 180.98 0110b - 211.87 0111b - 245.2 1000b - 288.31 1001b - 336.02 1010b - 394.34 1011b - 462.71 1100b - 548.04 1101b - 650.13 1110b - 771.65 1111b - 918.12
20-17 AGC_WBD_THR2	AGC WBD Step2 threshold Threshold value(unit: mV) for AGC WBD step2. The actual threshold voltage is as same as the description in AGC_WBD_THR1.
16-14	AGC WBD Step1 Gain Decreas Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
AGC_WBD_STEP1_SZ	Specify the gain index decrease value when AGC WBD step1 triggered. Actual gain step is: AGC_WBD_STEP1_SZ + 1
13-11 AGC_WBD_STEP2_SZ	AGC WBD Step2 Gain Decreas Value Specify the gain index decrease value when AGC WBD step2 triggered. Actual gain step is: AGC_WBD_STEP2_SZ + 1
10 AGC_FAST_EN	AGC Fast Magitude Mode Enable 0b - Disable fast magnitude mode 1b - Enable fast magnitude mode
9 AGC_FAST_STEP_UP_EN	AGC Fast Magitude Mode Step Up Enable AGC step up enable in fast magnitude mode. This bit only take effect with AGC_FAST_EN=1. If this bit cleared, the fast magnitude mode can only make AGC gain index step down. 0b - Fast magnitude mode can only make AGC gain index step down 1b - Fast magnitude mode can make AGC gain index step down or step up
8 AGC_SLOW_EN	AGC Slow Magitude Mode Enable 0b - Disable AGC slow magnitude mode 1b - Enable AGC slow magnitude mode
7 AGC_DELTA_SLOW_EN	AGC Delta Slow Magitude Mode Enable AGC delta slow magnitude mode enable. Note: The AGC delta slow magnitude mode can only work with AGC slow magnitude mode enabled(AGC_SLOW_EN=1). 0b - Disable AGC delta slow magnitude mode 1b - Enable AGC delta slow magnitude mode
6-4 AGC_DELTA_SLOW_STEP	AGC Delta Slow Mode Gain Step Up Value Gain index change for a big step up when delta slow mode triggered
3-2 AGC_HOLD_EN	AGC Hold Mode Enable AGC hold mode enable and control 00b - Disable AGC hold mode 01b - AGC hold when preamble found 10b - AGC hold when AGC hold timeout matched 11b - AGC hold when preamble found or hold timeout matched
1-0	AGC Unhold Enalbe AGC unhold feature enable and config.

Table continues on the next page...

Table continued from the previous page...

Field	Function
AGC_UNHOLD_FEAT_EN	AGC_UNHOLD_FEAT_EN[0]: When AGC in HOLD mode and the selected mag values exceed the range of "agc_hold_mag +/- ipr_hold_mag_margin", AGC will exit from HOLD mode. NOTE: See AGC_TIMING2[AGC_UNHOLD_MAG_SRC_SEL] and AGC_TIMING2[AGC_UNHOLD_MAG_CNT_SEL] for more info AGC_UNHOLD_FEAT_EN[1]: If AGC does not enter into FREEZE mode or receive done in time, AGC will exit HOLD mode. The unhold timeout value is set by AGC_UNHOLD_TMEOUT NOTE: AGC_UNHOLD_FEAT_EN=3 is a valid configuration

55.4.7.4.3.1.39 AGC Control Status (AGC_CTRL_STAT)

Offset

Register	Offset
AGC_CTRL_STAT	94h

Function

AGC control and status register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AGC_STATUS			AGC_GAIN_CHANGE_STATUS			AGC_GAI...	AGC_GAIN_IDX				AGC_PREV_GAIN_IDX				AGC_SOF...
W																
Reset	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AGC_SOF...	AGC_SOF...	AGC_GAIN_IDX_ST...		AGC_FREEZE_EN		AGC_UNF...	AGC_CAL...	AGC_PHY...	AGC_PHY...	AGC_INIT_IDX				AGC_MAX_IDX	
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0

Fields

Field	Function
31-29 AGC_STATUS	AGC FSM Status Indicate the current state of AGC state machine. This field will be 0 when RX_DIG or AGC not be enabled. 000b - AGC_IDLE

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - AGC_WB_ONLY 010b - AGC_WB_MAG 011b - AGC_WB_DEBOUNCE 100b - AGC_MAG_ONLY 101b - AGC_HOLD 110b - AGC_FREEZE 111b - AGC_WAIT_GAIN_SETTLE
28-26 AGC_GAIN_CHANGE_STATUS	AGC Gain Change Status Indicate the latest AGC gain index change reason. This field will be 0 when RX_DIG or AGC not be enabled. 000b - No gain change 001b - Gain decreased by WBD step1 010b - Gain decreased by WBD step2 011b - Gain decreased by fast mode 100b - Gain increased by fast mode 101b - Gain decreased by slow mode 110b - Gain increased by slow mode 111b - Gain increased by delta slow mode
25 AGC_GAIN_CHANGE	AGC Gain Change Indicate the AGC gain is changing. NOTE: This is a pulse signal, software may not read it correctly. To capture this pulse accurately, please use DTEST function.
24-21 AGC_GAIN_IDX	AGC Gain Index Indicate current AGC gain index. This field will be reset to AGC_INIT_IDX when AGC or RX_DIG disabled(rx_dig_en deassert) . To get the AGC gain index value using for previous packet receiving, enable the AGC gain index store function(set a non-zero value for AGC_GAIN_STORE field) and read the AGC_PREV_GAIN_IDX field.
20-17 AGC_PREV_GAIN_IDX	AGC Previous Gain Index Indicate previous AGC stable gain index. To use this status field, user should set a non-zero value for AGC_GAIN_IDX_STORE. The value of this field will be kept until next AGC gain store trigger occurs. That means RX_DIG disable or RX_DIG initial will not reset this field. See AGC_GAIN_IDX and AGC_GAIN_IDX_STOR for more info.
16-15 AGC_SOFT_RST_SRC_SEL	PHY AGC Soft Reset Sel 00b - Disable PHY AGC soft reset function 01b - Use posedge phy_soft_rst to reset AGC

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Use negedge phy_soft_rst to reset AGC 11b - Use negedge phy_agc_freeze_trig to reset AGC
14 AGC_SOFT_RST_GAIN_SEL	PHY AGC Soft Reset Gain Sel 0b - AGC keep current gain index when PHY AGC soft reset trigged, 1b - AGC return to AGC_INIT_IDX when PHY AGC soft reset trigged,
13-12 AGC_GAIN_IDX_STORE	Enable and config AGC gain index stroe function. The stored AGC gain index can be read from AGC_PREV_GAIN_IDX field 00b - AGC gain index stroe function is disabled 01b - Store AGC gain index when AGC enter into HOLD mode 10b - Store AGC gain index when AGC enter into FREEZE mode 11b - Store AGC gain index when AA matched
11-10 AGC_FREEZE_EN	AGC Freeze Mode Enable AGC Freeze mode enable and configure 00b - Disable AGC freeze mode 01b - AGC freeze when AA/SFD matched 10b - AGC freeze when AGC freeze timeout matched 11b - AGC freeze when AA/SFD matched or freeze timeout matched
9 AGC_UNFREEZE_FEAT_EN	AGC Unfreeze Feature Enable 0b - AGC unfreeze function is disabled 1b - AGC will exit FREEZE mode when AGC_UNFREEZE_TMEOUT matched and aa_found not be asserted
8 AGC_CALC_MAG_IN_FRZ	AGC Calucate Magnitude In Freeze Mode RXDIG continues to calculate the magnitude value after AGC freezed. NOTE: If NB_RSSI module is expected to start measurement by the asseration of aa_match, pd_found, or sdf_detect, and one of those signal set as AGC Freeze mode trigger, setting this bit is required or the NB-RSSI moudel cannot work when AGC freezed.
7 AGC_PHY_FREEZE_TRIG_SEL	AGC PHY Freeze Trigger Select 0b - PHY_AGC_FREEZE_TRIG is select as AGC freeze trig. 1b - PHY_AGC_HOLD_TRIG is select as AGC freeze trig.
6 AGC_PHY_HOLD_TRIG_SEL	AGC PHY Hold Trigger Select 0b - PHY_AGC_HOLD_TRIG is select as AGC hold trig. 1b - PHY_AGC_FREEZE_TRIG is select as AGC hold trig.
5-2	AGC Initial Gain Index

Table continues on the next page...

Table continued from the previous page...

Field	Function
AGC_INIT_IDX	Initial AGC gain index value. Note: the AGC_INIT_IDX should not greater than 11. Also see AGC_MAX_IDX for more infomation.
1-0 AGC_MAX_IDX	AGC Max Gain Index Actual AGC max gain index is: 11 - AGC_MAX_IDX. When reaches the max gain index AGC will not perform gain index step up.

55.4.7.4.3.1.40 AGC Timing Control 0 (AGC_TIMING0)

Offset

Register	Offset
AGC_TIMING0	98h

Function

AGC timing control 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	AGC_WBD_INIT_WAIT							0	AGC_MAG_INIT_WAIT						
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AGC_GAIN_STEP_WAIT						AGC_WBD_STEP1_TIMEOUT		AGC_WBD_STEP2_TIMEOUT						AGC_DELTA_SLOW_...	
W																
Reset	0	1	0	0	0	0	1	1	1	0	0	1	1	0	0	0

Fields

Field	Function
31 —	Reserved
30-24 AGC_WBD_INIT_WAIT	AGC WBD Mode Initial Wait Time When RXDIG enabled, the AGC will wait AGC_WBD_INIT_WAIT reference clock cycles than enable WB detect function. NOTE: The programmed value should not less than 2 and not larger than AGC_MAG_INIT_WAIT

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 —	Reserved
22-16 AGC_MAG_INIT_WAIT	AGC Magnitude Mode Initial Wait Time When RXDIG enabled, the AGC will wait AGC_MAG_INIT_WAIT reference clock cycles than enable FAST and/or SLOW mode. NOTE: The programmed value should not less than max(8'd2, AGC_WBD_INIT_WAIT)
15-10 AGC_GAIN_STEP_WAIT	AGC Gain Change Wait Time Determine the duration of AGC WAIT_GAIN_STEL state. That means the number of reference clock cycles to wait for analog stable after AGC gain index changed.
9-7 AGC_WBD_STEP1_TIMEOUT	AGC WBD Timeout WBD step1 timeout value. This timeout counter will start work at the point of AGC WBD initialization done. If AGC_WBD_AUTO_DIS[1]=1 and no WB trigger occurs, AGC will disable WB detect function. The actual timeout duration(Unit is microsecond) is: AGC_WBD_STEP1_TIMEOUT + 1
6-2 AGC_WBD_STEP2_TIMEOUT	AGC WBD Step2 Timeout Indicating the WBD step2 enable duration, counted by reference clock cycles. If no WBD step2 trigger occurs, the AGC will disable WBD function. NOTE: The programmed value should be no less than 2.
1-0 AGC_DELTA_SLOW_WAIT	AGC Delta Slow Mode Timing In delta slow mode, AGC will compare current slow_mag(slow_mag[N]) to the previous Kth(Here K = AGC_DELTA_SLOW_WAIT+1)slow mag(slow_mag[N-k]) and check whether the delta slow mag(slow_mag[N-K] - slow_mag[N]) exceed DELTA_SLOW_THR. If yes, the AGC will issue a delta slow trig and increas the gain index by AGC_DELATA_SLOW_STEP.

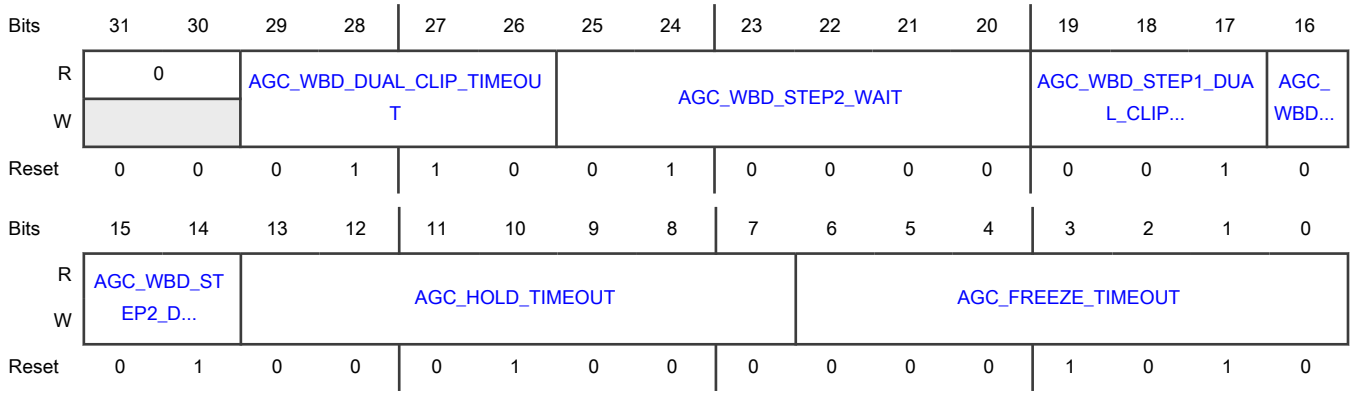
55.4.7.4.3.1.41 AGC Timing Control 1 (AGC_TIMING1)

Offset

Register	Offset
AGC_TIMING1	9Ch

Function
AGC timing control 1

Diagram



Fields

Field	Function
31-30 —	Reserved
29-26 AGC_WBD_DUAL_CLIP_TIMEOUT	Indicate the max duration, count by reference clk, for WBD debounce. If WBD not be triggered again in this time, debounce failed. If AGC_WBD_EN=2, the programmed AGC_WBD_DUAL_CLIP_TIMEOUT value should be no large than AGC_WBD_STEP2_TIMEOUT. This field has no effect when AGC_WBD_STEP1_DUAL_CLIP_EN || AGC_WBD_STEP2_DUAL_CLIP_EN=0.
25-20 AGC_WBD_STEP2_WAIT	AGC Gain Change Wait For WBD step2 Gain stable wait duration, count by referenct clock, for AGC WBD step2.
19-17 AGC_WBD_STEP1_DUAL_CLIP_WAIT	AGC WBD step1 Debounce Wait Time Cooldown duration for WBD step1.
16-14 AGC_WBD_STEP2_DUAL_CLIP_WAIT	AGC WBD step2 Debounce Wait Time Cooldown duration for WBD step2.
13-7 AGC_HOLD_TIMEOUT	AGC HOLD Mode Wait Time If AGC_HOLD_EN[1] = 1, AGC will trans to hold state when hold timeout matched. The actual hold timeout duration can be calculated as: hold_timeout_duration(microsecond) = AGC_HOLD_TIMEOUT + 1 The hold_timeout counter will start work at the point of AGC initialization done, and which will be reset when every gain index changed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-0 AGC_FREEZE_TIMEOUT	AGC FREEZE Mode Wait Time If AGC_FREEZE_EN[1] = 1, AGC will trans to FREEZE state when freeze timeout matched. The actual freeze timeout duration can be calculated as: freeze_timeout_duration(microsecond) = AGC_FREEZE_TIMEOUT + 1 If HOLD function enabled, the freeze_timeout counter will start work when AGC entered into HOLD mode,or it will start counting at the point of AGC initialization done. Similar to the hold_timeout counter, the freeze_timeout counter will be reset when every gain index changes.

55.4.7.4.3.1.42 AGC Timing Control 2 (AGC_TIMING2)

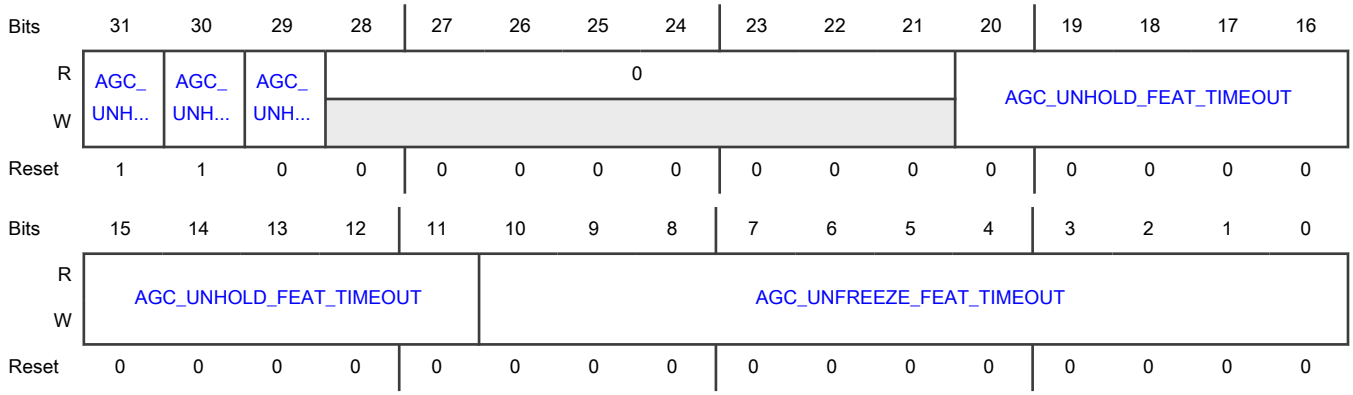
Offset

Register	Offset
AGC_TIMING2	A0h

Function

AGC timing control 2

Diagram



Fields

Field	Function
31 AGC_UNHOLD_MAG_SRC	AGC Magnitude Unhold Feature Source Selection Choose fast_mag or slow_mag for AGC magnitude unhold feature source. NOTE: This bit only takes effect when AGC_CTRL[AGC_UNHOLD_FEAT_EN][0]=1 0b - fast_mag 1b - slow_mag

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 AGC_UNHOLD_MAG_CNT	AGC Unhold Magnitude Count Selection This bit controls how many times the selected magnitude value exceeds unhold margin will unhold the AGC. 0: Once the selected magnitude value exceeds unhold margin, AGC will exit HOLD mode 1: Need two consecutive selected magnitude values exceeds unhold margin to let AGC exit HOLD mode NOTE: This bit only takes effect when AGC_CTRL[AGC_UNHOLD_FEAT_EN][0]=1
29 AGC_UNHOLD_GAIN_CHG	AGC Gain Index Change When UNHOLD If enabled, AGC will do gain index change immediately when AGC unhold by magnitude value exceed the hold margin. If the "mag > hold_mag + AGC_THR_MIS[HOLD_MARGIN_THR]", AGC will decrease the gain index by 1, while if "mag < hold_mag - AGC_THR_MIS[HOLD_MARGIN_THR]", AGC will increase the gain index by 1.
28-21 —	Reserved
20-11 AGC_UNHOLD_FEAT_TIMEOUT	AGC Unhold Feature Timeout When AGC_UNHOLD_EN[1]=1, AGC will exit HOLD mode and enter into MAG_ONLY mode if AA not be found in N microseconds. Where N equal to "AGC_UNHOLD_TIMEOUT + 1". If AGC_SOFT_RST_GAIN_SEL=1 and AGC currently at the non-initila gain index, AGC will exit HOLD mode with changing the gain index to AGC_INIT_IDX, thus the next state will be WAIT_GAIN_CHANGE followed by MAG_ONLY
10-0 AGC_UNFREEZE_FEAT_TIMEOUT	AGC Unfreeze Feature Timeout When AGC_UNFREEZE_EN, AGC will exit FREEZE mode and enter into MAG_ONLY mode if AA not be found in N microseconds. Where N equal to "AGC_UNFREEZE_TIMEOUT + 1". If AGC_SOFT_RST_GAIN_SEL=1 and AGC currently at the non-initila gain index, AGC will exit FREEZE mode with changing the gain index to AGC_INIT_IDX, thus the next state will be WAIT_GAIN_CHANGE followed by MAG_ONLY

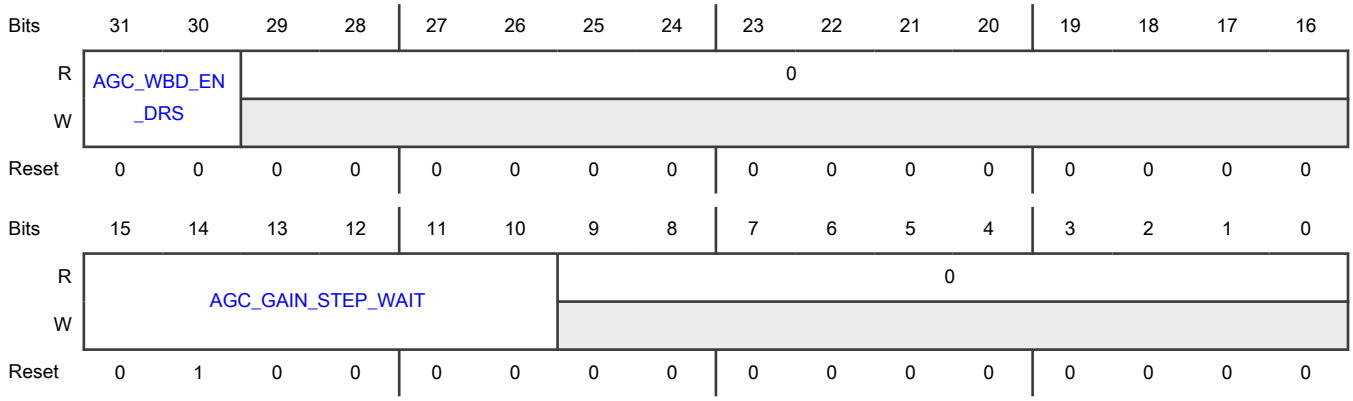
55.4.7.4.3.1.43 AGC Timing Control 0 DRS (AGC_TIMING0_DRS)

Offset

Register	Offset
AGC_TIMING0_DRS	A4h

Function
AGC timing control 0 for datarate_config_sel=1

Diagram



Fields

Field	Function
31-30 AGC_WBD_EN _DRS	DRS version of AGC_CTRL[AGC_WBD_EN]
29-16 —	Reserved
15-10 AGC_GAIN_ST EP_WAIT	AGC Gain Change Wait Time AGC_GAIN_STEP_WAIT for datarate_config_sel=1
9-0 —	Reserved

55.4.7.4.3.1.44 AGC Timing Control 1 DRS (AGC_TIMING1_DRS)

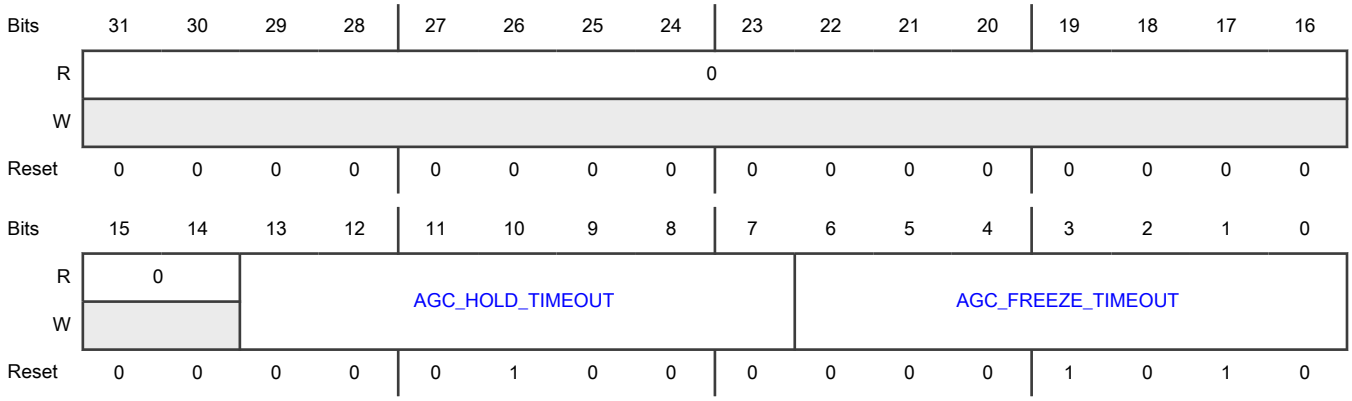
Offset

Register	Offset
AGC_TIMING1_DRS	A8h

Function

AGC timing control 1 for datarate_config_sel=1

Diagram



Fields

Field	Function
31-14 —	Reserved
13-7 AGC_HOLD_TIMEOUT	AGC HOLD Mode Wait Time AGC_HOLD_TIMEOUT for datarate_config_sel=1
6-0 AGC_FREEZE_TIMEOUT	AGC FREEZE Mode Wait Time AGC_FREEZE_TIMEOUT for datarate_config_sel=1

55.4.7.4.3.1.45 AGC Timing Control 2 DRS (AGC_TIMING2_DRS)

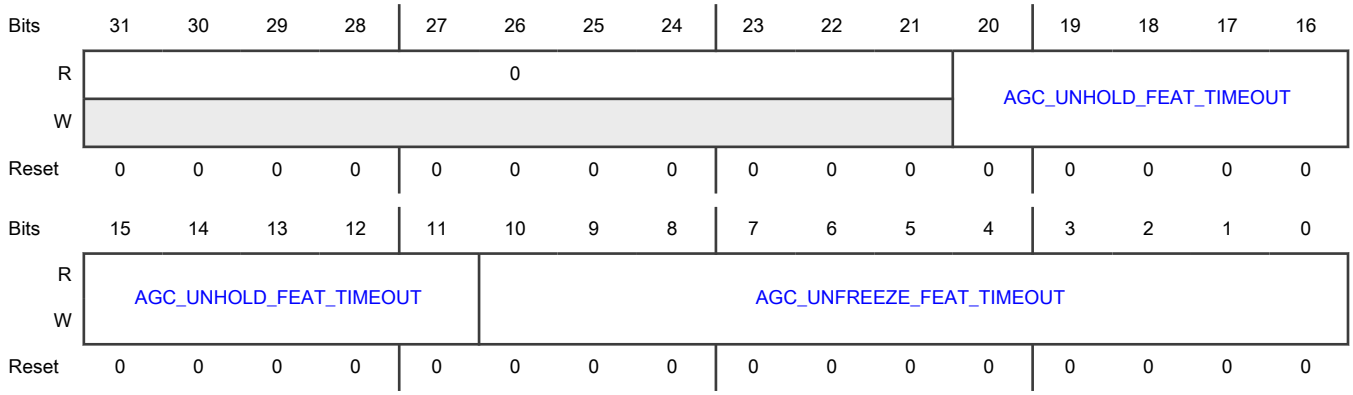
Offset

Register	Offset
AGC_TIMING2_DRS	ACh

Function

AGC timing control 2 for datarate_config_sel=1

Diagram



Fields

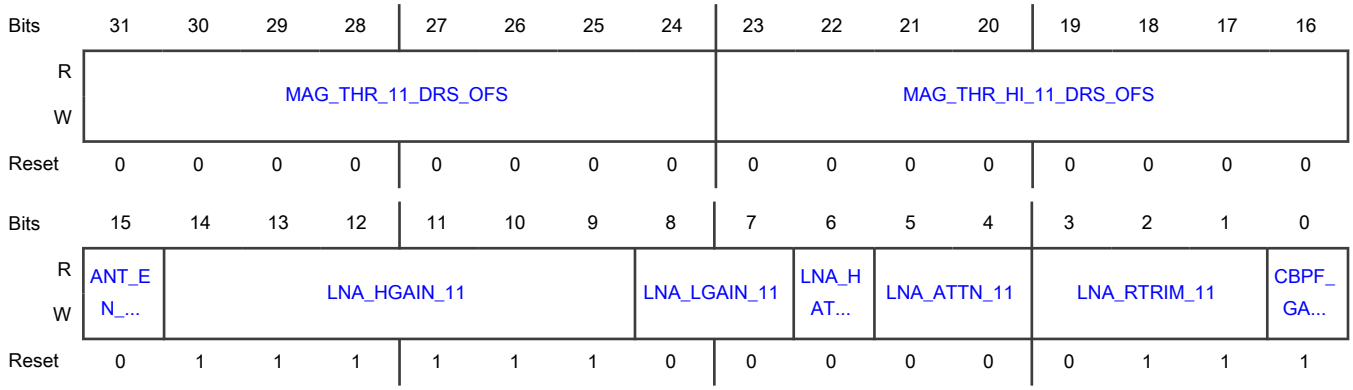
Field	Function
31-21 —	Reserved
20-11 AGC_UNHOLD_FEAT_TIMEOUT	AGC Unhold Feature Timeout AGC_UNHOLD_FEAT_TIMEOUT for datarate_config_sel=1
10-0 AGC_UNFREEZE_FEAT_TIMEOUT	AGC Unfreeze Feature Timeout AGC_UNFREEZE_FEAT_TIMEOUT for datarate_config_sel=1

55.4.7.4.3.1.46 AGC IDX11 Gain Config (AGC_IDX11_GAIN_CFG)

Offset

Register	Offset
AGC_IDX11_GAIN_CFG	B0h

Diagram



Fields

Field	Function
31-24 MAG_THR_11_DRS_OFS	Mag Thresh High DRS for AGC Gain Index 11 Magnitude threshold offset for AGC gain index 11 when datarate_config_sel = 1. Check the "AGC PHY Magnitude Threshold" chapter for more information.
23-16 MAG_THR_HI_11_DRS_OFS	Mag Thresh High DRS for AGC Gain Index 11 Magnitude threshold high offset for AGC gain index 11 when datarate_config_sel = 1. Check the "AGC PHY Magnitude Threshold" chapter for more information.
15 ANT_EN_RLOAD_11	ANT_EN_RLOAD_11 ldo_ant_en_rload during gain index=11
14-9 LNA_HGAIN_11	LNA_HGAIN_11 LNA high gain slices during gain index=11
8-7 LNA_LGAIN_11	LNA_LGAIN_11 LNA low gain slices during gain index=11
6 LNA_HATTN_11	LNA_HATTN_11 LNA high gain capacitor attenuation during gain index=11
5-4 LNA_ATTN_11	LNA_ATTN_11 LNA low gain capacitor attenuation during gain index=11
3-1 LNA_RTRIM_11	LNA_RTRIM_11 LNA RTFE matching resistor adjustment value during gain index=11
0 CBPF_GAIN_11	CBPF_GAIN_11 CBPF gain select during gain index=11

Table continues on the next page...

Table continued from the previous page...

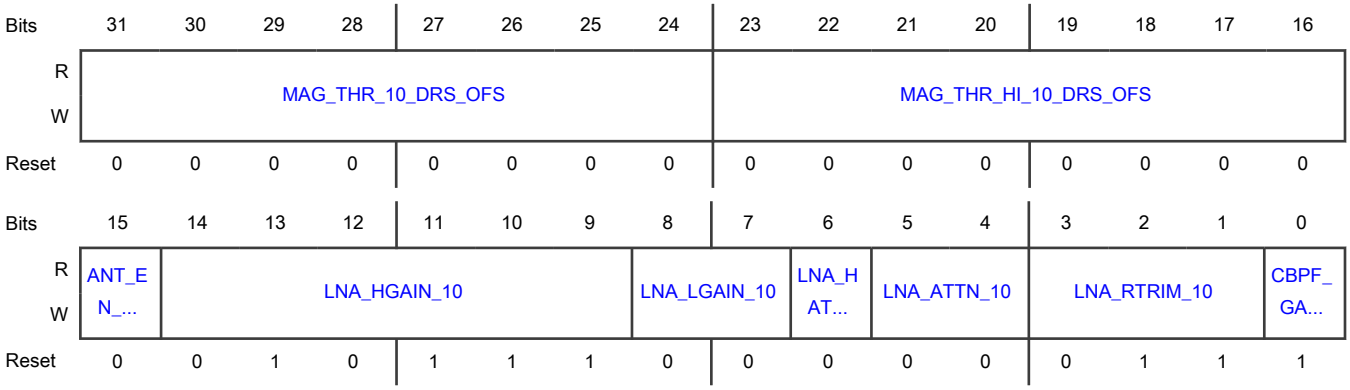
Field	Function
	0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.47 AGC IDX10 Gain Config (AGC_IDX10_GAIN_CFG)

Offset

Register	Offset
AGC_IDX10_GAIN_CFG	B4h

Diagram



Fields

Field	Function
31-24 MAG_THR_10_DRS_OFS	Magnitude threshold offset for AGC gain index 10 when datarate_config_sel = 1
23-16 MAG_THR_HI_10_DRS_OFS	Magnitude threshold high offset for AGC gain index 10 when datarate_config_sel = 1
15 ANT_EN_RLOAD_10	ANT_EN_RLOAD_10 ldo_ant_en_rload during gain index=10
14-9 LNA_HGAIN_10	LNA_HGAIN_10 LNA high gain slices during gain index=10

Table continues on the next page...

Table continued from the previous page...

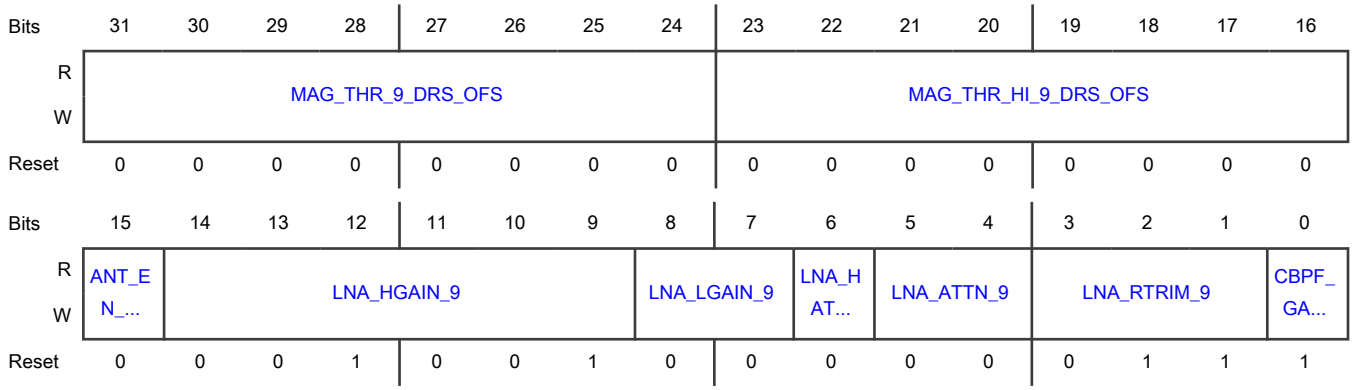
Field	Function
8-7 LNA_LGAIN_10	LNA_LGAIN_10 LNA low gain slices during gain index=10
6 LNA_HATTN_10	LNA_HATTN_10 LNA high gain capacitor attenuation during gain index=10
5-4 LNA_ATTN_10	LNA_ATTN_10 LNA low gain capacitor attenuation during gain index=10
3-1 LNA_RTRIM_10	LNA_RTRIM_10 LNA RTFE matching resistor adjustment value during gain index=10
0 CBPF_GAIN_10	CBPF_GAIN_10 CBPF gain select during gain index=10 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.48 AGC IDX9 Gain Config (AGC_IDX9_GAIN_CFG)

Offset

Register	Offset
AGC_IDX9_GAIN_CFG	B8h

Diagram



Fields

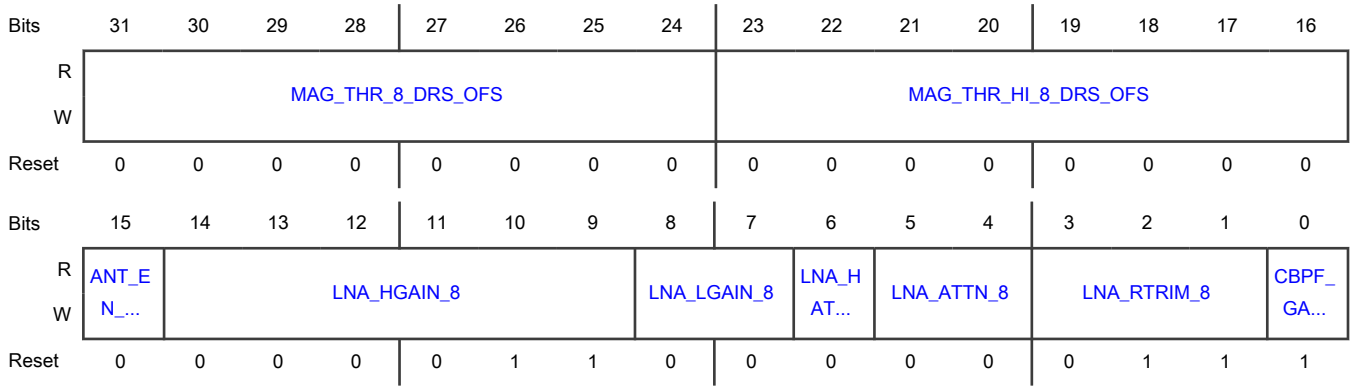
Field	Function
31-24 MAG_THR_9_DRS_OFS	Magnitude threshold offset for AGC gain index 9 when datarate_config_sel = 1
23-16 MAG_THR_HI_9_DRS_OFS	Magnitude threshold high offset for AGC gain index 9 when datarate_config_sel = 1
15 ANT_EN_RLOAD_9	ANT_EN_RLOAD_9 ldo_ant_en_rload during gain index=9
14-9 LNA_HGAIN_9	LNA_HGAIN_9 LNA high gain slices during gain index=9
8-7 LNA_LGAIN_9	LNA_LGAIN_9 LNA low gain slices during gain index=9
6 LNA_HATTN_9	LNA_HATTN_9 LNA high gain capacitor attenuation during gain index=9
5-4 LNA_ATTN_9	LNA_ATTN_9 LNA low gain capacitor attenuation during gain index=9
3-1 LNA_RTRIM_9	LNA_RTRIM_9 LNA RTFE matching resistor adjustment value during gain index=9
0 CBPF_GAIN_9	CBPF_GAIN_9 CBPF gain select during gain index=9 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.49 AGC IDX8 Gain Config (AGC_IDX8_GAIN_CFG)

Offset

Register	Offset
AGC_IDX8_GAIN_CFG	BCh

Diagram



Fields

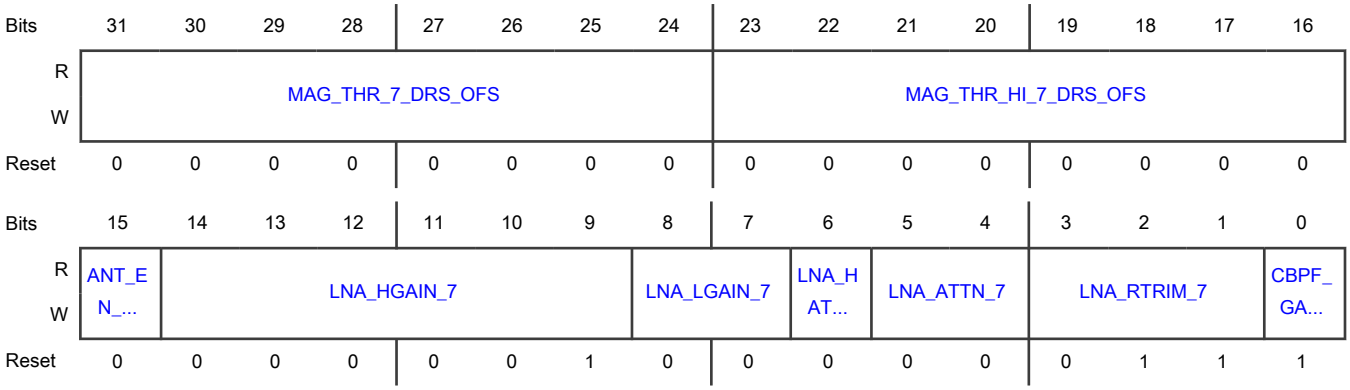
Field	Function
31-24 MAG_THR_8_DRS_OFS	Magnitude threshold offset for AGC gain index 8 when datarate_config_sel = 1
23-16 MAG_THR_HI_8_DRS_OFS	Magnitude threshold high offset for AGC gain index 8 when datarate_config_sel = 1
15 ANT_EN_RLOAD_8	ANT_EN_RLOAD_8 ldo_ant_en_rload during gain index=8
14-9 LNA_HGAIN_8	LNA_HGAIN_8 LNA high gain slices during gain index=8
8-7 LNA_LGAIN_8	LNA_LGAIN_8 LNA low gain slices during gain index=8
6 LNA_HATTN_8	LNA_HATTN_8 LNA high gain capacitor attenuation during gain index=8
5-4 LNA_ATTN_8	LNA_ATTN_8 LNA low gain capacitor attenuation during gain index=8
3-1 LNA_RTRIM_8	LNA_RTRIM_8 LNA RTFE matching resistor adjustment value during gain index=8
0 CBPF_GAIN_8	CBPF_GAIN_8 CBPF gain select during gain index=8 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.50 AGC IDX7 Gain Config (AGC_IDX7_GAIN_CFG)

Offset

Register	Offset
AGC_IDX7_GAIN_CFG	C0h

Diagram



Fields

Field	Function
31-24 MAG_THR_7_DRS_OFS	Magnitude threshold offset for AGC gain index 7 when datarate_config_sel = 1
23-16 MAG_THR_HI_7_DRS_OFS	Magnitude threshold high offset for AGC gain index 7 when datarate_config_sel = 1
15 ANT_EN_RLOAD_7	ANT_EN_RLOAD_7 ldo_ant_en_rload during gain index=7
14-9 LNA_HGAIN_7	LNA_HGAIN_7 LNA high gain slices during gain index=7
8-7 LNA_LGAIN_7	LNA_LGAIN_7 LNA low gain slices during gain index=7
6 LNA_HATTN_7	LNA_HATTN_7 LNA high gain capacitor attenuation during gain index=7
5-4	LNA_ATTN_7

Table continues on the next page...

Table continued from the previous page...

Field	Function
LNA_ATTN_7	LNA low gain capacitor attenuation during gain index=7
3-1 LNA_RTRIM_7	LNA_RTRIM_7 LNA RTFE matching resistor adjustment value during gain index=7
0 CBPF_GAIN_7	CBPF_GAIN_7 CBPF gain select during gain index=7 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.51 AGC IDX6 Gain Config (AGC_IDX6_GAIN_CFG)

Offset

Register	Offset
AGC_IDX6_GAIN_CFG	C4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAG_THR_6_DRS_OFS								MAG_THR_HI_6_DRS_OFS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ANT_EN_...	LNA_HGAIN_6						LNA_LGAIN_6		LNA_HAT...	LNA_ATTN_6		LNA_RTRIM_6			CBPF_GA...
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1

Fields

Field	Function
31-24 MAG_THR_6_DRS_OFS	Magnitude threshold offset for AGC gain index 6 when datarate_config_sel = 1
23-16 MAG_THR_HI_6_DRS_OFS	Magnitude threshold high offset for AGC gain index 6 when datarate_config_sel = 1

Table continues on the next page...

Table continued from the previous page...

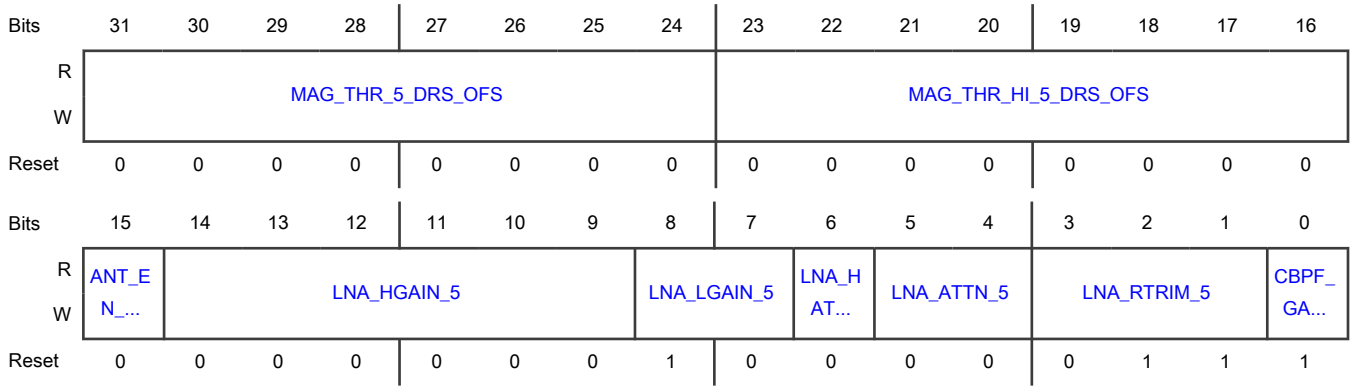
Field	Function
15 ANT_EN_RLOAD_6	ANT_EN_RLOAD_6 ldo_ant_en_rload during gain index=6
14-9 LNA_HGAIN_6	LNA_HGAIN_6 LNA high gain slices during gain index=6
8-7 LNA_LGAIN_6	LNA_LGAIN_6 LNA low gain slices during gain index=6
6 LNA_HATTN_6	LNA_HATTN_6 LNA high gain capacitor attenuation during gain index=6
5-4 LNA_ATTN_6	LNA_ATTN_6 LNA low gain capacitor attenuation during gain index=6
3-1 LNA_RTRIM_6	LNA_RTRIM_6 LNA RTFE matching resistor adjustment value during gain index=6
0 CBPF_GAIN_6	CBPF_GAIN_6 CBPF gain select during gain index=6 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.52 AGC IDX5 Gain Config (AGC_IDX5_GAIN_CFG)

Offset

Register	Offset
AGC_IDX5_GAIN_CFG	C8h

Diagram



Fields

Field	Function
31-24 MAG_THR_5_DRS_OFS	Magnitude threshold offset for AGC gain index 5 when datarate_config_sel = 1
23-16 MAG_THR_HI_5_DRS_OFS	Magnitude threshold high offset for AGC gain index 5 when datarate_config_sel = 1
15 ANT_EN_RLOAD_5	ANT_EN_RLOAD_5 ldo_ant_en_rload during gain index=5
14-9 LNA_HGAIN_5	LNA_HGAIN_5 LNA high gain slices during gain index=5
8-7 LNA_LGAIN_5	LNA_LGAIN_5 LNA low gain slices during gain index=5
6 LNA_HATTN_5	LNA_HATTN_5 LNA high gain capacitor attenuation during gain index=5
5-4 LNA_ATTN_5	LNA_ATTN_5 LNA low gain capacitor attenuation during gain index=5
3-1 LNA_RTRIM_5	LNA_RTRIM_5 LNA RTFE matching resistor adjustment value during gain index=5
0 CBPF_GAIN_5	CBPF_GAIN_5 CBPF gain select during gain index=5 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.53 AGC IDX4 Gain Config (AGC_IDX4_GAIN_CFG)

Offset

Register	Offset
AGC_IDX4_GAIN_CFG	CCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAG_THR_4_DRS_OFS								MAG_THR_HI_4_DRS_OFS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ANT_EN_...	LNA_HGAIN_4						LNA_LGAIN_4		LNA_HAT...	LNA_ATTN_4		LNA_RTRIM_4			CBPF_GA...
Reset	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1

Fields

Field	Function
31-24 MAG_THR_4_DRS_OFS	Magnitude threshold offset for AGC gain index 4 when datarate_config_sel = 1
23-16 MAG_THR_HI_4_DRS_OFS	Magnitude threshold high offset for AGC gain index 4 when datarate_config_sel = 1
15 ANT_EN_RLOAD_4	ANT_EN_RLOAD_4 ldo_ant_en_rload during gain index=4
14-9 LNA_HGAIN_4	LNA_HGAIN_4 LNA high gain slices during gain index=4
8-7 LNA_LGAIN_4	LNA_LGAIN_4 LNA low gain slices during gain index=4
6 LNA_HATTN_4	LNA_HATTN_4 LNA high gain capacitor attenuation during gain index=4
5-4	LNA_ATTN_4

Table continues on the next page...

Table continued from the previous page...

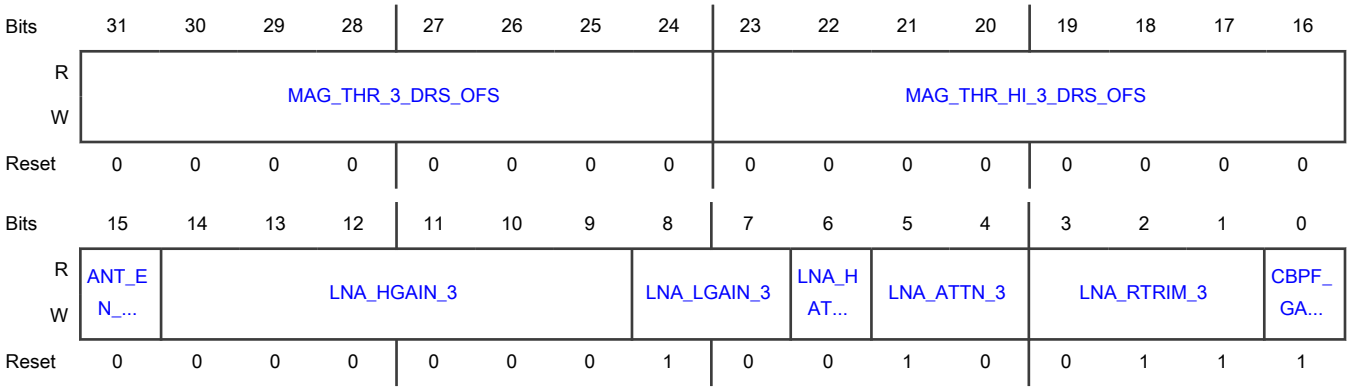
Field	Function
LNA_ATTN_4	LNA low gain capacitor attenuation during gain index=4
3-1 LNA_RTRIM_4	LNA_RTRIM_4 LNA RTFE matching resistor adjustment value during gain index=4
0 CBPF_GAIN_4	CBPF_GAIN_4 CBPF gain select during gain index=4 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.54 AGC IDX3 Gain Config (AGC_IDX3_GAIN_CFG)

Offset

Register	Offset
AGC_IDX3_GAIN_CFG	D0h

Diagram



Fields

Field	Function
31-24 MAG_THR_3_DRS_OFS	Magnitude threshold offset for AGC gain index 3 when datarate_config_sel = 1
23-16 MAG_THR_HI_3_DRS_OFS	Magnitude threshold high offset for AGC gain index 3 when datarate_config_sel = 1

Table continues on the next page...

Table continued from the previous page...

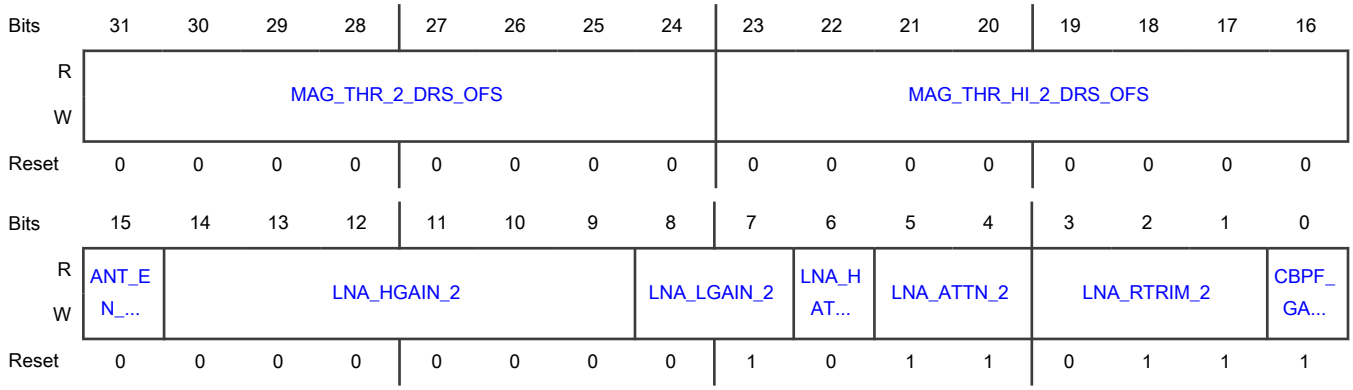
Field	Function
15 ANT_EN_RLOAD_3	ANT_EN_RLOAD_3 ldo_ant_en_rload during gain index=3
14-9 LNA_HGAIN_3	LNA_HGAIN_3 LNA high gain slices during gain index=3
8-7 LNA_LGAIN_3	LNA_LGAIN_3 LNA low gain slices during gain index=3
6 LNA_HATTN_3	LNA_HATTN_3 LNA high gain capacitor attenuation during gain index=3
5-4 LNA_ATTN_3	LNA_ATTN_3 LNA low gain capacitor attenuation during gain index=3
3-1 LNA_RTRIM_3	LNA_RTRIM_3 LNA RTFE matching resistor adjustment value during gain index=3
0 CBPF_GAIN_3	CBPF_GAIN_3 CBPF gain select during gain index=3 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.55 AGC IDX2 Gain Config (AGC_IDX2_GAIN_CFG)

Offset

Register	Offset
AGC_IDX2_GAIN_CFG	D4h

Diagram



Fields

Field	Function
31-24 MAG_THR_2_DRS_OFS	Magnitude threshold offset for AGC gain index 2 when datarate_config_sel = 1
23-16 MAG_THR_HI_2_DRS_OFS	Magnitude threshold high offset for AGC gain index 2 when datarate_config_sel = 1
15 ANT_EN_RLOAD_2	ANT_EN_RLOAD_2 ldo_ant_en_rload during gain index=2
14-9 LNA_HGAIN_2	LNA_HGAIN_2 LNA high gain slices during gain index=2
8-7 LNA_LGAIN_2	LNA_LGAIN_2 LNA low gain slices during gain index=2
6 LNA_HATTN_2	LNA_HATTN_2 LNA high gain capacitor attenuation during gain index=2
5-4 LNA_ATTN_2	LNA_ATTN_2 LNA low gain capacitor attenuation during gain index=2
3-1 LNA_RTRIM_2	LNA_RTRIM_2 LNA RTFE matching resistor adjustment value during gain index=2
0 CBPF_GAIN_2	CBPF_GAIN_2 CBPF gain select during gain index=2 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.56 AGC IDX1 Gain Config (AGC_IDX1_GAIN_CFG)

Offset

Register	Offset
AGC_IDX1_GAIN_CFG	D8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAG_THR_1_DRS_OFS								MAG_THR_HI_1_DRS_OFS							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ANT_EN_...	LNA_HGAIN_1						LNA_LGAIN_1		LNA_HAT...	LNA_ATTN_1		LNA_RTRIM_1			CBPF_GA...
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	1	1	0

Fields

Field	Function
31-24 MAG_THR_1_DRS_OFS	Magnitude threshold offset for AGC gain index 1 when datarate_config_sel = 1
23-16 MAG_THR_HI_1_DRS_OFS	Magnitude threshold high offset for AGC gain index 1 when datarate_config_sel = 1
15 ANT_EN_RLOAD_1	ANT_EN_RLOAD_1 ldo_ant_en_rload during gain index=1
14-9 LNA_HGAIN_1	LNA_HGAIN_1 LNA high gain slices during gain index=1
8-7 LNA_LGAIN_1	LNA_LGAIN_1 LNA low gain slices during gain index=1
6 LNA_HATTN_1	LNA_HATTN_1 LNA high gain capacitor attenuation during gain index=1
5-4	LNA_ATTN_1

Table continues on the next page...

Table continued from the previous page...

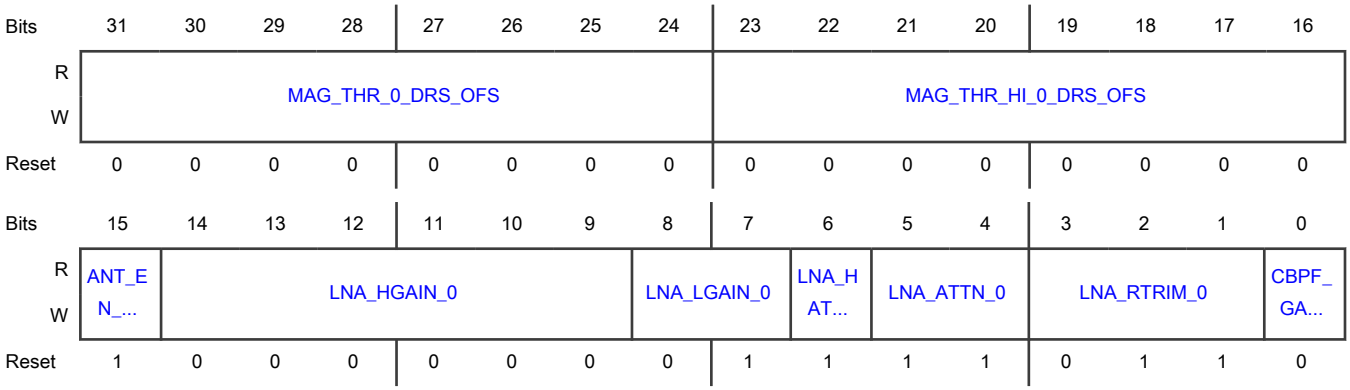
Field	Function
LNA_ATTN_1	LNA low gain capacitor attenuation during gain index=1
3-1 LNA_RTRIM_1	LNA_RTRIM_1 LNA RTFE matching resistor adjustment value during gain index=1
0 CBPF_GAIN_1	CBPF_GAIN_1 CBPF gain select during gain index=1 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.57 AGC IDX0 Gain Config (AGC_IDX0_GAIN_CFG)

Offset

Register	Offset
AGC_IDX0_GAIN_CFG	DCh

Diagram



Fields

Field	Function
31-24 MAG_THR_0_DRS_OFS	Magnitude threshold offset for AGC gain index 0 when datarate_config_sel = 1
23-16 MAG_THR_HI_0_DRS_OFS	Magnitude threshold high offset for AGC gain index 0 when datarate_config_sel = 1

Table continues on the next page...

Table continued from the previous page...

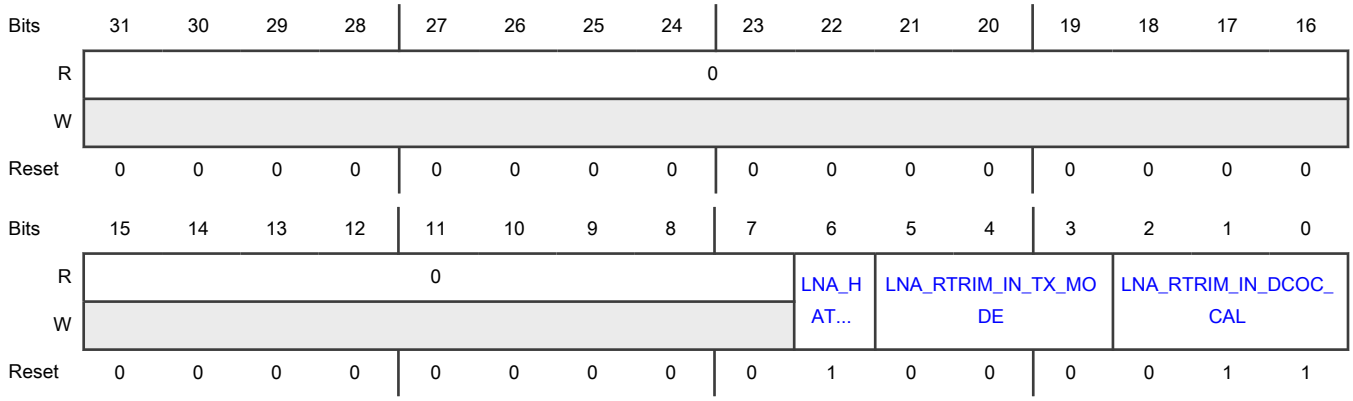
Field	Function
15 ANT_EN_RLOAD_0	ANT_EN_RLOAD_0 ldo_ant_en_rload during gain index=0
14-9 LNA_HGAIN_0	LNA_HGAIN_0 LNA high gain slices during gain index=0
8-7 LNA_LGAIN_0	LNA_LGAIN_0 LNA low gain slices during gain index=0
6 LNA_HATTN_0	LNA_HATTN_0 LNA high gain capacitor attenuation during gain index=0
5-4 LNA_ATTN_0	LNA_ATTN_0 LNA low gain capacitor attenuation during gain index=0
3-1 LNA_RTRIM_0	LNA_RTRIM_0 LNA RTFE matching resistor adjustment value during gain index=0
0 CBPF_GAIN_0	CBPF_GAIN_0 CBPF gain select during gain index=0 0b - -6 dB 1b - 0 dB

55.4.7.4.3.1.58 AGC Miscellaneous Gain Config (AGC_MIS_GAIN_CFG)

Offset

Register	Offset
AGC_MIS_GAIN_CFG	E0h

Diagram



Fields

Field	Function
31-7 —	Reserved
6 LNA_HATTN_IN_TX_MODE	LNA high gain capacitor attenuation value in TX mode
5-3 LNA_RTRIM_IN_TX_MODE	LNA RTFE matching resistor adjustment value in TX mode
2-0 LNA_RTRIM_IN_DCOC_CAL	LNA RTFE matching resistor adjustment value during DCOC calibration phase.

55.4.7.4.3.1.59 AGC IDX11 Gain Value (AGC_IDX11_GAIN_VAL)

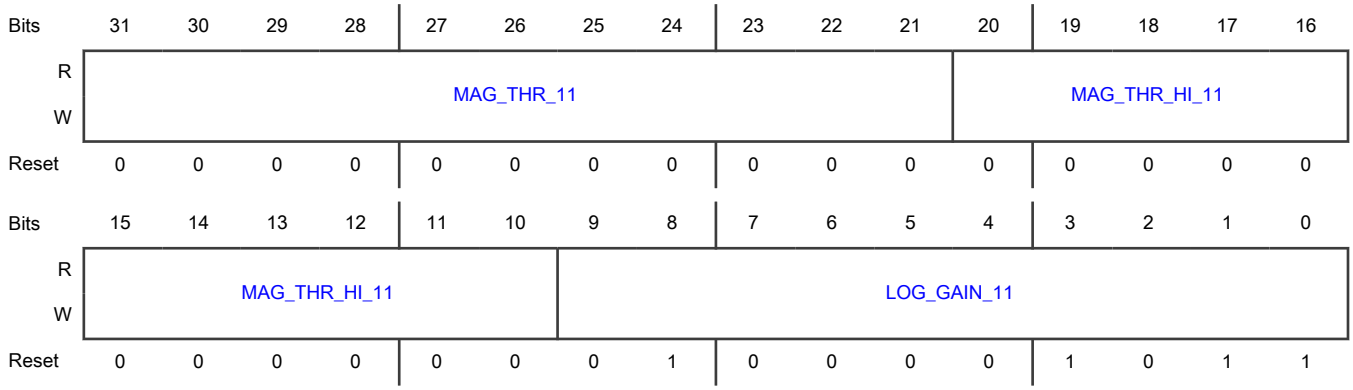
Offset

Register	Offset
AGC_IDX11_GAIN_VAL	E4h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_11	Magnitude threshold for AGC gain index 11. Check the "AGC PHY Magnitude Threshold" chapter for more information.
20-10 MAG_THR_HI_11	Magnitude threshold high for AGC gain index 11. Check the "AGC PHY Magnitude Threshold" chapter for more information.
9-0 LOG_GAIN_11	LOG_GAIN_11 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.60 AGC_IDX10_GAIN_VAL (AGC_IDX10_GAIN_VAL)

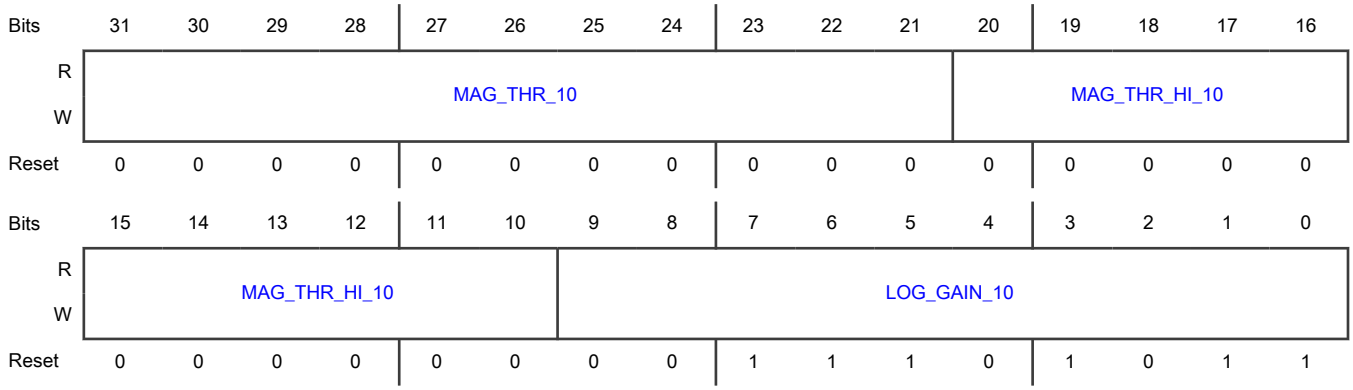
Offset

Register	Offset
AGC_IDX10_GAIN_VAL	E8h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_10	Magnitude threshold for AGC gain index 10
20-10 MAG_THR_HI_10	Magnitude threshold high for AGC gain index 10
9-0 LOG_GAIN_10	LOG_GAIN_10 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.61 AGC_IDX9_GAIN_VAL (AGC_IDX9_GAIN_VAL)

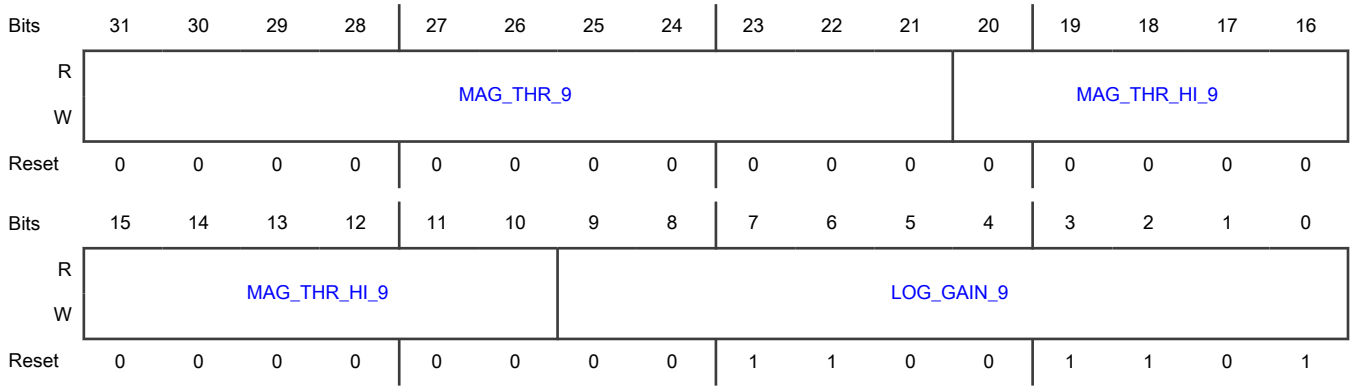
Offset

Register	Offset
AGC_IDX9_GAIN_VAL	ECh

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_9	Magnitude threshold for AGC gain index 9
20-10 MAG_THR_HI_9	Magnitude threshold high for AGC gain index 9
9-0 LOG_GAIN_9	LOG_GAIN_9 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.62 AGC_IDX8_GAIN_VAL (AGC_IDX8_GAIN_VAL)

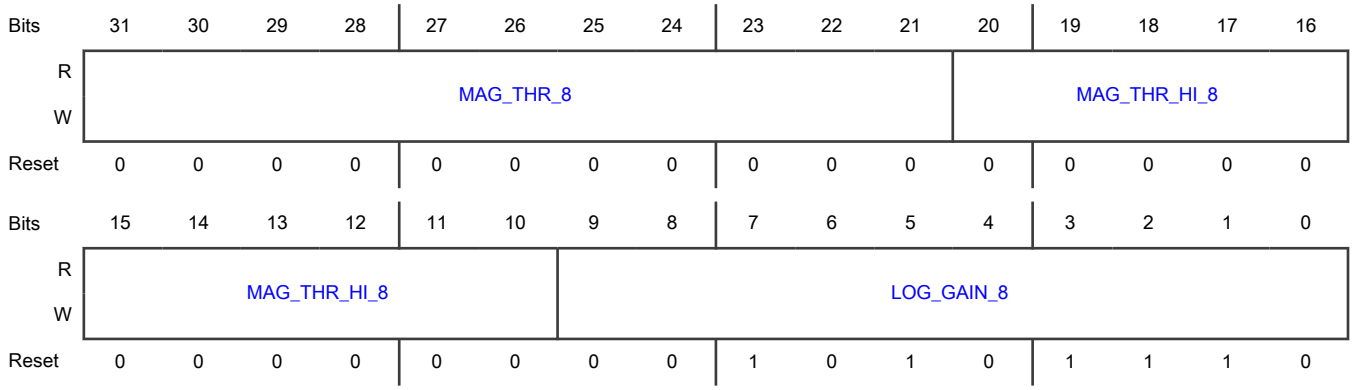
Offset

Register	Offset
AGC_IDX8_GAIN_VAL	F0h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_8	Magnitude threshold for AGC gain index 8
20-10 MAG_THR_HI_8	Magnitude threshold high for AGC gain index 8
9-0 LOG_GAIN_8	LOG_GAIN_8 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.63 AGC_IDX7_GAIN_VAL (AGC_IDX7_GAIN_VAL)

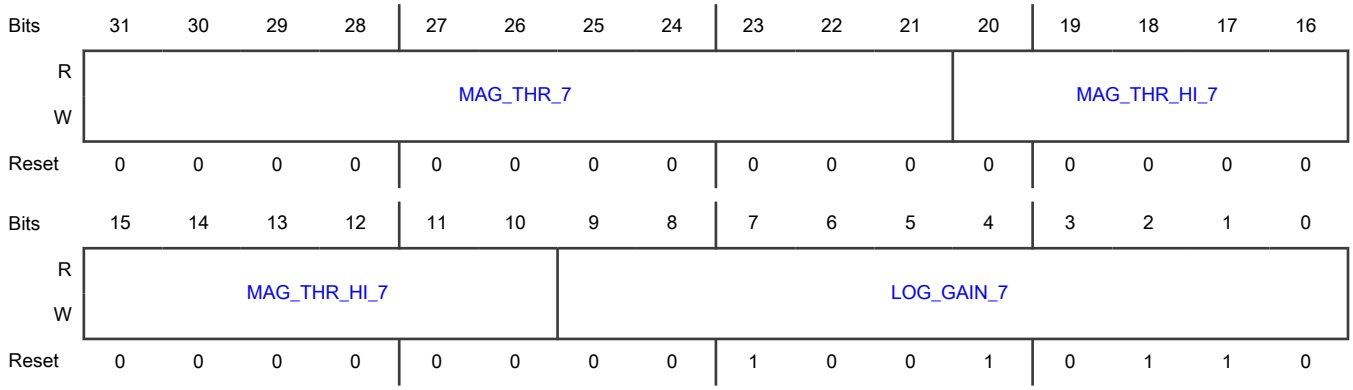
Offset

Register	Offset
AGC_IDX7_GAIN_VAL	F4h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_7	Magnitude threshold for AGC gain index 7
20-10 MAG_THR_HI_7	Magnitude threshold high for AGC gain index 7
9-0 LOG_GAIN_7	LOG_GAIN_7 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.64 AGC_IDX6_GAIN_VAL (AGC_IDX6_GAIN_VAL)

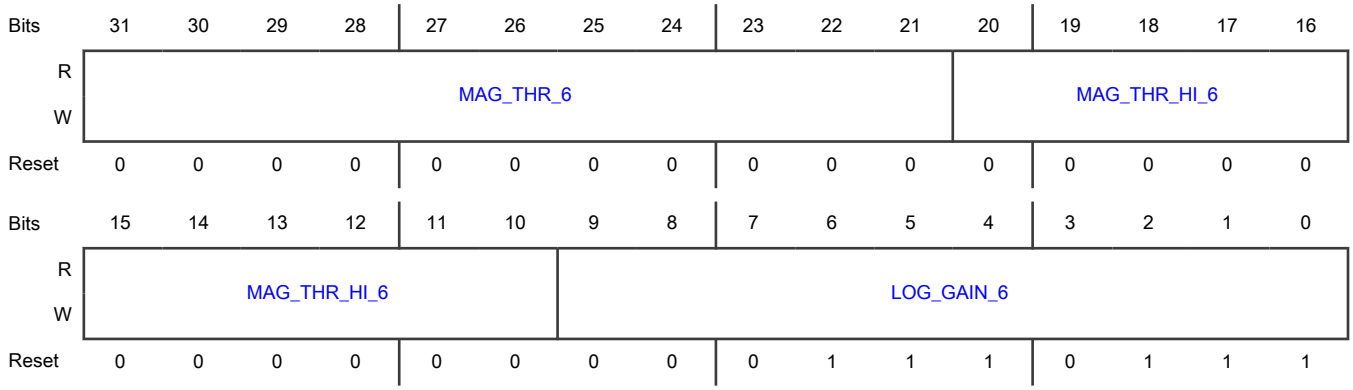
Offset

Register	Offset
AGC_IDX6_GAIN_VAL	F8h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_6	Magnitude threshold for AGC gain index 6
20-10 MAG_THR_HI_6	Magnitude threshold highfor AGC gain index 6
9-0 LOG_GAIN_6	LOG_GAIN_6 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.65 AGC_IDX5_GAIN_VAL (AGC_IDX5_GAIN_VAL)

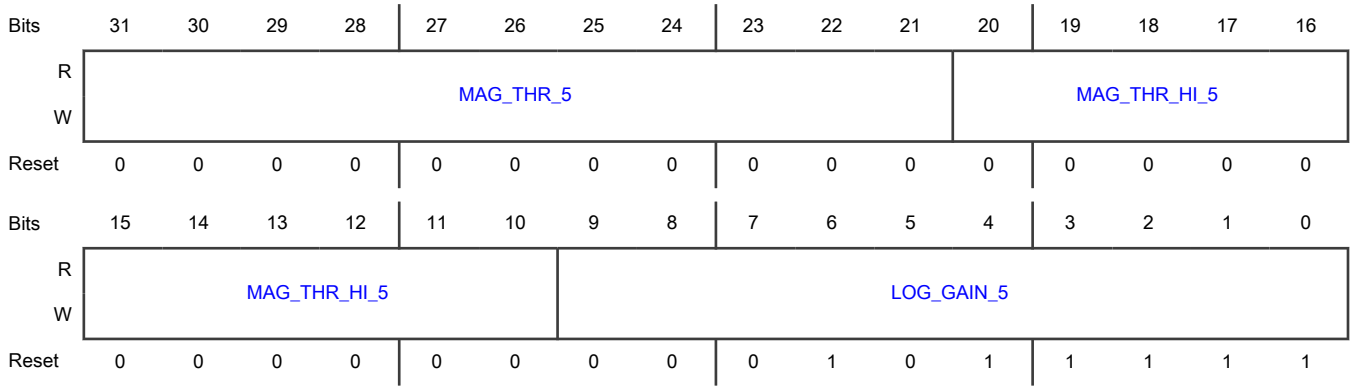
Offset

Register	Offset
AGC_IDX5_GAIN_VAL	FCh

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_5	Magnitude threshold for AGC gain index 5
20-10 MAG_THR_HI_5	Magnitude threshold high for AGC gain index 5
9-0 LOG_GAIN_5	LOG_GAIN_5 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.66 AGC_IDX4_GAIN_VAL (AGC_IDX4_GAIN_VAL)

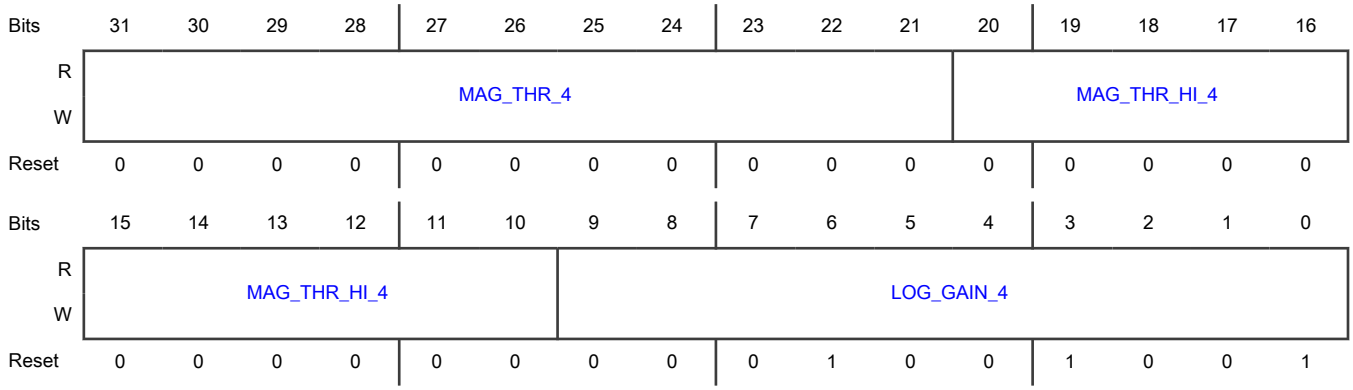
Offset

Register	Offset
AGC_IDX4_GAIN_VAL	100h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_4	Magnitude threshold for AGC gain index 4
20-10 MAG_THR_HI_4	Magnitude threshold high for AGC gain index 4
9-0 LOG_GAIN_4	LOG_GAIN_4 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.67 AGC_IDX3_GAIN_VAL (AGC_IDX3_GAIN_VAL)

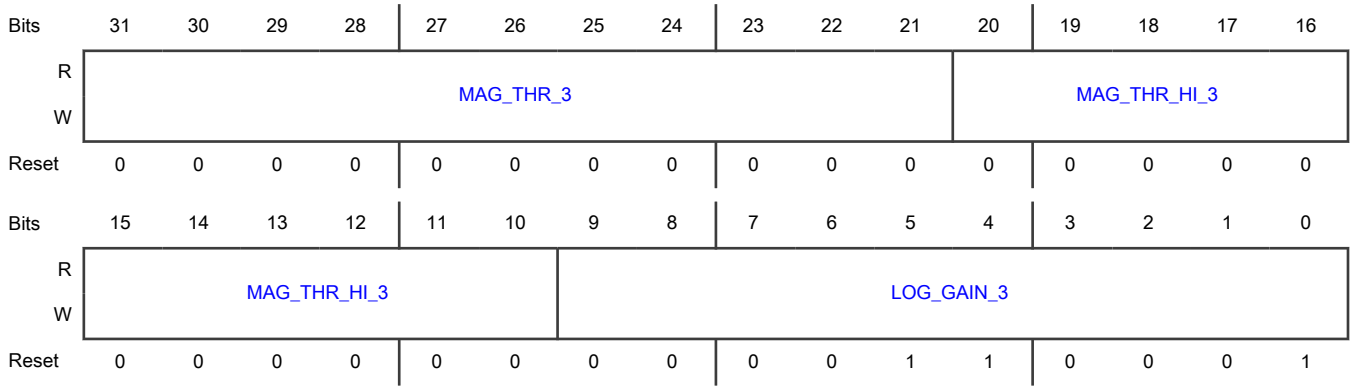
Offset

Register	Offset
AGC_IDX3_GAIN_VAL	104h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_3	Magnitude threshold for AGC gain index 3
20-10 MAG_THR_HI_3	Magnitude threshold high for AGC gain index 3
9-0 LOG_GAIN_3	LOG_GAIN_3 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.68 AGC_IDX2_GAIN_VAL (AGC_IDX2_GAIN_VAL)

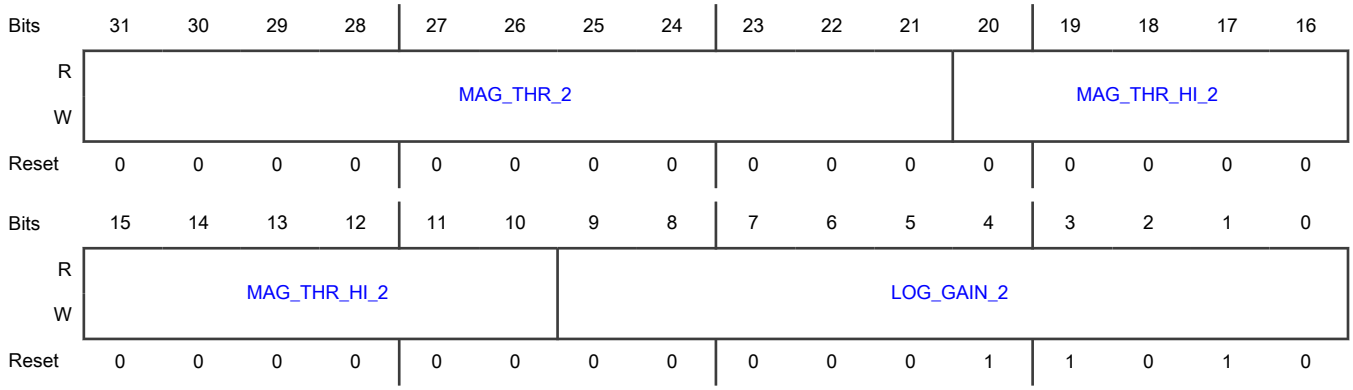
Offset

Register	Offset
AGC_IDX2_GAIN_VAL	108h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_2	Magnitude threshold for AGC gain index 2
20-10 MAG_THR_HI_2	Magnitude threshold high for AGC gain index 2
9-0 LOG_GAIN_2	LOG_GAIN_2 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.69 AGC_IDX1_GAIN_VAL (AGC_IDX1_GAIN_VAL)

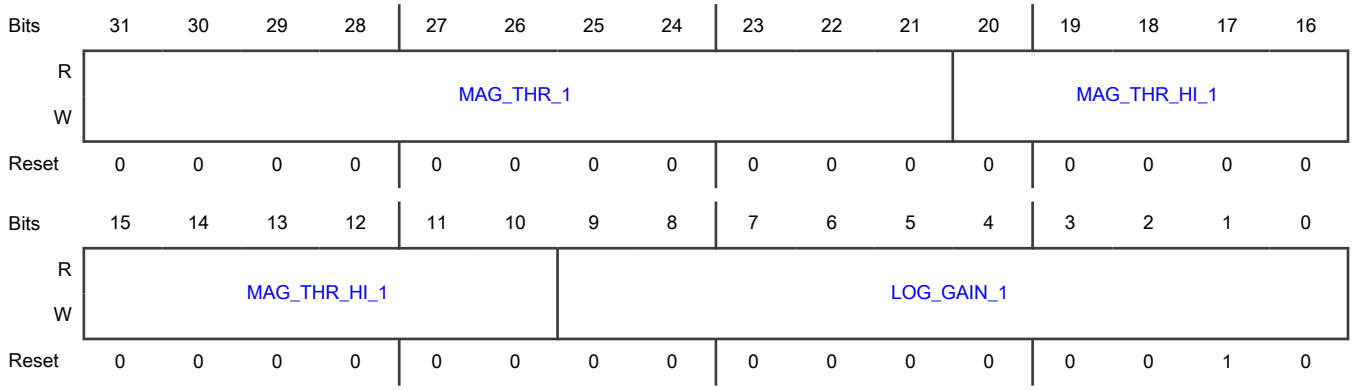
Offset

Register	Offset
AGC_IDX1_GAIN_VAL	10Ch

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_1	Magnitude threshold for AGC gain index 1
20-10 MAG_THR_HI_1	Magnitude threshold high for AGC gain index 1
9-0 LOG_GAIN_1	LOG_GAIN_1 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.70 AGC_IDX0_GAIN_VAL (AGC_IDX0_GAIN_VAL)

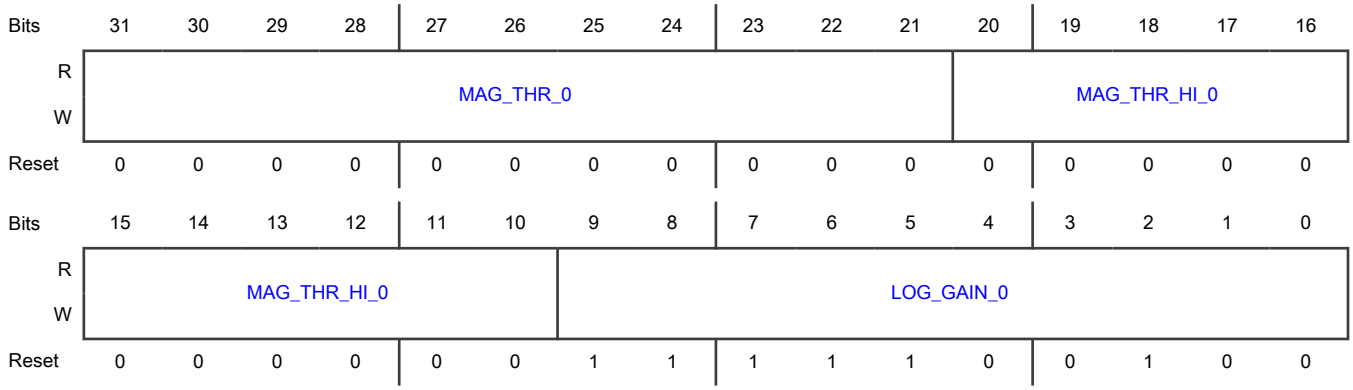
Offset

Register	Offset
AGC_IDX0_GAIN_VAL	110h

Function

AGC_IDXn_GAIN_VAL(where n can be 11~0) registers store the analog gain values during AGC_GAIN_IDX=n. These values using for WB-RSSI and NB-RSSI calculation and also can be used by PHY or software.

Diagram



Fields

Field	Function
31-21 MAG_THR_0	Magnitude threshold for AGC gain index 0
20-10 MAG_THR_HI_0	Magnitude threshold high for AGC gain index 0
9-0 LOG_GAIN_0	LOG_GAIN_0 LOG_GAIN_n represent the logarithmic gain values store in s7.2 format.

55.4.7.4.3.1.71 AGC Fast Mode Threshold (AGC_THR_FAST)

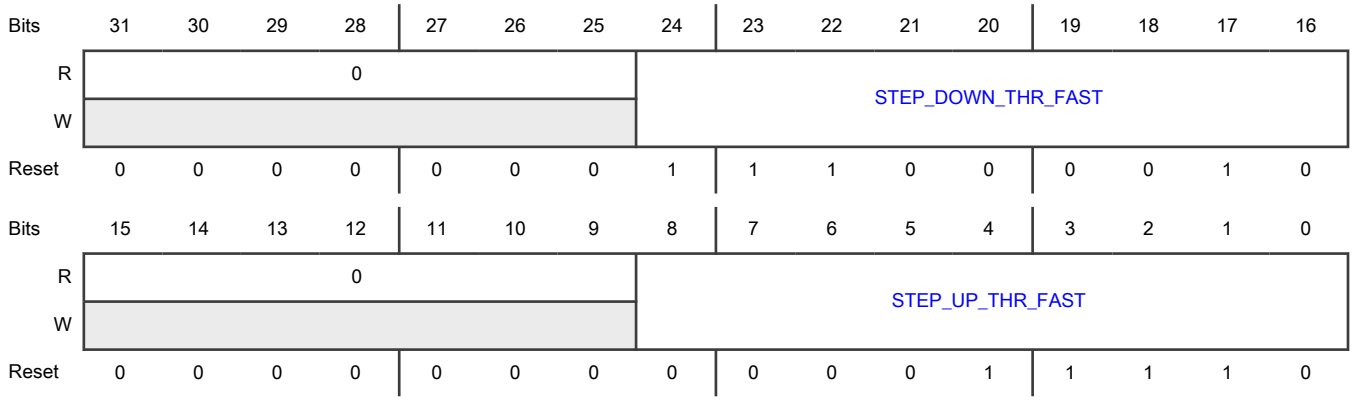
Offset

Register	Offset
AGC_THR_FAST	114h

Function

The threshold for FAST_MAG mode. NOTE: All gain index share the same FAST_MAG mode threshold value.

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 STEP_DOWN_THR_FAST	STEP_DOWN_THR_FAST
15-9 —	Reserved
8-0 STEP_UP_THR_FAST	STEP_UP_THR_FAST

55.4.7.4.3.1.72 AGC Fast Mode Threshold DRS (AGC_THR_FAST_DRS)

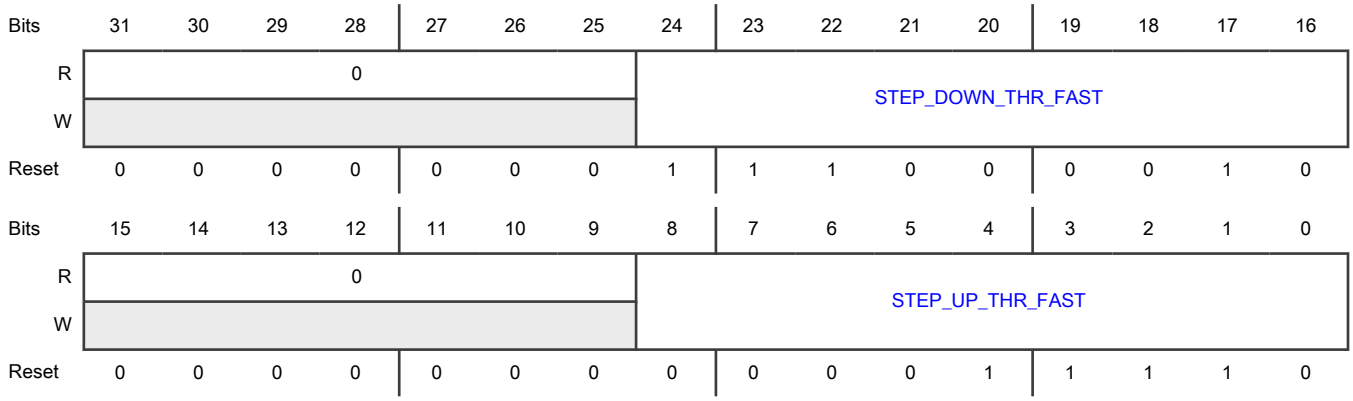
Offset

Register	Offset
AGC_THR_FAST_DRS	118h

Function

AGC_THR_FAST for datarate_config_sel=1

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 STEP_DOWN_THR_FAST	STEP_DOWN_THR_FAST
15-9 —	Reserved
8-0 STEP_UP_THR_FAST	STEP_UP_THR_FAST

55.4.7.4.3.1.73 AGC IDX11 Slow Mode Threshold (AGC_IDX11_THR)

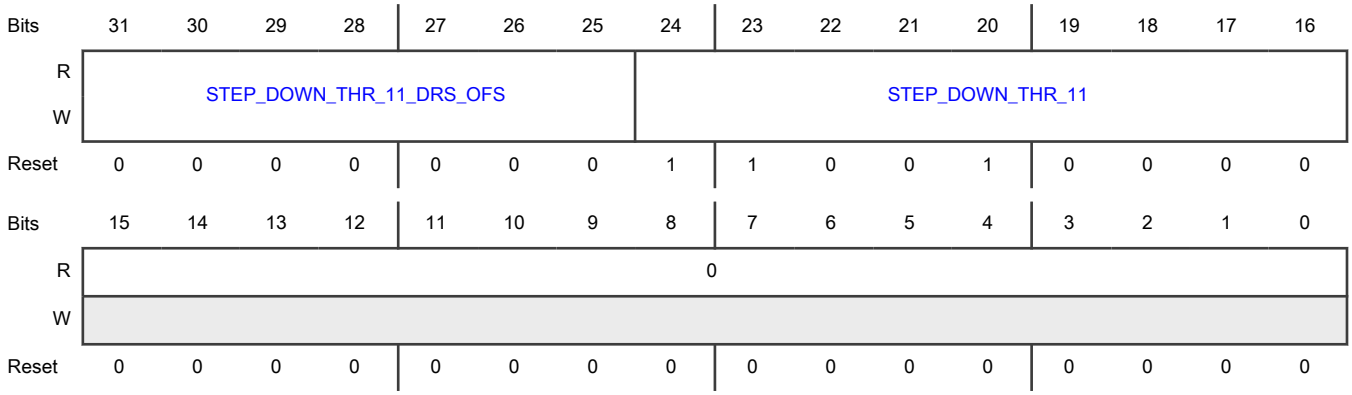
Offset

Register	Offset
AGC_IDX11_THR	11Ch

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. When slow_mag > STEP_DOWN_THR, AGC will decrease the gain index. When slow_mag < STEP_UP_THR, AGC will increase the gain index. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_11_DRS_OFS	STEP_DOWN_THR_11 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 11 is "STEP_DOWN_THR_11 + STEP_DOWN_THR_11_DRS_OFS". NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_11	STEP_DOWN_THR_11 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-0 —	Reserved

55.4.7.4.3.1.74 AGC IDX10 Slow Mode Threshold (AGC_IDX10_THR)

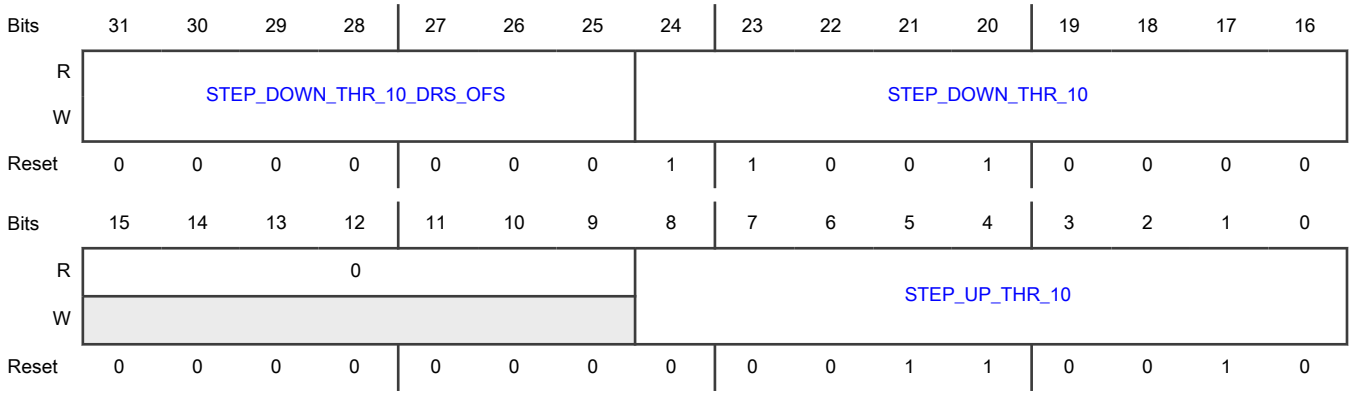
Offset

Register	Offset
AGC_IDX10_THR	120h

Function

See AGC_IDX11_THR for more information.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_10_DRS_OFS	STEP_DOWN_THR_10 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 10 is "STEP_DOWN_THR_10 + STEP_DOWN_THR_10_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_10	STEP_DOWN_THR_10 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_10	STEP_UP_THR_10 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.75 AGC IDX9 Slow Mode Threshold (AGC_IDX9_THR)

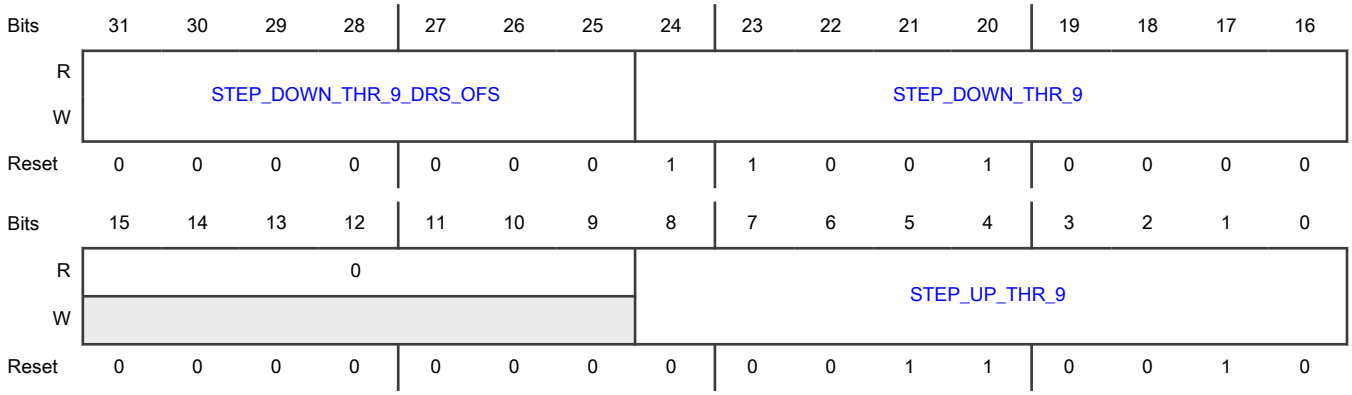
Offset

Register	Offset
AGC_IDX9_THR	124h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_9_DRS_OFS	STEP_DOWN_THR_9 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 9 is "STEP_DOWN_THR_9 + STEP_DOWN_THR_9_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_9	STEP_DOWN_THR_9 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_9	STEP_UP_THR_9 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.76 AGC IDX8 Slow Mode Threshold (AGC_IDX8_THR)

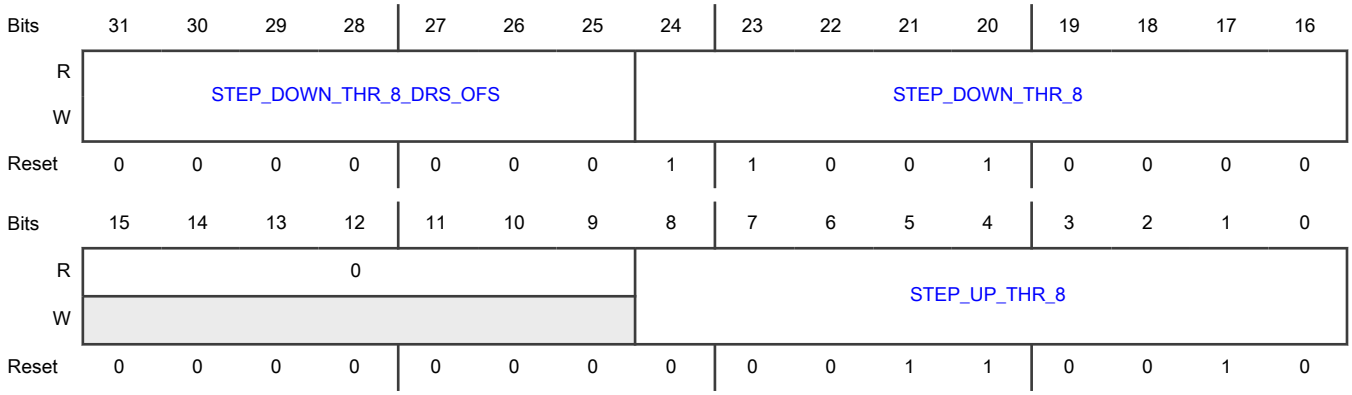
Offset

Register	Offset
AGC_IDX8_THR	128h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_8_DRS_OFS	STEP_DOWN_THR_8 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 8 is "STEP_DOWN_THR_8 + STEP_DOWN_THR_8_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_8	STEP_DOWN_THR_8 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_8	STEP_UP_THR_8 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.77 AGC IDX7 Slow Mode Threshold (AGC_IDX7_THR)

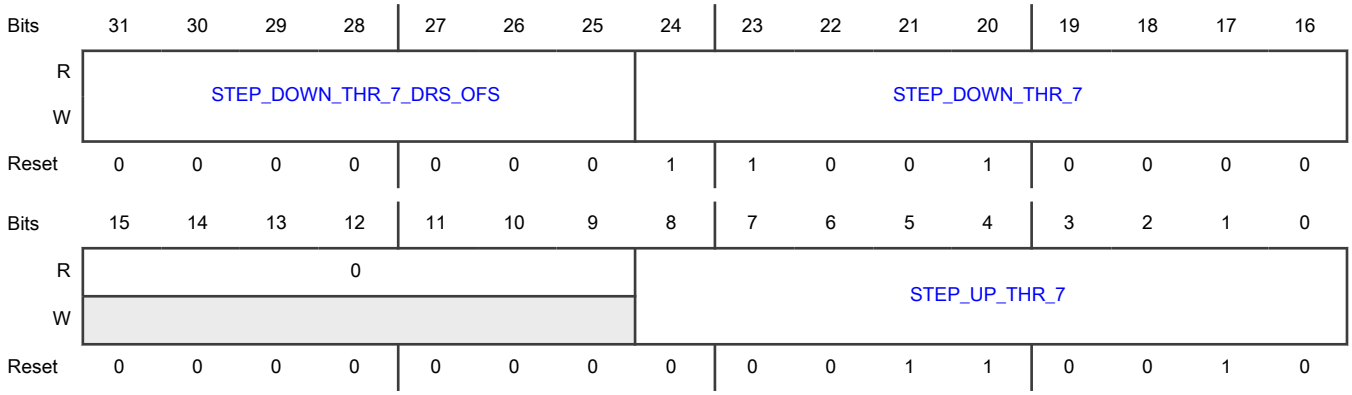
Offset

Register	Offset
AGC_IDX7_THR	12Ch

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_7_DRS_OFS	STEP_DOWN_THR_7 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 7 is "STEP_DOWN_THR_7 + STEP_DOWN_THR_7_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_7	STEP_DOWN_THR_7 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_7	STEP_UP_THR_7 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.78 AGC IDX6 Slow Mode Threshold (AGC_IDX6_THR)

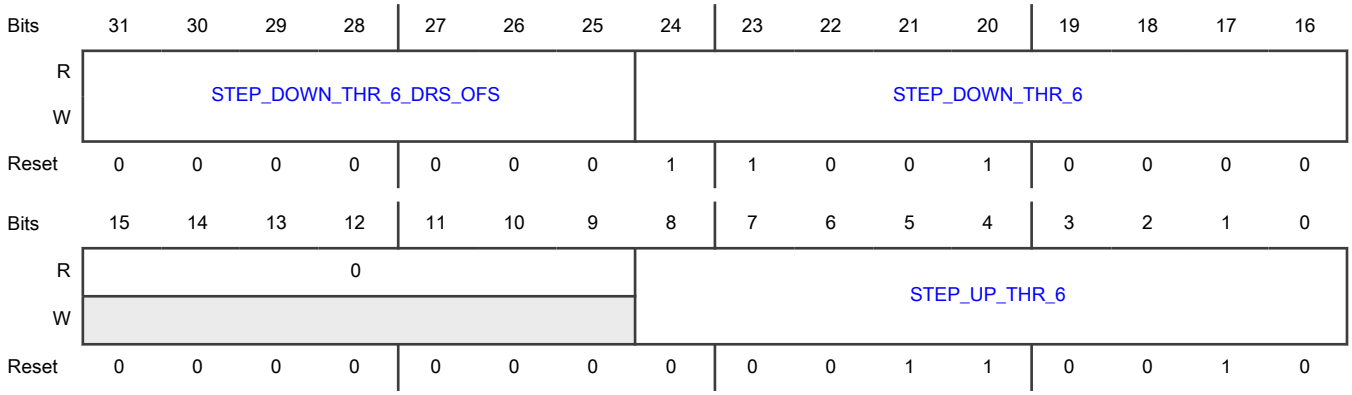
Offset

Register	Offset
AGC_IDX6_THR	130h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_6_DRS_OFS	STEP_DOWN_THR_6 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 6 is "STEP_DOWN_THR_6 + STEP_DOWN_THR_6_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_6	STEP_DOWN_THR_6 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_6	STEP_UP_THR_6 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.79 AGC IDX5 Slow Mode Threshold (AGC_IDX5_THR)

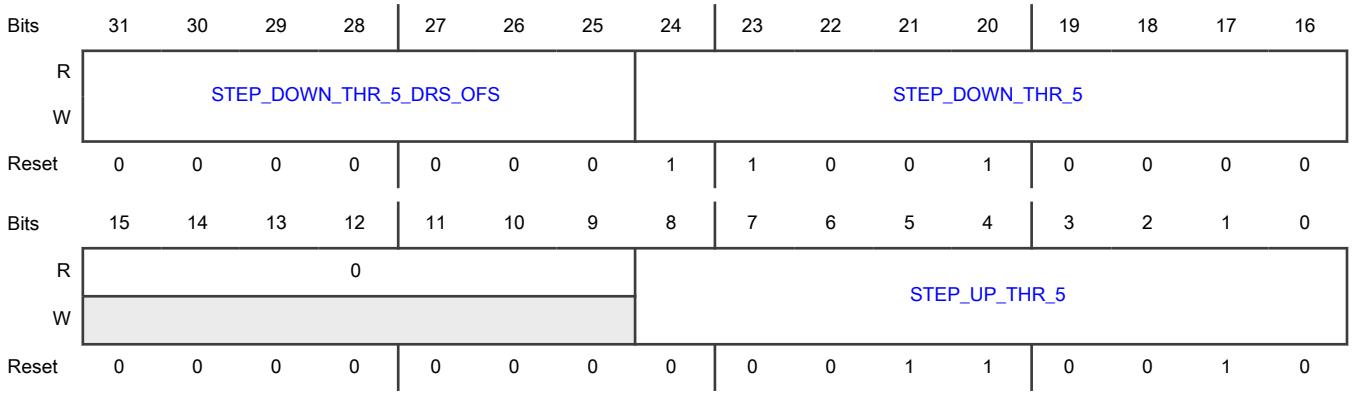
Offset

Register	Offset
AGC_IDX5_THR	134h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_5_DRS_OFS	STEP_DOWN_THR_5 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 5 is "STEP_DOWN_THR_5 + STEP_DOWN_THR_5_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_5	STEP_DOWN_THR_5 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_5	STEP_UP_THR_5 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.80 AGC IDX4 Slow Mode Threshold (AGC_IDX4_THR)

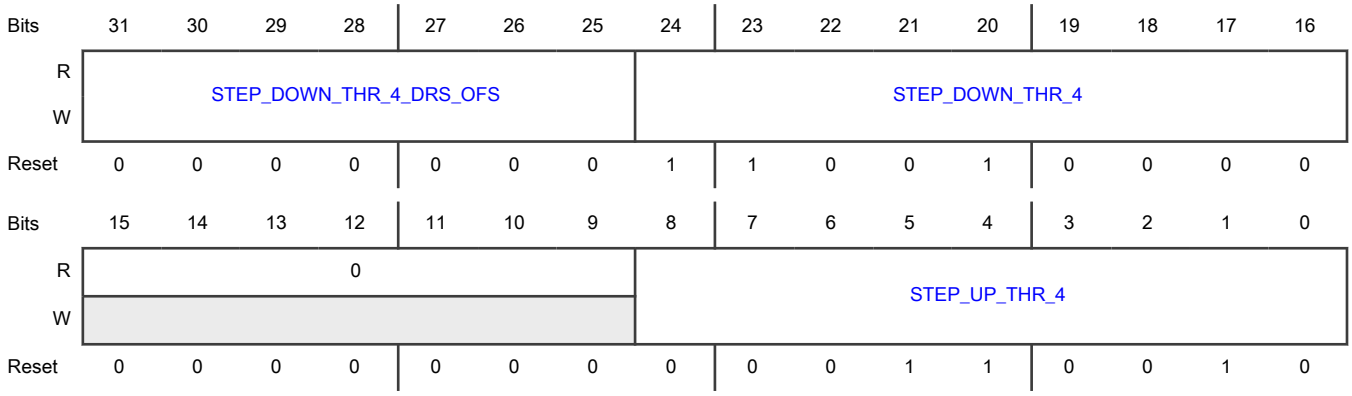
Offset

Register	Offset
AGC_IDX4_THR	138h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_4_DRS_OFS	STEP_DOWN_THR_4 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 4 is "STEP_DOWN_THR_4 + STEP_DOWN_THR_4_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_4	STEP_DOWN_THR_4 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_4	STEP_UP_THR_4 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.81 AGC IDX3 Slow Mode Threshold (AGC_IDX3_THR)

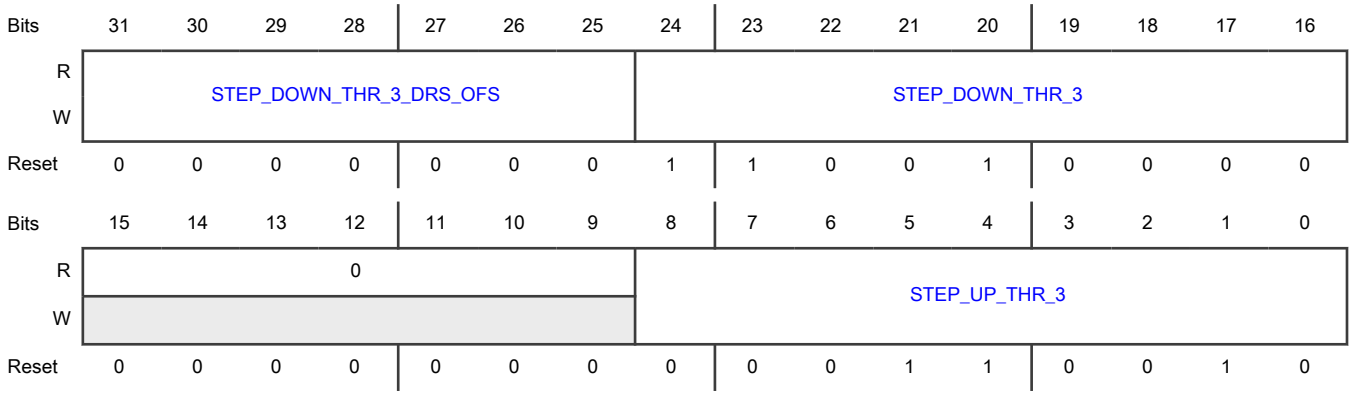
Offset

Register	Offset
AGC_IDX3_THR	13Ch

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_3_DRS_OFS	STEP_DOWN_THR_3 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 3 is "STEP_DOWN_THR_3 + STEP_DOWN_THR_3_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_3	STEP_DOWN_THR_3 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_3	STEP_UP_THR_3 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.82 AGC IDX2 Slow Mode Threshold (AGC_IDX2_THR)

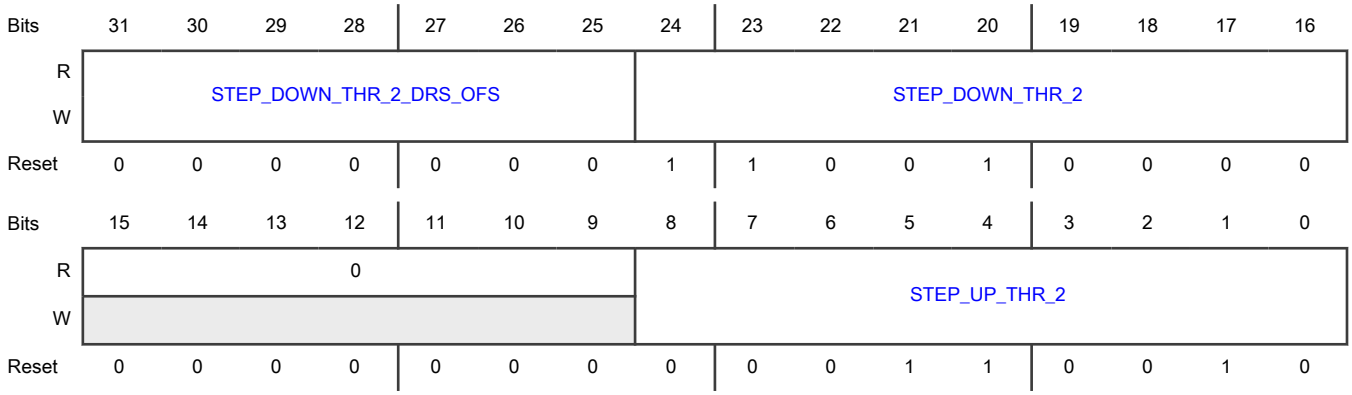
Offset

Register	Offset
AGC_IDX2_THR	140h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_2_DRS_OFS	STEP_DOWN_THR_2 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 2 is "STEP_DOWN_THR_2 + STEP_DOWN_THR_2_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_2	STEP_DOWN_THR_2 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_2	STEP_UP_THR_2 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.83 AGC IDX1 Slow Mode Threshold (AGC_IDX1_THR)

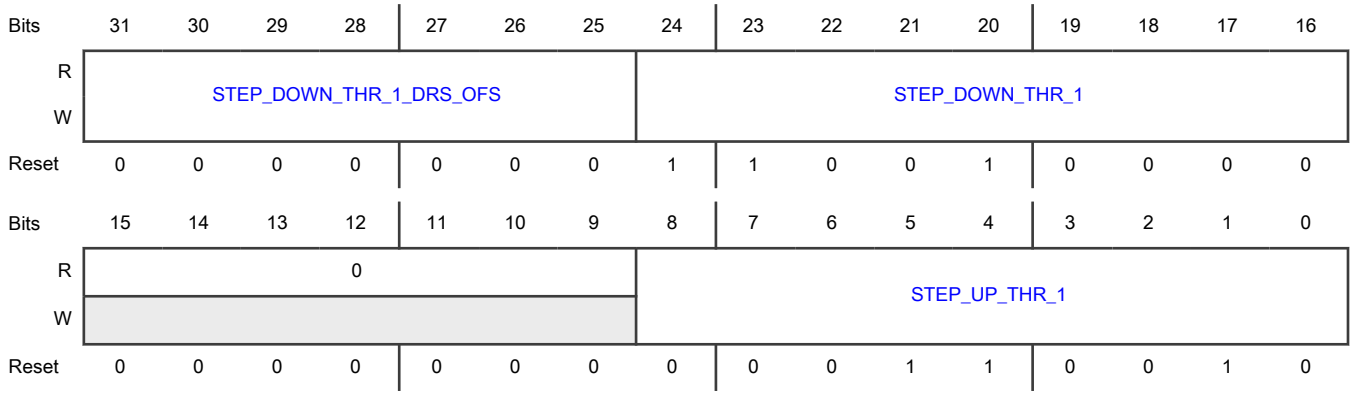
Offset

Register	Offset
AGC_IDX1_THR	144h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

Field	Function
31-25 STEP_DOWN_THR_1_DRS_OFS	STEP_DOWN_THR_1 DRS Offset When datarate_sel=1, the actual slow mode step down threshold in gain idx 1 is "STEP_DOWN_THR_1 + STEP_DOWN_THR_1_DRS_OFS" NOTE: 1. "STEP_DOWN_THR_*_DRS_OFS" is a signed value. 2. the "STEP_DOWN_THR_* + STEP_DOWN_THR_*_DRS_OFS".should in the range of 0 to 511.
24-16 STEP_DOWN_THR_1	STEP_DOWN_THR_1 When current AGC_GAIN_IDX=n, if slow_mag > STEP_DOWN_THR_n, AGC will decrease the gain index.
15-9 —	Reserved
8-0 STEP_UP_THR_1	STEP_UP_THR_1 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.84 AGC IDX0 Slow Mode Threshold (AGC_IDX0_THR)

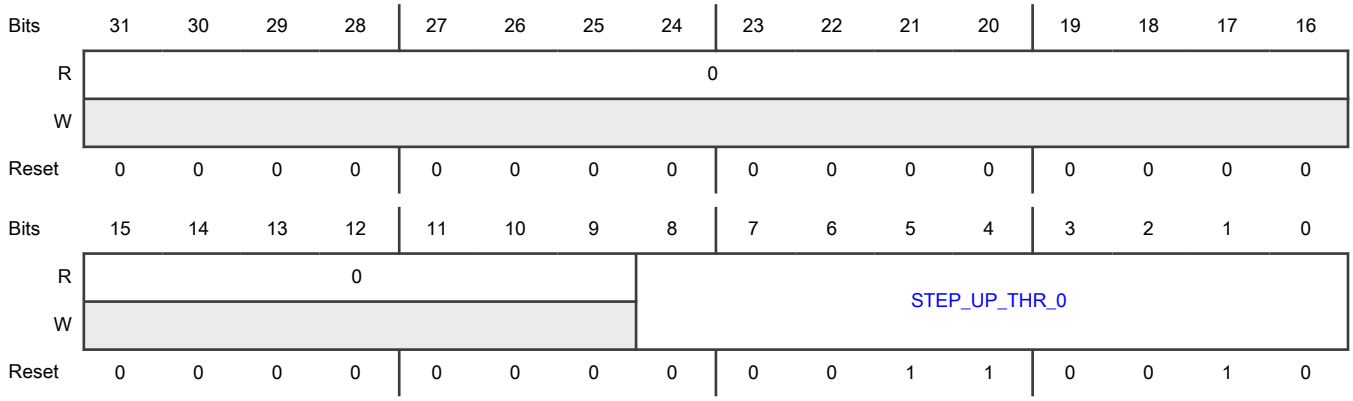
Offset

Register	Offset
AGC_IDX0_THR	148h

Function

AGC_IDXn_THR registers store the threshold values for AGC slow mode. Each AGC gain index has it's own step down and step up threshold. NOTE: There's no STEP_UP_THR for gain index 11 and no STEP_DOWN_THR for gain index 0.

Diagram



Fields

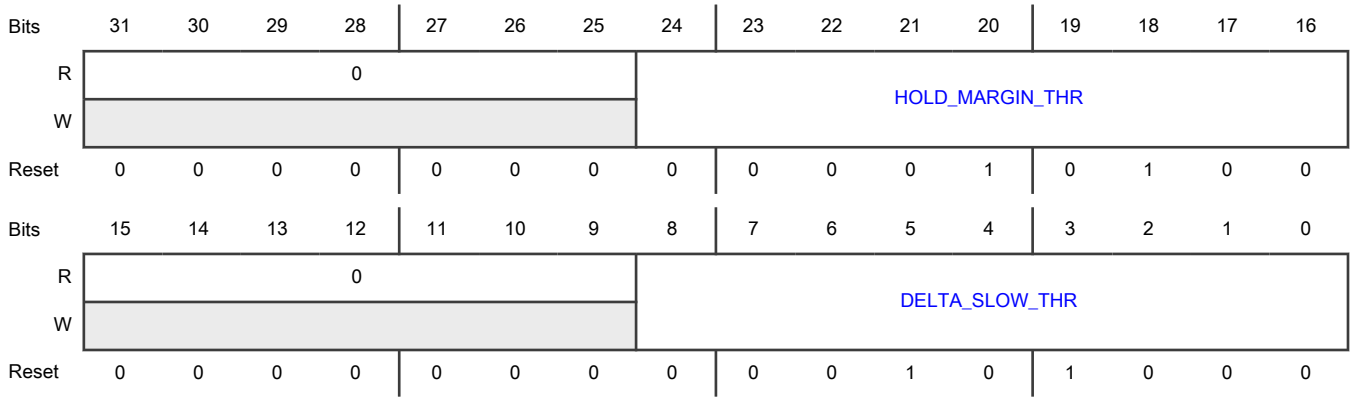
Field	Function
31-9 —	Reserved
8-0 STEP_UP_THR_0	STEP_UP_THR_0 When current AGC_GAIN_IDX=n, if slow_mag < STEP_UP_THR_n, AGC will increase the gain index.

55.4.7.4.3.1.85 AGC Miscellaneous Thresholds (AGC_THR_MIS)

Offset

Register	Offset
AGC_THR_MIS	14Ch

Diagram



Fields

Field	Function
31-25 —	Reserved
24-16 HOLD_MARGIN_THR	STEP_UP_THR_VLG2large If AGC unhold function enabled, AGC will record the last slow magnitude value(call as hold_mag) before enter into HOLD mode and continue measuring the slow magnitude. If there are two consecutive slow_mag values match the equation "abs(slow_mag-hold_mag) > HOLD_MARGIN_THR", AGC will exit HOLD mode.
15-9 —	Reserved
8-0 DELTA_SLOW_THR	STEP_UP_THR_VLG2 The threshold value for delta slow mode. Check the description of AGC_CTRL[AGC_DELTA_SLOW_EN] and AGC_CTRL[AGC_DELTA_SLOW_STEPS] for more information.

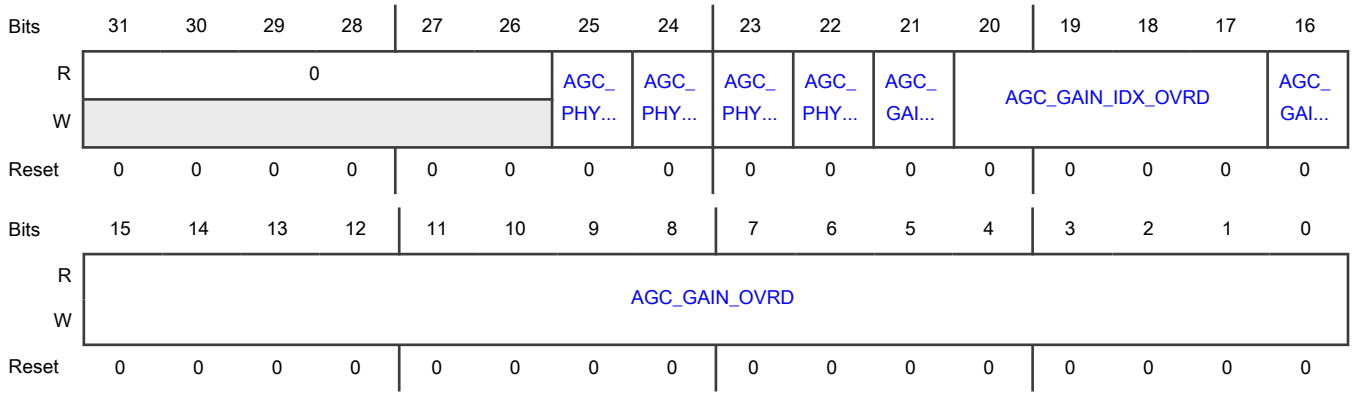
55.4.7.4.3.1.86 AGC Override Control (AGC_OVRD)

Offset

Register	Offset
AGC_OVRD	150h

Function
AGC override function control.

Diagram



Fields

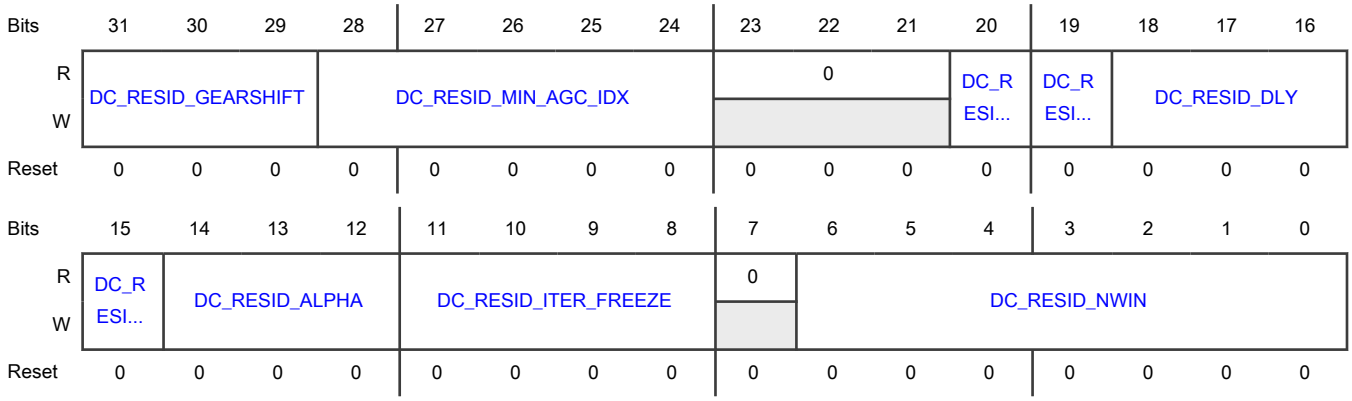
Field	Function
31-26 —	Reserved
25 AGC_PHY_FREEZE_OVRD_EN	PHY_FREEZE_TRIG signal override enable
24 AGC_PHY_FREEZE_OVRD	PHY_FREEZE_TRIG signal override value
23 AGC_PHY_HOLD_OVRD_EN	PHY_HOLD_TRIG signal override enable
22 AGC_PHY_HOLD_OVRD	PHY_HOLD_TRIG signal override value
21 AGC_GAIN_IDX_OVRD_EN	AGC gain index override enable.
20-17 AGC_GAIN_IDX_OVRD	AGC gain index override value.
16 AGC_GAIN_OVRD_EN	AGC gain config override enable
15-0 AGC_GAIN_OVRD	AGC gain config override values. NOTE: AGC Gain Override function has higher priority than AGC Gain Index override function

55.4.7.4.3.1.87 DC Residual Control (DC_RESID_CTRL)

Offset

Register	Offset
DC_RESID_CTRL	154h

Diagram



Fields

Field	Function
31-29 DC_RESID_GEARSHIFT	DC Residual Gearshift If gearshifting is enabled (DC_RESID_GS_EN = 1), the DC residual will use DC_RESID_ALPHA for DC_RESID_GEARSHIFT windows (DC_RESID_NWIN samples) then will use DC_RESID_ALPHA+1 for the remaining windows.
28-24 DC_RESID_MIN_AGC_IDX	DC Residual Minimum AGC Table Index Specifies the minimum AGC table index value at which DC residual can be enabled. E.g., if this is 5'd0, then the DC residual is enabled for all AGC gain table indexes (assuming RX_DC_RESID_EN is set) if this is 5'd20, then tracking is enabled for AGC gain table indexes 20-26 (assuming RX_DC_RESID_EN is set).
23-21 —	Reserved
20 DC_RESID_EXT_DC_EN	DC Residual External DC Enable 0b - External DC disable. The DC Residual activates at a delay specified by DC_RESID_DLY after an AGC gain change pulse. The DC Residual is initialized with a DC offset of 0. 1b - External DC enable. The DC residual activates after the DCOC's tracking hold timer expires. The DC Residual is initialized with the DC estimate from the DCOC tracking estimator.
19 DC_RESID_SECOND_RUN_EN	DC Residual Second Run Enable This bit enables additional DC residual compensation (primarily intended for Long Range) after the normal DC residual operation has completed. 0b - Second Run disabled 1b - Second Run enabled
18-16 DC_RESID_DLY	DC Residual Delay The delay from activation of the DC residual block to the time when it starts processing. For a 32MHz reference clock, the delay in microseconds matches the value programmed. For other clock frequencies,

Table continues on the next page...

Table continued from the previous page...

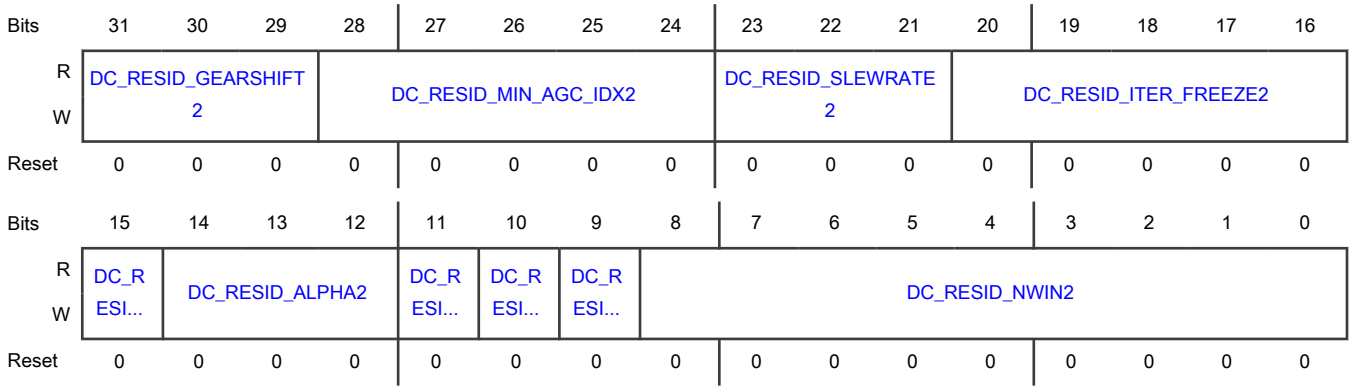
Field	Function
	the delay is scaled based on the clock period. Supported values: 0-7. This delay is only used when EXT_DC_EN=0.
15 DC_RESID_GS_EN	DC Residual Gearshift Enable Controls whether or not gearshifting is enabled. See description of DC_RESID_GEARSHIFT for more information on gearshifting. 0b - Gearshifting disabled 1b - Gearshifting enabled
14-12 DC_RESID_ALPHA	DC Residual Alpha The Alpha parameter controls the rate at which the DC estimate is updated. 000b - Update factor is 1 001b - Update factor is 1/2 010b - Update factor is 1/4 + 1/8 011b - Update factor is 1/4 100b - Update factor is 1/8 + 16 101b - Update factor is 1/8 110b - Update factor is 1/16 + 1/32 111b - Update factor is 1/16
11-8 DC_RESID_ITER_FREEZE	DC Residual Iteration Freeze Number of windows of DC_RESID_NWIN samples before the DC is frozen. Supported values: 1-8
7 —	Reserved
6-0 DC_RESID_NWIN	DC Residual NWIN Number of samples in a window.

55.4.7.4.3.1.88 DC Residual Control2 (DC_RESID_CTRL2)

Offset

Register	Offset
DC_RESID_CTRL2	158h

Diagram



Fields

Field	Function
31-29 DC_RESID_GEARSHIFT2	DC Residual Gearshift, for Second Run If gearshifting is enabled (DC_RESID_GS2_EN = 1) for Second Run, the DC residual will use DC_RESID_ALPHA2 for DC_RESID_GEARSHIFT2 windows (DC_RESID_NWIN2 samples) then will use DC_RESID_ALPHA2+1 for the remaining windows.
28-24 DC_RESID_MIN_AGC_IDX2	DC Residual Minimum AGC Table Index, for Second Run Specifies the minimum AGC table index value at which DC residual can be enabled while Second Run is active. E.g., if this is 5'd0, then the DC residual is enabled for all AGC gain table indexes (assuming RX_DC_RESID_EN is set) if this is 5'd20, then tracking is enabled for AGC gain table indexes 20-26 (assuming RX_DC_RESID_EN is set).
23-21 DC_RESID_SLEWRATE2	DC Residual Slewrate, for Second Run If Slewrate control is enabled (DC_RESID_SR2_EN = 1), changes in the DC correction will be limited to this maximum absolute value, while Second Run is active.
20-16 DC_RESID_ITER_FREEZE2	DC Residual Iteration Freeze, for Second Run Number of windows of DC_RESID_NWIN samples before the DC is frozen. This is used only when Second Run is active.
15 DC_RESID_GS2_EN	DC Residual Gearshift Enable, for Second Run Controls whether or not gearshifting is enabled for Second Run. See description of DC_RESID_GEARSHIFT2 for more information on gearshifting. 0b - Gearshifting disabled for Second Run 1b - Gearshifting enabled for Second Run
14-12 DC_RESID_ALPHA2	DC Residual Alpha, for Second Run The Alpha parameter controls the rate at which the DC estimate is updated. The update factor is shown below. This is used only when the Second Run is active. 000b - Update factor is 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Update factor is 1/2 010b - Update factor is 1/4 + 1/8 011b - Update factor is 1/4 100b - Update factor is 1/8 + 16 101b - Update factor is 1/8 110b - Update factor is 1/16 + 1/32 111b - Update factor is 1/16
11 DC_RESID_SR2_EN	DC Residual Slewrate Enable, for Second Run When DC Residual is performing Second Run, if this bit is set, then changes in the DC correction will be limited by DC_RESID_SLEWRATE after Gearshifting.
10 DC_RESID_CC_EN	DC Residual Continuous Correction Enable When DC Residual is performing Second Run, if this bit is set, then the DC_RESID_ITER_FREEZE2 will be ignored and the DC Residual will perform continuous corrections.
9 DC_RESID_PHY_STOP_EN	DC Residual PHY Stop Enable When DC Residual is performing Second Run, if this bit is set, then when the PHY's aa found uncoded signal is asserted, further DC correction updates will be stopped.
8-0 DC_RESID_NWIN2	DC Residual NWIN, for Second Run Number of samples in a window. This is used only when DC_RESID_SECOND_RUN_EN=1.

55.4.7.4.3.1.89 DC Residual Control DataRate1 (DC_RESID_CTRL_DRS)

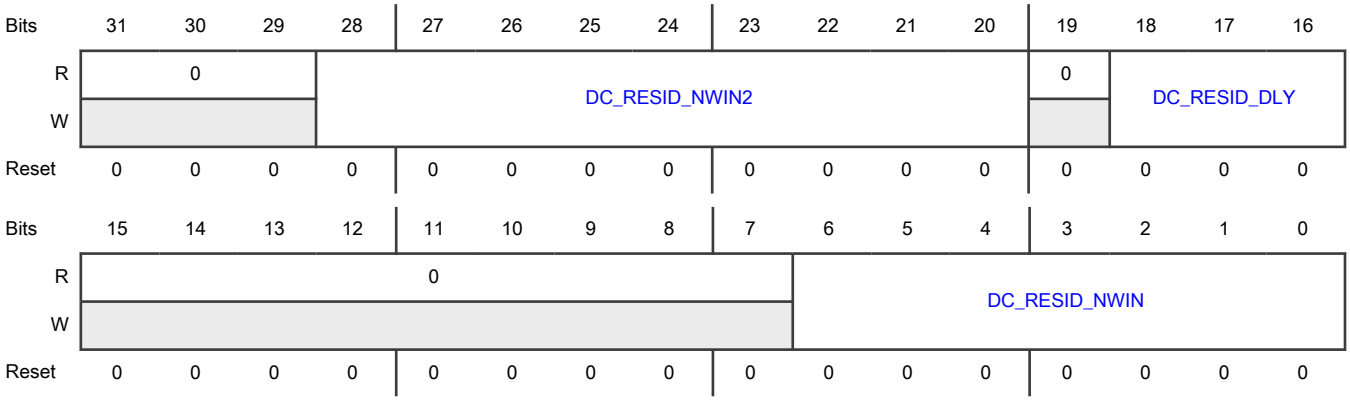
Offset

Register	Offset
DC_RESID_CTRL_DRS	15Ch

Function

The bits in the _DRS registers are only used when the Bluetooth LE signals to request a 2Mbps data rate, or the Generic LL signals using datarate_config_sel to request an alternate data rate.

Diagram



Fields

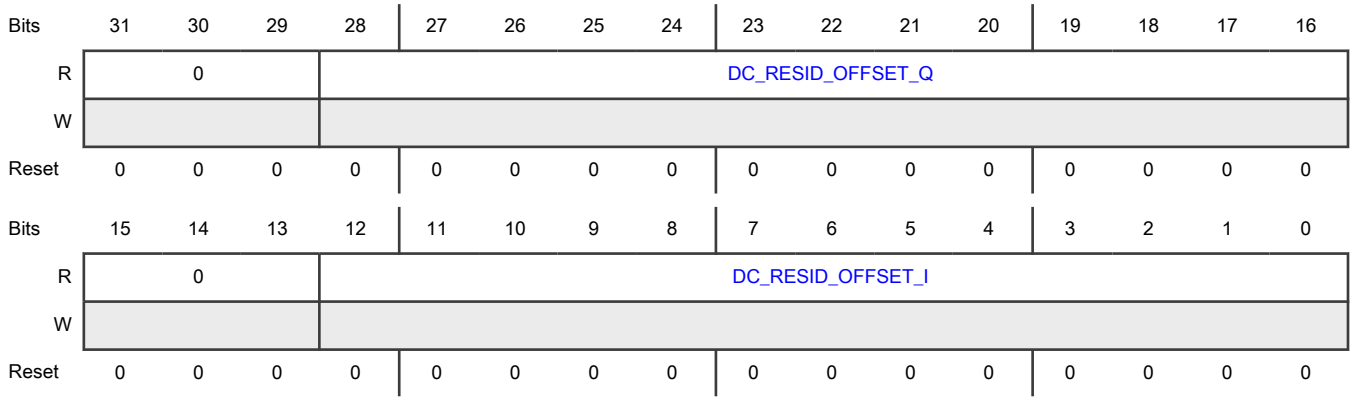
Field	Function
31-29 —	Reserved
28-20 DC_RESID_NWIN2	DC Residual NWIN, for Second Run Number of samples in a window. This is used only when DC_RESID_SECOND_RUN_EN=1.
19 —	Reserved
18-16 DC_RESID_DLY	DC Residual Delay The delay from activation of the DC residual block to the time when it starts processing. For a 32MHz reference clock, the delay in microseconds matches the value programmed. For other clock frequencies, the delay is scaled based on the clock period. Supported values: 0-7. This delay is only used when EXT_DC_EN=0.
15-7 —	Reserved
6-0 DC_RESID_NWIN	DC Residual NWIN Number of samples in a window.

55.4.7.4.3.1.90 DC Residual Estimate (DC_RESID_EST)

Offset

Register	Offset
DC_RESID_EST	160h

Diagram



Fields

Field	Function
31-29 —	Reserved
28-16 DC_RESID_OFFSET_Q	DC Residual Offset Q Reflects the current DC residual offset estimate for Q channel. Format is s11.1. This is provided for debug and characterization purposes only.
15-13 —	Reserved
12-0 DC_RESID_OFFSET_I	DC Residual Offset I Reflects the current DC residual offset estimate for I channel. Format is s11.1. This is provided for debug and characterization purposes only.

55.4.7.4.3.1.91 DfT tone analyzer (DFT_TONE_ANALYZER0)

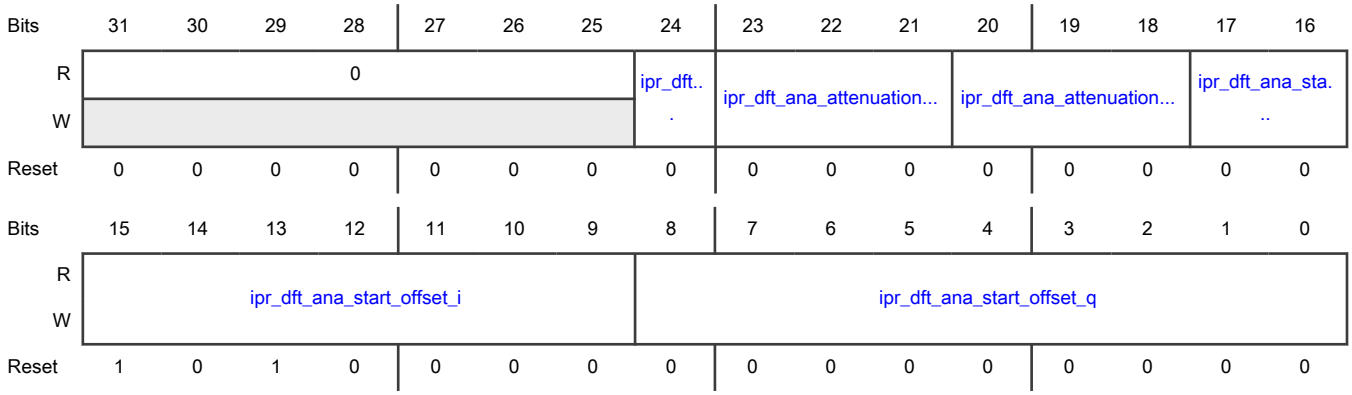
Offset

Register	Offset
DFT_TONE_ANALYZER0	164h

Function

1 of 4 registers required

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ipr_dft_ana_en	Enable for DfT tone analyzer Enable DFT Tone Analyzer module
23-21 ipr_dft_ana_attenuation_i	Check description of ipr_dft_ana_attenuation_q
20-18 ipr_dft_ana_attenuation_q	Tone Attenuation For Q Path Full amplitude of tone generator inside the DFT Tone Analyzer is 1024. By set ipr_dft_ana_attenuation_i and/or ipr_dft_ana_attenuation_q can scale down the amplitude for I and/or Q path. The actual scale factor is calculated as: i_scale = 2^(-ipr_dft_ana_attenuation_i), q_scale = 2^(-ipr_dft_ana_attenuation_q),
17-9 ipr_dft_ana_start_offset_i	I Initial Phase Check description of ipr_dft_ana_start_offset_q
8-0 ipr_dft_ana_start_offset_q	Q Initial Phase ipr_dft_ana_start_offset_i and ipr_dft_ana_start_offset_q are using for control the initial phase of I and Q of tone generator inside the DFT tone analyzer. By setting different phase of I and Q, the tone generator can produce positive freq or negtive freq waveform. Default freq of tone generator is positive. To generage a negtive frequence, set ipr_dft_ana_start_offset_i = 0x50 and ipr_dft_ana_start_offset_q = 0xA0.

55.4.7.4.3.1.92 DfT tone analyzer (DFT_TONE_ANALYZER1)

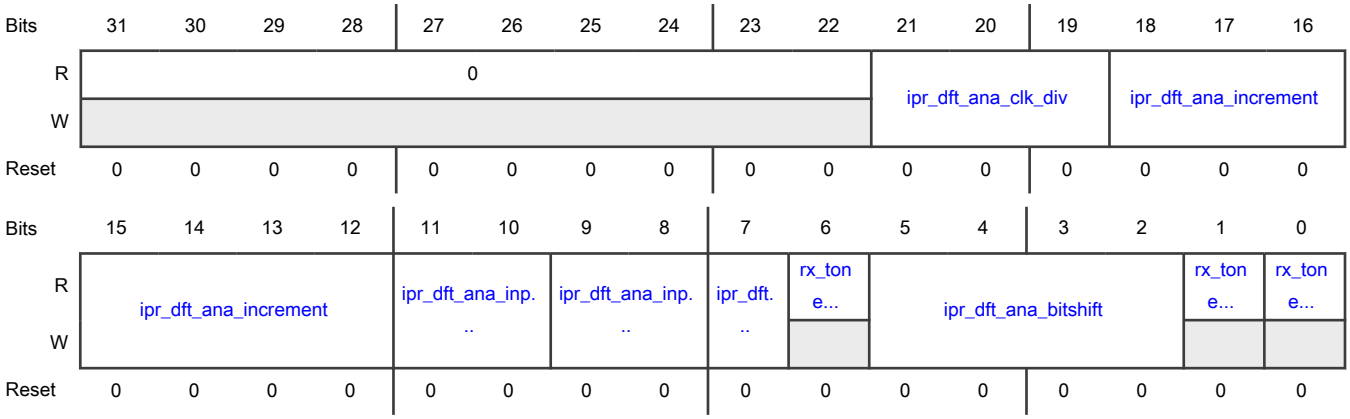
Offset

Register	Offset
DFT_TONE_ANALYZER1	168h

Function

1 of 4 registers required

Diagram



Fields

Field	Function
31-22 —	Reserved
21-19 ipr_dft_ana_clk_div	Clock divider factor for tone generator. Check ipr_dft_ana_increment for more information. 000b - ref_clk 001b - ref_clk div 2 010b - ref_clk div 4 011b - ref_clk div 8 100b - ref_clk div 16
18-12 ipr_dft_ana_increment	ipr_dft_ana_increment and ipr_dft_ana_clk_div determines the tone frequence. Tone_Freq = (ref_clk_freq * ipr_dft_ana_increment) / (320 * 2^(ipr_dft_ana_clk_div))
11-10	Should always set 0 for this field

Table continues on the next page...

Table continued from the previous page...

Field	Function
ipr_dft_ana_input_sel_1	
9-8 ipr_dft_ana_input_sel_2	Should always set 2 for this field
7 ipr_dft_ana_start	Set 1 to trig the DFT Tone Analyzer. NOTE: Before set this bit, make sure the ipr_dft_ana_en asserted.
6 rx_tone_ana_done	Read only status bit. Indicating the tone analyze finished.
5-2 ipr_dft_ana_bitshift	Accumulaotr input valude scale down factor.
1 rx_tone_ana_bitshift_ovf	Indicate the accumulator input has overflow during measurement.
0 rx_tone_ana_accu_ovf	Indicate the accumulator has overflow during measurement.

55.4.7.4.3.1.93 DfT tone analyzer (DFT_TONE_ANALYZER2)

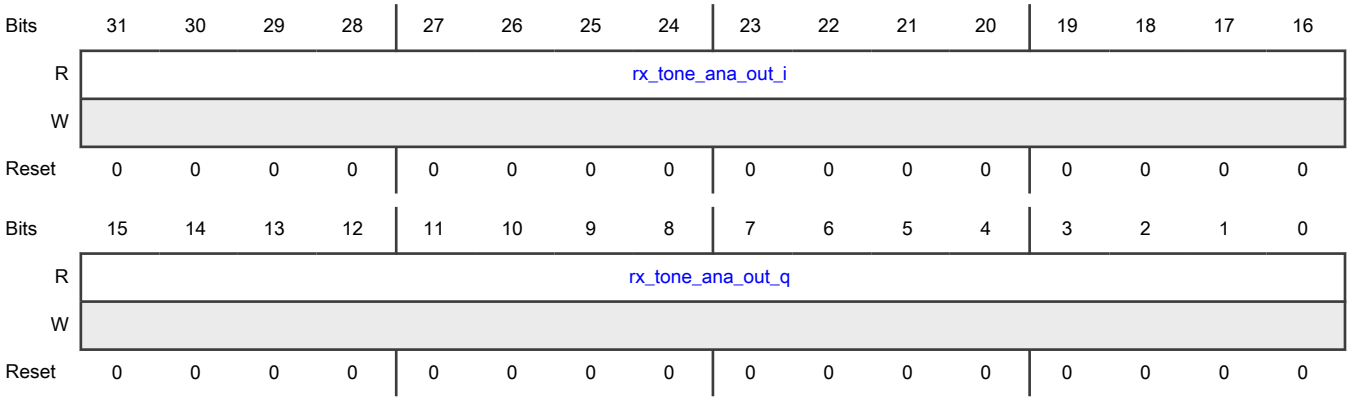
Offset

Register	Offset
DFT_TONE_ANALYZER2	16Ch

Function

1 of 4 registers required

Diagram



Fields

Field	Function
31-16 rx_tone_ana_out_i	Accumulator I-path result
15-0 rx_tone_ana_out_q	Accumulator Q-path result

55.4.7.4.3.1.94 DfT tone analyzer (DFT_TONE_ANALYZER3)

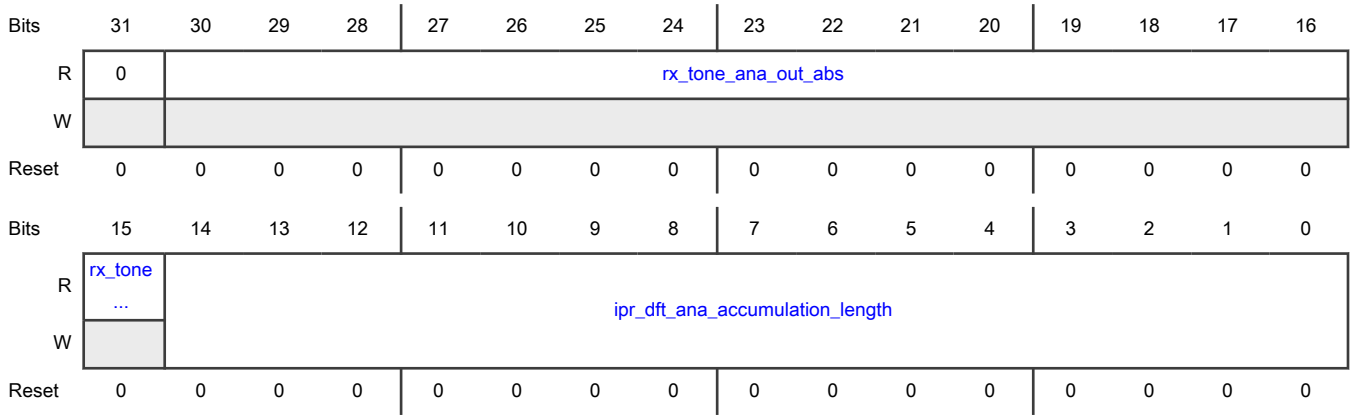
Offset

Register	Offset
DFT_TONE_ANALYZER3	170h

Function

1 of 4 registers required

Diagram



Fields

Field	Function
31 —	Reserved
30-15 rx_tone_ana_out_abs	Tone Analyzer Result
14-0 ipr_dft_ana_accumulation_length	Accumulation length

55.4.7.4.3.1.95 DCOC Digital Correction Result (DCOC_DIG_CORR_RESULT)

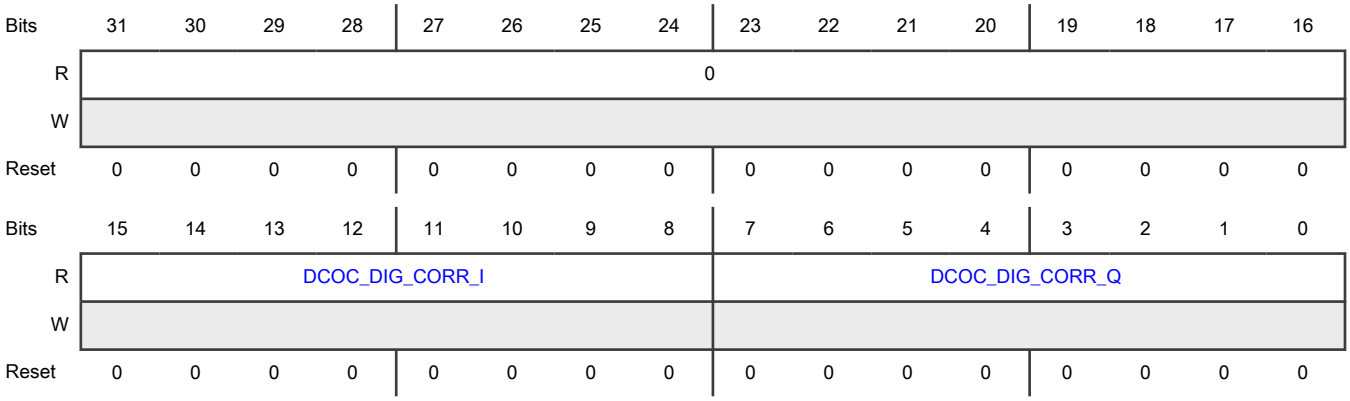
Offset

Register	Offset
DCOC_DIG_CORR_RESULT	174h

Function

This read-only register stores the residual DC value after DCOC ADC and CBPF DAC calibration done. This register updates at the end of each non-fast RX warmup.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DCOC_DIG_CORR_I	DCOC Q-Channel Residual After Calibration
7-0 DCOC_DIG_CORR_Q	DCOC I-Channel Residual After Calibration

55.4.7.5 Zigbee PHY

55.4.7.5.1 Introduction

This block guide has the details for the Zigbee Demodulator function feature registers, which include multi CTRL registers.

55.4.7.5.2 Memory Map and Register Definition

The Zigbee Demodulator module memory map and detailed descriptions of all its registers are as follows.

55.4.7.5.2.1 XCVR_ZBDEMOD register descriptions

55.4.7.5.2.1.1 XCVR_ZBDEMOD_ADDR memory map

XCVR_ZBDEMOD base address: 48A0_7500h

Offset	Register	Width (In bits)	Access	Reset value
0h	802.15.4 DEMOD CORRELATOR CONTROL (CORR_CTRL)	32	RW	0000_0482h
4h	802.15.4 DEMOD PN TYPE (PN_TYPE)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8h	802.15.4 DEMOD PN CODE (PN_CODE)	32	RW	744A_C39Bh
Ch	802.15.4 DEMOD SYMBOL SYNC CONTROL (SYNC_CTRL)	32	RW	0000_0008h
10h	802.15.4 CCA/LQI SOURCE (CCA_LQI_SRC)	32	RW	0000_000Ch
14h	FAD CORRELATOR THRESHOLD (FAD_LPPS_THR)	32	RW	6060_6082h
18h	802.15.4 AFC STATUS (ZBDEM_AFC)	32	RW	0000_0001h
1Ch	CCA MODE 2 CONTROL REGISTER (CCA2_CTRL)	32	RW	0000_FF05h
20h	CCA MODE 2 CONTROL REGISTER (CCA2_THRESH)	32	RW	0040_0080h
24h	CCA MODE 2 STATUS REGISTER (CCA2_STATUS)	32	R	0000_0000h
28h	802.15.4 DEMOD CORRELATOR CONTROL2 (CORR_CTRL2)	32	RW	0000_00FFh

55.4.7.5.2.1.2 802.15.4 DEMOD CORRELATOR CONTROL (CORR_CTRL)

Offset

Register	Offset
CORR_CTRL	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RX_MAX_PREAMBLE								RX_MAX_CORR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ZBDEM_C...	0			MAX_COR...	CORR_NVAL			CORR_VT							
W																
Reset	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0

Fields

Field	Function
31-24 RX_MAX_PREAMBLE	RX_MAX_PREAMBLE Max correlator during preamble-- max correlator value found during the preamble.

Table continues on the next page...

Table continued from the previous page...

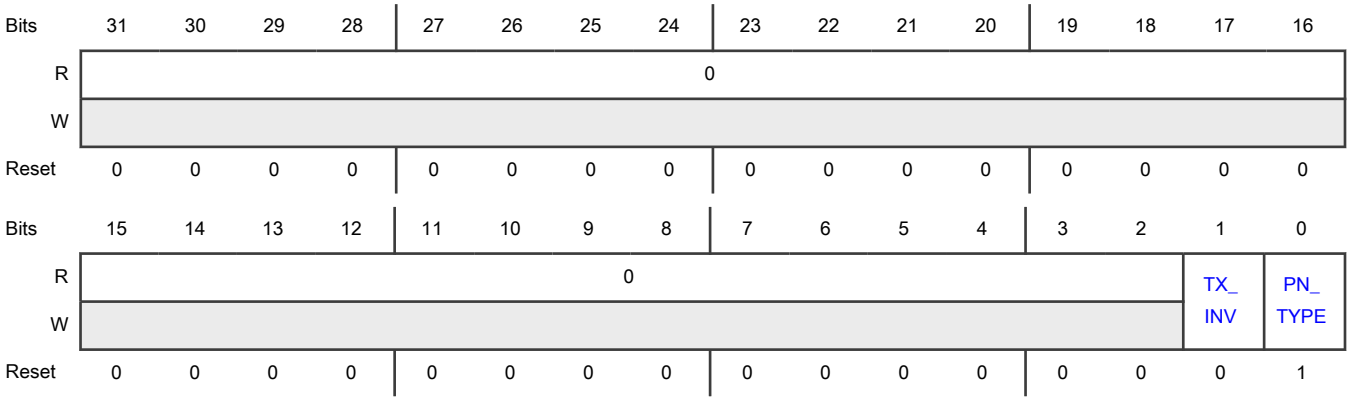
Field	Function
23-16 RX_MAX_CORR	RX_MAX_CORR Max correlator after preamble-- max correlator value found in packet after the preamble (refreshed every symbol rate if MAX_CORR_EN=1).
15 ZBDEM_CLK_ON	Force 802.15.4 Demodulator Clock On 0b - Normal Operation 1b - Force 802.15.4 Demodulator Clock On (debug purposes only)
14-12 —	Reserved
11 MAX_CORR_EN	MAX_CORR_EN Max correlator after preamble enable-- Enable the refresh of the max corr register.
10-8 CORR_NVAL	CORR_NVAL Number of consecutively detected zero-symbols required to declare a preamble detected.
7-0 CORR_VT	CORR_VT Correlator threshold, defines the sensitivity of demod during the preamble search state.

55.4.7.5.2.1.3 802.15.4 DEMOD PN TYPE (PN_TYPE)

Offset

Register	Offset
PN_TYPE	4h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 TX_INV	TX_INV test mode to invert the transmission
0 PN_TYPE	PN_TYPE PN Type - Pseudo Noise Chip Code Type (for 802.15.4 = 1)

55.4.7.5.2.1.4 802.15.4 DEMOD PN CODE (PN_CODE)

Offset

Register	Offset
PN_CODE	8h

Function
Pseudo Noise Chip Code Seed Value

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PN_MSB															
Reset	0	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PN_LSB															
Reset	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1

Fields

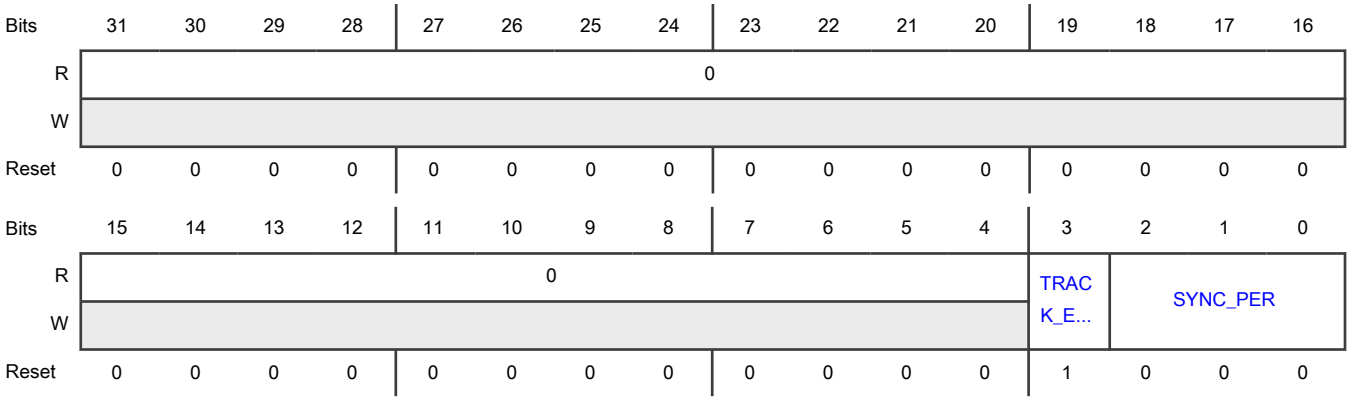
Field	Function
31-16 PN_MSB	PN_MSB PN_CODE MS half
15-0 PN_LSB	PN_LSB PN_CODE LS half

55.4.7.5.2.1.5 802.15.4 DEMOD SYMBOL SYNC CONTROL (SYNC_CTRL)

Offset

Register	Offset
SYNC_CTRL	Ch

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TRACK_ENABLE	TRACK_ENABLE 0b - symbol timing synchronization tracking disabled in Rx frontend 1b - symbol timing synchronization tracking enabled in Rx frontend (default)
2-0 SYNC_PER	Symbol Sync Tracking Period determines update rate for symbol timing, per equation. An early/late measurement is made every 2^SYNC_PER[2:0] symbols. Valid range of SYNC_PER[2:0] is 0 to 4.

55.4.7.5.2.1.6 802.15.4 CCA/LQI SOURCE (CCA_LQI_SRC)

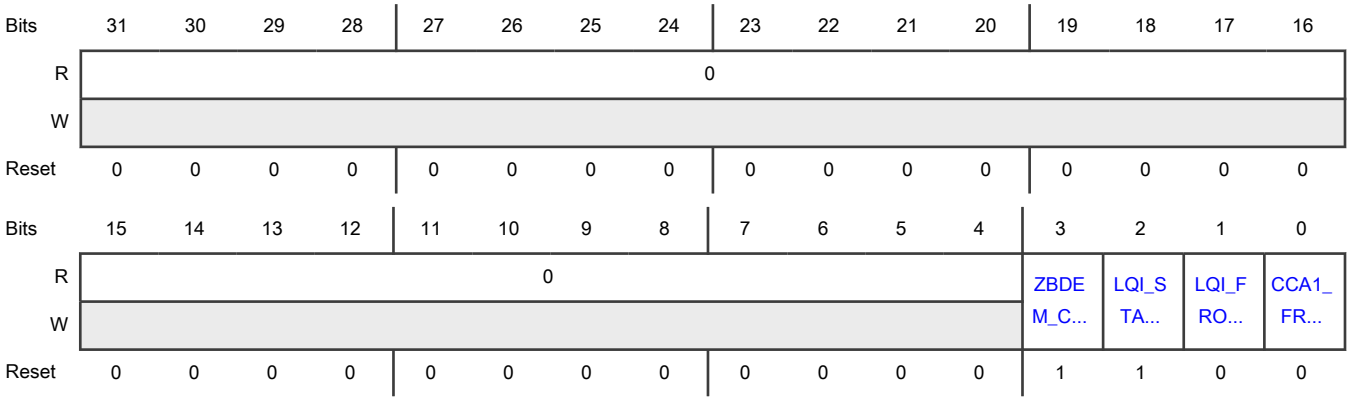
Offset

Register	Offset
CCA_LQI_SRC	10h

Function

Selects the Source of CCA and LQI Information Provided to the 802.15.4 Link Layer

Diagram



Fields

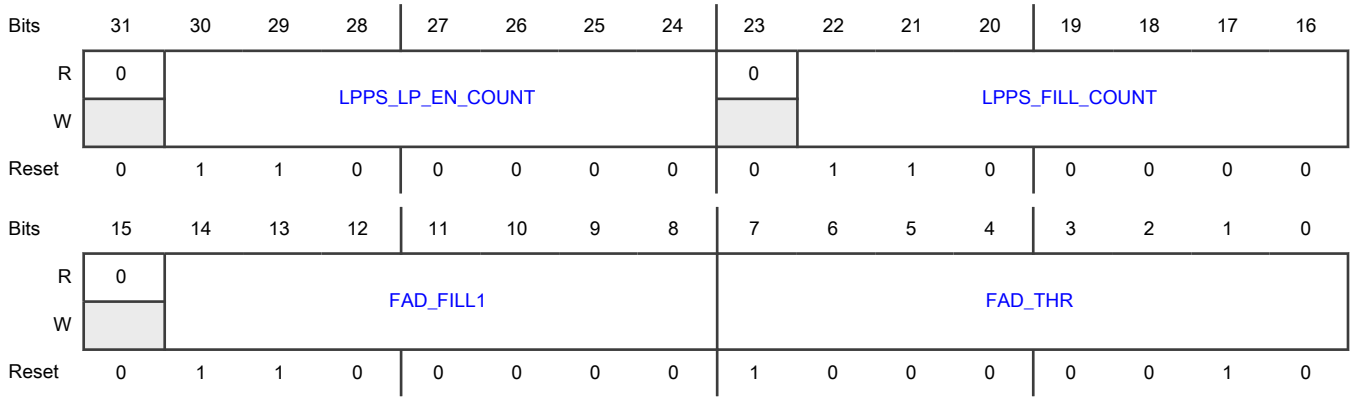
Field	Function
31-4 —	Reserved
3 ZBDEM_CCA_CLK_ON	802.15.4 Demodulator CCA Clock Enable 802.15.4 Demodulator CCA Clock Enable. This bit should be left in its default state (1) if any CCA Mode (Modes 1, 2, and 3), Energy Detect, or LQI uses the 802.15.4 Demodulator to compute its value. If 802.15.4 Demodulator is not used to compute any of these, this bit can be cleared to save power.
2 LQI_START_AT_SFD	Select Start Point for LQI Computation 0b - Start LQI computation at Preamble Detection (similar to previous NXP 802.15.4 products) 1b - Start LQI computation at SFD (Start of Frame Delimiter) Detection
1 LQI_FROM_RX_DIG	Selects the Source of LQI (Link Quality Indicator) Information Provided to the 802.15.4 Link Layer 0b - Use the LQI information computed internally in the 802.15.4 Demod 1b - Use the LQI information computed by the RX Digital
0 CCA1_FROM_RX_DIG	Selects the Source of CCA1 (Clear Channel Assessment Mode 1) Information Provided to the 802.15.4 Link Layer 0b - Use the CCA1 information computed internally in the 802.15.4 Demod 1b - Use the CCA1 information computed by the RX Digital

55.4.7.5.2.1.7 FAD CORRELATOR THRESHOLD (FAD_LPPS_THR)

Offset

Register	Offset
FAD_LPPS_THR	14h

Diagram



Fields

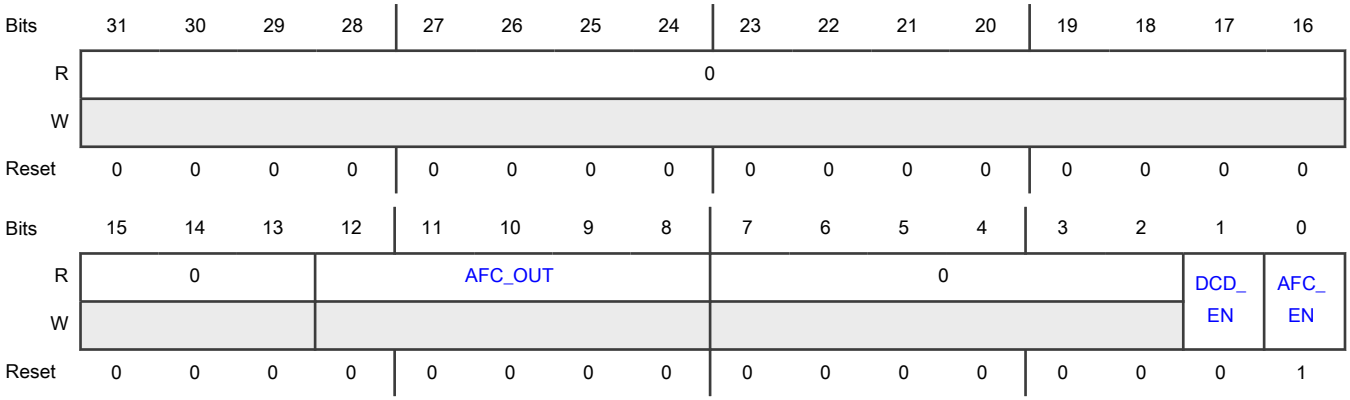
Field	Function
31 —	Reserved
30-24 LPPS_LP_EN_COUNT	LPPS_LP_EN high time Parameter used to configure the duration of time lpps_lp_enable is in high state
23 —	Reserved
22-16 LPPS_FILL_COUNT	Wait duration after lpps_lp_enable is de-asserted Parameter used to configure the wait duration used for valid samples acquisition after *lpps_lp_enable* is de-asserted. The overall wait duration expressed in base-band samples is determined using formula: *lpps_fill_duration=lpps_fill_count+lpps_tsm_warmup_duration* where *lpps_tsm_warmup_duration* is the duration of the fast TSM warmup, used for the LPPS mode, expressed in samples.
15 —	Reserved
14-8 FAD_FILL1	Pre-detection buffer filling duration Interval used to wait for valid samples acquisition after an antenna switch command has been issued.
7-0 FAD_THR	FAD_THR Correlator threshold at which the FAD will select the antenna.

55.4.7.5.2.1.8 802.15.4 AFC STATUS (ZBDEM_AFC)

Offset

Register	Offset
ZBDEM_AFC	18h

Diagram



Fields

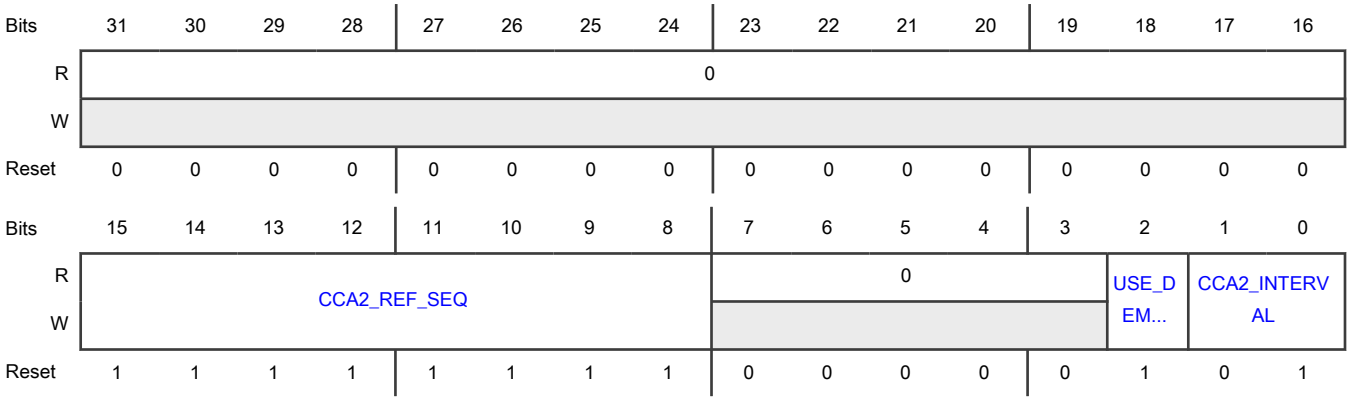
Field	Function
31-13 —	Reserved
12-8 AFC_OUT	AFC_OUT AFC Result from the last received packet. Format is Signed, Two's Complement. Each LSB represents 15KHz of Frequency Offset
7-2 —	Reserved
1 DCD_EN	DCD_EN 0b - NCD Mode (default) 1b - DCD Mode
0 AFC_EN	AFC_EN the AFC Function 0b - AFC is disabled 1b - AFC is enabled

55.4.7.5.2.1.9 CCA MODE 2 CONTROL REGISTER (CCA2_CTRL)

Offset

Register	Offset
CCA2_CTRL	1Ch

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 CCA2_REF_SEQ	CCA Mode 2 Sequence Address Selection of the sequence addresses to be used.
7-3 —	Reserved
2 USE_DEMOD_CCA2	Selects CCA Mode 2 Computation Engine Selects between demodulator-based or standalone CCA Mode 2 Computation Engine 0b - Use standalone (new) CCA Mode 2 Engine, decoupled from demodulator 1b - Use 802.15.4 demodulator-based (legacy) CCA Mode 2 Engine (default)
1-0 CCA2_INTERVAL	CCA Mode 2 Measurement Window Duration Specifies the CCA2 measurement window duration: 00b - 64 µs 01b - 128 µs 10b - 256 µs 11b - 512 µs

55.4.7.5.2.1.10 CCA MODE 2 CONTROL REGISTER (CCA2_THRESH)

Offset

Register	Offset
CCA2_THRESH	20h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								CCA2_SYM_THRESH							
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CCA2_CNT_THRESH							
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-16 CCA2_SYM_THRESH	CCA Mode 2 Symbol Threshold Threshold used to detect repetition sequence addresses corresponding to correlation products that exceed CCA2_CORR_THRESH[7:0]. **NOTE** CCA2_CORR_THRESH[7:0] resides in 802.15.4 address space
15-10 —	Reserved
9-0 CCA2_CNT_THRESH	CCA Mode 2 Count Threshold Threshold to compare count of correlation products exceeding CCA2_CORR_THRESH[7:0]. **NOTE** CCA2_CORR_THRESH[7:0] resides in 802.15.4 address space

55.4.7.5.2.1.11 CCA MODE 2 STATUS REGISTER (CCA2_STATUS)

Offset

Register	Offset
CCA2_STATUS	24h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						CCA2_CNT_SYM									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CCA2_CH...	CCA2_CO...	CCA2_CNT_MAX									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-16 CCA2_CNT_SYM	CCA Mode 2 Repetition Sequence Addresses Count Actual count of repetition sequence addresses corresponding to correlation products that exceed CCA2_CORR_THRESH[7:0]. **NOTE** CCA2_CORR_THRESH[7:0] resides in 802.15.4 address space
15-12 —	Reserved
11 CCA2_CHANNEL_STATE	CCA Mode 2 Channel State Indicates whether the channel is idle or busy; 0=idle, 1=busy. Asserted in any of the following cases: 1. CCA mode 2 measurement finds the channel busy when CCA2 duration expires 2. *rx_preamble_det* signal is asserted (note that, following this event, if *rx_preamble_det* is deasserted and *zb_cca2_complete* is low, *zb_cca2_channel_state* is deasserted also) 3. *rx_sfd_detect* signal is asserted
10	CCA Mode 2 Measurement Complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
CCA2_COMPLETE	Indicates when a CCA mode 2 measurement is complete. Asserted in any of the following cases: 1. CCA mode 2 measurement is completed (when CCA2 duration has expired) 2. rx_sfd_detect signal is asserted
9-0 CCA2_CNT_MAX	CCA Mode 2 Maximum Count Actual count of correlation products exceeding CCA2_CORR_THRESH[7:0] during the measurement interval. **NOTE** CCA2_CORR_THRESH[7:0] resides in 802.15.4 address space

55.4.7.5.2.1.12 802.15.4 DEMOD CORRELATOR CONTROL2 (CORR_CTRL2)

Offset

Register	Offset
CORR_CTRL2	28h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								EARLY_PD_THRESH							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Fields

Field	Function
31-8 —	Reserved
7-0 EARLY_PD_THRESH	EARLY_PD_THRESH Early preamble detection threshold, defines the sensitivity of demod during the early preamble search state.

55.4.7.6 2.4GHz PHY

55.4.7.6.1 About this module

55.4.7.6.1.1 Introduction

This documents describes the PHY and its operation.

modules of the 2.4GHz PHY receiver supporting the demodulation of a wide range of use cases employing FSK modulation:

- acquisition (timing and frequency synchronization)
- symbol demodulation
- interfacing with the PrePHY, RBME and LL

The Following protocols are supported:

- Non-coded Bluetooth LE
- Long range Bluetooth LE
- custom FSK modulation schemes
- MSK

55.4.7.6.1.2 Features

The 2.4GHz PHY includes the following features:

- support preamble detect
- support address match
- support Bluetooth LE long range
- support Bluetooth LE none coded up to 4 address
- support demodulate to RBME/LL
- support custom FSK modulation schemes
- support MSK
- high-accuracy CFO estimation using known sequences embedded in the PDU

55.4.7.6.1.3 Modes and Operations

2.4GHz PHY supports the modes and operations described in the indicated sections:

- Operating modes
- Operations

55.4.7.6.1.4 PHY block diagram and integration

Figure 309 shows the interface signals and the integration of the PHY module with other blocks of the radio:

- Front-end module
- Receiver
- Radio bit manipulation engine
- Link-layer
- Antenna selection logic for the estimation of the angle-of-arrival

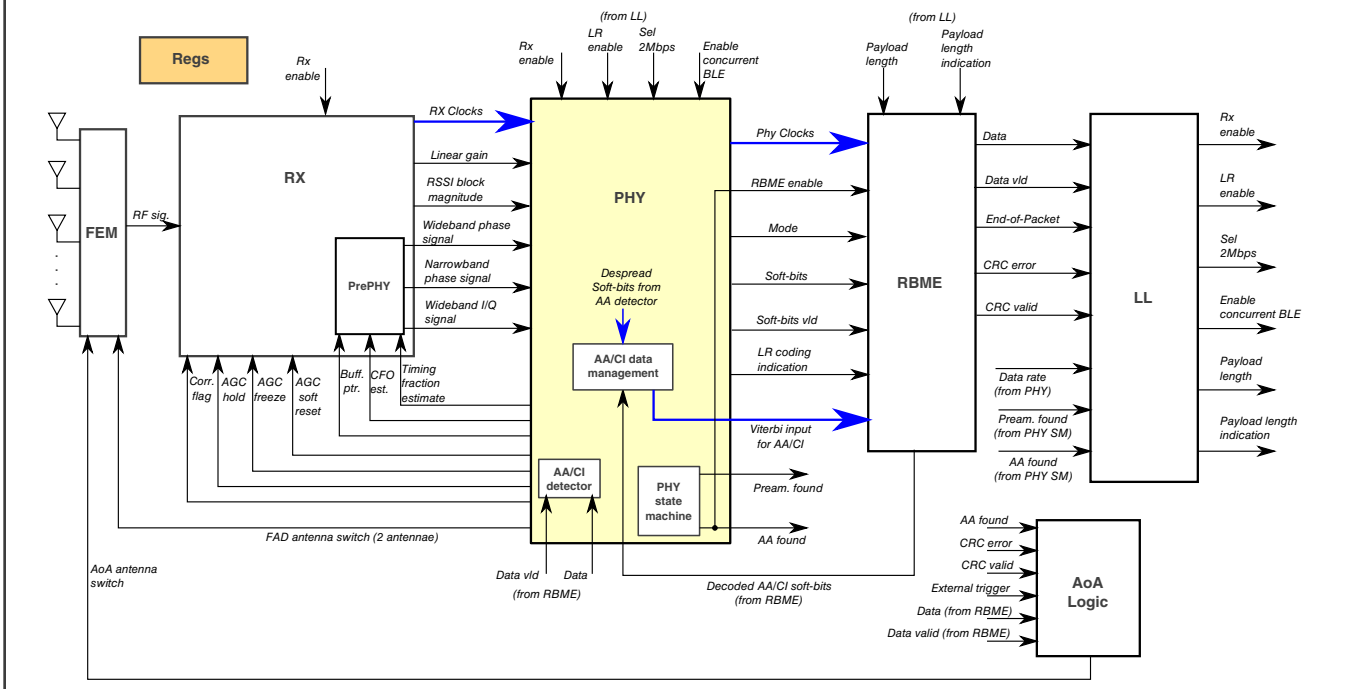


Figure 309. PHY integration block diagram

Figure 310 shows the block diagram of the PHY module. It consists of the following components:

- State-machine and clocks responsible with:
  - generation of control signals
  - generation of clock signals for the PHY processing blocks
- Time/frequency synchronization, responsible with the frame time synchronization and CFO estimation for non-coded packets, that consists of:
  - Power threshold responsible with channel energy sensing
  - Phase domain preamble detection
  - Discriminated phase (FM) domain preamble detection
  - AA detection
- Time/frequency synchronization, responsible with the frame time synchronization and CFO estimation for LR packets, that consists of:
  - LR preamble detection
  - LR AA search
- GFSK demodulator, responsible with the symbol demodulation at the over-the-air baud rate
- AA/CI data parallel to serial conversion, responsible with the data and data valid generation for the RBME module, on the AA/CI portion of the long range Bluetooth LE packets
- AA/CI detection, responsible with the decoded AA matching and CI field extraction for the long range Bluetooth LE packets

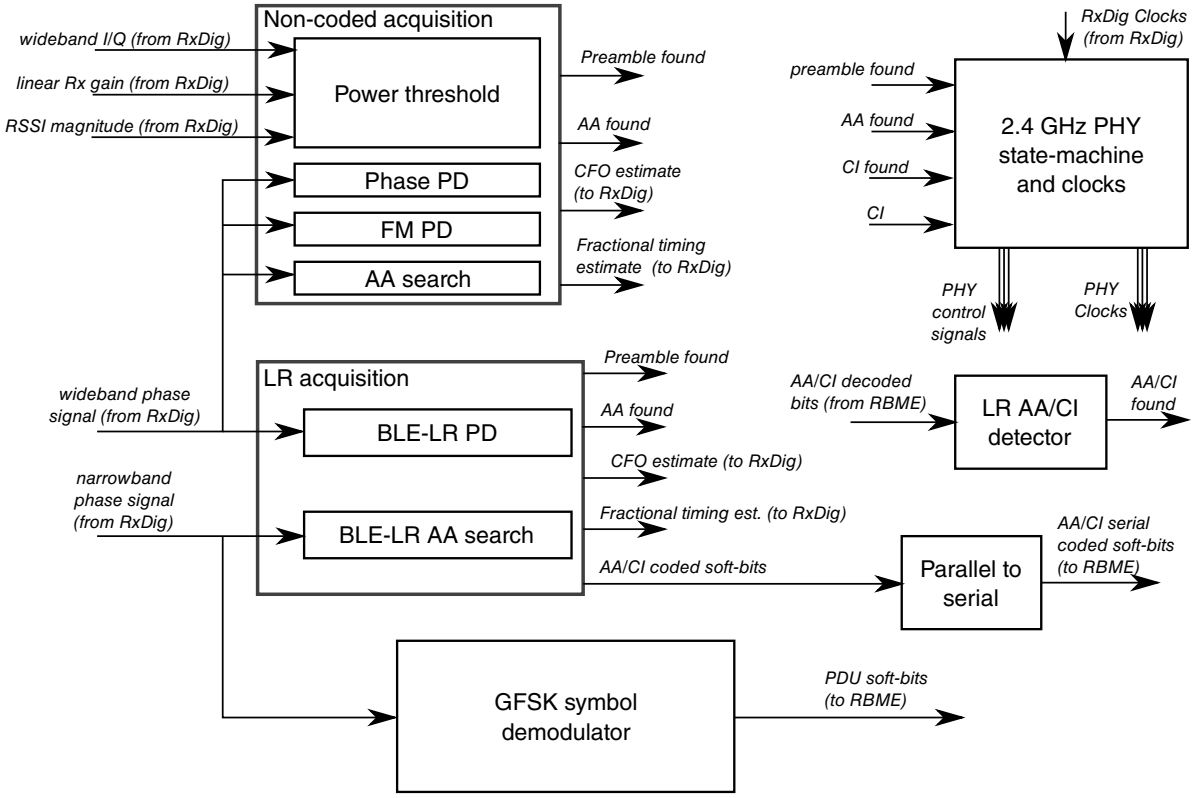


Figure 310. PHY block diagram

A "look-back" buffer provides the required latency of the frequency correction and timing acquisition engines. A pointer is provided from the PHY to the PrePHY via the interface illustrated in Figure 309. The pointer to the IQ buffer has the role of tapping the CFO mixer at the correct sample in the IQ buffer so that the CFO and the fractional timing are properly compensated. The pointer to the buffer is configured by the PHY registers and it is different for different use-cases and configurations. In general the pointer needs to be chosen to account for the following:

- Latency of a detection event, produced by the acquisition block, which is followed by:
  - the enable of the narrowband path
  - the enable of the blocks consuming the output of the narrowband path
- Channel filter pre-filling with enought history samples so that the output is computed using only CFO/fractional timing compensated samples
- The number of history samples has to be equal with at least half of the size of the CHF
- Enough history timing/frequency error compensated samples must be available for the FSK symbol demodulator
- The history needed by the FSK symbol demodulator is 2-symbols worth of samples
- A fine-tune adjustment that accounts for different latencies in the system, intended to ensure a smooth transition when switching from using the wide-band samples to using the narrow-band samples

The size of the IQ buffer needs to be chosen such that it covers for the worst-case scenario. The overall latency of the narrowband path equals the latency of the IQ buffer plus the channel filter latency plus the latencies introduced by the CFO mixer and timing frequency correction interpolator.

55.4.7.6.1.5 Overview

The PHY is made up of several sub-modules. These modules can be viewed as members of two classes: datapath (the flow of the signal and the resultant bits) and control (enable/ disable logic and state control).

55.4.7.6.1.5.1 PHY Datapath

There are two data paths used by the PHY. The wide-band path and the narrow-band path. The PHY uses the one of the two data paths at a time or both of them simultaneously depending on the process(acquisition or payload demodulation) and use-case(coded or non-coded).

The signal of interest is passed through the PHY along the data paths. The overall objective is to process phase samples that are presented and create bits that are sent out. The significant sub-systems are here presented in generally chronological order as the signal passes through.

Initially, the phase samples are input, sequentially, at the sampling rate, one at a time. The first step is to buffer the wide-band samples, so they can be used all at once for certain computations.

The buffer of samples is then presented to the acquisition logic. The acquisition module is responsible for detecting the presence of a signal and then indicating the timing and frequency offset when a signal is detected.

Up to 4 AA's can be searched for in the case of non-coded packets and one AA can be searched for in the case of long range Bluetooth LE packets. When an match occurs, the bit timing (the sample offset at which each bit is optimally demodulated) is recorded.The symbol timing information thus determined is used by the symbol demodulator block to perform payload demodulation.

When a packet is detected, the CFO estimate is presented to the corrector and the correction is applied to the incoming samples. In this fashion, the samples leaving the corrector, which is placed on the narrowband data path, will be optimally corrected for any frequency offset before beginning demodulation.

Demodulation is performed on frequency corrected samples, provided by the narrowband data path, and after a packet is detected, the demodulator produces soft-bits information, at the symbol rate, which is presented to the RBME block. More details are provided in the functional description section.

55.4.7.6.1.5.2 PHY Control

The control of the PHY blocks is realized from a single place denoted the PHY state-machine. the purpose of this block is to provide:

1. triggers(data valid signals)
2. enable/disable signals
3. indications to the processing blocks of the PHY module

Another aspect of the PHY stat machine is to fast antenna diversity(FAD) preamble search feature. FAD is a mechanism that allows antenna selection on fly.

A more detailed description of the PHY control is provided in the functional description section.

55.4.7.6.2 Interface

The PHY interacts with the system in a few distinct ways. The interfaces can be grouped into 5 categories: configuration, signal input, demodulation output, miscellaneous and debug.

55.4.7.6.2.1 PHY Configuration Interface

The PHY is configured by static signals that are observed and used throughout the receive process. It is important that these signals (typically originating from IP registers) remain static during a receive operation to avoid unpredictable behavior. Some examples of these types of configurations are PD thresholds, AA to search for, etc.

55.4.7.6.2.2 Signal Input Interface

The PHY processes the inputs signal from a set of signed, 6-bit phase inputs. There are three class of input data, wide band, narrow band, and raw IQ.

55.4.7.6.2.3 Demodulation Output

The demodulator produces soft bits in fixed-point format which are presented to the RBME block.

55.4.7.6.2.4 Miscellaneous

There are various other outputs from the PHY that are mostly intended for hand shake with other modules. A few significant signals are described here:

- Preamble found: indicates that Preamble was found.
- CFO estimate: Frequency offset as determined by the preamble detect engine.
- FRAC estimate: Fractional timing error estimate computed in the AA detector;
- Data rate:identified received data rate.
- AA found: indicates that an acceptable address synchronization has occurred.
- AA id: indicates which access address was found.
- corr flag: Indicates to the LQI calculator that the correlation magnitude, exceeds a specified threshold value.
- ant_out: indicates the requested antenna when FAD is enabled. May be ignored when FAD is not enabled.

55.4.7.6.2.5 Debug

There are various other outputs from the PHY that are mostly intended for debug. A few significant signals are described here:

- dtest_*: various outputs send to DTEST module, and then send to GPIO pin.
- phy_dma_mux: muxed one of five DMA data send to DMA module, then finially dumped to RAM

55.4.7.6.3 Memory Map and register definition

This section includes the 2.4GHz PHY module memory map and detailed descriptions of all registers.

55.4.7.6.3.1 2.4GHz PHY REGISTERS register descriptions

55.4.7.6.3.1.1 IPR memory map

XCVR_2P4GHz_PHY base address: 48A0_7600h

Offset	Register	Width (In bits)	Access	Reset value
0h	PHY Uncoded Preamble Detect Config 0 (FSK_PD_CFG0)	32	RW	0000_0000h
4h	PHY Uncoded Preamble Detect Config 1 (FSK_PD_CFG1)	32	RW	0000_0000h
8h	PHY Uncoded Preamble Detect Config 2 (FSK_PD_CFG2)	32	RW	0000_0000h
Ch	(FSK_PD_PH0)	32	RW	0000_0000h
10h	(FSK_PD_PH1)	32	RW	0000_0000h
14h	(FSK_PD_RO_PH2)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	(FSK_PD_RO_PH3)	32	R	0000_0000h
1Ch	(FSK_PD_RO_PH4)	32	R	0000_0000h
20h	(FSK_PD_RO_PH5)	32	R	0000_0000h
24h	PHY Uncoded Config 0 (FSK_CFG0)	32	RW	0000_0000h
28h	PHY Uncoded Config 1 (FSK_CFG1)	32	RW	0000_0000h
2Ch	PHY Uncoded Config 2 (FSK_CFG2)	32	RW	0000_0000h
30h	PHY Uncoded Config 3 (FSK_CFG3)	32	R	0000_0000h
34h	PHY Uncoded Power Threshold Config (FSK_PT)	32	RW	0000_0000h
38h	PHY Uncoded FAD Control (FSK_FAD_CTRL)	32	RW	0000_0000h
3Ch	PHY Uncoded FAD Config (FSK_FAD_CFG)	32	RW	0000_0000h
40h	PHY Uncoded Status (FSK_STAT)	32	R	0000_0000h
44h	PHY Long Range Preamble Detect Config (LR_PD_CFG)	32	RW	0000_0000h
48h	(LR_PD_PH0)	32	RW	0000_0000h
4Ch	(LR_PD_PH1)	32	RW	0000_0000h
50h	(LR_PD_PH2)	32	RW	0000_0000h
54h	(LR_PD_PH3)	32	RW	0000_0000h
58h	(LR_PD_RO_PH4)	32	R	0000_0000h
5Ch	(LR_PD_RO_PH5)	32	R	0000_0000h
60h	(LR_PD_RO_PH6)	32	R	0000_0000h
64h	(LR_PD_RO_PH7)	32	R	0000_0000h
68h	(LR_PD_RO_PH8)	32	R	0000_0000h
6Ch	(LR_PD_RO_PH9)	32	R	0000_0000h
70h	(LR_PD_RO_PH10)	32	R	0000_0000h
74h	(LR_PD_RO_PH11)	32	R	0000_0000h
78h	(LR_PD_RO_PH12)	32	R	0000_0000h
7Ch	(LR_PD_RO_PH13)	32	R	0000_0000h
80h	(LR_PD_RO_PH14)	32	R	0000_0000h
84h	(LR_PD_RO_PH15)	32	R	0000_0000h
88h	(LR_PD_RO_PH16)	32	R	0000_0000h
8Ch	PHY Long Range AA Config (LR_AA_CFG)	32	RW	0000_0000h
90h	PHY Long Range Status (LR_STAT)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
94h	PHY State Machine Config (SM_CFG)	32	RW	0000_0000h
98h	PHY Misc Config (MISC)	32	RW	0000_0000h
9Ch	PHY Status 0 (STAT0)	32	R	0000_0000h
A0h	PHY Status 1 (STAT1)	32	R	0000_0000h
A4h	PHY Status 2 (STAT2)	32	R	0000_0000h
A8h	PHY PrePHY Misc Config (PREPHY_MISC)	32	RW	0000_0000h
ACh	PHY Demodulator Control 0 (DMD_CTRL0)	32	RW	0000_0000h
B0h	PHY Dmodulator Control 1 (DMD_CTRL1)	32	RW	0000_0000h
B4h	PHY Demodulator Control 2 (DMD_CTRL2)	32	RW	0000_0111h
B8h	(DMD_WAVE0_REG0)	32	RW	0000_0000h
BCh	(DMD_WAVE0_REG1)	32	RW	0000_0000h
C0h	(DMD_WAVE0_REG2)	32	RW	0000_0000h
C4h	(DMD_WAVE1_REG0)	32	RW	0000_0000h
C8h	(DMD_WAVE1_REG1)	32	RW	0000_0000h
CCh	(DMD_WAVE1_REG2)	32	RW	0000_0000h
D0h	(DMD_WAVE2_REG0)	32	RW	0000_0000h
D4h	(DMD_WAVE2_REG1)	32	RW	0000_0000h
D8h	(DMD_WAVE2_REG2)	32	RW	0000_0000h
DCh	(DMD_WAVE3_REG0)	32	RW	0000_0000h
E0h	(DMD_WAVE3_REG1)	32	RW	0000_0000h
E4h	(DMD_WAVE3_REG2)	32	RW	0000_0000h
E8h	(DMD_WAVE4_REG0)	32	RW	0000_0000h
ECh	(DMD_WAVE4_REG1)	32	RW	0000_0000h
F0h	(DMD_WAVE4_REG2)	32	RW	0000_0000h
F4h	(DMD_WAVE5_REG0)	32	RW	0000_0000h
F8h	(DMD_WAVE5_REG1)	32	RW	0000_0000h
FCh	(DMD_WAVE5_REG2)	32	RW	0000_0000h
100h	(DMD_WAVE6_REG0)	32	RW	0000_0000h
104h	(DMD_WAVE6_REG1)	32	RW	0000_0000h
108h	(DMD_WAVE6_REG2)	32	RW	0000_0000h
10Ch	(DMD_WAVE7_REG0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

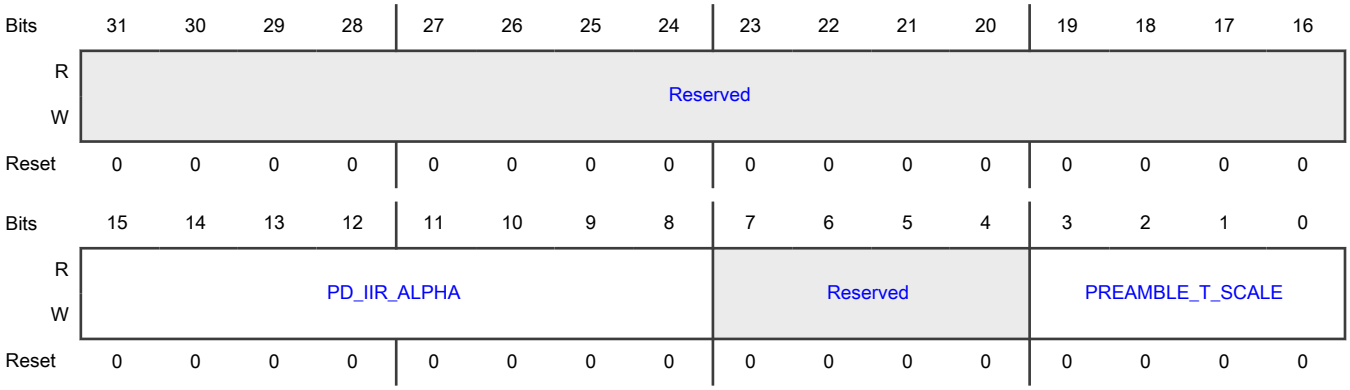
Offset	Register	Width (In bits)	Access	Reset value
110h	(DMD_WAVE7_REG1)	32	RW	0000_0000h
114h	(DMD_WAVE7_REG2)	32	RW	0000_0000h
164h	PHY Demodulator Based SFD Confirmation control register. (DMDAA_CTRL)	32	RW	0000_0009h
168h	High resolution Time-Of-Flight calculation Status. (RTT_STAT)	32	R	5401_0001h
16Ch	PHY RTT control register. (RTT_CTRL)	32	RW	0000_0000h
170h	PHY RTT reference register. (RTT_REF)	32	RW	0000_0000h

55.4.7.6.3.1.2 PHY Uncoded Preamble Detect Config 0 (FSK_PD_CFG0)

Offset

Register	Offset
FSK_PD_CFG0	0h

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 PD_IIR_ALPHA	Forgetting factor used by the complex correlations smoothing leaky integrator.
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 PREAMBLE_T_SCALE	Scaling factor used for fractional time estimation during preamble search.

55.4.7.6.3.1.3 PHY Uncoded Preamble Detect Config 1 (FSK_PD_CFG1)

Offset

Register	Offset
FSK_PD_CFG1	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								PREAMBLE_PATTERN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 PREAMBLE_PATTERN	8-bit preamble pattern used in FM-domain preamble detector.

55.4.7.6.3.1.4 PHY Uncoded Preamble Detect Config 2 (FSK_PD_CFG2)

Offset

Register	Offset
FSK_PD_CFG2	8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								PD_THRESH_ACQ_1_3_2M							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PD_THRESH_ACQ_1_3_1M							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

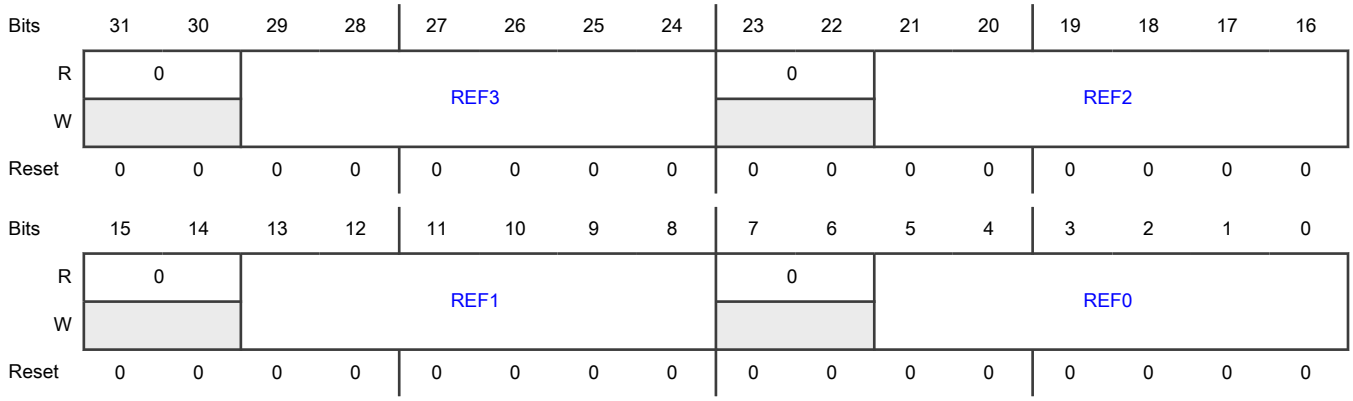
Field	Function
31-24 —	Reserved
23-16 PD_THRESH_ACQ_1_3_2M	Preamble detect threshold for acq mode 1 and 3 at data rate 2Mbps Preamble detect correlation threshold (ufix8en8). This threshold is used to define high correlation for the preamble detect logic.
15-8 —	Reserved
7-0 PD_THRESH_ACQ_1_3_1M	Preamble detect threshold for acq mode 1 and 3 at data rate 1Mbps Preamble detect correlation threshold (ufix8en8). This threshold is used to define high correlation for the preamble detect logic.

55.4.7.6.3.1.5 (FSK_PD_PH0)

Offset

Register	Offset
FSK_PD_PH0	Ch

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Uncoded preamble reference waveform sample 3 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Uncoded preamble reference waveform sample 2 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Uncoded preamble reference waveform sample 1 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Uncoded preamble reference waveform sample 0 (sfix6en5)

55.4.7.6.3.1.6 (FSK_PD_PH1)

Offset

Register	Offset
FSK_PD_PH1	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Uncoded preamble reference waveform sample 7 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Uncoded preamble reference waveform sample 6 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Uncoded preamble reference waveform sample 5 (sfix6en5)
7-6 —	Reserved
5-0	Uncoded preamble reference waveform sample 4 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

Field	Function
REF0	

55.4.7.6.3.1.7 (FSK_PD_RO_PH2)

Offset

Register	Offset
FSK_PD_RO_PH2	14h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Uncoded preamble reference waveform sample 19 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Uncoded preamble reference waveform sample 18 (sfix6en5)
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 REF1	Uncoded preamble reference waveform sample 17 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Uncoded preamble reference waveform sample 16 (sfix6en5)

55.4.7.6.3.1.8 (FSK_PD_RO_PH3)

Offset

Register	Offset
FSK_PD_RO_PH3	18h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Uncoded preamble reference waveform sample 23 (sfix6en5)
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 REF2	Uncoded preamble reference waveform sample 22 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Uncoded preamble reference waveform sample 21 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Uncoded preamble reference waveform sample 20 (sfix6en5)

55.4.7.6.3.1.9 (FSK_PD_RO_PH4)

Offset

Register	Offset
FSK_PD_RO_PH4	1Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-24 REF3	Uncoded preamble reference waveform sample 27 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Uncoded preamble reference waveform sample 26 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Uncoded preamble reference waveform sample 25 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Uncoded preamble reference waveform sample 24 (sfix6en5)

55.4.7.6.3.1.10 (FSK_PD_RO_PH5)

Offset

Register	Offset
FSK_PD_RO_PH5	20h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Uncoded preamble reference waveform sample 31 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Uncoded preamble reference waveform sample 30 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Uncoded preamble reference waveform sample 29 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Uncoded preamble reference waveform sample 28 (sfix6en5)

55.4.7.6.3.1.11 PHY Uncoded Config 0 (FSK_CFG0)

Offset

Register	Offset
FSK_CFG0	24h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			AA_ACQ_1_2_3_THRESH_2M					Reserved	BLE_NTW_ADR_THR			HAMMING_AA_LOW_PWR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			AA_ACQ_1_2_3_THRESH_1M					Reserved	MSK2FSK...	MSK_EN	Reserved	Reserved	FSK_BIT...	AA_OUT_...	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 AA_ACQ_1_2_3_THRESH_2M	For 2Mbps data rate, correlation threshold applicable to AA detection; uses ufix5_En5 fixed-point format.
23 —	Reserved
22-20 BLE_NTW_ADR_THR	Maximum hamming distance from the given AA pattern that may still be accepted as a match; valid range [0,7]. This threshold value are performed on lower power case.
19-16 HAMMING_AA_LOW_PWR	Maximum hamming distance from the given AA pattern that may still be accepted as a match; valid range [0,7]. This threshold value are performed on lower power case.
15-13 —	Reserved
12-8 AA_ACQ_1_2_3_THRESH_1M	For 1Mbps data rate, Correlation threshold applicable to AA detection; uses ufix5_En5 fixed-point format.
7 —	Reserved
6 MSK2FSK_SEED	Last bit of preamble. It is used as seed of the MSK to FSK transformation to derive the FM access address pattern when MSK is enabled
5 MSK_EN	Configures PHY for MSK decoding.
4 —	Reserved
3 —	Reserved
2	This applies at the demodulator, so it affects both AA and the data portions of the packet. 0b - Normal demodulation. 1b - Invert demodulated bits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FSK_BIT_INVERT	0b - Normal demodulation 1b - Invert demodulated bits
1 AA_OUT_SEL	Specifies which AA bits to be played-back to the LL: 0b - output the received AA bits 1b - output the programmed AA bits
0 —	Reserved

55.4.7.6.3.1.12 PHY Uncoded Config 1 (FSK_CFG1)

Offset

Register	Offset
FSK_CFG1	28h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				SYNCTSCALE				Reserved				OVERH_INV			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	OVERH_INV				Reserved				OVERH							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 SYNCTSCALE	Scaling factor used for fractional time estimation during AA search; represented in ufix4_En3 format.
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-11 OVERH_INV	Reciprocal of modulation index; represented in ufix9_En7 format.
10-9 —	Reserved
8-0 OVERH	Modulation index; represented in ufix9_En6 format.

55.4.7.6.3.1.13 PHY Uncoded Config 2 (FSK_CFG2)

Offset

Register	Offset
FSK_CFG2	2Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MAG_WIN				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 MAG_WIN	Indicates the forgetting factor used in received signal level measurement; forgetting factor used is 2^(-mag_window). Note that mag_window values higher than 8 are not typically used.
27-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-0 —	Reserved

55.4.7.6.3.1.14 PHY Uncoded Config 3 (FSK_CFG3)

Offset

Register	Offset
FSK_CFG3	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

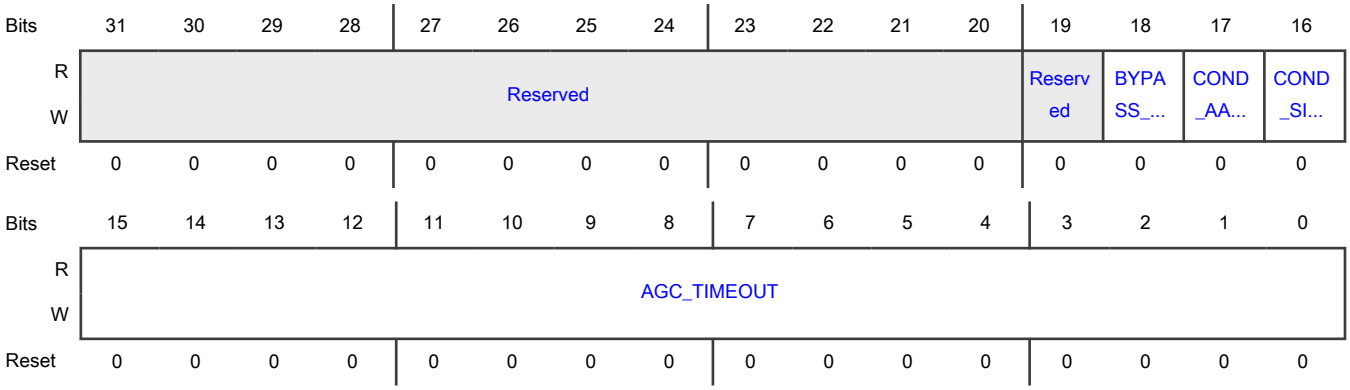
Field	Function
31-28 —	Reserved
27-16 —	Reserved
15-12 —	Reserved
11-0 —	Reserved

55.4.7.6.3.1.15 PHY Uncoded Power Threshold Config (FSK_PT)

Offset

Register	Offset
FSK_PT	34h

Diagram



Fields

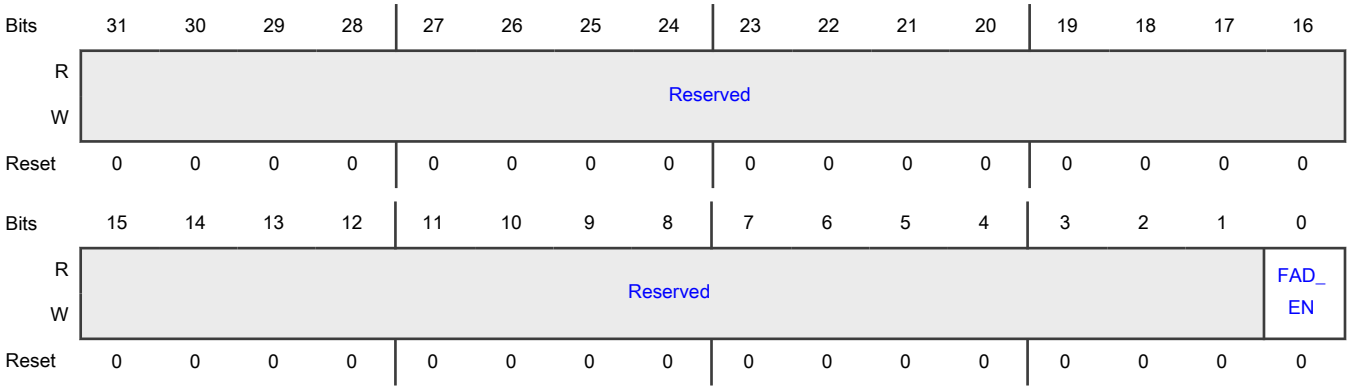
Field	Function
31-20 —	Reserved
19 —	Reserved
18 BYPASS_WITH_RSSI	Bypass signal power measurement with RSSI measurement; 0b - no 1b - yes
17 COND_AA_BUFF_EN	Enables special condition for enabling AA detector buffer; 0b - disable. 1b - enable.
16 COND_SIG_PRST_EN	Enables special conditioning of signal detection; 0b - disable. 1b - enable.
15-0 AGC_TIMEOUT	Time-out, applicable to special conditioning of signal power detection in the Power threshold block, after each AGC gain adjustment. It is expressed in number of samples.

55.4.7.6.3.1.16 PHY Uncoded FAD Control (FSK_FAD_CTRL)

Offset

Register	Offset
FSK_FAD_CTRL	38h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FAD_EN	Enables FAD; 0b - disable. 1b - enable.

55.4.7.6.3.1.17 PHY Uncoded FAD Config (FSK_FAD_CFG)

Offset

Register	Offset
FSK_FAD_CFG	3Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	WIN_SEARCH_PD							Reserved	WIN_FAD_SEARCH_PD						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	WIN_FAD_WAIT_PD							Reserved	WIN_FAD_WAIT_SYNCH						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

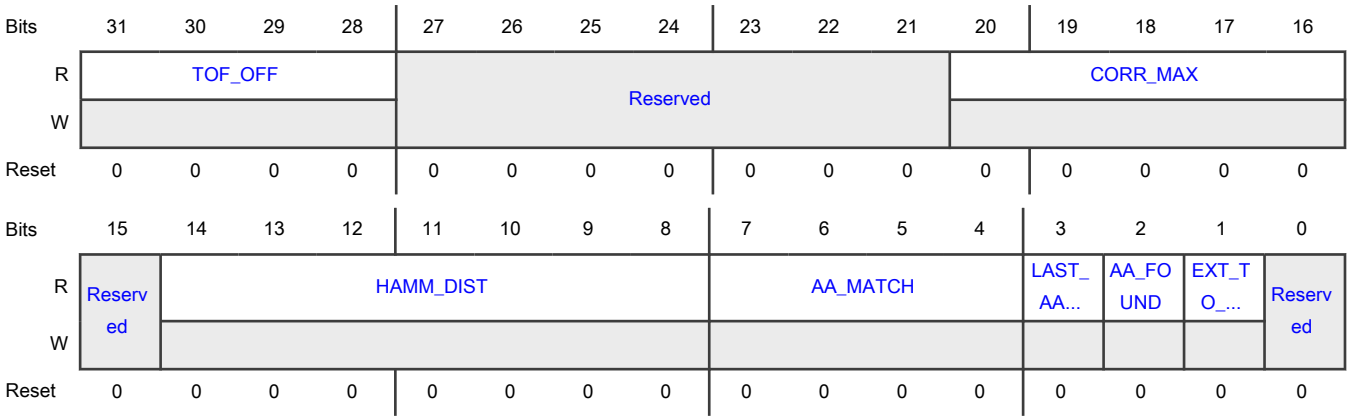
Field	Function
31 —	Reserved
30-24 WIN_SEARCH_PD	Time-window to match preamble pattern on samples coming from the currently selected antenna (refered to as T0 in the PHY state-machine section).
23 —	Reserved
22-16 WIN_FAD_SEARCH_PD	Time-window to match preamble pattern on samples coming from the previously selected antenna (refered to as T1 in the PHY state-machine section).
15 —	Reserved
14-8 WIN_FAD_WAIT_PD	Time-window to wait for clean samples if PD was not found after antenna switch (refered to as T2 in the PHY state-machine section).
7 —	Reserved
6-0 WIN_FAD_WAIT_SYNCH	Time-window to wait for clean samples, before transitioning to AA search PHY state, if PD was found after antenna switch (refered to as T3 in the PHY state-machine section).

55.4.7.6.3.1.18 PHY Uncoded Status (FSK_STAT)

Offset

Register	Offset
FSK_STAT	40h

Diagram



Fields

Field	Function
31-28 TOF_OFF	Timing offset for use in time-of-flight calculation. It is a modulo 16 value which is modified from PHY enabled until AA is found. This provides a fine, sample-time resolution, timing information. Time unit is sample rate, for example, 0.25us for 1M data rate.
27-21 —	Reserved
20-16 CORR_MAX	Indicates the correlation witnessed when AA match occurred {MiHai}In my opinion this should be the maximum AA correlation magnitude, which should be sticky from the moment AA detection occurs until the next PHY reset.
15 —	Reserved
14-8 HAMM_DIST	Indicates the hamming distance witnessed when AA match occurred.
7-4 AA_MATCH	Indicates which non-coded AA has matched. This will clear when the PHY is re-initialized.
3	reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
LAST_AA_BIT	
2 AA_FOUND	Indicates that a uncoded AA detect is active.
1 EXT_TO_MODES_13	Reserved
0 —	Reserved

55.4.7.6.3.1.19 PHY Long Range Preamble Detect Config (LR_PD_CFG)

Offset

Register	Offset
LR_PD_CFG	44h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														NO_PEAKS	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			FREQ_TH					CORR_TH							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17-16 NO_PEAKS	Number of consecutive correlation values that have to exceed the PD correlation threshold,for the same preamble phase, to assert preamble found;

Table continues on the next page...

Table continued from the previous page...

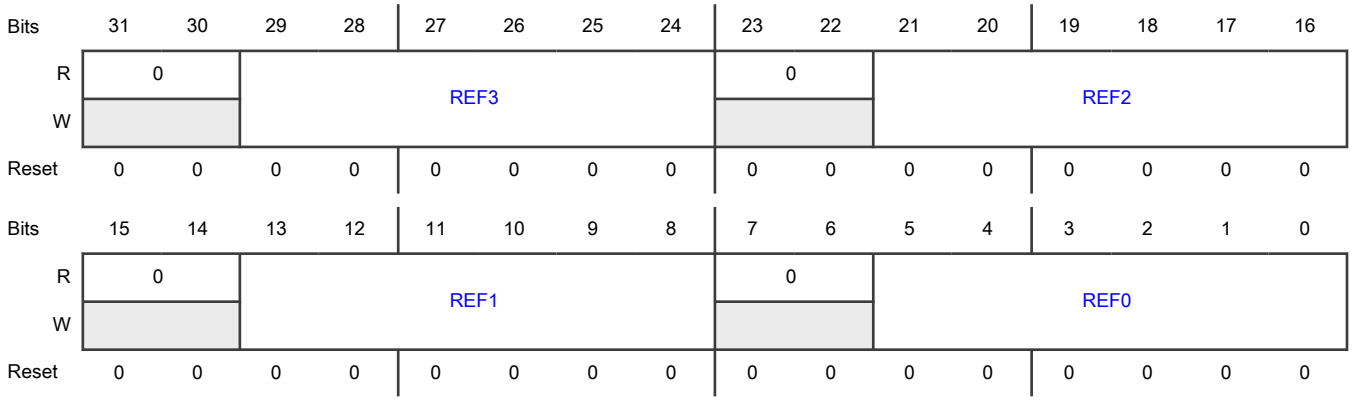
Field	Function
	00b - 2 peaks; 01b - 3 peaks; 10b - 4 peaks; 11b - 5 peaks;
15-13 —	Reserved
12-8 FREQ_TH	Threshold used to compare CFO estimates in the LR preamble detector; uses ufix5_En5 format.
7-0 CORR_TH	Correlation threshold applicable to preamble detection; uses (0,8,8) fixed-point format.

55.4.7.6.3.1.20 (LR_PD_PH0)

Offset

Register	Offset
LR_PD_PH0	48h

Diagram



Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

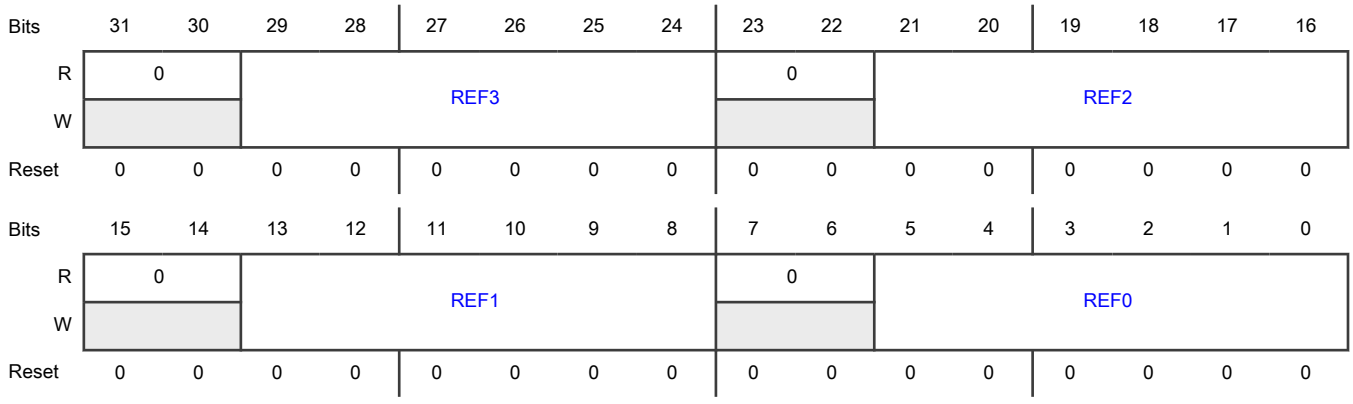
Field	Function
29-24 REF3	Long range preamble reference waveform sample 3 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 2 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 1 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 0 (sfix6en5)

55.4.7.6.3.1.21 (LR_PD_PH1)

Offset

Register	Offset
LR_PD_PH1	4Ch

Diagram



Fields

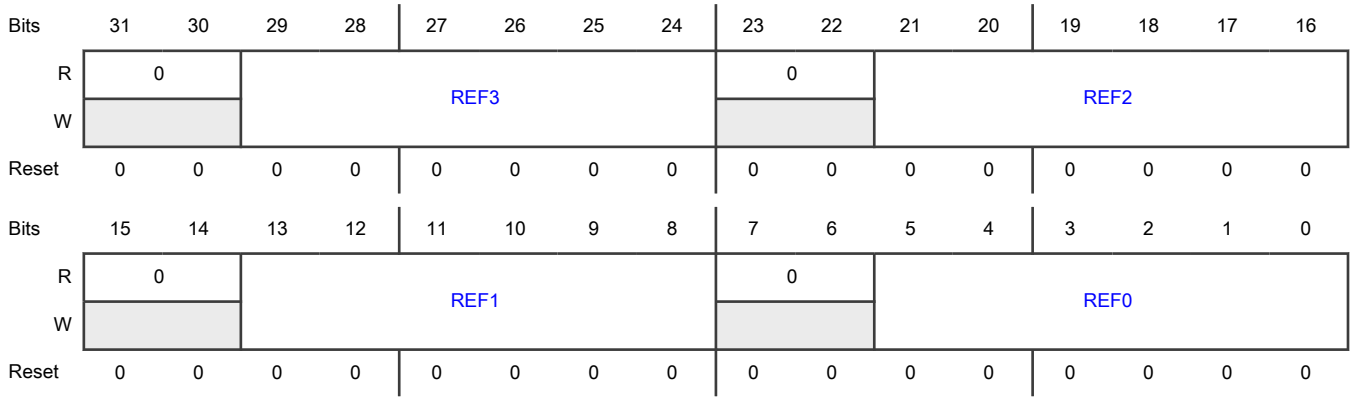
Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 7 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 6 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 5 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 4 (sfix6en5)

55.4.7.6.3.1.22 (LR_PD_PH2)

Offset

Register	Offset
LR_PD_PH2	50h

Diagram



Fields

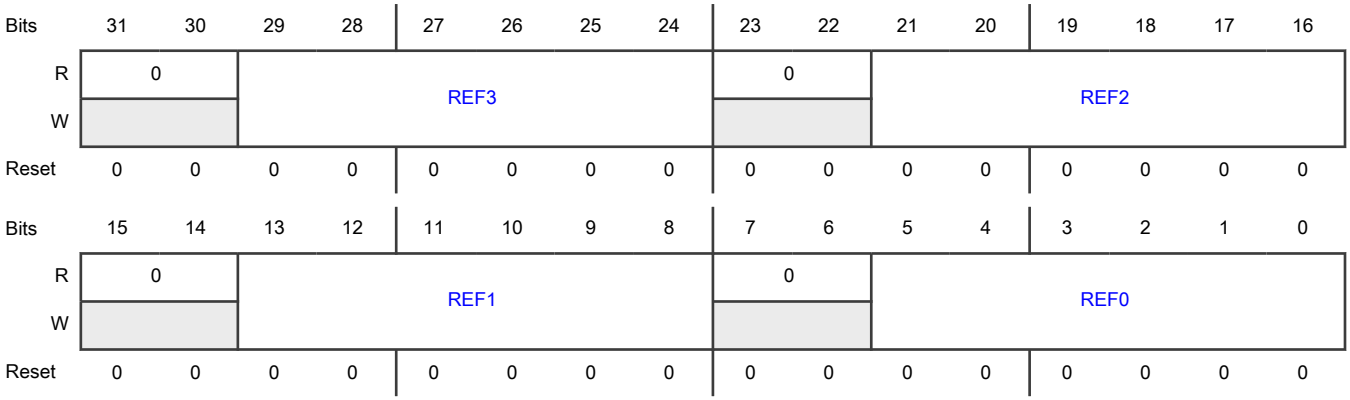
Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 11 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 10 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 9 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 8 (sfix6en5)

55.4.7.6.3.1.23 (LR_PD_PH3)

Offset

Register	Offset
LR_PD_PH3	54h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 15 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 14 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 13 (sfix6en5)
7-6 —	Reserved
5-0	Long range preamble reference waveform sample 12 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

Field	Function
REF0	

55.4.7.6.3.1.24 (LR_PD_RO_PH4)

Offset

Register	Offset
LR_PD_RO_PH4	58h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 19 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 18 (sfix6en5)
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 REF1	Long range preamble reference waveform sample 17 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 16 (sfix6en5)

55.4.7.6.3.1.25 (LR_PD_RO_PH5)

Offset

Register	Offset
LR_PD_RO_PH5	5Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 23 (sfix6en5)
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 REF2	Long range preamble reference waveform sample 22 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 21 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 20 (sfix6en5)

55.4.7.6.3.1.26 (LR_PD_RO_PH6)

Offset

Register	Offset
LR_PD_RO_PH6	60h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-24 REF3	Long range preamble reference waveform sample 27 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 26 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 25 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 24 (sfix6en5)

55.4.7.6.3.1.27 (LR_PD_RO_PH7)

Offset

Register	Offset
LR_PD_RO_PH7	64h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 31 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 30 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 29 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 28 (sfix6en5)

55.4.7.6.3.1.28 (LR_PD_RO_PH8)

Offset

Register	Offset
LR_PD_RO_PH8	68h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 35 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 34 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 33 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 32 (sfix6en5)

55.4.7.6.3.1.29 (LR_PD_RO_PH9)

Offset

Register	Offset
LR_PD_RO_PH9	6Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 39 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 38 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 37 (sfix6en5)
7-6 —	Reserved
5-0	Long range preamble reference waveform sample 36 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

Field	Function
REF0	

55.4.7.6.3.1.30 (LR_PD_RO_PH10)

Offset

Register	Offset
LR_PD_RO_PH10	70h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 43 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 42 (sfix6en5)
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 REF1	Long range preamble reference waveform sample 41 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 40 (sfix6en5)

55.4.7.6.3.1.31 (LR_PD_RO_PH11)

Offset

Register	Offset
LR_PD_RO_PH11	74h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 47 (sfix6en5)
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 REF2	Long range preamble reference waveform sample 46 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 45 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 44 (sfix6en5)

55.4.7.6.3.1.32 (LR_PD_RO_PH12)

Offset

Register	Offset
LR_PD_RO_PH12	78h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-24 REF3	Long range preamble reference waveform sample 51 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 50 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 49 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 48 (sfix6en5)

55.4.7.6.3.1.33 (LR_PD_RO_PH13)

Offset

Register	Offset
LR_PD_RO_PH13	7Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 55 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 54 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 53 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 52 (sfix6en5)

55.4.7.6.3.1.34 (LR_PD_RO_PH14)

Offset

Register	Offset
LR_PD_RO_PH14	80h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 59 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 58 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 57 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 56 (sfix6en5)

55.4.7.6.3.1.35 (LR_PD_RO_PH15)

Offset

Register	Offset
LR_PD_RO_PH15	84h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 63 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 62 (sfix6en5)
15-14 —	Reserved
13-8 REF1	Long range preamble reference waveform sample 61 (sfix6en5)
7-6 —	Reserved
5-0	Long range preamble reference waveform sample 60 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

Field	Function
REF0	

55.4.7.6.3.1.36 (LR_PD_RO_PH16)

Offset

Register	Offset
LR_PD_RO_PH16	88h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		REF3						0		REF2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		REF1						0		REF0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 REF3	Long range preamble reference waveform sample 67 (sfix6en5)
23-22 —	Reserved
21-16 REF2	Long range preamble reference waveform sample 66 (sfix6en5)
15-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 REF1	Long range preamble reference waveform sample 65 (sfix6en5)
7-6 —	Reserved
5-0 REF0	Long range preamble reference waveform sample 64 (sfix6en5)

55.4.7.6.3.1.37 PHY Long Range AA Config (LR_AA_CFG)

Offset

Register	Offset
LR_AA_CFG	8Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		AA_LR_CORR_GAIN						Reserved				ACCESS_ADDR_HAM			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		AA_HAM_THRESH						AA_COR_THRESH							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 AA_LR_CORR_GAIN	AA correlator gain. Format ufix6en3. This gain is applied to soft bits from the demodulator before they are used for address search synchronization.
23-21	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
20-16 ACCESS_ADDR_HAM	Threshold use to compare the Hamming distance, between the reference AA sequence and the received Viterbi decoded AA sequence.
15-14 —	Reserved
13-8 AA_HAM_THRESH	Threshold use to compare the Hamming distance, between reference coded sequence and received coded sequence, in the long-range AA correlator.
7-0 AA_COR_THRESH	Threshold use to compare the correlation magnitude in the long-range AA correlator.

55.4.7.6.3.1.38 PHY Long Range Status (LR_STAT)

Offset

Register	Offset
LR_STAT	90h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMAG_MAX								AA_CORR_MAX							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	CODED_HAMM_DIST							CI	AA_FOUND	DECODED_HAMM_DIST					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

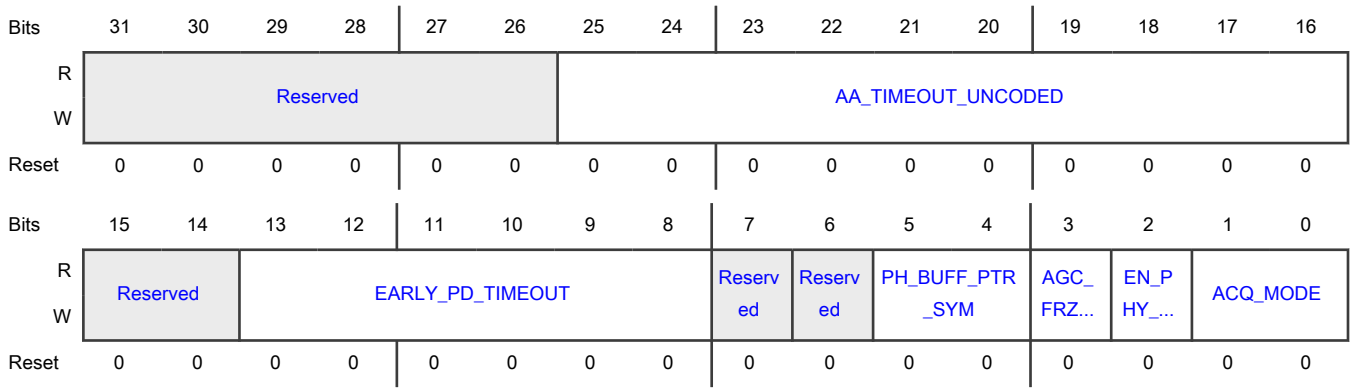
Field	Function
31-24 CMAG_MAX	Indicates the maximum preamble correlation magnitude during preamble found
23-16 AA_CORR_MAX	Indicates the AA correlation magnitude witnessed when AA match occurred
15 —	Reserved
14-8 CODED_HAMM_DIST	Hamming distance between the coded reference sequence and the coded received sequence. This hamming distance is computed in the "Bluetooth LE long range AA correlator" block.
7 CI	CI received.
6 AA_FOUND	Indicates that a AA detect is active for both LR and uncoded.
5-0 DECODED_HAMM_DIST	Hamming distance between the reference sequence and the Viterbi decoded received sequence This is computed in the "AA/CI Detector" block

55.4.7.6.3.1.39 PHY State Machine Config (SM_CFG)

Offset

Register	Offset
SM_CFG	94h

Diagram



Fields

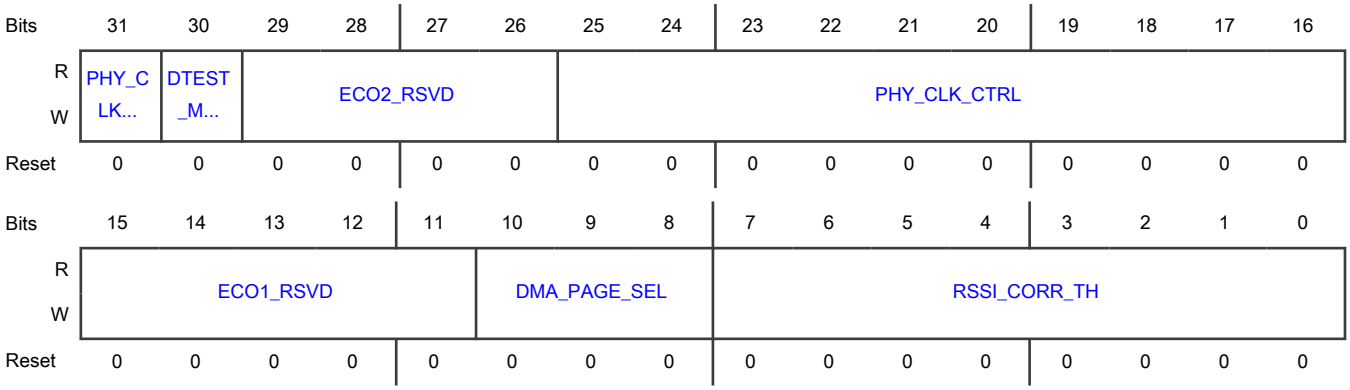
Field	Function
31-26 —	Reserved
25-16 AA_TIMEOUT_UNCODED	Time-out value for access address search for uncoded packets
15-14 —	Reserved
13-8 EARLY_PD_TIMEOUT	Time-out used to reset the AGC state-machine for the eventuality that an "PD found early" event occurs but it is not followed by an "PD found" event
7 —	Reserved
6 —	Reserved
5-4 PH_BUFF_PTR_SYM	Phase buffer size to demodulator, long range only. for system 32MHz case, it should be 0, for system 26MHz case, it should be 1
3 AGC_FRZ_ON_PD_FOUND_ACQ1_LR	Specfies AGC freeze condition for non-coded acq.1 and Bluetooth LE long range. 0b - AGC freeze on AA found. 1b - AGC freeze asserted on PD found.
2 EN_PHY_SM_EXT_RST	Enable PHY state-machine reset on the external reset port; Reserved, should keep 0. 0b - Reset is not allowed. 1b - Reset is allowed.
1-0 ACQ_MODE	Acquisition mode for non-coded reception 00b - Reserved 01b - Use preamble and verify a correlation peak, the synch at the symbol rate as symbol timing is established by the preamble acquisition 10b - Use synch only (which may incorporate part of the preamble) 11b - Use mainly the sync detection: Use a low threshold on the preamble detector and launch the synch detection only if the preamble has shown a recent peak

55.4.7.6.3.1.40 PHY Misc Config (MISC)

Offset

Register	Offset
MISC	98h

Diagram



Fields

Field	Function
31 PHY_CLK_ON	Force PHY clock ON Forces the system to keep the PHY clock on
30 DTEST_MUX_EN	Reserved Should be programed as reset value 0.
29-26 ECO2_RSVD	Reserved it should be programed as reset value 0
25-16 PHY_CLK_CTRL	Enables various clock gating features. Bits are individually decoded, so any combination is allowable. • 0000000001b - Gate off PHY clock when phy_en is not asserted. • 0000000010b - Gate off uncoded preamble detect clock when pd_enable is not asserted (internal signal). • 0000000100b - Gate off uncoded AA synchronizer clock when synchronizer is not in use. • 0000001000b - Gate off m2n demodulator clock when demodulator is not in use. • 0000010000b - Gate off long range preamble detect clock when pd_enable is not asserted (internal signal). • 0000100000b - Gate off long range AA synchronizer clock when synchronizer is not in use. • 0001000000b - Gate off m3c demodulator clock when demodulator is not in use.

Table continues on the next page...

Table continued from the previous page...

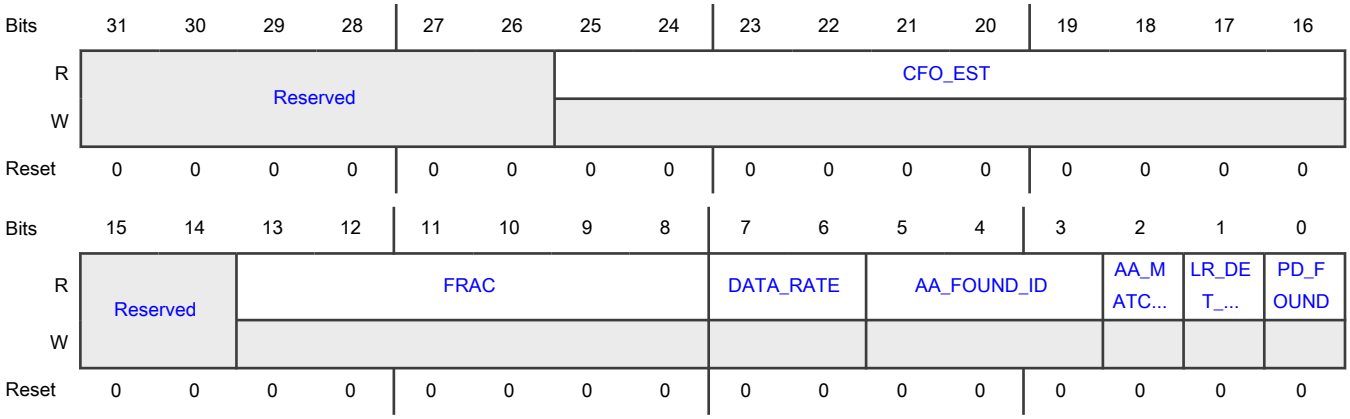
Field	Function
	• 0010000000b - Gate off AA CI detect clock when it is not in use • 0100000000b - Gate off Viterbi input formating clock when it is not in use • 1000000000b - Gate off phy state machine clock dynamicly when it is not toggle
15-11 ECO1_RSVD	Reserved Must be programed as reset value 0.
10-8 DMA_PAGE_SEL	Select which DMA page is send out 000b - Select DMA PAGE 0 for M3C with cfo; 001b - Select DMA PAGE 1 for M3C with magnitude; 010b - Select DMA PAGE 2 for un-coded; 011b - Select DMA PAGE 3 for Long Range Preampble Detect; 100b - Select DMA PAGE 4 for Long Range AA Detect;
7-0 RSSI_CORR_TH	Threshold use to compare a correlation magnitude value, computed in the acquisition block, in order to determine the correlation flag value provided by the PHY to the LQI computation block. Format is ufix8_En8

55.4.7.6.3.1.41 PHY Status 0 (STAT0)

Offset

Register	Offset
STAT0	9Ch

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 CFO_EST	Indicates the currently estimated CFO. Format is sfix10_en9(sign extend form sfix8_en9)
15-14 —	Reserved
13-8 FRAC	Indicates the fractional timing estimate determined in the acquisition block. Format is sfix6_en5(sign extend from sfix3_En2).
7-6 DATA_RATE	Indicates the data rate of received bit 00b - 1Mbps 01b - 2Mbps 10b - 125kbps 11b - 500kbps
5-3 AA_FOUND_ID	Indicates which AA was matched for LR and uncode 000b - uncoded address 0 matched 001b - uncoded address 1 matched 010b - uncoded address 2 matched 011b - uncoded address 3 matched 100b - long range address matched
2 AA_MATCHED	Indicates AA was matched for LR or uncoded
1 LR_DET_FLAG	Indicates Bluetooth LE long range was detected
0 PD_FOUND	PD_FOUND for LR or uncoded

55.4.7.6.3.1.42 PHY Status 1 (STAT1)

Offset

Register	Offset
STAT1	A0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	AA_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AA_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 AA_BITS	AA bits either received or programed

55.4.7.6.3.1.43 PHY Status 2 (STAT2)

Offset

Register	Offset
STAT2	A4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						CNT_AGC_RST		CNT_AACI_TIMEOUT		CNT_LRAA_TIMEOUT		CNT_UNCAA_TIMEO...		CNT_ANT_SW	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9-8 CNT_AGC_RST	Count of AGC soft reset event
7-6 CNT_AACI_TIMEOUT	Count of long range AACI detect timeout event
5-4 CNT_LRAA_TIMEOUT	Count of lang range AA search timeout event
3-2 CNT_UNCAA_TIMEOUT	Count of uncoded AA search timeout event
1-0 CNT_ANT_SW	Count of uncoded ANT switch event when FAD was enabled.

55.4.7.6.3.1.44 PHY PrePHY Misc Config (PREPHY_MISC)

Offset

Register	Offset
PREPHY_MISC	A8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				BUFF_PTR_GFSK				Reserved				BUFF_PTR_LR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-13 —	Reserved
12-8 BUFF_PTR_GFSK	Pointer to the PrePHY IQ buffer for the reception of the uncoded packets.
7-5 —	Reserved
4-0 BUFF_PTR_LR	Pointer to the PrePHY IQ buffer for the reception of the long-range packets.

55.4.7.6.3.1.45 PHY Demodulator Control 0 (DMD_CTRL0)

Offset

Register	Offset
DMD_CTRL0	ACh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	DEMOD_MOD		DREP_SI...	FERR_TR...	TERR_TR...	FED_ERR_SCALE		REPEAT_FACTOR				DREP_SCALE_FREQ			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved				FED_ACT_WIN		Reserved		Reserved				TED_ACT_WIN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29	Determines the number of taps used by the demodulator correlators;

Table continues on the next page...

Table continued from the previous page...

Field	Function
DEMOD_MOD	00b - use 12 taps 01b - use 4 taps 10b - use 7 taps 11b - use 13 taps
28 DREP_SINE_EN	Flag used to enable the non-linear operation in the de-repeater.
27 FERR_TRK_EN	Enables frequency tracking in the demodulator.
26 TERR_TRK_EN	Enables time tracking in the demodulator.
25-23 FED_ERR_SCALE	Scaling factor used by the freqency tracking loop. Value used to scale the frequency drift estimation in the frequency tracking loop. data format is ufix3_en2
22-20 REPEAT_FACTOR	Repetition factor used by the de-repeater. Positive integer which specifies the repetition factor, i.e. the ratio between the PHY sampling rate (at the input of the demodulator) and the 4x oversampling sampling rate used by the demodulator. Data format is ufix3
19-16 DREP_SCALE_FREQ	Frequency domain signal scaling factor used by the de-repeater. Value used to scale the discriminated phase inside the de-repeater (the de-repeater is used to drop the sampling rate by "REPEAT_FACTOR" times inside the demodulator). Data format is ufix4_en2.
15-14 —	Reserved
13-10 —	Reserved
9-8 FED_ACT_WIN	Active window size for the frequency tracking mechanism, expressed in symbols. • 00b - 8. • 01b - 16. • 10b - 32. • 11b - 64.
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-2 —	Reserved
1-0 TED_ACT_WIN	Active window size for the time tracking mechanism, expressed in symbols. • 00b - 8. • 01b - 16. • 10b - 32. • 11b - 64.

55.4.7.6.3.1.46 PHY Dmodulator Control 1 (DMD_CTRL1)

Offset

Register	Offset
DMD_CTRL1	B0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TED_IMM...	Reserved	TTRK_INT_RANGE						TED_IDLE_WIN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FED_IMM...	Reserved	TED_ERR_SCALE						FED_IDLE_WIN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TED_IMM_MEAS_EN	Specifies whether the time tracking starts with an active window; 0b - start with idle window 1b - start with active window
30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-26 TTRK_INT_RANGE	Timing error correction interpolation range, expressed in samples. The value must equal or bigger than 1.
25-16 TED_IDLE_WIN	Idle window size for the time tracking mechanism, expressed in symbols.
15 FED_IMM_MEAS_EN	Specifies whether the frequency tracking starts with an active window; 0b - start with idle window 1b - start with active window
14 —	Reserved
13-10 TED_ERR_SCALE	Scaling factor used by the time tracking loop. Value used to scale the time drift estimation in the frequency tracking loop. Data format is sfix4_en2
9-0 FED_IDLE_WIN	Idle window size for the frequency tracking mechanism, expressed in symbols.

55.4.7.6.3.1.47 PHY Demodulator Control 2 (DMD_CTRL2)

Offset

Register	Offset
DMD_CTRL2	B4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				WAIT_DMD_CLKEN_ADJ				WAIT_VIA_AFTER_AA_ADJ				WAIT_DMD_LR_ADJ			
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1

Fields

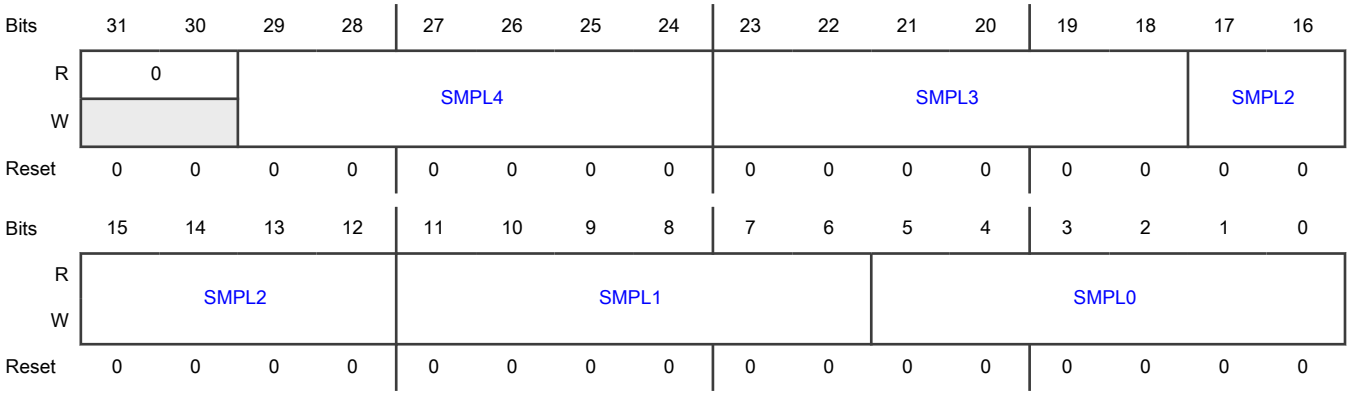
Field	Function
31-12 —	Reserved
11-8 WAIT_DMD_CLKEN_ADJ	Reserved Must be programed as reset value 1.
7-4 WAIT_VIA_AFTER_AA_ADJ	Reserved Must be programed as reset value 1.
3-0 WAIT_DMD_LR_ADJ	Reserved Must be programed as reset value 1.

55.4.7.6.3.1.48 (DMD_WAVE0_REG0)

Offset

Register	Offset
DMD_WAVE0_REG0	B8h

Diagram



Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-24 SMPL4	Demodulator waveform 0 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 0 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 0 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 0 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 0 sample 0 (sfix6en5)

55.4.7.6.3.1.49 (DMD_WAVE0_REG1)

Offset

Register	Offset
DMD_WAVE0_REG1	BCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		SMPL9						SMPL8						SMPL7	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SMPL7				SMPL6				SMPL5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-24 SMPL9	Demodulator waveform 0 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 0 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 0 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 0 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 0 sample 5 (sfix6en5)

55.4.7.6.3.1.50 (DMD_WAVE0_REG2)

Offset

Register	Offset
DMD_WAVE0_REG2	C0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														SMPL12	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SMPL12				SMPL11				SMPL10							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved

Table continues on the next page...

Table continued from the previous page...

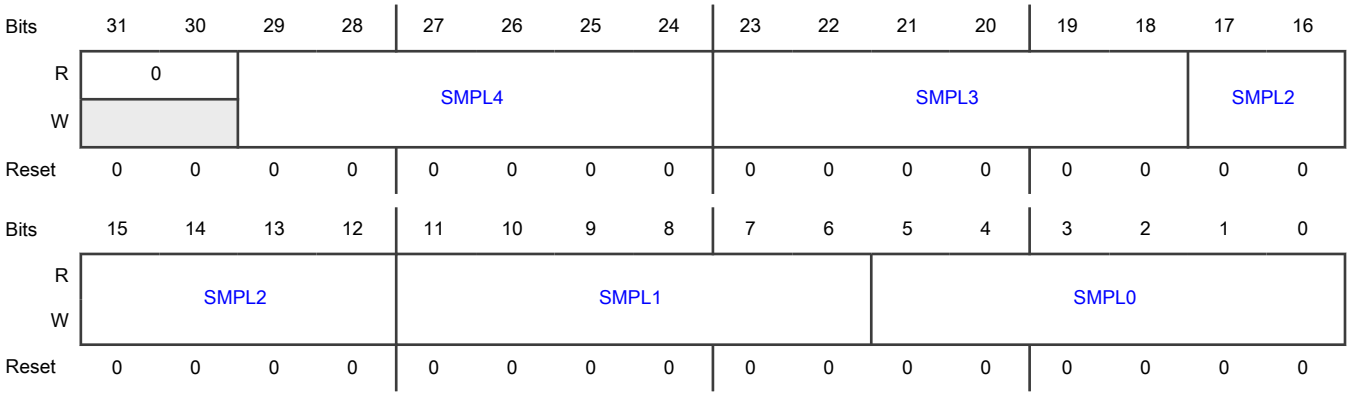
Field	Function
17-12 SMPL12	Demodulator waveform 0 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 0 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 0 sample 10 (sfix6en5)

55.4.7.6.3.1.51 (DMD_WAVE1_REG0)

Offset

Register	Offset
DMD_WAVE1_REG0	C4h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 1 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 1 sample 3 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-12 SMPL2	Demodulator waveform 1 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 1 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 1 sample 0 (sfix6en5)

55.4.7.6.3.1.52 (DMD_WAVE1_REG1)

Offset

Register	Offset
DMD_WAVE1_REG1	C8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		SMPL9						SMPL8						SMPL7	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SMPL7				SMPL6				SMPL5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 1 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 1 sample 8 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

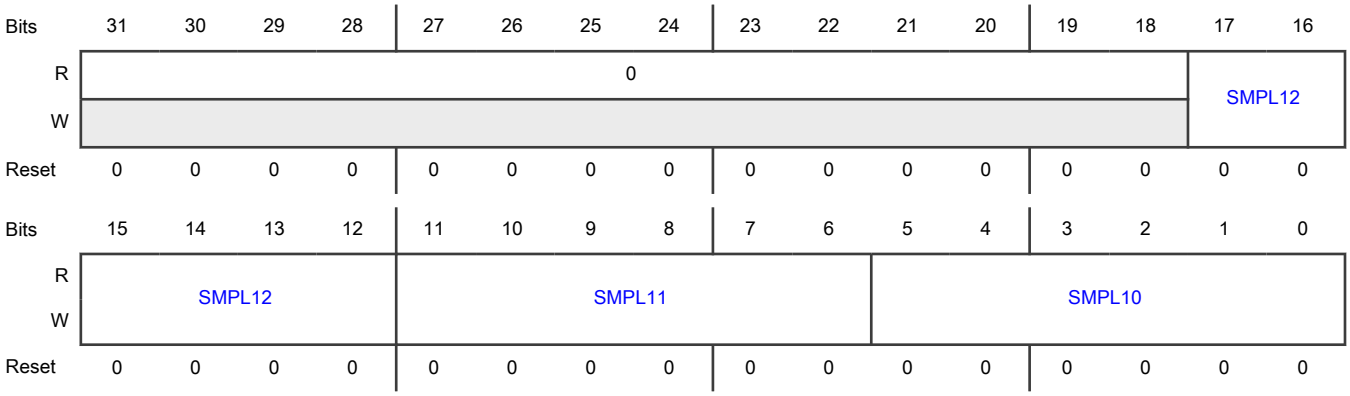
Field	Function
17-12 SMPL7	Demodulator waveform 1 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 1 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 1 sample 5 (sfix6en5)

55.4.7.6.3.1.53 (DMD_WAVE1_REG2)

Offset

Register	Offset
DMD_WAVE1_REG2	CCh

Diagram



Fields

Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 1 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 1 sample 11 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

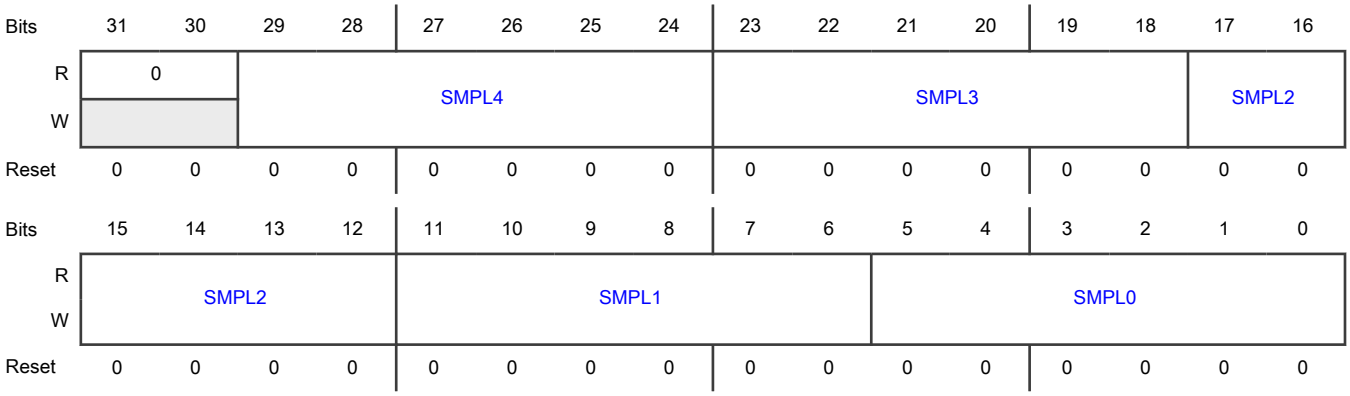
Field	Function
5-0 SMPL10	Demodulator waveform 1 sample 10 (sfix6en5)

55.4.7.6.3.1.54 (DMD_WAVE2_REG0)

Offset

Register	Offset
DMD_WAVE2_REG0	D0h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 2 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 2 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 2 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 2 sample 1 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

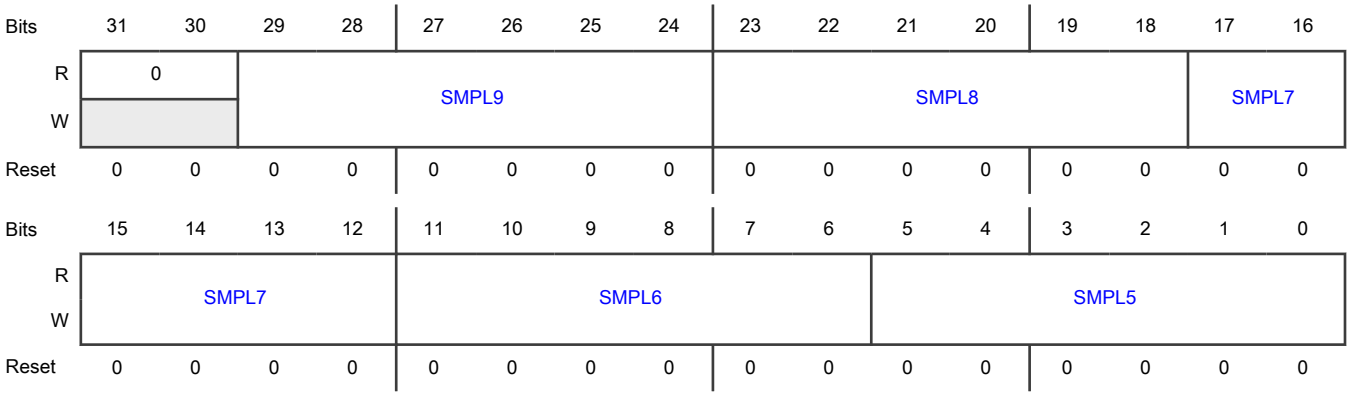
Field	Function
5-0 SMPL0	Demodulator waveform 2 sample 0 (sfix6en5)

55.4.7.6.3.1.55 (DMD_WAVE2_REG1)

Offset

Register	Offset
DMD_WAVE2_REG1	D4h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 2 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 2 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 2 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 2 sample 6 (sfix6en5)

Table continues on the next page...

Table continued from the previous page...

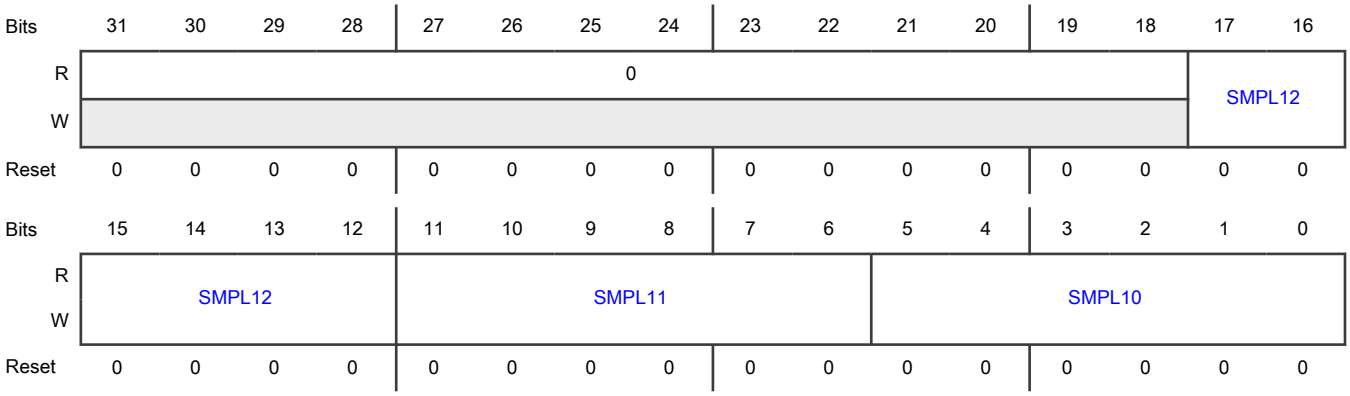
Field	Function
5-0 SMPL5	Demodulator waveform 2 sample 5 (sfix6en5)

55.4.7.6.3.1.56 (DMD_WAVE2_REG2)

Offset

Register	Offset
DMD_WAVE2_REG2	D8h

Diagram



Fields

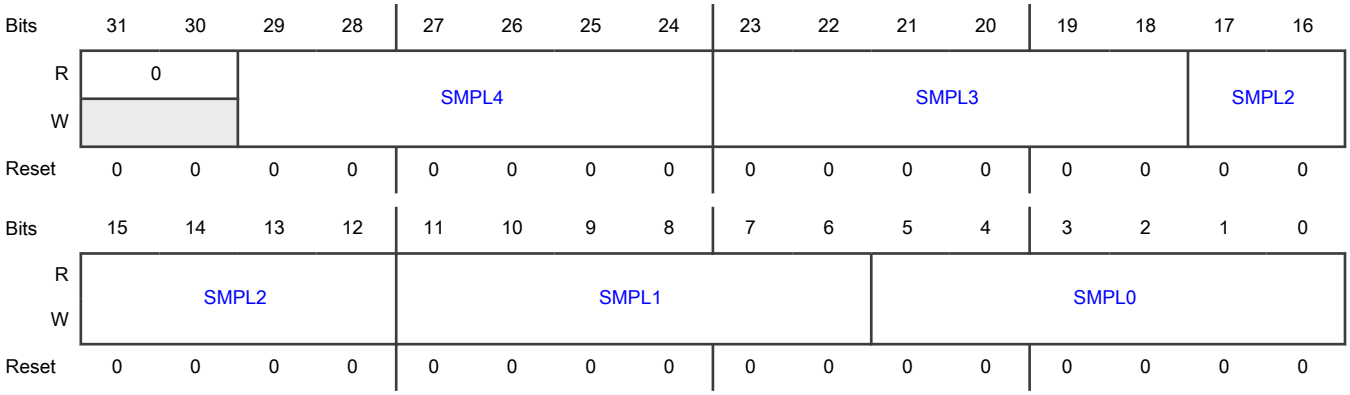
Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 2 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 2 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 2 sample 10 (sfix6en5)

55.4.7.6.3.1.57 (DMD_WAVE3_REG0)

Offset

Register	Offset
DMD_WAVE3_REG0	DCh

Diagram



Fields

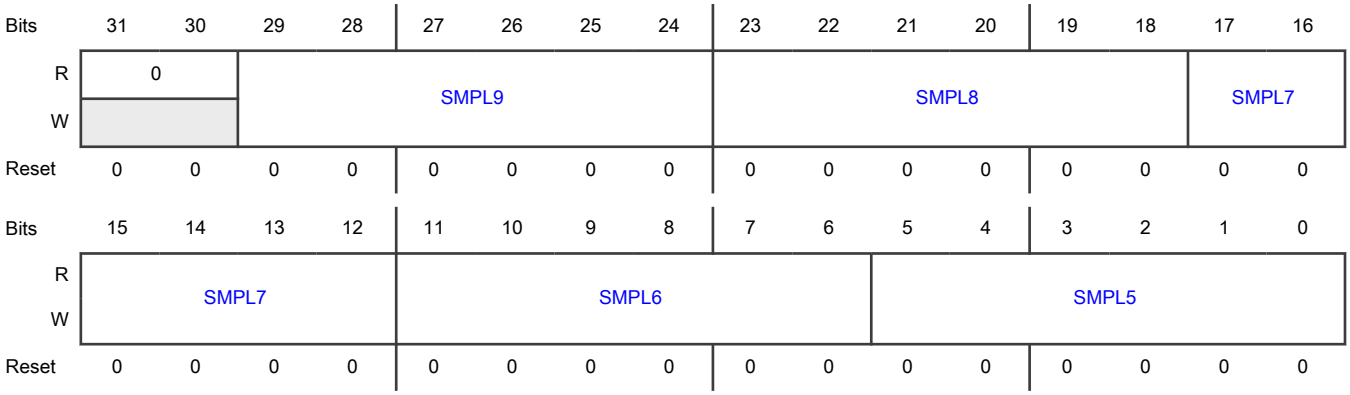
Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 3 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 3 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 3 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 3 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 3 sample 0 (sfix6en5)

55.4.7.6.3.1.58 (DMD_WAVE3_REG1)

Offset

Register	Offset
DMD_WAVE3_REG1	E0h

Diagram



Fields

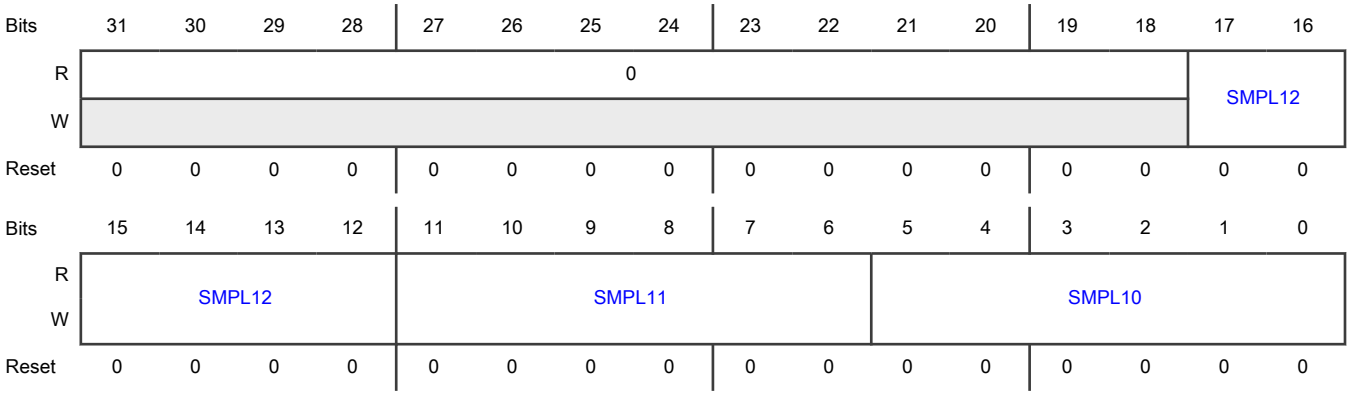
Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 3 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 3 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 3 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 3 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 3 sample 5 (sfix6en5)

55.4.7.6.3.1.59 (DMD_WAVE3_REG2)

Offset

Register	Offset
DMD_WAVE3_REG2	E4h

Diagram



Fields

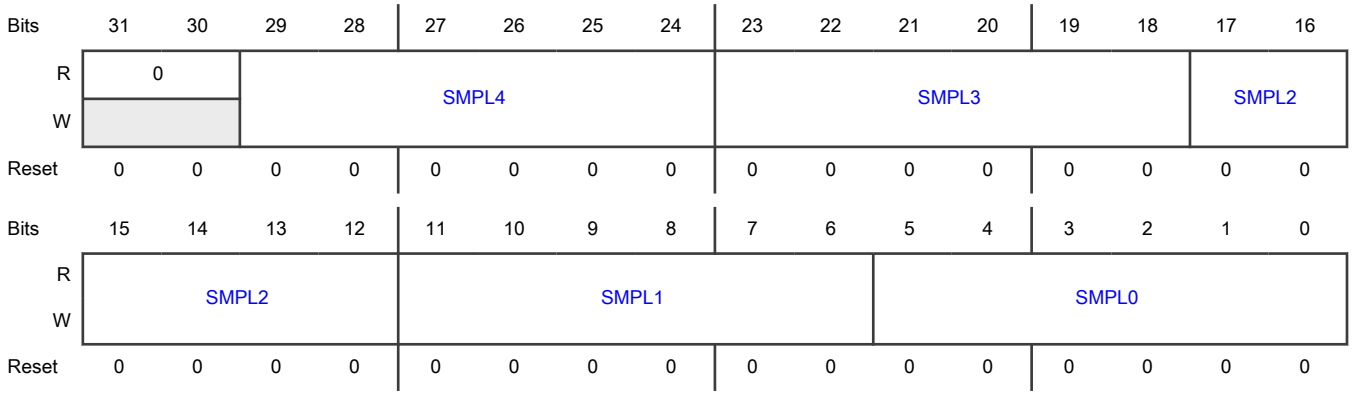
Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 3 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 3 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 3 sample 10 (sfix6en5)

55.4.7.6.3.1.60 (DMD_WAVE4_REG0)

Offset

Register	Offset
DMD_WAVE4_REG0	E8h

Diagram



Fields

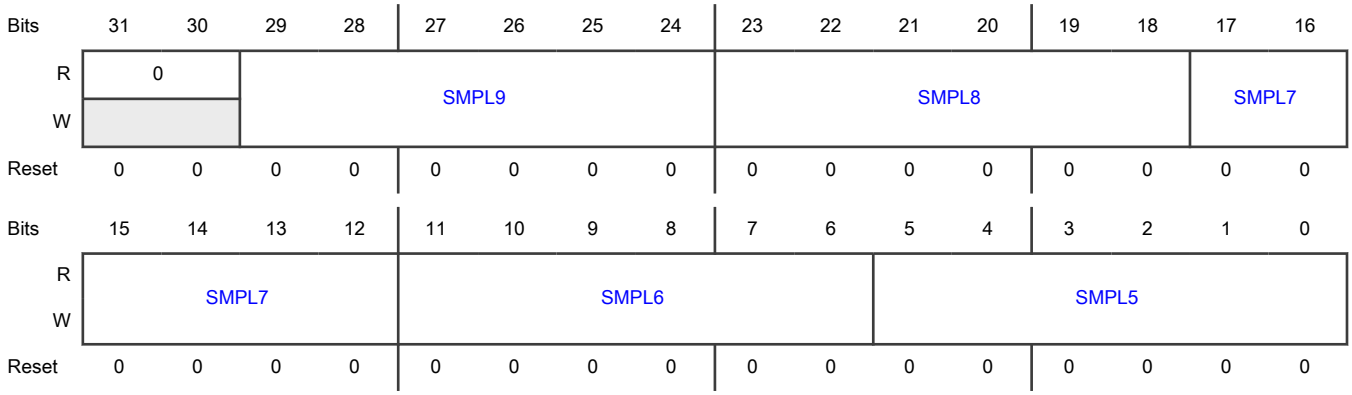
Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 4 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 4 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 4 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 4 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 4 sample 0 (sfix6en5)

55.4.7.6.3.1.61 (DMD_WAVE4_REG1)

Offset

Register	Offset
DMD_WAVE4_REG1	ECh

Diagram



Fields

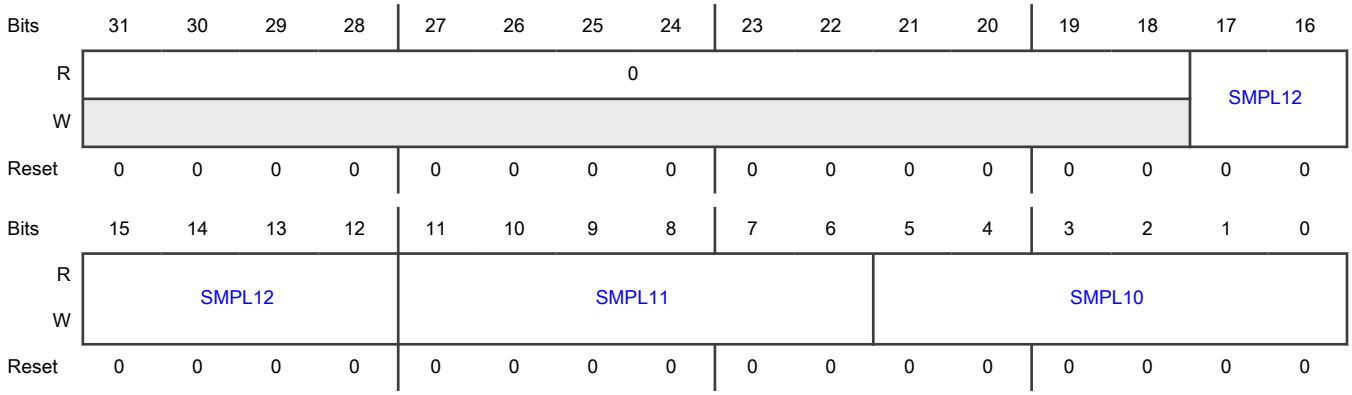
Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 4 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 4 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 4 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 4 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 4 sample 5 (sfix6en5)

55.4.7.6.3.1.62 (DMD_WAVE4_REG2)

Offset

Register	Offset
DMD_WAVE4_REG2	F0h

Diagram



Fields

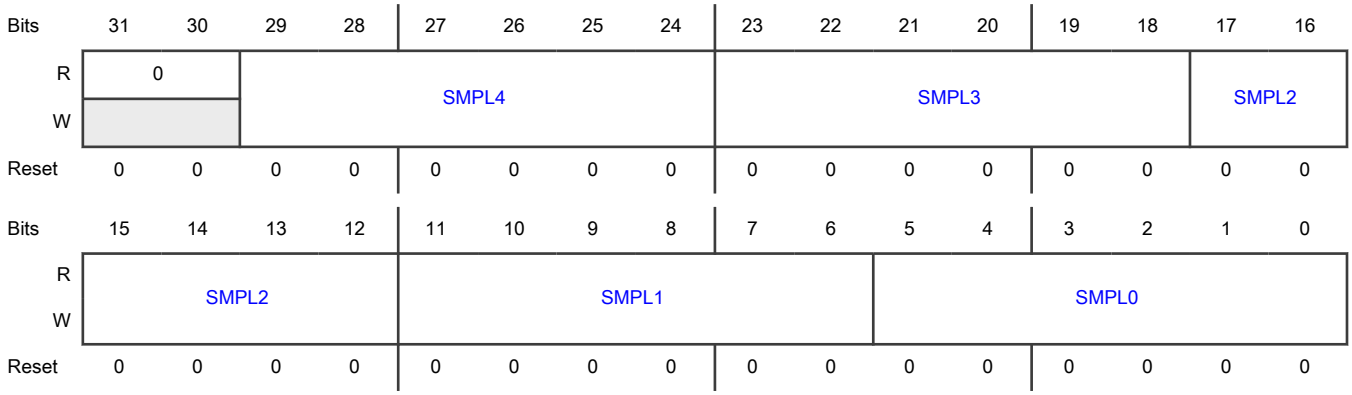
Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 4 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 4 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 4 sample 10 (sfix6en5)

55.4.7.6.3.1.63 (DMD_WAVE5_REG0)

Offset

Register	Offset
DMD_WAVE5_REG0	F4h

Diagram



Fields

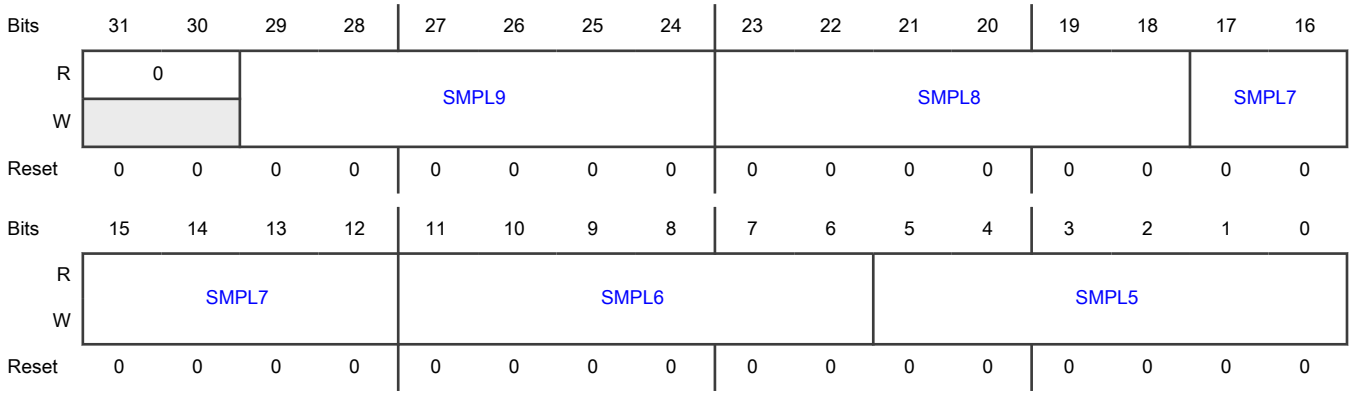
Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 5 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 5 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 5 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 5 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 5 sample 0 (sfix6en5)

55.4.7.6.3.1.64 (DMD_WAVE5_REG1)

Offset

Register	Offset
DMD_WAVE5_REG1	F8h

Diagram



Fields

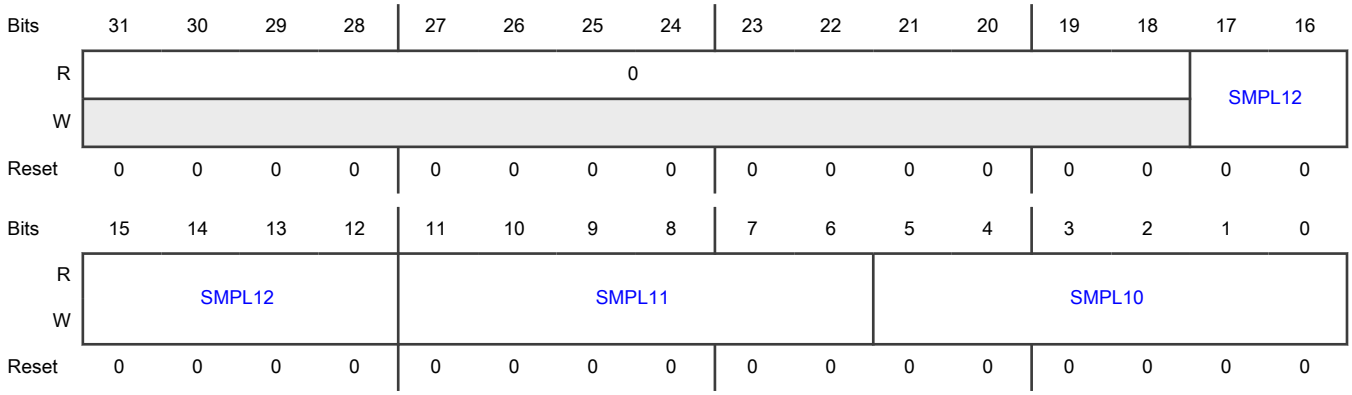
Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 5 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 5 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 5 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 5 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 5 sample 5 (sfix6en5)

55.4.7.6.3.1.65 (DMD_WAVE5_REG2)

Offset

Register	Offset
DMD_WAVE5_REG2	FCh

Diagram



Fields

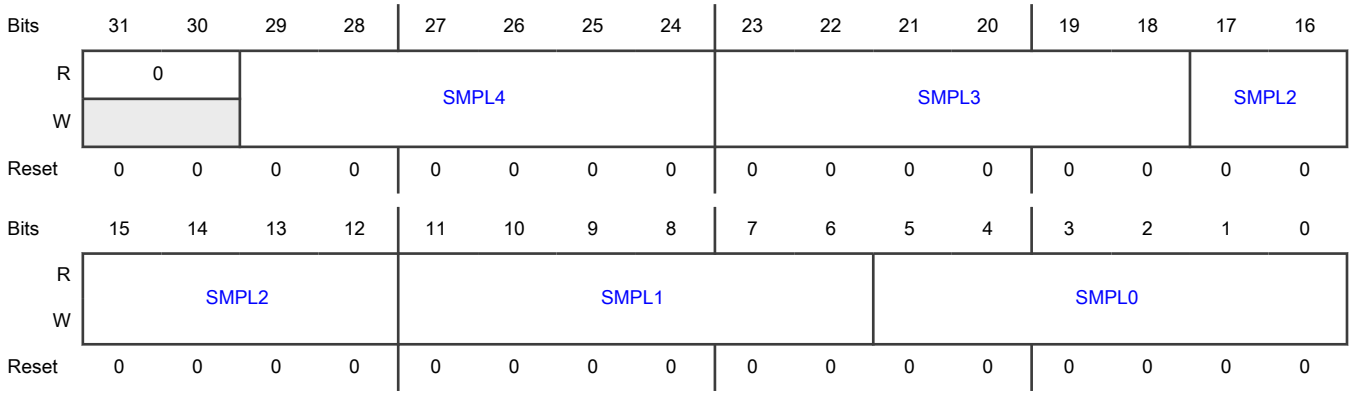
Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 5 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 5 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 5 sample 10 (sfix6en5)

55.4.7.6.3.1.66 (DMD_WAVE6_REG0)

Offset

Register	Offset
DMD_WAVE6_REG0	100h

Diagram



Fields

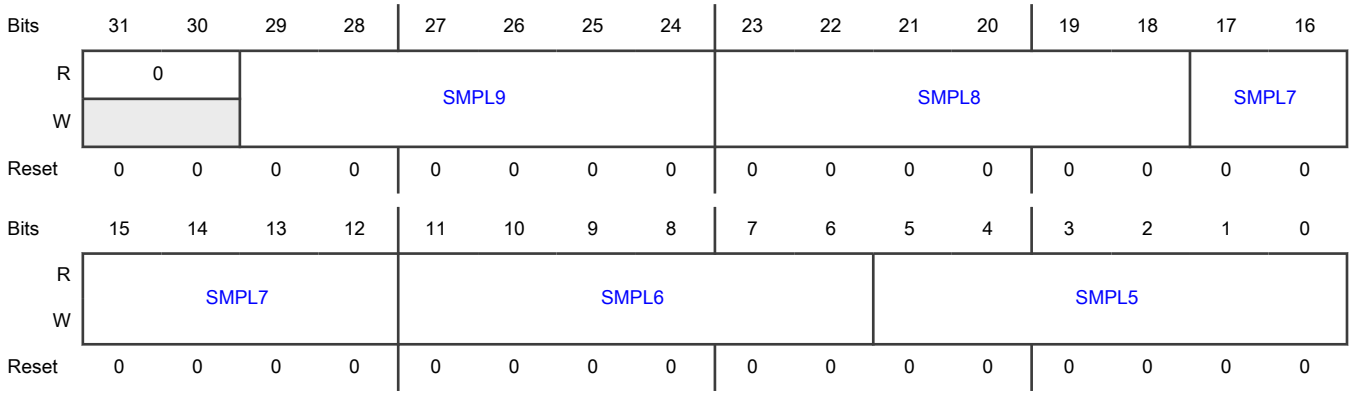
Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 6 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 6 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 6 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 6 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 6 sample 0 (sfix6en5)

55.4.7.6.3.1.67 (DMD_WAVE6_REG1)

Offset

Register	Offset
DMD_WAVE6_REG1	104h

Diagram



Fields

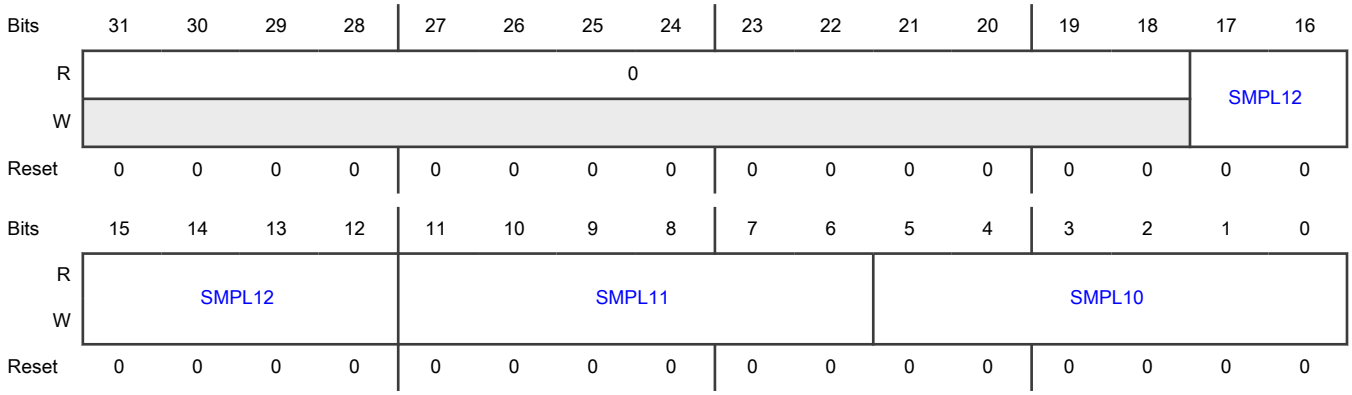
Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 6 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 6 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 6 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 6 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 6 sample 5 (sfix6en5)

55.4.7.6.3.1.68 (DMD_WAVE6_REG2)

Offset

Register	Offset
DMD_WAVE6_REG2	108h

Diagram



Fields

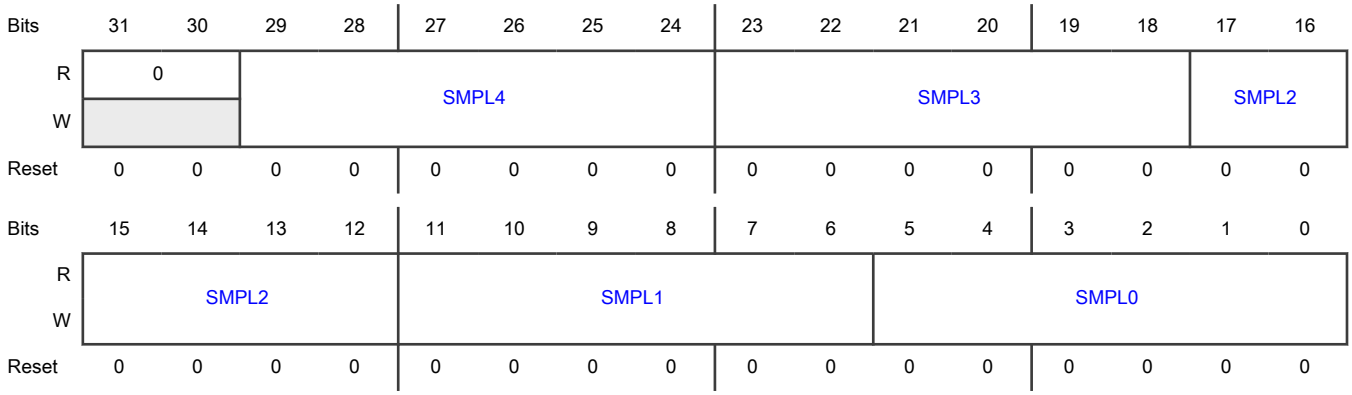
Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 6 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 6 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 6 sample 10 (sfix6en5)

55.4.7.6.3.1.69 (DMD_WAVE7_REG0)

Offset

Register	Offset
DMD_WAVE7_REG0	10Ch

Diagram



Fields

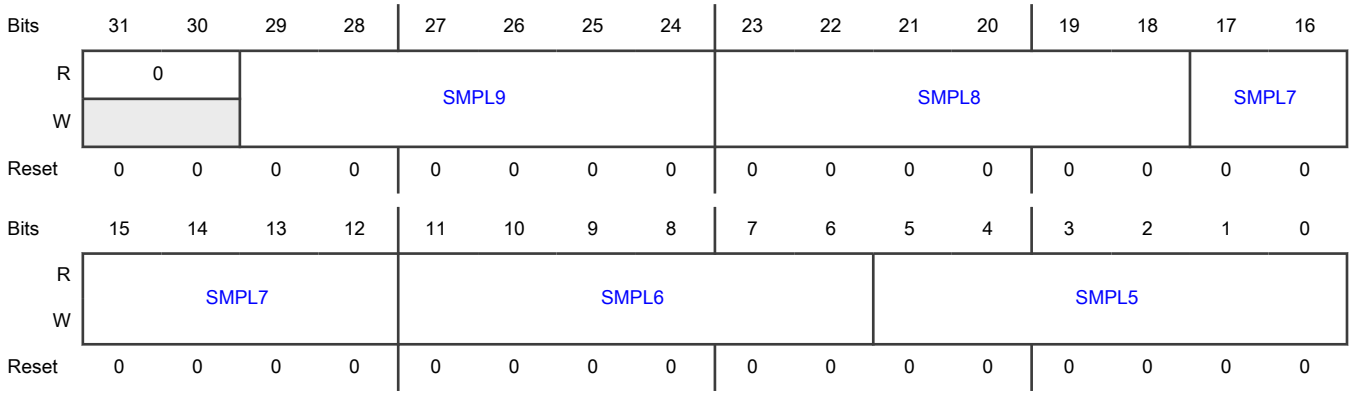
Field	Function
31-30 —	Reserved
29-24 SMPL4	Demodulator waveform 7 sample 4 (sfix6en5)
23-18 SMPL3	Demodulator waveform 7 sample 3 (sfix6en5)
17-12 SMPL2	Demodulator waveform 7 sample 2 (sfix6en5)
11-6 SMPL1	Demodulator waveform 7 sample 1 (sfix6en5)
5-0 SMPL0	Demodulator waveform 7 sample 0 (sfix6en5)

55.4.7.6.3.1.70 (DMD_WAVE7_REG1)

Offset

Register	Offset
DMD_WAVE7_REG1	110h

Diagram



Fields

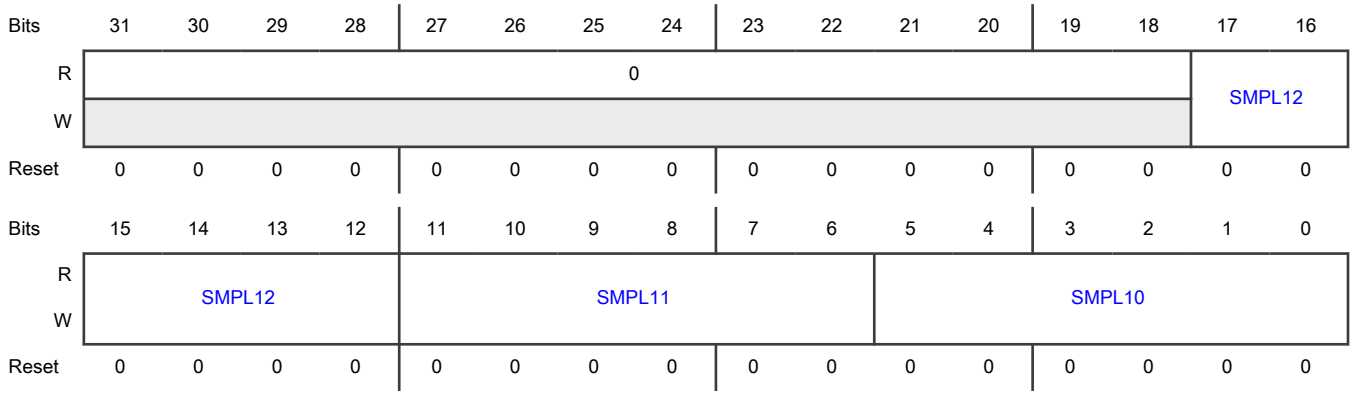
Field	Function
31-30 —	Reserved
29-24 SMPL9	Demodulator waveform 7 sample 9 (sfix6en5)
23-18 SMPL8	Demodulator waveform 7 sample 8 (sfix6en5)
17-12 SMPL7	Demodulator waveform 7 sample 7 (sfix6en5)
11-6 SMPL6	Demodulator waveform 7 sample 6 (sfix6en5)
5-0 SMPL5	Demodulator waveform 7 sample 5 (sfix6en5)

55.4.7.6.3.1.71 (DMD_WAVE7_REG2)

Offset

Register	Offset
DMD_WAVE7_REG2	114h

Diagram



Fields

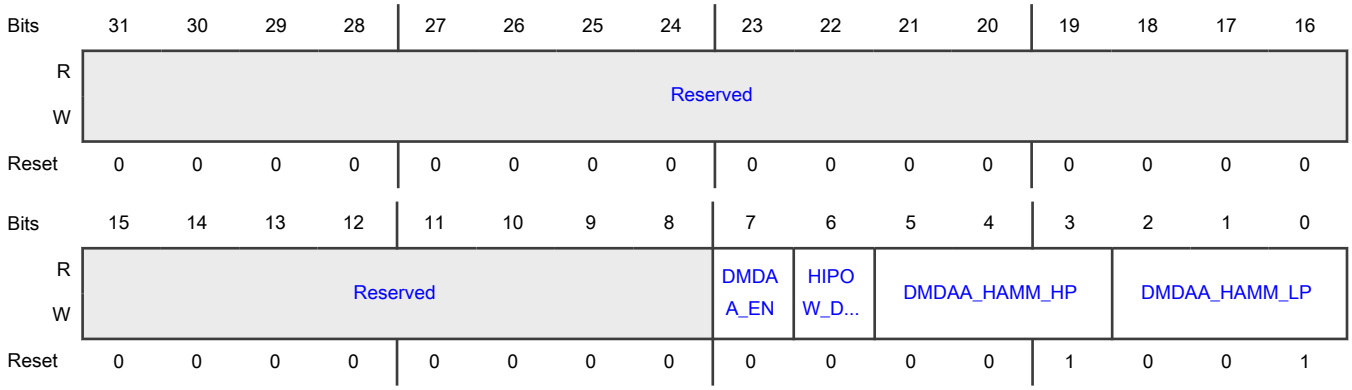
Field	Function
31-18 —	Reserved
17-12 SMPL12	Demodulator waveform 7 sample 12 (sfix6en5)
11-6 SMPL11	Demodulator waveform 7 sample 11 (sfix6en5)
5-0 SMPL10	Demodulator waveform 7 sample 10 (sfix6en5)

55.4.7.6.3.1.72 PHY Demodulator Based SFD Confirmation control register. (DMDAA_CTRL)

Offset

Register	Offset
DMDAA_CTRL	164h

Diagram



Fields

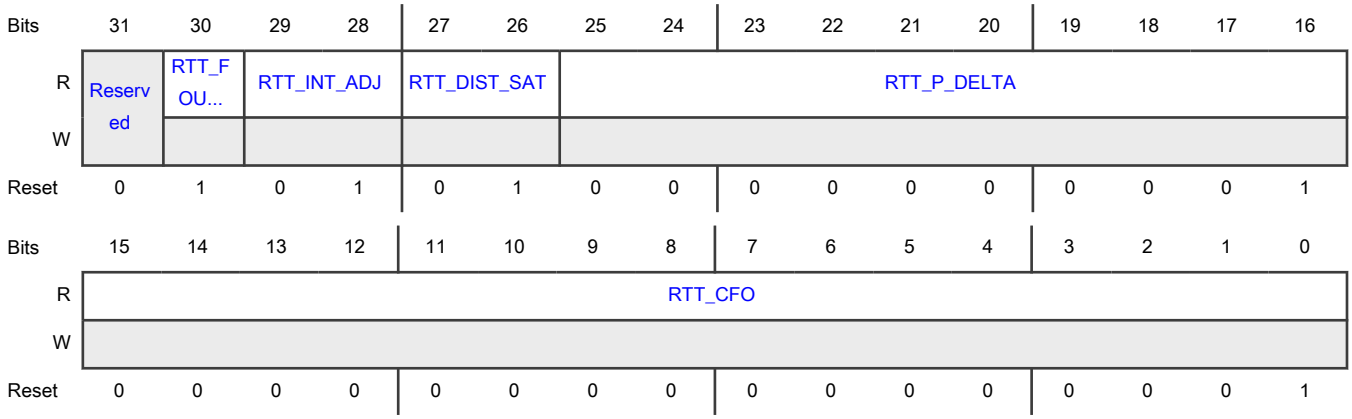
Field	Function
31-8 —	Reserved
7 DMDAA_EN	Enables Demodulator Based SFD Confirmation; DMDAA can perform a secondrdy AA hamming check in parallel to AA search which high precision, which is statred when AA search matched. By default, DMDAA is only applied when signal power near Sensitivity. 0b - disable 1b - enable
6 HIPOW_DIS_OVRD	Override the feature: disable DMDAA when power sensitivity is higher; 0b - disable override, DMDAA disabled when power is high 1b - enable override, DMDAA enabled when power is high
5-3 DMDAA_HAMM_HP	Maximum hamming distance from the given AA pattern that may still be accepted as a match in high power case; valid range [0,7].
2-0 DMDAA_HAMM_LP	Maximum hamming distance from the given AA pattern that may still be accepted as a match in low power case; valid range [0,7].

55.4.7.6.3.1.73 High resolution Time-Of-Flight calculation Status. (RTT_STAT)

Offset

Register	Offset
RTT_STAT	168h

Diagram



Fields

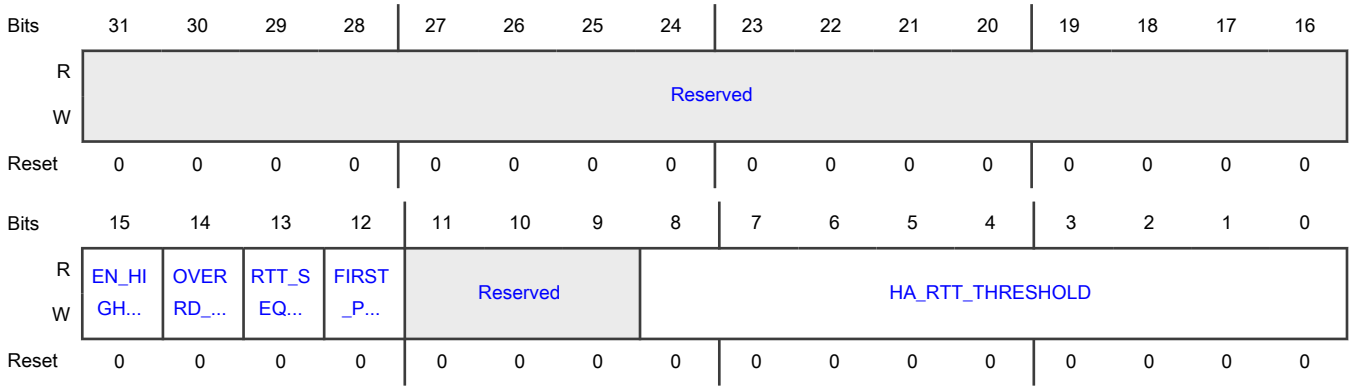
Field	Function
31 —	Reserved
30 RTT_FOUND	Flag that indicates that the HARTT operation is done and a valid PN pattern was detected.
29-28 RTT_INT_ADJ	An integer adjustment of the timing which takes a value different of 0 when the early-late mechanism in the HARTT block chooses a peak different of the one chosen in the acquisition module (possible values are {-1,0,+1}).
27-26 RTT_DIST_SAT	Computed Hamming distance saturated to 2 bits, format is ufix2.
25-16 RTT_P_DELTA	Difference between the squared correlation magnitude values, pm-pp provided by the HARTT block, format is sfix10En9.
15-0 RTT_CFO	The high accuracy CFO computed by the HARTT block through the CORDIC algorithm.

55.4.7.6.3.1.74 PHY RTT control register. (RTT_CTRL)

Offset

Register	Offset
RTT_CTRL	16Ch

Diagram



Fields

Field	Function
31-16 —	Reserved
15 EN_HIGH_ACC_RTT	enables the use of the HA RTT block;
14 OVERRD_PROGR_AA	Enables overriding the programmed AA bits with the PN sequence used by RTT;
13 RTT_SEQ_LEN	can be either 32 (when 0) or 64 bits (when 1) depending on the RTT configuration;
12 FIRST_PDU_BIT	is programmed by software - used for regular packets high accuracy RTT;
11-9 —	Reserved
8-0 HA_RTT_THRESHOLD	threshold used to validate a HA RTT result. Format ufix9en9

55.4.7.6.3.1.75 PHY RTT reference register. (RTT_REF)

Offset

Register	Offset
RTT_REF	170h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								FM_REF_111							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FM_REF_110								FM_REF_010							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 FM_REF_111	Contextual values used to derive the FM reference ha_rtt_threshold . Threshold used to validate a HA RTT result. Format sfix8en7.
15-8 FM_REF_110	Contextual values used to derive the FM reference ha_rtt_threshold . Threshold used to validate a HA RTT result. Format sfix8en7.
7-0 FM_REF_010	Contextual values used to derive the FM reference ha_rtt_threshold . Threshold used to validate a HA RTT result. Format sfix8en7.

55.4.7.6.4 Functional description

The following sections describe functional details of the 2.4GHz PHY module.

55.4.7.6.4.1 Protocol Configuration

The PHY has several configurable parameters that can be changed based on the desired protocol behavior. The details of these parameters are described in the following sections and a summary table is provided here for quick reference.

Table 434. FSK schemes supported by the demodulator

Scheme	baud rate (kbps) R	Data rate (kbps) DR	modulation index *h*	Frequency pulse	LO error (ppm)
Bluetooth LE	1000	1000	0.50	Gaussian; BT=0.5	62.50
Bluetooth LE LR 125	1000	125	0.50	Gaussian; BT=0.5	62.50
Bluetooth LE LR 500	1000	500	0.50	Gaussian; BT=0.5	62.50
GFSK_05_07-500	500	500	0.70	Gaussian; BT=0.5	41.67
GFSK_03_05-1000	1000	1000	0.50	Gaussian; BT=0.3	41.67
MSK_05-1000	500	500	0.50	MSK	41.67
GFSK_05_05-250	250	250	0.50	Gaussian; BT=0.5	20.83
GFSK_05_10-250	250	250	1.00	Gaussian; BT=0.5	20.83
GFSK_07_05-1000	1000	1000	0.50	Gaussian; BT=0.7	62.50
GFSK_05_05-2000	2000	2000	0.50	Gaussian; BT=0.5	62.50
MSK_05-250	250	250	0.50	MSK	20.83
GFSK_05_032-2000	2000	2000	0.32	Gaussian; BT=0.5	62.5
FSK_05-500	500	500	0.50	Rectangular	41.67

55.4.7.6.4.2 Operating modes

This section describes all functional operation modes of the 2.4GHz PHY module.

55.4.7.6.4.2.1 Scan mode

Scan mode determines the way packets are being search for. As shown three modes of operation are defined for the 2.4 GHz PHY:

- Non-coded FSK search: look only for non-coded packets, while keeping all the Long range Bluetooth LE acquisition blocks disabled;
- Long range Bluetooth LE search: search only for long range Bluetooth LE packets while keeping all the non-coded acquisition blocks disabled;
- Simultaneous Non-coded and long range Bluetooth LE search: simultaneously look for long range and non-coded Bluetooth LE packets by enabling both long range and non-coded acquisition blocks.

Note that the PHY block is capable of performing simultaneous search of any non-code FSK use-case and long range Bluetooth LE but, from an use-cases perspective, this mode of operation is restricted to non-coded Bluetooth LE.

55.4.7.6.4.2.2 Acqusition mode

From a non-coded perspective there are three acquisition modes:

- Acquisition mode 1: both preamble and AA are used for packet detection and synchronization. The preamble detector is used to detect the packet and to realize symbol timing synchronization and, following a preamble detection, AA is being searched at symbol rate in order to get frequency and frame synchronization. This mode is suitable for packets that have long enough preambles (usually more than 6 octets) so that the symbol timing synchronization can be reliably realized;

- Acquisition mode 2: only the AA is used for packet detection and time/frequency synchronization. This acquisition mode is suitable to be used when the packet being searched has a short preamble;
- Acquisition mode 3: both preamble and AA are used for packet detection and synchronization. The preamble detector is used merely to detect the presence of a packet and, following a preamble detection, AA is being searched at sample rate in order to get frequency synchronization and symbol/frame timing synchronization. This acquisition mode is suitable for packets which short preamble, and it provides not only a lower power consumption but also an earlier packet presence indication than mode 2.

55.4.7.6.4.3 Operations

This section describes the 2.4GHz PHY module's operations.

55.4.7.6.4.3.1 PHY state-machine

The state-machine state diagram is depicted in Figure 311. As per the depiction in Scan mode three modes of operation are defined for the 2.4 GHz PHY:

- Non-coded FSK search: look only for non-coded packets, while keeping all the long range Bluetooth LE acquisition blocks disabled
- Long range Bluetooth LE search: search only for long range Bluetooth LE packets while keeping all the non-coded acquisition blocks disabled
- Simultaneous Non-coded and long range Bluetooth LE search: simultaneously look for long range Bluetooth LE and non-coded packets by enabling both long range and non-coded acquisition blocks

Note that the PHY block can simultaneously search for non-coded FSK packets and long range Bluetooth LE packets; however the only tested configurations are for simultaneous non-coded Bluetooth LE with long range Bluetooth LE.

The state-machine has 6 main states:

- State 0: Initialization state
  - Reset state used to recycle all the blocks in the PHY
- State 1: Acquisition state
  - Completely (for non-coded) or partially (for long range Bluetooth LE) responsible with time/frequency synchronization
    - More details about state1 are given in the sequel of this section
  - When the PHY is configured to perform simultaneous non-coded and long range Bluetooth LE search
    - Makes a decision whether to continue treating the packet as a non-coded or as a long range Bluetooth LE
- State 2: long range Bluetooth LE PD/AA
  - Only when a long range Bluetooth LE packet is detected
  - Responsible with the simultaneous LR preamble and AA search
  - It can only be entered following a long range Bluetooth LE preamble detection event
    - Keeps the preamble search to improve the CFO estimate
    - Extends the time used for the AA search process as required
- State 3: Long range Bluetooth LE AA/CI check
  - Only when a long range Bluetooth LE packet is detected
  - Tells the PHY and RBME to unpack the AA despread soft-bits
    - More details about these in the long range Bluetooth LE AA detection section
  - Turns on the Viterbi decoder
  - Looks for a match to the AA bits

- Extracts the CI information
  - The CI is further used to configure RBME for PSDU decoding.

- State 4: Demodulation & Decoding
  - handles the PSDU demodulation and decoding for both non-coded and long range Bluetooth LE packets
- State 5: RTT state
  - This state is used for the RTT operation (for more details please consult the description of the RTT module)
  - This represents an idle state for the PHY to transition to when the reception of an RTT packets is done
  - PHY exists this state following a PHY reset issued at a higher level in the hierarchy
  - (Figures below don't show state 5 as this state is not representative for regular communication packets exchanges)

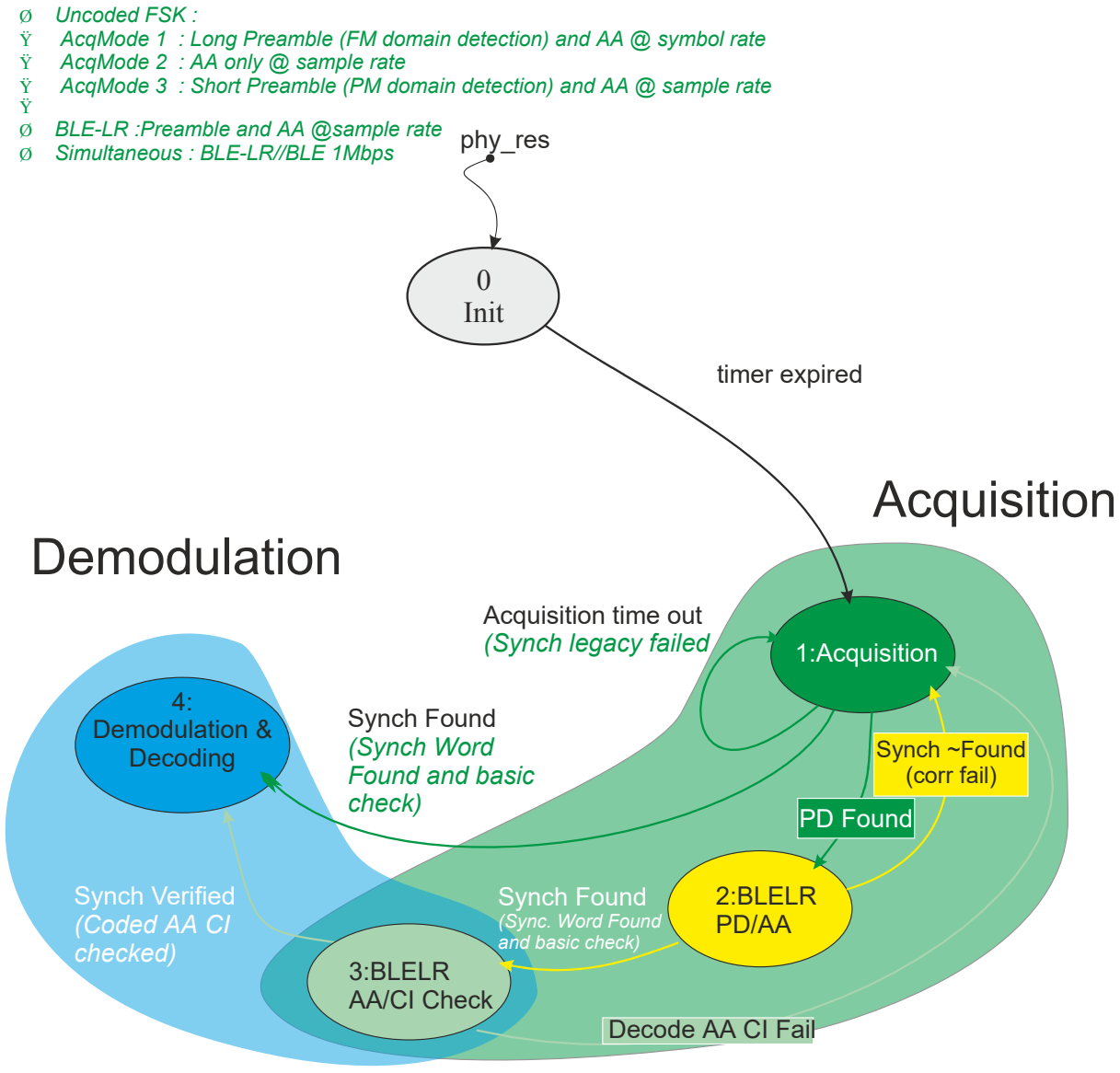


Figure 311. Top level view of the state-machine

A time out is used for the AA detection to allow for a prompt recovery. In long range and non-coded acquisition modes 1 and 3, AA detection runs in parallel with preamble detection in order to: 1) extend the AA search as needed and 2) to update the synchronization information. If neither a preamble detection nor an AA detection occur during the time-out period, the PHY state machine returns to the Acquisition state, disables AA detection and recycles all the PD block memory excepting the samples buffer.

Figure 311 illustrates two functions. The green areas are all about acquisition in a larger sense, meaning that all the processing concerning packets detection are associated to it and the blue region is about demodulation, responsible for PSDU processing and/or decoding in the RBME (Radio Bit Manipulation Engine).

For long range Bluetooth LE packets the preamble detection is responsible for packet acquisition and frequency synchronization (i.e. CFO estimation). Following a preamble found event: 1) the incoming samples are demodulated; 2) the resulting soft bits are correlated to the AA pattern; 3) finally, the correlation peak marks the location of the incoming frame and symbol timing is estimated. The soft-bits are now re-aligned to symbol timing and passed through the Viterbi decoder (State 2 of the SM) . The decoded bits are then matched to the AA pattern and the CI is extracted (State 3 of the SM) .

From a non-coded perspective as the depiction in Acqusition mode there are three acquisition modes:

- Acquisition mode 1
  - Both preamble and AA are used for packet detection and synchronization
  - The preamble detector detects the packet and synchronizes timing
  - Following a preamble detection, the AA is searched at the symbol rate
    - The AA detection outputs frequency and frame synchronization
  - This mode is suitable for packets that have long enough preambles (usually more than 6 octets)
- Acquisition mode 2
  - Only the AA is used for packet detection and time/frequency synchronization
  - This acquisition mode is suitable to be used when the packet being searched has a short preamble
- Acquisition mode 3
  - Both preamble and AA are used for packet detection and synchronization
  - The preamble detector is used merely to detect the presence of a packet
  - Following a preamble detection, the AA is searched at the sample rate
    - The AA detection outputs frequency synchronization and symbol/frame timing synchronization
  - This acquisition mode is
    - Suitable for packets which short preambles
    - Lowers power consumption and an earlier packet detection than mode 2

The figures below show detailed instantiations of the top-level state-machine, considering all the acquisition modes described earlier and all the operation modes. Figure 312 illustrates the state-machine instantiation for non-simultaneous non-coded acquisition modes or long range Bluetooth LE.

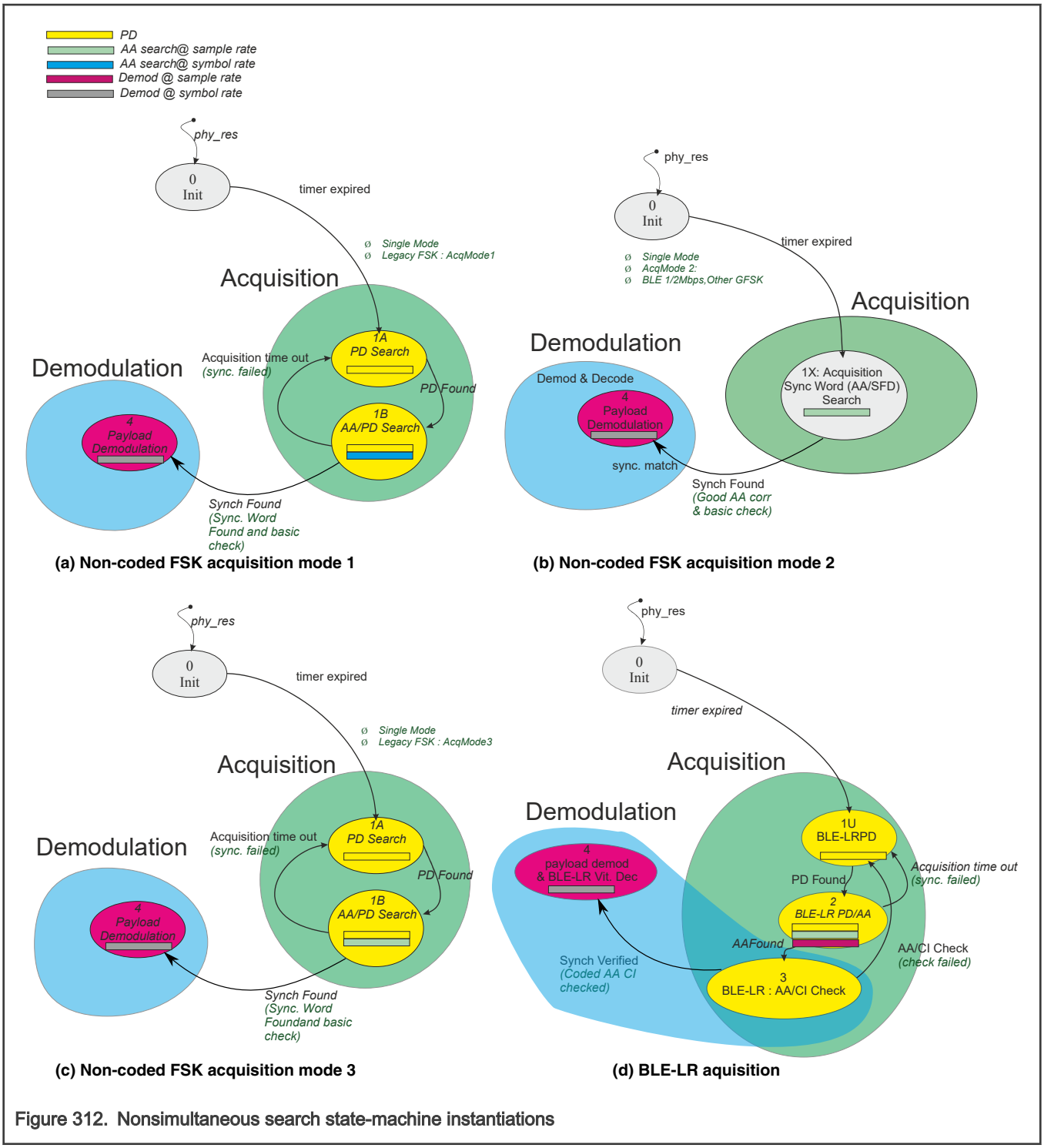
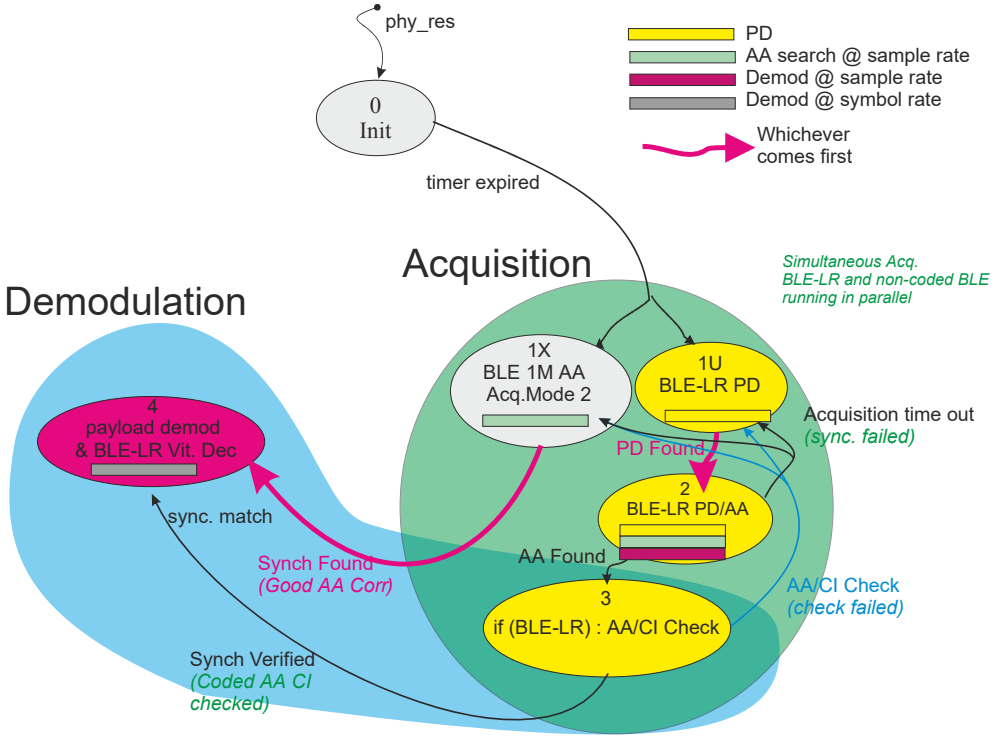
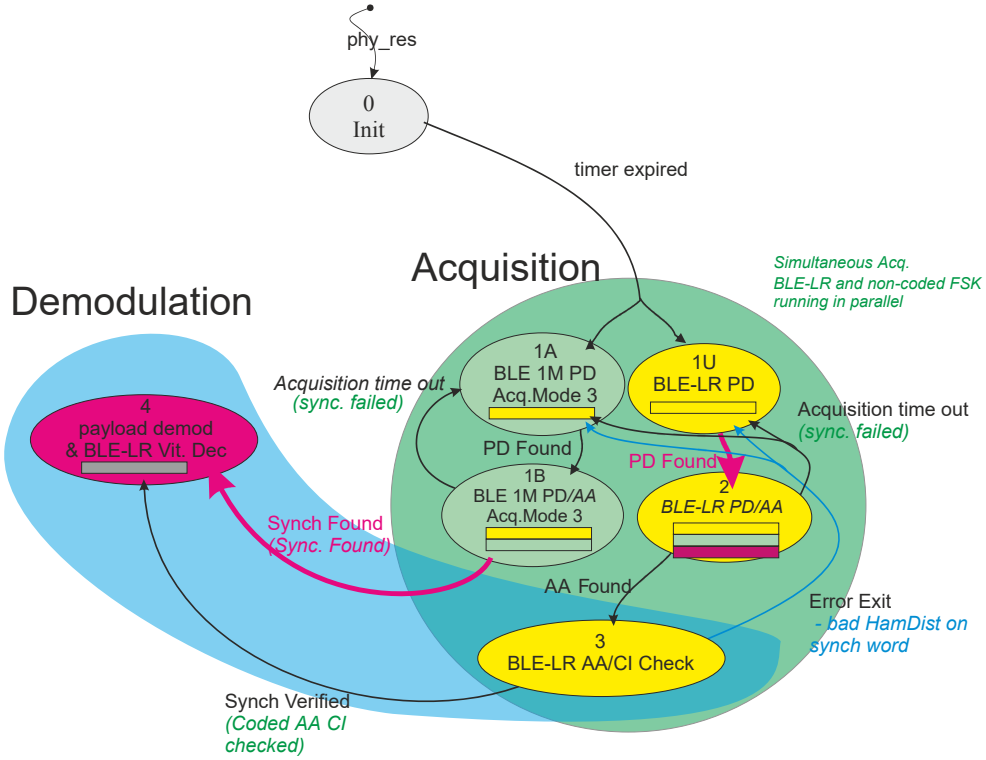


Figure 312. Nonsimultaneous search state-machine instantiations

The simultaneous search mode is intended for Bluetooth LE (non-coded or long range) packets. Since the non-coded Bluetooth LE packets have a one octet long preamble, only non-coded acquisition modes 2 and 3 are suitable for the simultaneous search. Figure 313 illustrates the instantiation of the state-machine for simultaneous search of long range Bluetooth LE and non-coded Bluetooth LE 1Mbps in acquisition modes 2 and 3.



(a) Simultaneous BLE-LR and non-coded BLE 1Mbps using acquisition mode 2



(b) Simultaneous BLE-LR and non-coded BLE 1Mbps using acquisition mode 3

Figure 313. Simultaneous search state-machine instantiations

Another aspect of the PHY state machine is the fast antenna diversity (FAD) preamble search feature. FAD is a mechanism that allows antenna selection on the fly. Two antennas and a single receiver chain are used, thus, the receiving chain is used in a time-shared manner. If a detection occurs, the current antenna is immediately selected. FAD involves using the preamble detection mechanism, i.e., one of the two antennas is selected when a preamble is first found. During the FAD process the receiver has to deal with blind spots due to transient behaviour of the different blocks on the receiver chain, so, the preamble must be fairly long to be able to support FAD. The minimal preamble length needed to support FAD depends on a series of factors:

- Receiver latency (without dynamic processes)
- Time required for dynamic control processes (in the sequel AGC only is considered)
- Number of samples needed by the preamble detector to reliably provide detection, timing and CFO estimate

From simulation it has been concluded that the preamble has to be at least 4 octets long. Due to the short preamble portions available for correlation, the FAD feature can only use the preamble detection engine used by acquisition mode 3 in a non-simultaneous search. Note that, the FAD state machine is coded in state 1A, in Figure 312, subfigure (c). A more detailed illustration of the FAD state-machine is provided in Figure 314.

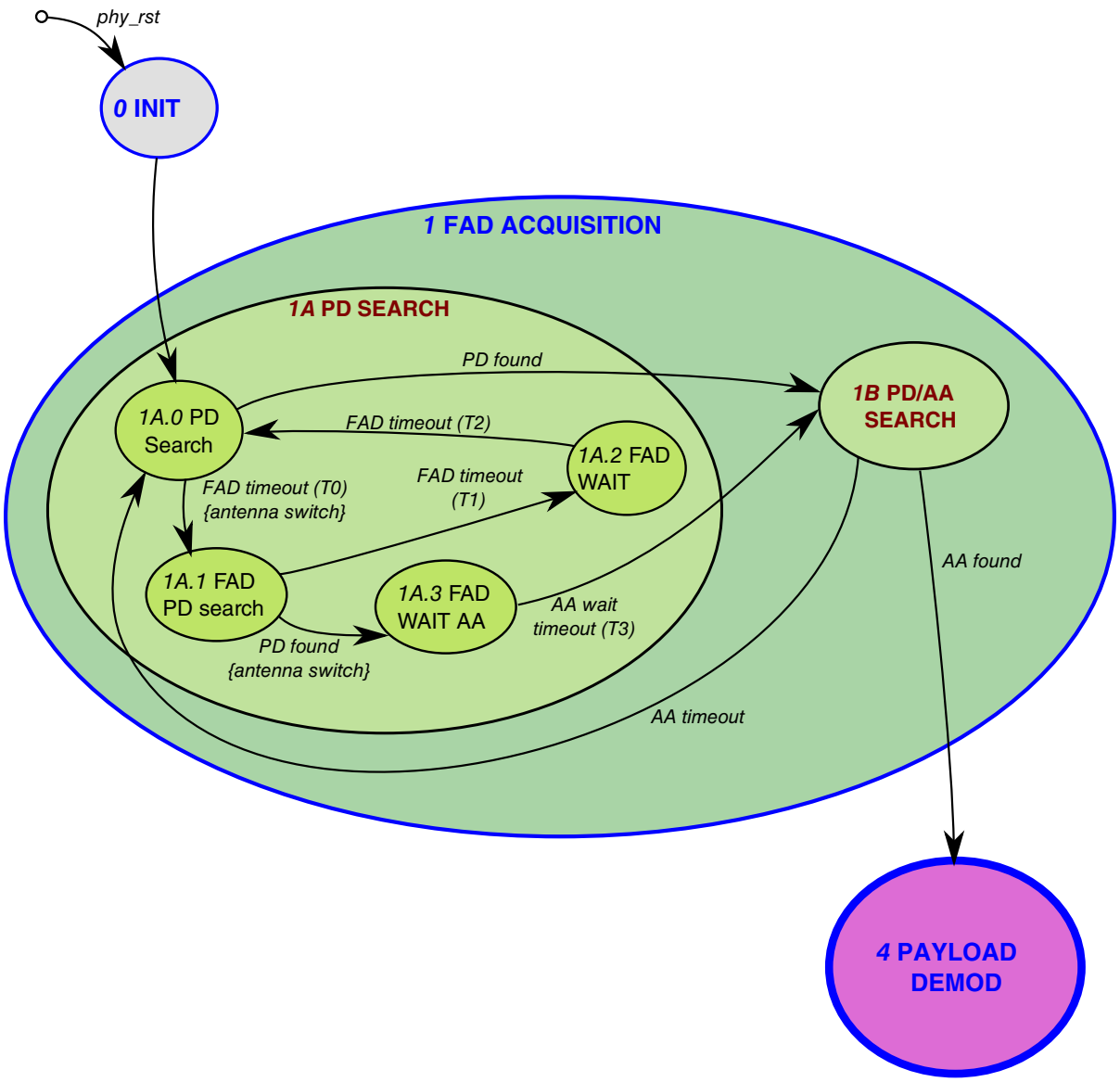


Figure 314. FAD detailed state-machine

The FAD states are:

- FAD sub-state 0 (PD Search)
  - Preamble detection over samples received from the currently selected antenna
    - Note that when the FAD is disabled in acquisition mode 3 this is the only attainable sub-state in "PD search"
- FAD sub-state 1 (FAD PD search)
  - Preamble detection over samples received from the previously selected antenna
    - Note that this is possible due to the propagation latency in the receiver
- FAD sub-state 2 (FAD wait)
  - The state machine waits for transients if a preamble pattern was not detected after antenna switch
- FAD sub-state 3 (FAD wait AA)
  - Before transitioning to AA search PHY state
  - The state machine waits for transients if a preamble pattern was detected after antenna switch

Time constants T_0, T_1 and T_2 from Figure 314 represent the dwell durations the FAD states having the same index and are illustrated in the time diagram from Figure 315. These values are configurable from registers. Figure 315 does not show T_3 because the antenna is not switched after the preamble is found so transition to the PHY state 1B occurs from FAD sub-state 0.

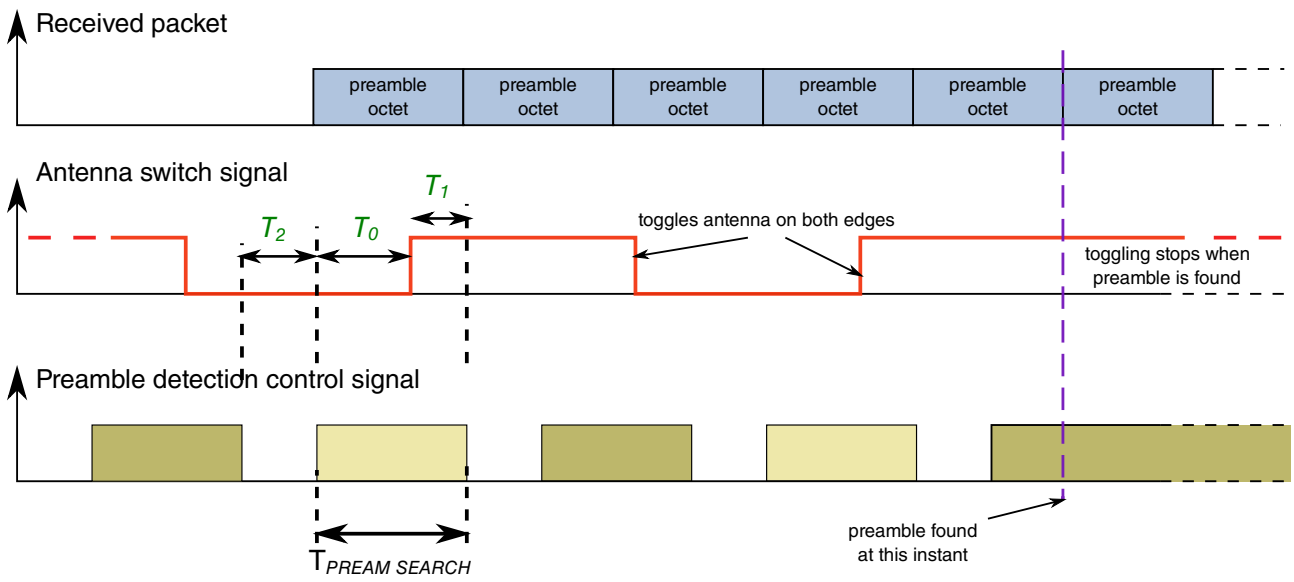


Figure 315. FAD process time diagram

Table 435 presents the recommended settings for FAD.

Table 435. Recommended parameters for FAD

Parameter name	Value
search_pd_window[6:0]	32
fad_search_pd_window[6:0]	12
fad_wait_pd_window[6:0]	12
fad_wait_synch_window[6:0]	12

55.4.7.6.4.3.2 Timing and frequency synchronization

55.4.7.6.4.3.2.1 General description

Figure 316 illustrates the block diagram of the timing/frequency synchronization (acquisition) block. It may be configured to operate in any one of the following three operation modes:

1. Acquisition for non-coded use cases only
2. Acquisition for long range Bluetooth LE only
3. Simultaneous acquisition for non-coded and long range Bluetooth LE use cases

The main inputs to the block are:

1. Wideband phase signal - phase domain signal not compensated for CFO
2. Wideband IQ signal - IQ domain signal not compensated for CFO
3. Narrowband phase signal: phase domain signal compensated for CFO and filtered using the channel filter

Acquisition for non-coded and longe range Bluetooth LE use cases is handled by separate blocks. The *Power Threshold* block provides two outputs that are used by these blocks to:

1. Disqualify any detection events that may occur when the input signal level is below a specified level because they are deemed unreliable
2. Enable Hamming distance check during AA detection when the input signal level is above a specified threshold

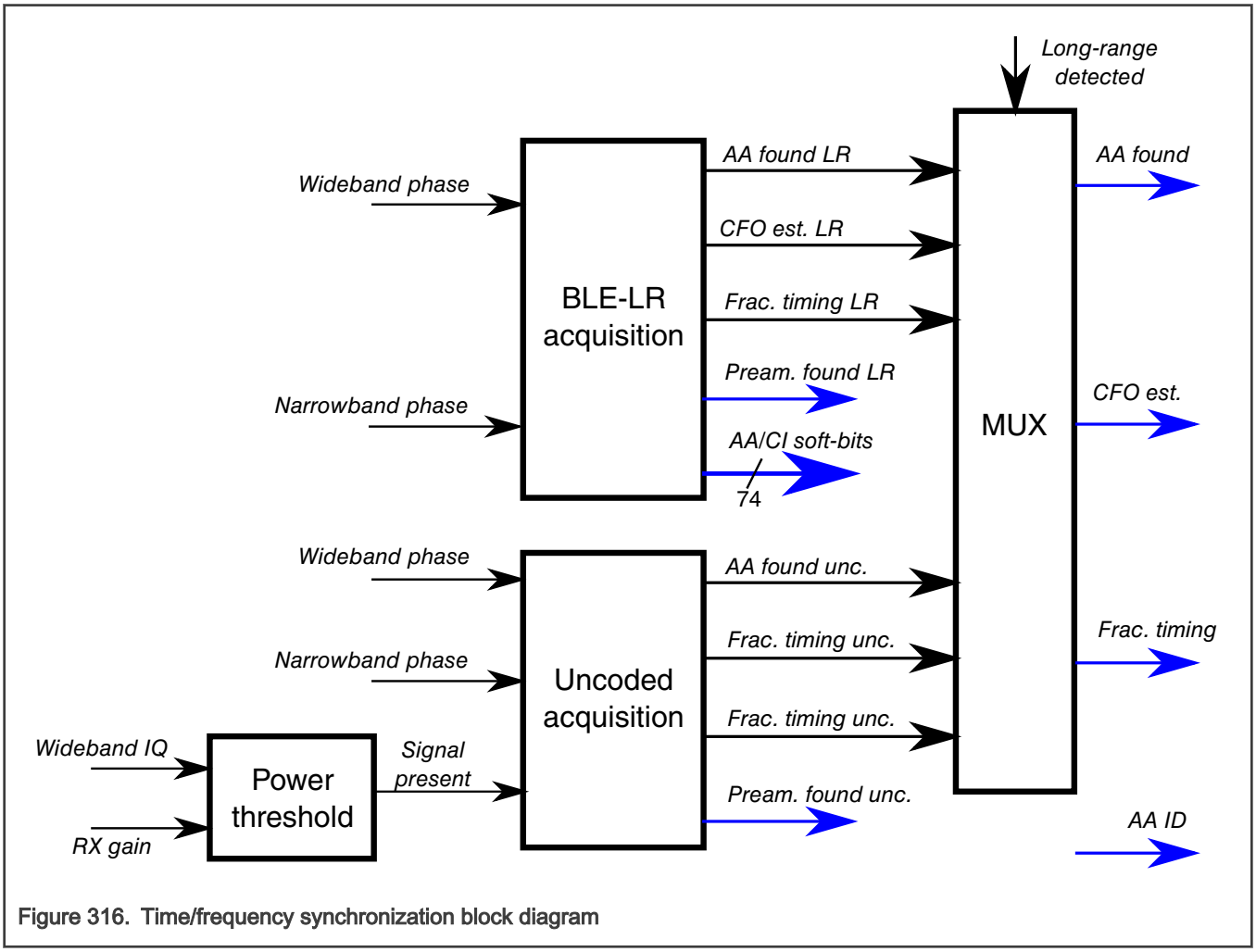


Figure 316. Time/frequency synchronization block diagram

Acquisition for non-coded and longe range Bluetooth LE use cases is described later in this section.

55.4.7.6.4.3.2.2 Acquisition for non-coded use cases

55.4.7.6.4.3.2.2.1 Functional Description

Figure 317 illustrates the three main processing stages involved in the processing of non-coded use cases.

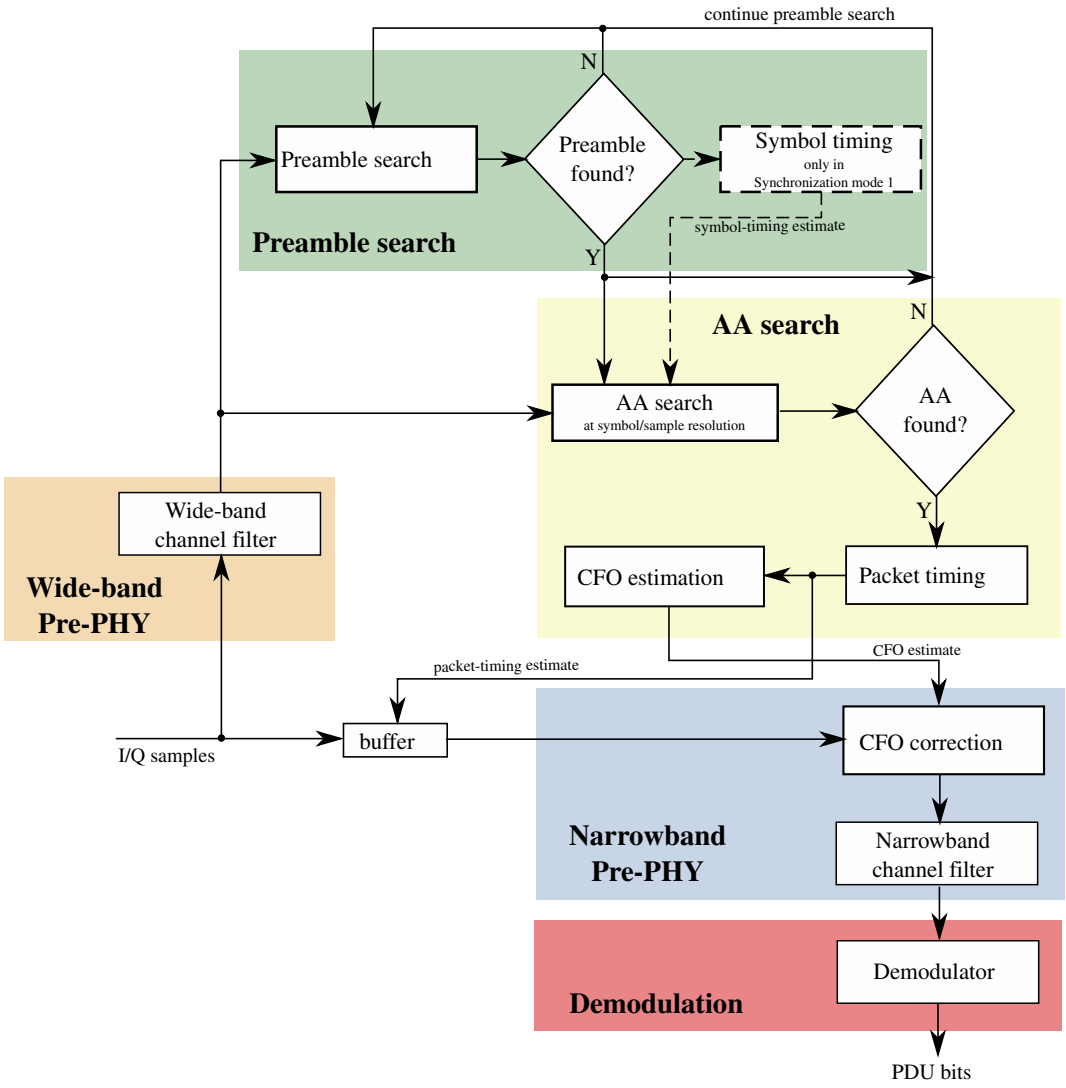


Figure 317. Processing stages of non-coded use cases

55.4.7.6.4.3.2.2.1.1 Preamble detection

During the preamble detection stage, the input phase values are processed to detect the preamble pattern and, if detected, AA search is triggered. No frequency acquisition is performed during this step but a timing estimate may be produced. The process shall continue even after starting AA search and is only terminated when AA is found or if *phy_enable* is de-asserted by the MAC.

55.4.7.6.4.3.2.2.1.2 AA search

During AA search, the PHY searches for one or more AA's to (i) achieve packet timing synchronization, and (ii) produce a CFO estimate. If the preamble detection stage is also active, its symbol timing estimate may be used during this stage to reduce the search complexity. The receiver can simultaneously search for up to four unique AA's of the same size. The number of AA's to be searched and their patterns need to be programmed *a priori* by the MAC.

This stage gets terminated upon either (i) a successful AA detection and synchronization following which the PHY proceeds to the demodulation stage, or (ii) if the *phy_enable* signal is de-asserted by the MAC.

The quality of the timing and CFO estimates obtained from AA search is strongly dependent on the correlation properties of the AA pattern used. So it is important to choose an AA with good correlation properties; refer to AA requirements & guidelines for related notes.

55.4.7.6.4.3.2.2.1.3 Demodulation

During the demodulation stage, using the timing & frequency estimates from the preceding stages, the PHY begins to demodulate the PDU and produces bits which are then sent to the MAC for further processing. In case there is significant timing and/or frequency drift in the PDU, the demodulator can simultaneously perform timing and/or frequency tracking as well.

The demodulator continues its operation until the *phy_enable* signal is de-asserted by the MAC.

55.4.7.6.4.3.2.2.2 AA requirements & guidelines

To ensure good synchronization performance, it is important that the AA used during the connection possesses good auto-correlation and cross-correlation properties. In general terms, if the auto-correlation of a candidate AA pattern, for a non-zero lag, exceeds a pre-defined threshold, then it should not be used. A suggested threshold value is 0.40 but recommended to be no more than 0.50.

For AA's of sizes 3 and 4 octets and For AA's of sizes 1 and 2 octets list some requirements which will reduce the likelihood of generating a candidate AA with poor auto-correlation properties.

55.4.7.6.4.3.2.2.2.1 For AA's of sizes 3 and 4 octets

The following requirements applicable to AA's of sizes 3 and 4 octets are adopted from Bluetooth specification v5.2; vol 6, part B.

The AA shall:

1. have no more than six consecutive 0's or 1's.
2. not be the advertising channel packet's AA.
3. not be a sequence that differs from the advertising channel packet's AA by only one bit.
4. not have all octets identical.
5. have no more than:
   - twenty-four bit transitions in case of 4-octet AA's.
   - eighteen bit transitions in case of 3-octet AA's.
6. have a minimum of two transitions in the most significant six bits.

The following additional requirements ensure better synchronization performance. Each AA shall:

1. contain no more than two contiguous instances of a bit sequence of length seven or more bits.
2. not contain two instances of a bit sequence, contiguous or otherwise, of length
   - thirteen or more bits in case of 4-octet AA's.
   - ten or more bits in case of 3-octet AA's.
3. have no more than four consecutive 0's or 1's within the least significant octet.
4. have no more than 6 transitions in the least significant octet.
5. have at least one transition in the most significant four bits.

55.4.7.6.4.3.2.2.2.2 For AA's of sizes 1 and 2 octets

Below are the requirements applicable to AA's of sizes 1 and 2 octets. Each AA shall:

1. have no more than four consecutive 0's or 1's.
2. not be the advertising channel packet's AA.
3. not be a sequence that differs from the advertising channel packet's AA by only one bit.

4. in case of 2 octets AA's, the two octets shall not be identical.
5. have no more than:
   - eleven bit transitions in case of 2-octet AA's.
   - five bit transitions in case of 1-octet AA's.
6. have a minimum of two transitions in the most significant four bits.
7. contain no more than two contiguous instances of a bit sequence of length five or more bits (not applicable to 1-octet AA's).
8. not contain two instances of a bit sequence, contiguous or otherwise, of length:
   - six or more bits in case of 2-octet AA's.
   - four bits in case of 3-octet AA's.
9. have no more than three consecutive 0's or 1's within the least significant octet.
10. no more than 6 transitions in the least significant octet.

55.4.7.6.4.3.2.2.3 Synchronization for non-coded use cases

For non-coded use cases, three different modes of synchronization are provided to support the general packet structure. Each mode involves one or two of the following three processes.

1. Correlation-based preamble detection in frequency-domain (Freq-PD)
2. Correlation-based preamble detection in phase-domain (Phase-PD)
3. Correlation-based AA detection (AAD)

55.4.7.6.4.3.2.2.3.1 Synchronization modes - functional description, state machines and implementation

55.4.7.6.4.3.2.2.3.1.1 Synchronization mode 1

In ACQ mode 1, the Freq-PD and AAD processes are used to achieve synchronization. This mode is only useful, and is especially recommended, when a long preamble is available because it can significantly reduce power consumption.

55.4.7.6.4.3.2.2.3.1.1.1 Functional description

Figure 318 shows a functional block diagram of the PHY in mode 1. First, the preamble is searched in the frequency-domain (*i.e.*, using phase-discriminated inputs) and when detected, a symbol-timing estimate is produced. The symbol-timing estimate is then used to begin AA search at symbol resolution thereby achieving significant reduction in computational complexity compared to searching the AA at sample resolution. Upon detecting an AA, an AA-based CFO estimate and a packet-timing estimate are generated.

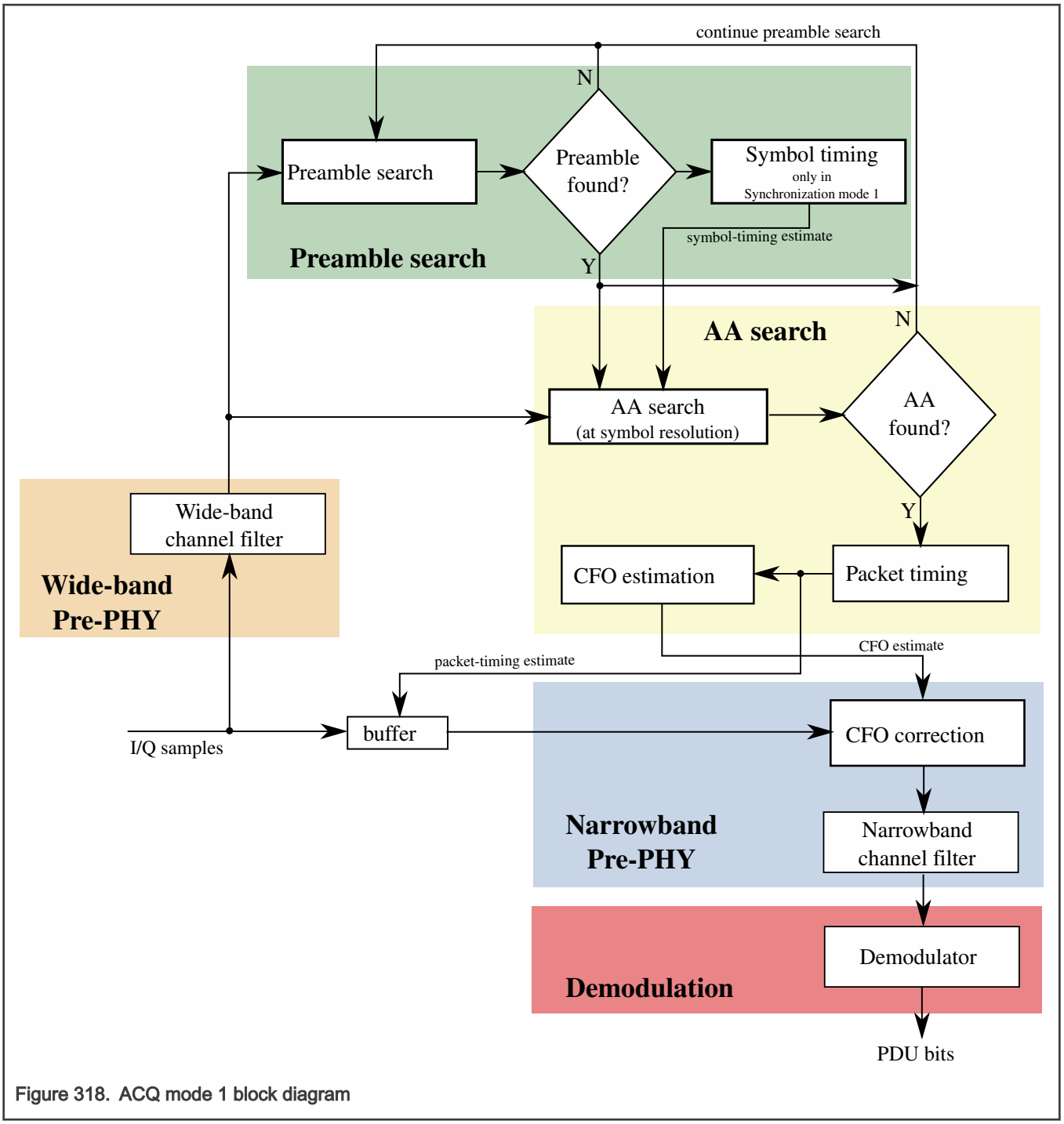


Figure 318. ACQ mode 1 block diagram

55.4.7.6.4.3.2.2.3.1.1.2 State machine

Figure 312 shows the state-machine of the PHY operating in mode 1.

55.4.7.6.4.3.2.2.3.1.2 Synchronization mode 2

In ACQ mode 2, only the AAD process is used to achieve synchronization. This ACQ mode may be used regardless of the length of the preamble but is recommended only when the preamble is short.

55.4.7.6.4.3.2.2.3.1.2.1 Functional description

Figure 319 shows a functional block diagram of the PHY in ACQ mode 2. The receiver begins searching for the AA and, upon detection, produces packet-timing and CFO estimates.

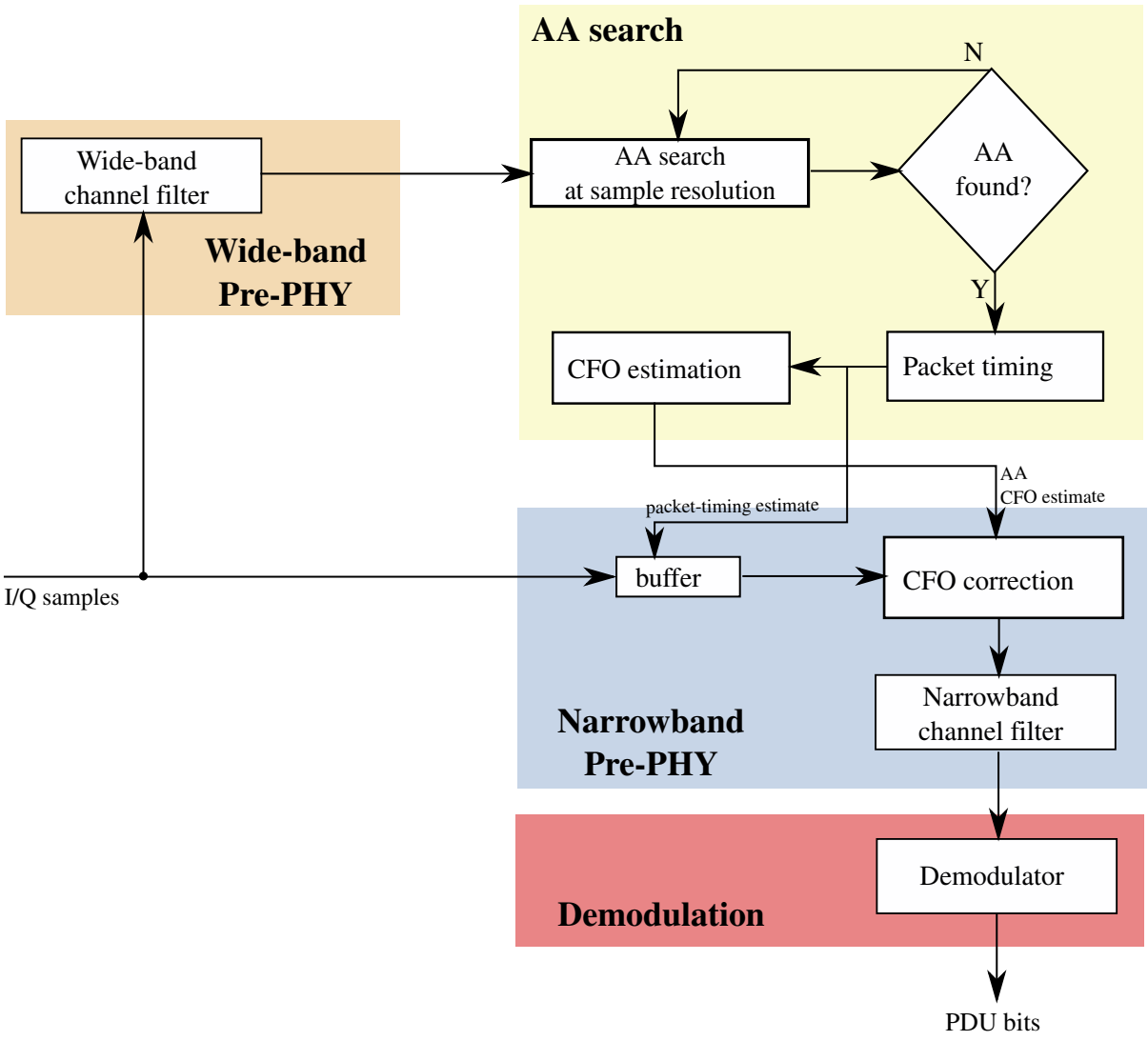


Figure 319. ACQ mode 2 block diagram

55.4.7.6.4.3.2.2.3.1.2.2 State machine

Figure 312 shows the state-machine of the PHY operating in mode 2.

55.4.7.6.4.3.2.2.3.1.3 Synchronization mode 3

In ACQ mode 3, the Phase-PD and AAD processes are used to achieve synchronization. But , unlike ACQ mode 1, this mode can be used when the preamble is short to quickly detect the arrival of a packet.

55.4.7.6.4.3.2.2.3.1.3.1 Functional description

Figure 320 shows a functional block diagram of the PHY in ACQ mode 3. First, the preamble is searched to identify the presence of a packet on the air but no timing or CFO estimate is generated. Like ACQ mode 2, the AA is searched at sample resolution and upon detecting an AA, an AA-based CFO estimate and a packet-timing estimate are generated.

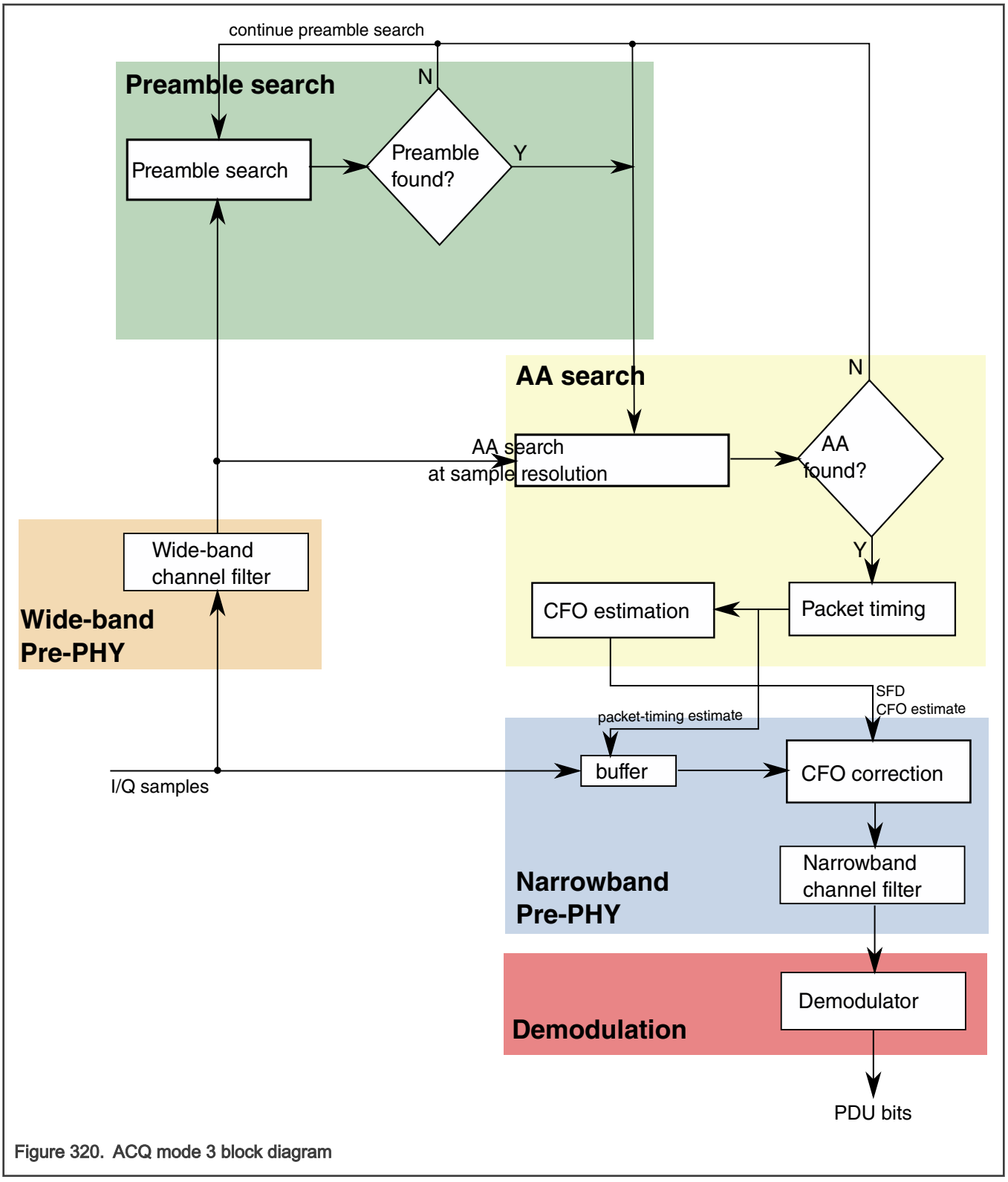


Figure 320. ACQ mode 3 block diagram

55.4.7.6.4.3.2.2.3.1.3.2 State machine

Figure 312 shows the state-machine of the PHY operating in mode 3.

55.4.7.6.4.3.2.2.3.2 Main component blocks

This section presents the details of the main component blocks used as the building blocks for realizing the various ACQ modes. There are six of them as listed below.

1. Power threshold
2. Pre-processor
3. Freq-PD preamble search engine
4. Phase-PD preamble search engine
5. AA search engine
6. Interpolator

55.4.7.6.4.3.2.2.3.2.1 Power threshold

The purpose of this block is to help avoid false preamble/AA detections in the respective search engines when the input signal is purely noise. The fixed-point formats of the block's inputs as well as those used at each internal stage are shown within parentheses.

It measures the input signal magnitude by passing the received I/Q samples through a single-pole IIR filter whose forgetting factor, α, is programmable. This magnitude is compared against two input thresholds - a low power threshold and a high power threshold which are both expected to be adjusted based on the AGC gain setting and the associated NF. If the measured signal magnitude is not less than the low threshold, then the boolean output *signal_present_low* gets asserted; if it is also not less than the high threshold, then both the boolean outputs, *signal_present_low* and *signal_present_high*, get asserted.

The block may alternatively use the narrowband RSSI measured by the RxDIG instead of the signal magnitude computed internally for comparing against the low and high power thresholds. This behavior is configurable through the input parameter BypassWithRSSI.

55.4.7.6.4.3.2.2.3.2.1.1 Register parameters

Table 436 lists the register parameters used by the power threshold block.

Table 436. Registers used by the power threshold block

Parameter	Notes
mag_window[3:0]	Indicates the forgetting factor used in received signal level measurement; forgetting factor used is 2^{-mag_window}. Note that *mag_window* values higher than 8 are not typically used.
pow_th_low[10:0]	Low power threshold specifies a power level below which the signal is considered too weak for reliable aquisition; uses (0,11,11) fixed-point format.
pow_th_high[10:0]	High power threshold specifies a power level above which the signal is considered strong enough that a stricter Hamming tolerance may be applied during acquisition; uses (0,11,11) fixed-point format.
bypass_with_rssi	Bypass signal power measurement with RSSI measurement; 0→no, 1→yes.
cond_sig_prst_enable	Enables special conditioning of signal power detection in the Power threshold block; 0→disable, 1→enable.

55.4.7.6.4.3.2.2.3.2.2 Pre-processor

It is used as the first step, in both Freq-PD and AAD processes, to pre-process the input I/Q samples and its output is then passed on to the respective search block.

55.4.7.6.4.3.2.2.3.2.2.1 Register parameters

Table 437 lists the only register parameter used by the pre-processor. It is used to obtain *1/2h* in the appropriate fixed-point format for providing as input to the pre-processor.

Table 437. Registers used by the pre-processor

Parameter	Notes
recip_mod_idx[8:0]	Reciprocal of modulation index; represented in (0,9,7) format.

55.4.7.6.4.3.2.2.3.2.3 Freq-PD preamble search engine

A block diagram of the Freq-PD preamble search engine is shown in Figure 321. It is used for correlation-based preamble detection in the frequency-domain. The fixed-point formats used at various internal points are also shown within parentheses.

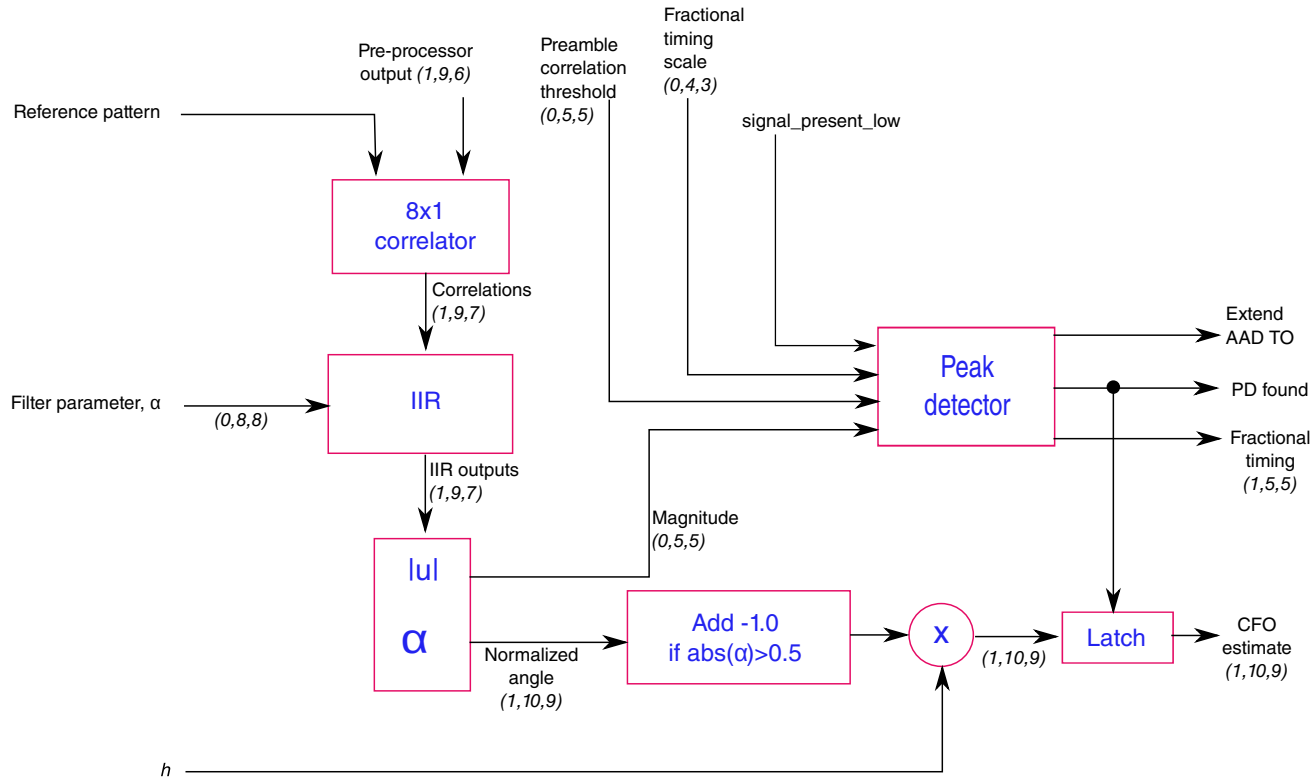


Figure 321. Freq-PD preamble search engine

An 8x1 binary reference pattern along with the pre-processor output phase values is provided as input to the engine. In the first step, each phase value is converted into its equivalent I/Q sample of unit magnitude and a correlation is performed between the I/Q values thus obtained and the reference pattern. The resultant complex correlation values are then passed through a single-pole IIR filter. The outputs of the IIR filter are long-term averages of its inputs and, therefore, are roughly equivalent to much longer correlations. Next, the magnitudes and angles of the complex IIR outputs are obtained using LUTs and passed to the timing detection and CFO estimation circuits respectively.

55.4.7.6.4.3.2.2.3.2.3.1 Register parameters

Table 438 lists the register parameters used by the preamble search engine.

Table 438. Registers used by the Freq-PD preamble search engine

Parameter	Notes
preamble_pattern[7:0]	8-bit preamble pattern used in FM-domain preamble detector.

Table continues on the next page...

Table 438. Registers used by the Freq-PD preamble search engine (continued)

Parameter	Notes
preamble_thresh[7:0]	Correlation threshold applicable to preamble detection; uses (0,5,5) fixed-point format which gets written in (0,8,8) format by zeroing out the LSB's.
preamble_frac_time_scale[3:0]	Scaling factor used for fractional time estimation during preamble search; represented in (0,4,3) format.
iir_alpha[7:0]	IIR filter parameter used during preamble detection; represented in (0,8,8) format.
mod_idx[10:0]	Modulation index; represented in (0,11,7) format.

Table 439 lists a suggested preamble correlation threshold value for each non-coded use case listed in Table 434. The values are applicable to ACQ mode 1 but it should be noted that they were optimized for the specific preamble length of 8 octets and, therefore, may need to be adjusted if a different preamble length is used.

Table 439. Suggested Freq-PD correlation thresholds for ACQ mode 1

Use case	Preamble threshold (fraction)	*preamble_thresh[7:0]* (hexadecimal notation)
Bluetooth LE	12/32	0x60
GFSK_05_07-500	14/32	0x70
GFSK_03_05-1000	12/32	0x60
MSK_05-500	10/32	0x50
GFSK_05_05-250	12/32	0x60
GFSK_05_10-250	16/32	0x80
GFSK_07_05-1000	16/32	0x80
GFSK_05_05-2000	16/32	0x80
MSK_05-250	14/32	0x70
GFSK_05_032-2000	16/32	0x80
FSK_05-500	10/32	0x50

Table 440 lists a suggested value for *preamble_frac_time_scale* applicable to all non_coded use cases. The length of the preamble does not influence this parameter and so the suggested value is applicable regardless of the preamble length.

Table 440. Suggested Freq-PD fractional timing scale for ACQ mode 1

Use case	Preamble frational timing scale (decimal)	*preamble_frac_time_scale[3:0]* (hexadecimal notation)
All use cases	1.50	0xC

Table 441 lists a suggested value for *iir_alpha* for each non-coded use case listed in Table 434. It was optimized for a preamble length of 8 octets and may need to be adjusted if the preamble length is different.

Table 441. Suggested Freq-PD IIR forgetting factor for ACQ mode 1

Use case	IIR parameter, α (fraction)	*iir_alpha[7:0]* (hexadecimal notation)
Bluetooth LE	6/32	0x30
GFSK_05_07-500	6/32	0x30
GFSK_03_05-1000	4/32	0x20
MSK_05-500	3/32	0x18
GFSK_05_05-250	6/32	0x30
GFSK_05_10-250	4/32	0x20
GFSK_07_05-1000	6/32	0x30
GFSK_05_05-2000	6/32	0x30
MSK_05-250	6/32	0x30
GFSK_05_032-2000	6/32	0x30
FSK_05-500	3/32	0x18

55.4.7.6.4.3.2.2.3.2.4 Phase-PD preamble search engine

A block diagram of the Phase-PD preamble search engine is shown in Figure 322. It is used for correlation-based preamble detection in the phase-domain. The fixed-point formats used at various internal points are also shown in the diagram.

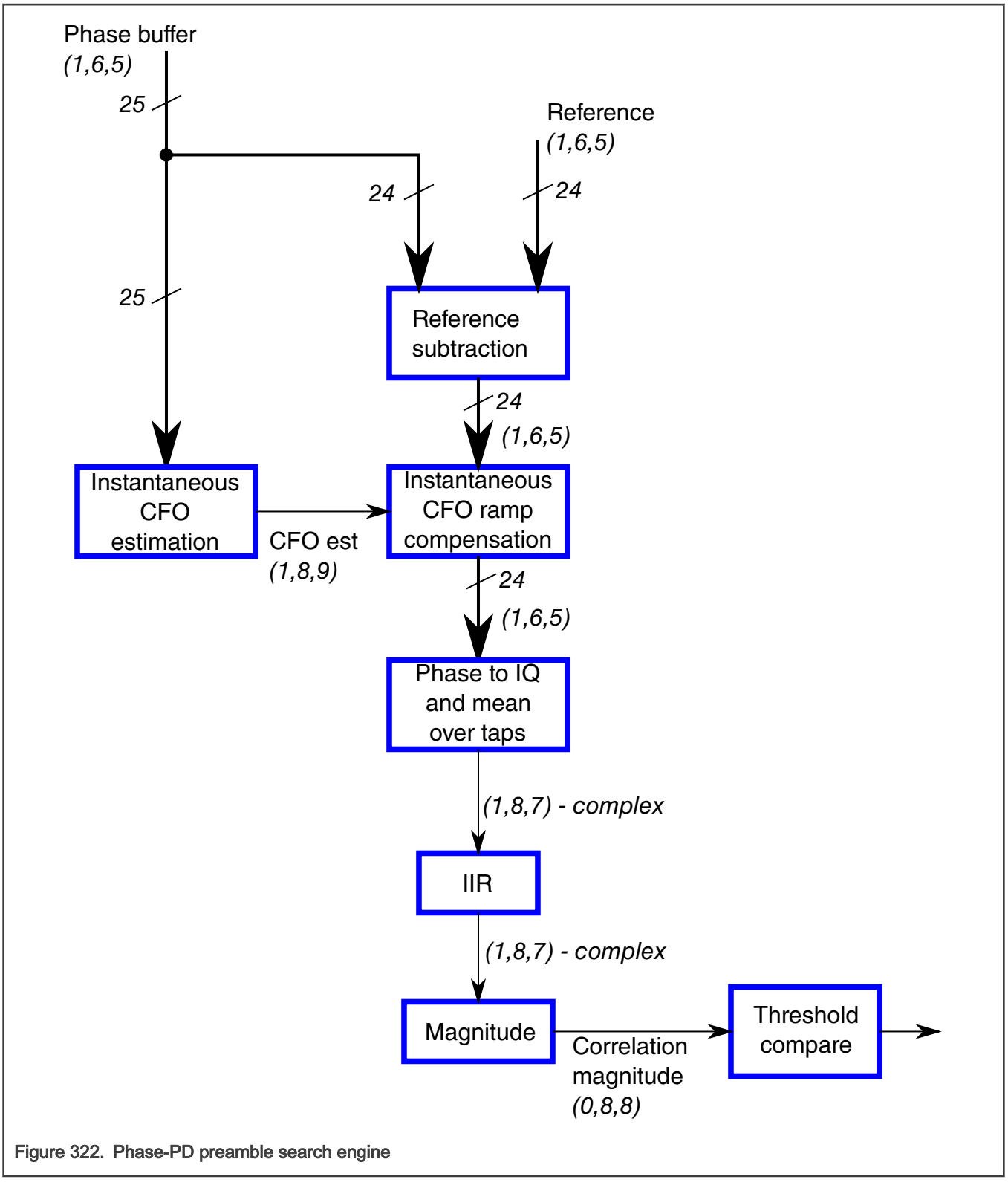


Figure 322. Phase-PD preamble search engine

An 24x1 phase-domain reference pattern along with the pre-processor output phase values is provided as input to the engine. Since the 24x1 vector is periodic with an 8x1 vector repeating three times, only 8 values need to be programmed, as shown in Table 442, in order to save register space. To be able to properly perform phase-domain correlation the phase ramp caused by CFO has to be compensated. In order to achieve that, an instantaneous CFO estimate is computed, by averaging differences between consecutive values in the phase buffer, and all correlation taps are simultaneously compensated by reconstructing the

ramp based upon the estimated CFO. As shown in Figure 322, the CFO is estimated on the input phase vector after the reference is subtracted. After the CFO compensation the phase values are converted to complex values and the taps are averaged over, thus transforming the vector input to a scalar ouput. An IIR can be optionally used, to average the complex correlation values, if preamble is longer. However it is expected that, for the use cases that use acquisition mode 3, the IIR will be bypassed.

Finally the complex correlation values are converted to real values through absolute operation ("Magitude" block) and compared with a threshold. Given the intrinsic behavior of the phase-domain correlator to output a large correlation magnitude when the buffer is filled with zeros (this happens when the preable detector is being reset), a timer is used to gate the output of the block untill the buffer gets filled with non-zeros samples.

55.4.7.6.4.3.2.2.3.2.4.1 Register parameters

Table 442 lists the register parameters used by the preamble search engine.

Table 442. Registers used by the Phase-PD search engine

Parameter	Notes
preamble_pattern[47:0]	8 reference phase values, 6-bits each; uses (1,6,5) fixed-point format.
preamble_thresh[7:0]	Correlation threshold applicable to preamble detection; uses (0,8,8) fixed-point format.
iir_alpha[7:0]	IIR filter parameter used during preamble detection; represented in (0,8,8) format.

Table 439 lists a suggested preamble correlation threshold value for each non-coded use case listed in Table 434.

Table 443. Suggested Phase-PD preamble correlation thresholds for ACQ mode 3

Use case	Preamble threshold (fraction)	*preamble_thresh[7:0]* (hexadecimal notation)
Bluetooth LE	246/256	0xF6
GFSK_05_07-500	215/256	0xD7
GFSK_03_05-1000	246/256	0xF6
MSK_05-500	240/256	0xF0
GFSK_05_05-250	235/256	0xEB
GFSK_05_10-250	215/256	0xD7
GFSK_07_05-1000	235/256	0xEB
GFSK_05_05-2000	235/256	0xEB
MSK_05-250	184/256	0xB8
GFSK_05_032-2000	246/256	0xF6
FSK_05-500	240/256	0xF0

Table 444 lists a suggested value for *iir_alpha* applicable to all use cases that use acquisition mode 3.

Table 444. Suggested Phase-PD IIR forgetting factor for ACQ mode 3

Use case	IIR parameter, α (fraction)	*iir_alpha[7:0]* (hexadecimal notation)
All use cases	1 (bypass)	0xFF

Table 445. Reference waveforms for phase-domain PD

Use case	Phase domain reference pattern (*preamble_pattern[47:0]*) in hexadecimal notation
Bluetooth LE	[0x03, 0x05, 0x03, 0x00, 0x3d, 0x3b, 0x3d, 0x00];
GFSK_05_07-500	[0x05, 0x07, 0x05, 0x00, 0x3b, 0x39, 0x3b, 0x00];
GFSK_03_05-1000	[0x02, 0x03, 0x02, 0x00, 0x3e, 0x3d, 0x3e, 0x00]
MSK_05-500	[0x02, 0x03, 0x02, 0x00, 0x3e, 0x3d, 0x3e, 0x00]
GFSK_05_05-250	[0x03, 0x05, 0x03, 0x00, 0x3d, 0x3b, 0x3d, 0x00]
GFSK_05_10-250	[0x07, 0x0a, 0x07, 0x00, 0x39, 0x36, 0x39, 0x00]
GFSK_07_05-1000	[0x04, 0x06, 0x04, 0x00, 0x3c, 0x3a, 0x3c, 0x00]
GFSK_05_05-2000	[0x03, 0x05, 0x03, 0x00, 0x3d, 0x3b, 0x3d, 0x00]
MSK_05-250	[0x02, 0x03, 0x02, 0x00, 0x3e, 0x3d, 0x3e, 0x00]
GFSK_05_032-2000	[0x02, 0x03, 0x02, 0x00, 0x3e, 0x3d, 0x3e, 0x00]
FSK_05-500	[0x02, 0x03, 0x02, 0x00, 0x3e, 0x3d, 0x3e, 0x00]

55.4.7.6.4.3.2.2.3.2.5 AA search engine

A block diagram of the AA search engine is shown in Figure 323. It is used for correlation-based AA detection in AAD. The fixed-point formats used at various stages are also shown within parentheses in the diagram.

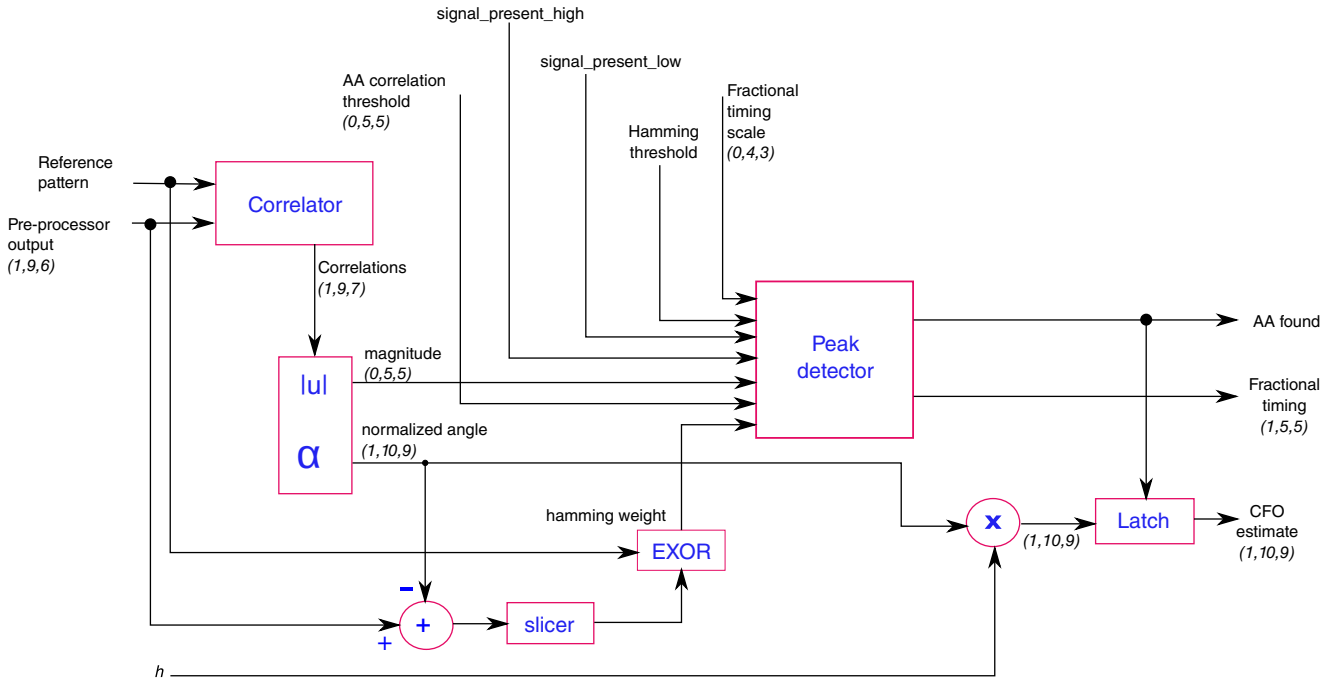


Figure 323. AA search engine

A binary AA reference pattern, along with the pre-processor output phase values, is provided as input to the engine. In the first step, each phase value is converted into its equivalent I/Q sample of unit magnitude and a correlation is performed between the I/Q values thus obtained and the reference pattern. Next, the magnitudes and angles of the complex correlations are obtained using LUTs and passed to the peak detector and CFO estimation circuits respectively. In addition, a hamming weight is also computed between

1. the reference pattern, and
2. the pre-processor output vector (corresponding to the current correlation) after removing the bias due to CFO.

55.4.7.6.4.3.2.2.3.2.5.1 Register parameters

Table 446 lists the register parameters used by the AA search engine.

Table 446. Registers used by the AA search engine

Parameter	Notes
aa_0[31:0]	AA pattern of length 1, 2, 3 or 4 octets used for AA search in ACQ path 0; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
aa_1[31:0]	AA pattern of length 1, 2, 3 or 4 octets used for AA search in ACQ path 1; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
aa_2[31:0]	AA pattern of length 1, 2, 3 or 4 octets used for AA search in ACQ path 2; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
aa_3[31:0]	AA pattern of length 1, 2, 3 or 4 octets used for AA search in ACQ path 3; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
size_aa[1:0]	Indicates length of active AA's; 0→1 octet, 1→2 octets, 2→3 octets, 3→4 octets.
aa_thresh[4:0]	Correlation threshold applicable to AA detection; uses (0,5,5) fixed-point format.
aa_frac_time_scale[3:0]	Scaling factor used for fractional time estimation during AA search; represented in (0,4,3) format.
search_aa_0	Indicates whether aa_0 should be searched during acquisition; 0→no, 1→yes.
search_aa_1	Indicates whether aa_1 should be searched during acquisition; 0→no, 1→yes.
search_aa_2	Indicates whether aa_2 should be searched during acquisition; 0→no, 1→yes.
search_aa_3	Indicates whether aa_3 should be searched during acquisition; 0→no, 1→yes.
hamming_aa_high_pwr[2:0]	Maximum hamming distance from the given AA pattern that may still be accepted as a match; valid range is [0,7].
hamming_aa_low_pwr[3:0]	Hamming distance tolerance applicable around sensitivity and at low power levels during AA detection; valid range is [0,15].
mod_idx[10:0]	Modulation index; represented in (0,11,7) format.

Table 447 lists the suggested AA correlation threshold values for ACQ modes 2 & 3 while Table 448 lists suggested AA correlation threshold values for ACQ mode 1 for the non-coded use cases listed in Table 434. Note that they are not influenced by the length of the preamble.

Table 447. Suggested AA correlation thresholds for ACQ modes 2 & 3

Use case	AA threshold (fraction)	*aa_thresh[4:0]* (hexadecimal notation)
Bluetooth LE	19/32	0x13
GFSK_05_07-500	16/32	0x10
GFSK_03_05-1000	15/32	0x0F
MSK_05-500	18/32	0x12

Table continues on the next page...

Table 447. Suggested AA correlation thresholds for ACQ modes 2 & 3 (continued)

Use case	AA threshold (fraction)	*aa_thresh[4:0]* (hexadecimal notation)
GFSK_05_05-250	18/32	0x12
GFSK_05_10-250	19/32	0x13
GFSK_07_05-1000	15/32	0x0F
GFSK_05_05-2000	17/32	0x11
MSK_05-250	17/32	0x11
GFSK_05_032-2000	17/32	0x11
FSK_05-500	15/32	0x0F

Table 448. Suggested AA correlation thresholds for ACQ mode 1

Use case	AA threshold (fraction)	*aa_thresh[4:0]* (hexadecimal notation)
Bluetooth LE	17/32	0x11
GFSK_05_07-500	15/32	0x0F
GFSK_03_05-1000	15/32	0x0F
MSK_05-500	13/32	0x0D
GFSK_05_05-250	17/32	0x11
GFSK_05_10-250	17/32	0x11
GFSK_07_05-1000	17/32	0x01
GFSK_05_05-2000	17/32	0x11
MSK_05-250	17/32	0x11
GFSK_05_032-2000	17/32	0x11
FSK_05-500	13/32	0x0D

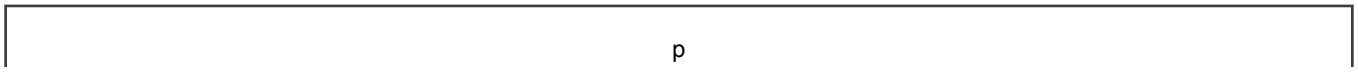
Table 449 lists a suggested value for *aa_frac_time_scale* applicable to all use cases regardless of the preamble length.

Table 449. Suggested AA fractional timing scale for ACQ modes 2 and 3

Use case	AA frational timing scale (decimal)	*aa_frac_time_scale[3:0]* (hexadecimal notation)
All use cases	1.875	0xF

55.4.7.6.4.3.2.2.3.2.6 Interpolator

A block diagram of the interpolator is shown in Figure 324. It is only used in ACQ mode 1 for interpolation in AAD. The fixed-point formats used at various internal stages are also shown within parentheses in the diagram.



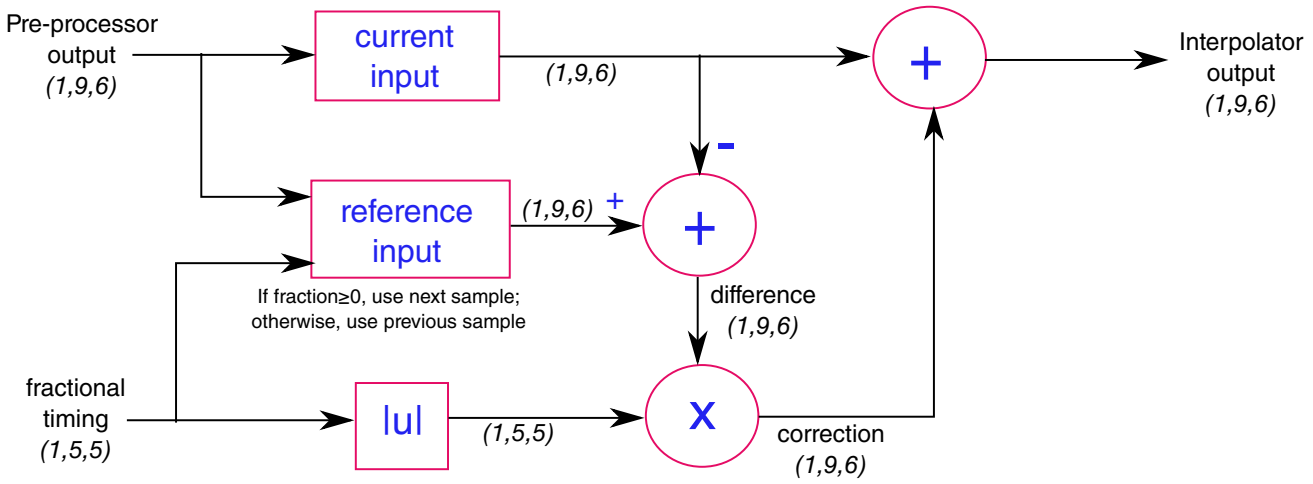


Figure 324. Interpolator

The interpolation in performed by using as reference the immediately preceding sample if the fractional timing estimated in PD is negative or the immediately succeeding sample if the fractional timing is non-negative. For interpolating each input sample, the difference between the itself and the reference is scaled by the fractional timing estimate and added back to itself.

55.4.7.6.4.3.2.3 Acquisition for BluetoothLE-LR

55.4.7.6.4.3.2.3.1 Longe range Bluetooth LE preamble detection

The purpose of the preamble detector is to detect the presence of a longe range Bluetooth LE packet and to estimate the CFO.

55.4.7.6.4.3.2.3.2 Long range Bluetooth LE AA detection

The soft-bits based AA detector performs AA detection for long range Bluetooth LE packets. It consists of the following components:

- Low complexity FSK symbol demodulator which produces the soft-bits, triggered at the sample rate
- Buffer which converts the scalar input to a vector used for correlation
- Long range Bluetooth LE AA correlator, used to perform long range Bluetooth LE AA detection after a long range Bluetooth LE preamble is detected

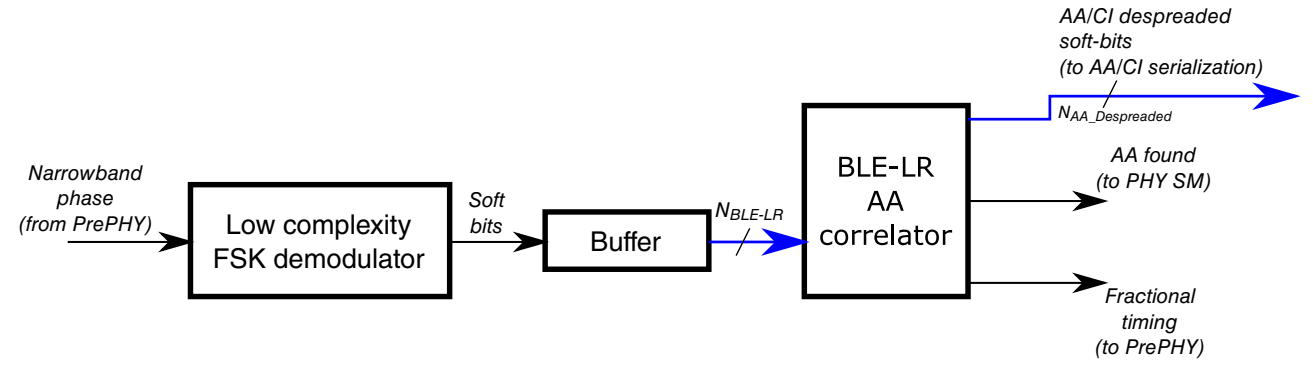


Figure 325. Soft-bits domain AA detector block diagram

It is being recalled here that the AA detection that uses soft-bits is done in parallel with continued preamble detection which has the purpose of improving the CFO estimate and extend the AA search time-out. These details are given in a more detailed manner in the PHY state-machine section. An important aspect to be mentioned is that the long range Bluetooth LE detector supports

detection of a single AA pattern. When AA detection starts, the low complexity FSK demodulator and the buffer are first enabled and the correlator is enabled after enough samples are acquired into the buffer to allow a valid correlation product.

55.4.7.6.4.3.2.3.2.1 Low-Complexity FSK Symbol Demodulation

The soft-bits are generated at the sample rate *i.e.*, $1/T_s$ and they are used during AA detection. The FSK symbol demodulator takes the CFO-corrected phase-domain samples as input and produces soft-bits as described below. Since the PHY has no knowledge of the PDU size, the process continues until the *activate_search* signal is de-asserted by the MAC.

55.4.7.6.4.3.2.3.2.2 Long range Bluetooth LE AA detection

The AA/CI despreaded soft-bits are latched when AA is found and they are further used by the long range Bluetooth LE AA/CI decoding mechanims, which is described in a different section.

In order to match the AA, the despreaded soft-bits are selected in the "Select AA portion" block and 64 despreaded soft bits are provided to the correlation and Hamming distance circuits. The correlator performs real values correlation, against AA corresponding reference waveform, over the input vector and the Hamming distance uses hard-bits to measure the distance between the received vector and the expected, coded AA.

55.4.7.6.4.3.3 Long range Bluetooth LE AA/CI decoding

The AA/CI decoding process is illustrated in Figure 326. This process corresponds to state 3 in the PHY state machine and involves both the PHY and the RBME modules. The chronological steps are indicated by the red arrow. Note that the arrow is meant to show data flow not control. The steps are:

- When AA is detected the detector provides a vector containing the despreaded soft-bits corresponding to the AA and CI portions
- The parallel to serial conversion block in the AA/CI decoding block starts producing codewords (consisting of the pairs of soft-bits {Soft-bit 1, Soft-bit 2}) and data valids
- The RBME engine decodes the inputted codewords and provides back data and data valids to the AA/CI detection block
- AA/CI detection is performing AA matching and CI extraction
  - If AA is matched the CI information is sent to the state-machine together with a CI valid indication
    - Following this event, the PHY state-machine transitions to payload demodulation
  - If AA is not matched the CI valid is not sent and a time-out occurs in the PHY state-machine
    - This further leads to returning to the acquisition state and to RBME disable

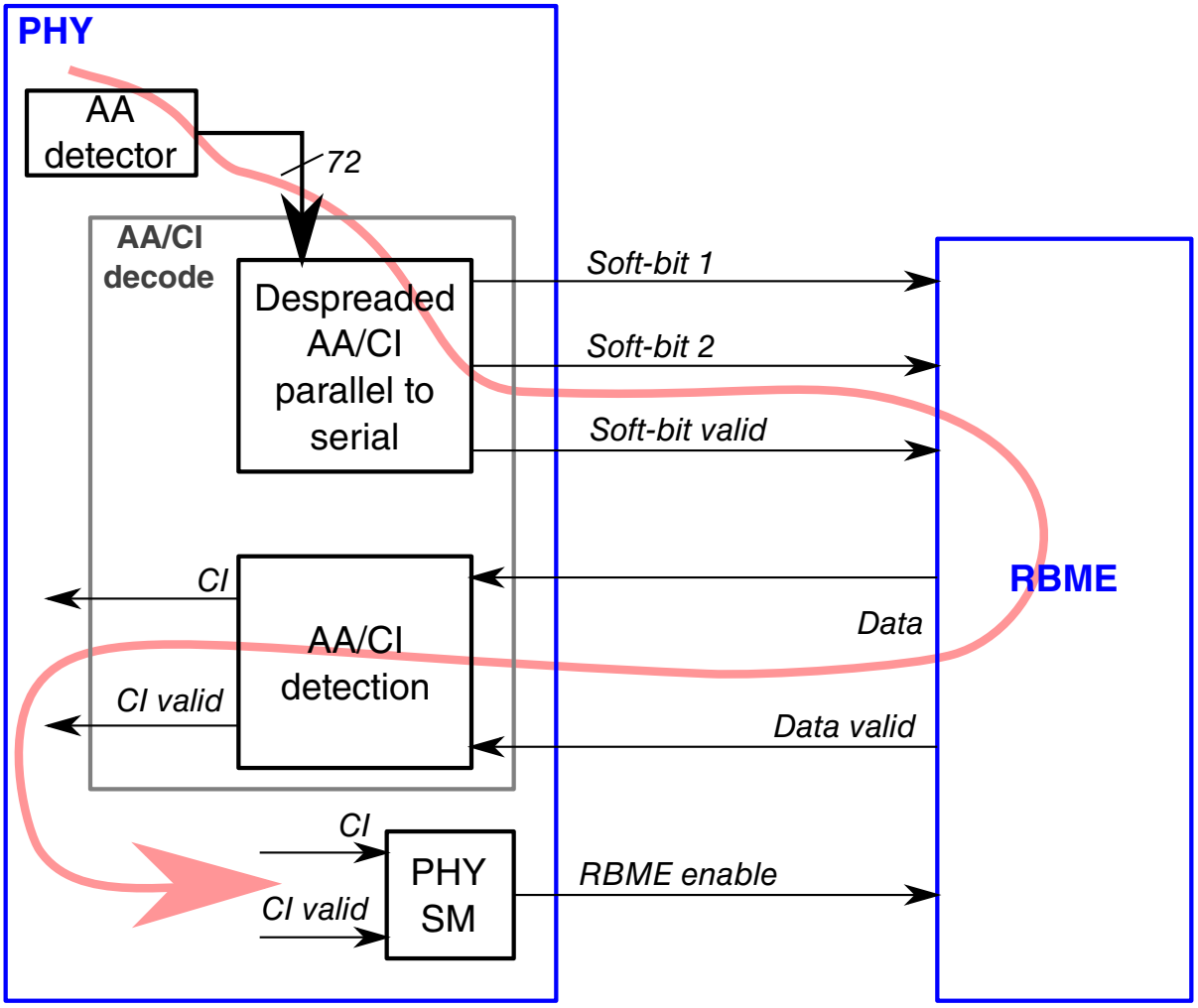


Figure 326. AA/CI decoding process

55.4.7.6.4.3.4 FSK payload demodulation

The demodulator consumes the received samples, at an oversampling rate (OSR) equal to 4, after the symbol timing and frame timing have been determined and after the CFO correction was applied. The I/Q samples are first converted to phase, by using the normalizer block.

The demodulator block supports:

- Symbol demodulation using input phase samples at OSR=4:
- Time tracking to compensate for residual timing error and for timing drift
  - Given that the timing drift needs to cover a various range of use-cases, from a maximum packet length perspective, the range is configurable via registers
- Frequency error detection and compensation to compensate for residual frequency error

Each supported feature will be detailed in the sequel of this section. Figure 327 shows the corresponding block diagram.

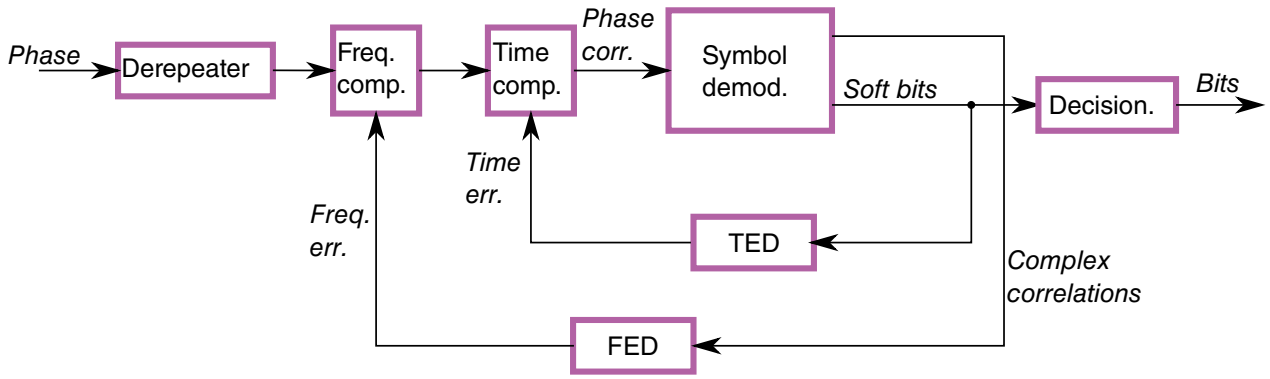


Figure 327. Demodulator block diagram

Symbol demodulation

The symbol demodulation mechanism will be referred to as M3c in the sequel. Figure 328 shows the block diagram of the symbol demodulator.

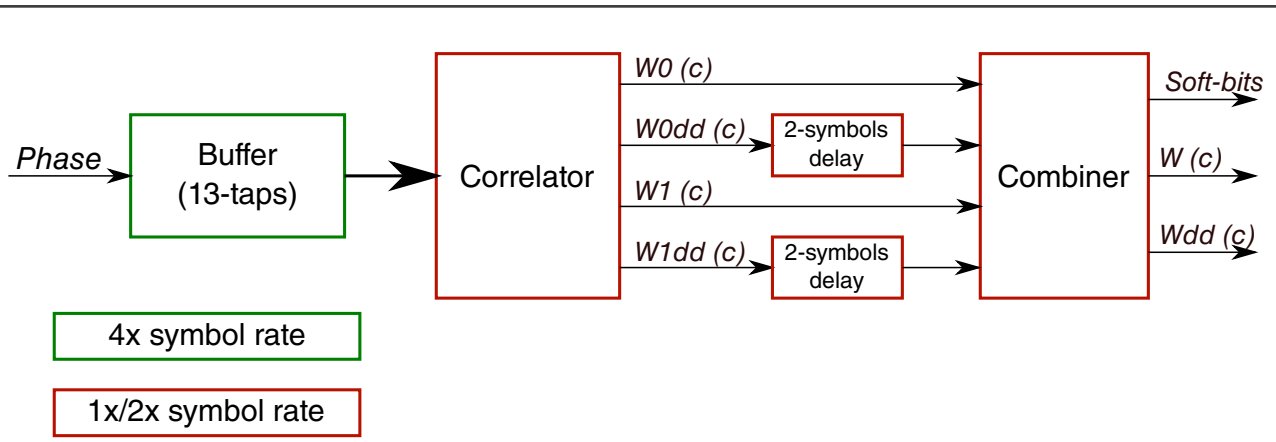


Figure 328. Symbol demodulator block diagram

From a correlation perspective, the symbol demodulator supports four modes:

- Mode 0: uses 12-taps (OSR in the adder tree is 4) and the reference phase vectors are shaped taking into consideration the modulation pulse and the receiver filtering
- Mode 1: uses 4-taps (OSR in the adder tree is 1) and no shaping is performed on the reference phase vectors
- Mode 2: uses 7-taps (OSR in the adder tree is 2) and no shaping is performed on the reference phase vectors
- Mode 4: uses 13-taps (OSR in the adder tree is 4) and no shaping is performed on the reference phase vectors

Some of the modes not only they provide a lower current drain in the correlation circuit but they can also provide a better performance.

Time/frequency error tracking

Following impairments need to be taken into consideration in the demodulator:

- Phase drift which is due to the residual carrier frequency offset, due to imperfect frequency synchronization in the acquisition block (initial frequency error)

- Crystal frequency drift which leads to carrier frequency offset accumulation over time (accumulated frequency error)
- Symbol timing residual error, due to imperfect time synchronization in the acquisition block (initial timing error)
- Timing drift due to misalignment of the frequency of the crystal at the transmitter and at the receiver (accumulated timing error)

The former two impairments are addressed through the frequency tracking mechanism whereas the latter two are handled by the timing tracking loop.

The frequency error tracking algorithm has two components: frequency error detection (FED) and frequency error correction (FED).

The most difficult use-case, from a frequency tracking perspective, is long range Bluetooth LE. This is because no frequency tracking is performed during the AA search. This means that there is a ~300 microseconds period between the moment when the CFO estimate is provided by the preamble detector and the start of the frequency tracking on the PDU portion. Long range Bluetooth LE 500 kbps is particularly affected due to the shorter duration of the informational symbol.

The timing error compensator provides:

1. integer compensation using a buffer
2. fractional compensation using a linear interpolator

The range of the timing error compensator is configurable and determines the latency of this block.

55.4.7.6.4.3.5 AA-confirmation block

The Bluetooth LE specification stipulates that a spec-compliant device be capable of detecting and rejecting a Bluetooth LE packet if the received AA is different from the desired AA even in only one bit position. But the AAD block requires a received powel level which is at least 4-5 dB above sensitivity to fulfil this stipulation as it is a simple correlation-based detector. However, by using a symbol demodulator, a mechanism can be created to meet the stipulation even at 0.2 dB above sensitivity. The AA-confirmation block is a manifestation of such a mechanism.

55.4.7.6.4.3.5.1 Functional description

The AA-confirmation block may be used as a supplementary step with any of the three ACQ modes. It can be enabled/disabled using the parameter *dmd_aa_enable* (see Table 450 for a list of all the parameters used by this block) but will begin its operation only after a successful AA detection has occurred.

A block diagram of the AA-confirmation block is shown in Figure 329. If enabled and a successful AA detection is declared by the AAD block, the AA-confirmation block obtains the symbol timing and CFO estimates produced by the synchronization stage; and simultaneously, a buffer containing the recent history of incoming samples is frozen (or disabled). The buffer then holds all the samples required by the symbol demodulator for a successful demodulation of the AA bits.

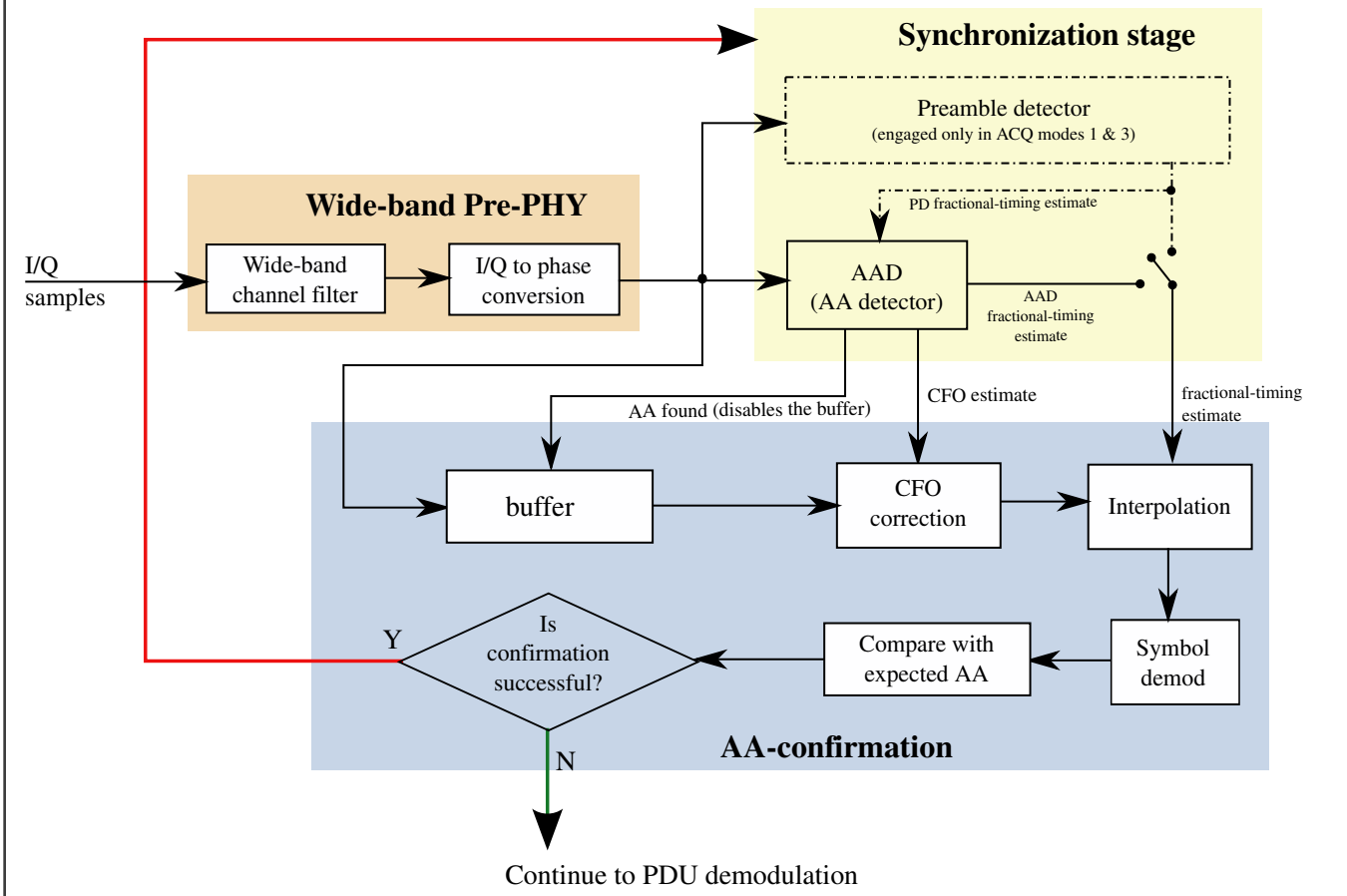


Figure 329. AA-confirmation functional block diagram

Next, the block begins operation by fetching successive samples from the frozen buffer and performs the following steps.

1. First, a CFO correction and timing interpolation is applied to compensate for the frequency offset and timing offset obtained from the synchronization stage.
2. The samples are then passed to a symbol demodulator which processes them and outputs one symbol decision for every *P* input samples.
3. The symbol decisions are then passed to a comparator which compares the demodulated sequence to the desired AA. If they match to within the Hamming distance specified, then AA-confirmation is declared successful and the PHY will continue with PDU demodulation; if not successful, then the AA-confirmation block sends a signal to the PHY state machine to resume acquisition.

55.4.7.6.4.3.5.2 Registers used by the AA-confirmation block

Table 450. AA-confirmation block parameters

Parameter	Notes
aa_0[31:0]	AA pattern of length 1, 2, 3 or 4 octets; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
aa_1[31:0]	AA pattern of length 1, 2, 3 or 4 octets; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.

Table continues on the next page...

Table 450. AA-confirmation block parameters (continued)

Parameter	Notes
aa_2[31:0]	AA pattern of length 1, 2, 3 or 4 octets; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
aa_3[31:0]	AA pattern of length 1, 2, 3 or 4 octets; 1, 2 and 3 octet patterns in bits [7:0], [15:0] and [23:0] respectively.
size_aa[1:0]	Indicates length of active AA's; 0→1 octet, 1→2 octets, 2→3 octets, 3→4 octets.
acq_mode[1:0]	Indicates the ACQ mode used for timing & frequency synchronization in non-coded use cases; 1→mode 1, 2→mode 2, 3→mode 3.
dmd_aa_enable	Enables demodulator-based AA-confirmation; 0→disable, 1→enable.
dmd_aa_hamming_high_pwr[2:0]	Hamming distance tolerance allowed at high power level during AA-confirmation; valid range is [0,7].
dmd_aa_hamming_low_pwr[2:0]	Hamming distance tolerance applicable around sensitivity and at low power levels during AA-confirmation; valid range is [0,7].
dmd_aa_hipow_disable_ovrd	Overrides the default disablement of AA-confirmation at high power levels; 0→no override, 1→override.

55.4.7.6.4.3.6 High accuracy RTT (round-trip time) block (HARTT)

The purpose of this block is to determine an accurate estimate of the timing information and also of the CFO, upon detection of an expected GFSK modulated bit pattern. From a detection perspective, it can be used in two possible cases:

1. Regular packets (i.e. packets that are composed of a preamble, an access address and a PDU)
2. RTT packets (i.e. packets that contain only a preamble and a 32 or a 64 bit PN sequence that does not bear any information bits)

The CFO is directly estimated by the HARTT block. On the other hand, from a timing estimation perspective, the time-stamp is not directly determined by the HARTT block. Squared correlation magnitudes around the peak of the main lobe are instead provided to the application, which needs to further use this information to compute the accurate timing.

55.4.7.6.4.3.6.1 Block description

55.4.7.6.4.3.6.1.1 Introduction

In principle, the HARTT block re-uses the arithmetic from the AA detection blocks in the acquisition module. The main difference is that it has a much lower duty-cycle (to save power) and a much higher resolution on the fixed-point formats (to achieve increased time/frequency accuracy). The expected accuracy is <2 ns for the timing estimation and <600 Hz for the CFO. The operation is triggered by an AA found event, provided by the AA detector in the acquisition block. A re-evaluation of the correlation peak is performed, assuming that the peak may not have been correctly detected during acquisition.

55.4.7.6.4.3.6.1.2 I/Os and settings

Note that all the outputs in the input/output signals table are packed in a 31-bit result (rtt_data[30:0]) and provided to the system memory via DMA. The results provided to software are sents as {rtt_data[30:0],1'b1} if a valid RTT pattern was detected and as{rtt_data[30:0],1'b0} when a valid RTT pattern is not detected in the expected time-slot.

Table 451. Input/output signals

Signal name	Type	Dimension	Description
fm_buffer[7:0]	I	134	The content of the FM buffer shared between acquisition and HARTT

Table continues on the next page...

Table 451. Input/output signals (continued)

Signal name	Type	Dimension	Description
rtt_bits	I	64	The bit sequence which is expected by the HARTT block when *overrd_progr_aa* is true
progr_aa	I	32	The AA programmed for the AA0 detector in the acquisition module
aa_found	I	1	Flag that indicates that AA was found in the AA0 detector
cfo_est[9:0]	I	1	The CFO estimate computed by the AA0 detector
PDelta[9:0]	O	1	Difference between the squared correlation magnitude values, *pm-pp*, provided by the HARTT block (format is sfix10En9)
rtt_int_adj[1:0]	O	1	An integer adjustment of the timing which takes a value different of 0 when the early-late mechanism in the HARTT block chooses a peak different of the one chosen in the acquisition module (format is ufix2En0 and possible values are {-1,0,+1})
ham_rtt_dist_sat[1:0]	O	1	Computed Hamming distance saturated to 2 bits (ufix2)
ha_rtt_cfo[15:0]	O	1	The high accuracy CFO computed by the HARTT block through the CORDIC algorithm
ha_rtt_found	O	1	Flag that indicates that the HARTT operation is done and a valid PN pattern was detected

Table 452. HARTT parameters

Parameter	Description
en_hartt	Enables the HARTT block
overrd_progr_aa	Overrides the programmed AA with an input sequence, which is used as reference for both the AA detector and for the HARTT correlation
rtt_seq_len	Specifies the correlation length applicable when *overrd_progr_aa* is true 0 - the correlation width is 32 bits 1 - the correlation width is 64 bits
first_pdu_bit	Allows providing to the HARTT block the value of the first bit of the PDU. This information is used to generate the HARTT correlator reference sequence when *overrd_progr_aa* is false
fm_ref_111[7:0]	Weight applied to the current symbol, in the HARTT correlation sequence, when both the adjacent bits are the same
fm_ref_110[7:0]	Weight applied to the current symbol, in the HARTT correlation sequence, when one of the adjacent bits is the same and the other one is different
fm_ref_010[7:0]	Weight applied to the current symbol, in the HARTT correlation sequence, when both the adjacent bits are different
ha_rtt_threshold[8:0]	Threshold used to compare p0 with in order to validate RTT found

55.4.7.6.4.3.6.1.3 Block diagram

Figure below shows the block diagram of the HARTT block, which has 4 main components:

- Control engine

- it is a state-machine which controls the behaviour of the correlator and of the combiner
- responsible of the generation of the enable/disable signal for the arithmetic
- it manages the generation of the reference sequences used by the correlator

- Correlator
  - responsible with computing the complex correlation values and the Hamming distances
  - the maximum supported correlation width is 32 bits
  - when the HARTT block is programmed to perform a 64-bit correlation, the correlator performs two 32-bits correlations
- Combiner
  - combines the results computed inside the correlator, when a 64-bits correlation is required
  - performs a peak selection based upon an early-late criterion
- IQ to phase conversion
  - determines the CFO using CORDIC algorithm

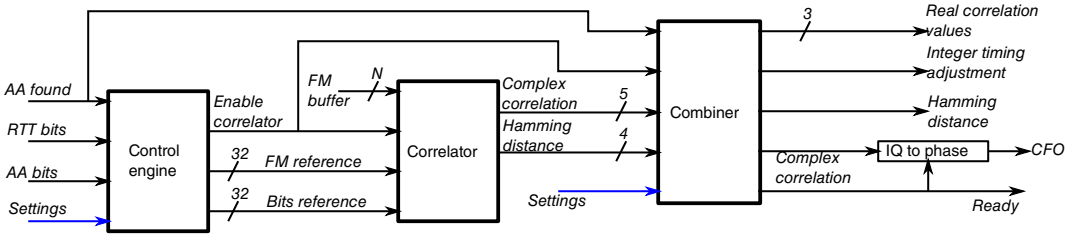


Figure 330. Block diagram

The "Ready" signal is asserted when the result is ready. One example of use of the HARTT block is to interface it with the RSM (ranging sequence manager) block which, when the "Ready" signal is asserted, manages the DMA transfer of the result, packed in a 32-bit word, to the packet RAM. When the HARTT block is used in conjunction with the RSM block, an interrupt is issued to soft-ware for every RSM step (RTT steps included).

More details about each component are provided in the following subsections.

55.4.7.6.4.3.6.1.3.1 Control engine

The state machine consists of 3 states. Two active states and an idle state. When the correlation width is 64 bits (i.e. both *overrd_progr_aa* and *rtt_seq_len* are "1"), both the active states are being used. Otherwise only state 0 is necessary. During each active state, the correlator arithmetic is run for a 5 samples worth of time.

Besides the main states there are secondary states. Their purpose is, they represent transitional states which are meant for preparing the reference sequences or for wait periods. The most complex operation of the state machine is when a 64-bit correlation is required. In this case, after the first 32-bit correlation completes, the state machine manages the change of the reference sequence and also it manages the timing of the second 32-bit chunk correlation.

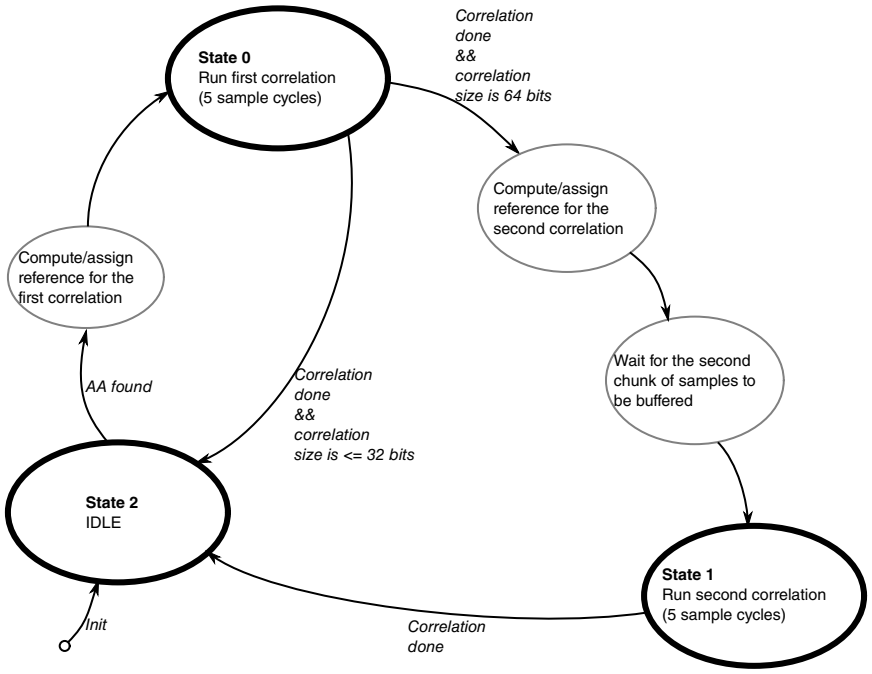


Figure 331. Control engine state-machine diagram

For each active state, an enable signal is sent to the correlator, which drives the computation of 5 complex correlation products at the PHY sampling rate. In order to compute the timing only 3 squared correlation magnitude values are required. However, 5 such values are being computed in order to be able to re-evaluate the position of the correlation peak. Computing 5 values allows moving the estimation of the peak timing by 1 sample at most.

The reference sequence used for correlation is computed by combining the reference symbols with the programmed FM weights. Table below shows an example of how the reference is being derived.

Table 453. Example of FM reference sequence computation

Bit	1	0	1	0	0	0	1	1	1
Bipolar symbol	+1	-1	+1	-1	-1	-1	+1	+1	+1
FM reference value	N/A	*-fm_ref_010*	*+fm_ref_010*	*-fm_ref_110*	*-fm_ref_111*	*-fm_ref_110*	*+fm_ref_110*	*+fm_ref_111*	N/A

A few comments on the reference waveforms:

- Reference values are the result of the multiplication between the bipolar symbols and the programmed weights
- Weights depend on the modulation scheme (more specifically they depend in the BT product and on the modulation index)
- Programmed reference values depend on the context
  - inter-symbol interference is considered negligible from symbols other than the immediate neighbours
  - in order to determine a certain value both the preceding and the succeeding bits need to be known
  - for the reason on the bullet above the values are indeterminate for the first and the last columns of the table above
- When determining an FM reference sequence corresponding to *N* bits, actually (*N*+2) bits need to be known
  - means that the last bit of the preamble and the first bit of the PDU (for regular packets) or the first bit of the post-padding (for the RTT packets) need to be known

— the best timing estimation is obtained when the sequence to be searched for starts and ends with a transition (i.e. the last bit of the preamble is the inverse of the first bit of the searched sequence and the first bit of the PDU/post-padding is the inverse of the last bit of the searched sequence)

- If *N*=64 the correlator needs to perform 2 correlation products of 32-bit width - the correct provisioning of the FM reference sequences, corresponding to the first and the second half, are managed by the control engine

Table below shows an example of FM reference values that correspond to BT=0.5/h=0.5 (RTT 1 Mbps) and BT=2/h=0.5 (RTT 2 Mbps).

Table 454. FM reference values for BT=0.5 / h=0.5

	fm_ref_111	*fm_ref_110*	*fm_ref_010*
Real value [rad]	*π*/2	1.22	0.88
Fixed point value [normalized]	0.5	0.387	0.281
Fixed point value [hex]	0x80	0x63	0x48

Table 455. FM reference values for BT=2 / h=0.5

	fm_ref_111	*fm_ref_110*	*fm_ref_010*
Real value [rad]	*π*/2	1.4	1.2
Fixed point value [normalized]	0.5	0.445	0.382
Fixed point value [hex]	0x80	0x72	0x61

55.4.7.6.4.3.6.1.3.2 Correlator

This sub-block is responsible with computing the complex correlation values and the Hamming distance. It is implemented in a similar fashion as the adder tree in the acquisition AA detectors. The difference is that more resolution is provided to the correlator taps in order to obtain a better timing accuracy.

55.4.7.6.4.3.6.1.3.3 Combiner

This sub-block is responsible with combining the complex correlation and Hamming distance computation results when a 64-bit correlation is required. It is also responsible with the re-evaluation of the correlation peak. The figure below shows a flowchart diagram of the combiner.

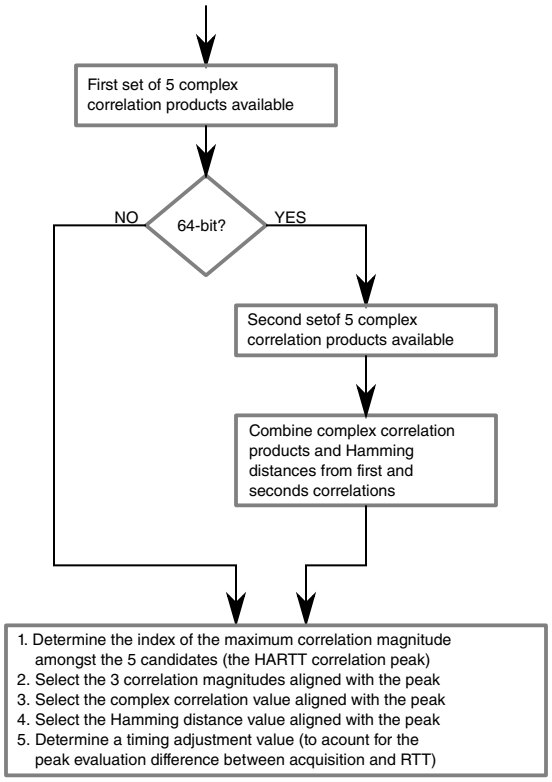


Figure 332. Combiner operation flowchart

The complex magnitude outputted by the combiner is further used by the CFO estimation algorithm.

55.4.7.6.4.3.6.1.3.4 Accurate CFO estimation

For a detailed description of the CFO estimation via CORDIC algorithm please consult the "Radio Frequency Fingerprinting (Fine CFO estimation)" section of the current chapter.

55.4.7.6.4.3.6.2 Integration considerations

As mentioned in the block description there are a few signals provided by the acquisition AA0 detector to the HARTT block. *aa_found* provides the timing reference and triggers the operation of the HARTT block. *cfo_est* is used by the Hamming distance computation for the FM symbols demodulation. Regarding the settings of the AA0 detector, when *overrd_progr_aa* is true the programmed AA is overwritten by the first half of the *rtt_bits* signal and the programmed AA length (a.k.a. SFD length) is overwritten by value 3 (which is the value corresponding to 32-bits).

Impact on other blocks in the PHY is described in the sequel.

55.4.7.6.4.3.6.2.1 Integration with the AA Hamming confirmation block

The purpose of the Hamming confirmation block is to provide a more reliable assessment on the Hamming distance than the one provided by the AA detectors in the acquisition module. When enabled, the operation of the AA Hamming confirmation block is triggered following the assertion of *aa_found*. After a while the AA detection is either confirmed or disconfirmed. When *overrd_progr_aa* is true, the programmed AA used by the AA Hamming confirmation block is overwritten by the first half of the *rtt_bits* signal.

When the operation of the confirmation block is enabled, the *ha_rtt_ready* indication from the HARTT is qualified by the AA confirmation event, which results in *ha_rtt_vld* signal.

55.4.7.6.4.3.6.2.2 Integration with the PHY state-machine

In order to support the reception of RTT packets (which have no PDU) a dedicated state is defined in the PHY state-machine, called RTT state. If *overrd_progr_aa* is true, the PHY SM transitions to the RTT state after AA is detected and confirmed.

RTT state is responsible to providing the control signals to all the blocks in the PHY, after an RTT packet is detected. It also has the role of extending the operation of the FM buffer (in the acquisition module) when *rtt_seq_len*=1 (i.e. the HARTT block searches for a 64-bit sequence).

55.4.7.6.4.3.6.3 Computation of the fractional timing (intended for the application)

The time-stamps, used for the round-trip time computation, are locally determined by the application using the information from HARTT. It has two components:

- an integer component, computed by combining the timing of "AA detection" event (issued by the radio) with the *rtt_int_adj* value provided by HARTT
- a fractional component, computed using the 3 values contained in the *pm_pp_p0* result provided by the HARTT block

Since the determination of the integer part is trivial, the computation of the fractional timing is not. The latter is addressed in the current section.

A few consideration on the main correlation lobe:

- When the data is oversampled it can be approximated by a sinc function
- A property of the sinc function is that peak of the sinc function can be estimated as $p=k*(\alpha-\gamma)$
  - where α and γ are the correlation output samples (squared correlation magnitudes) before and after the peak
  - p is the deviation from the center of the main lobe
  - α and γ are computed by the HARTT block and reported in the *pm_p0_pp* result (they represent the first and the last value of the result)
  - from now on the following notation will be used $\delta=(\alpha-\gamma)$
- k depends on the bit-sequence used for correlation
  - for RTT packets the sequence is randomly generated so it doesn't have to respect strict rules
  - there is a lot of variability of the used sequences leading to variability of k
  - flatter the main lobe larger the required k - sequences close to all-ones or all-zeros lead to very flat lobes
  - some sequences may have a periodical structure which leads to lobe smearing

55.4.7.6.4.3.6.3.1 Modelling of the fractional delay

Modelling of p (which is the fractional delay) as a first order polynomial was not satisfactory. Indeed a third order polynomial is used. In this case, k is replaced by a vector of 3 elements and the equation becomes:

$$p=k_1*\delta+k_2*\delta^2+k_3*\delta^3$$

Usually only the first and the third order terms are significant in the equation above. The coefficients k_1, k_2 and k_3 are related to the structure of the correlation sequence and they are computed for each sequence separately. Different sequences with similar properties will map on the same set of coefficients.

55.4.7.6.4.3.6.3.2 Dependence of the coefficients on the correlation sequence

More specifically, k_1, k_2 and k_3 depend on the context of each composing bit. To formalize this, the following notations are used for a certain sequence:

- c_{010} - count of the bits having both the adjacent neighbours logically inverted
- c_{011} - count of the bits having one of the adjacent neighbours logically inverted
- c_{111} - count of the bits having none of the adjacent neighbours logically inverted

The relation between the coefficients and the count consists in a third order polynomial. This is formalized in the equation below:

$[k_1, k_2, k_3]=[c_{010}, c_{011}, c_{111}, c_{010}^2, c_{011}^2, c_{111}^2, c_{010}^3, c_{011}^3, c_{111}^3]*\Gamma$

where **Γ** is the 9x3 transformation matrix.

This transformation is very useful because it greatly reduces the storage requirements. It also slightly increases the computation requirements because, first, the counts need to be determined for each sequence and second a multiplication is required to determine k_1, k_2 and k_3. **Γ** is unique for all possible sequences and it depends on the modulation scheme (modulation index - BT combination). Matrix **Γ** is pre-determined using intensive simulations and it is made available to the application from memory.

55.4.7.6.4.3.6.3.2.1 Γ matrix values for *N*=32, *R*=1 Mbps (h=0.5 / BT=0.5)

0.134045	-0.01074	-1.50855
-0.0689	0.084502	3.240881
0.140553	-0.0119	-0.9309
-0.00192	-0.00165	0.02838
0.00597	-0.00568	-0.18365
6.15E-05	-0.00107	0.016142
8.73E-06	8.06E-05	0.000279
-7.48E-05	9.96E-05	0.002691
0.000147	5.57E-05	-0.00151

55.4.7.6.4.3.6.3.2.2 Γ matrix values for *N*=32, *R*=2 Mbps (h=0.5 / BT=2)

0.049461	0.027639	-0.35573
0.023867	-0.03173	0.578603
0.083292	0.000394	-0.06136
-3.83E-05	-0.0017	0.010247
0.001268	0.001771	-0.0328
1.61E-03	0.001144	-0.02035
-3.67E-06	4.09E-05	-4.87E-05
-2.20E-05	-2.49E-05	0.000476
6.18E-05	-3.92E-05	0.000372

55.4.7.6.4.3.6.3.2.3 Γ matrix values for *N*=64, *R*=1 Mbps (h=0.5 / BT = 0.5)

0.065958	0.001162	-0.7422
-0.04655	0.011012	2.207578
0.068705	0.007808	-0.14743
-4.64E-04	-0.00057	0.004103
0.002112	-9.60E-05	-0.07448

Table continues on the next page...

Table continued from the previous page...

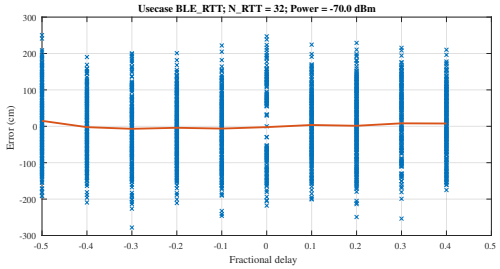
3.05E-04	-0.001	-0.02855
3.53E-06	1.25E-05	4.53E-06
-1.73E-05	-3.72E-06	0.000684
1.07E-05	2.17E-05	0.000561

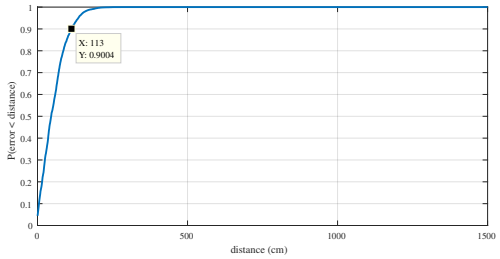
55.4.7.6.4.3.6.3.2.4 Γ matrix values for *N*=64, *R*=2 Mbps (h=0.5 / BT=2)

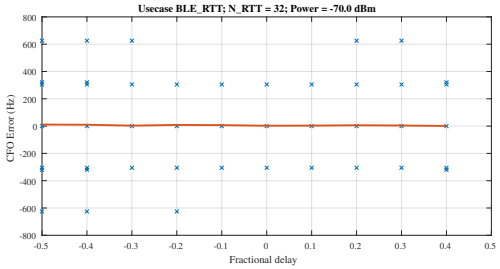
0.02593	-0.02971	-0.32042
0.007891	0.013627	0.634331
0.049023	0.005655	-0.13215
-3.35E-05	0.00173	0.007166
0.000397	-3.68E-04	-0.02022
-3.76E-05	-0.00047	-0.00146
4.63E-07	-3.24E-05	-9.79E-05
-3.06E-06	3.01E-06	0.000178
1.66E-05	1.27E-05	-5.94E-05

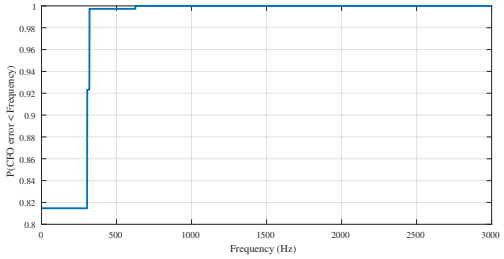
55.4.7.6.4.3.6.3.3 RTT block timing/frequency accuracy at hight SNR

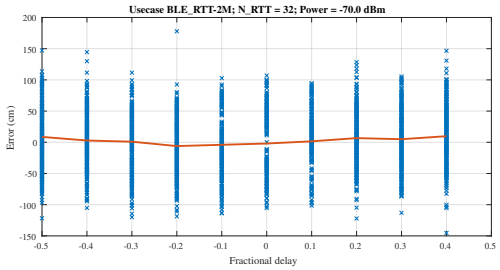
In the figures below each blue "x" dot is the distance estimate corresponding to one RTT measurement. The red line corresponds to the averaging of all the measurements corresponding to a certain fractional delay (i.e. averaging all the dots on the vertical). To determine the red line, which is intended to show that there is no bias in the estimation, 100 distance estimates were used (i.e. 100 distance estimates per fractional delay value). The fractional delay reflects the timing uncertainty due to the sample clock which is 4 MHz for 1Mbps and 8 MHz for 2Mbps. As an example, when the frequency of the sample clock is 4 MHz, a fractional delay of 0.25 corresponds to a timing delay of 0.25/(4 MHz) = 62.5 nanoseconds.

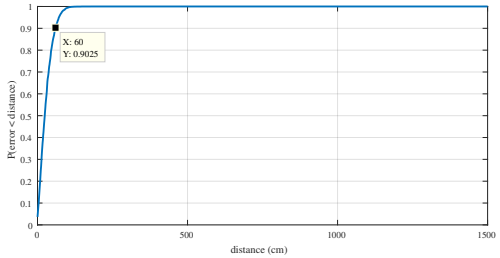

Usecase BLE_RTT; N_RTT = 32; Power = -70.0 dBm
Error (cm)
Fractional delay

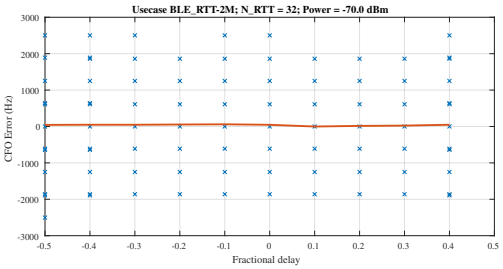

X: 113
Y: 0.9004
P(error < distance)
distance (cm)

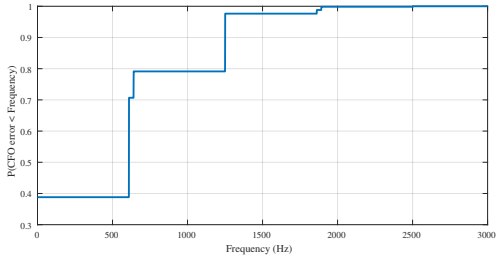

Usecase BLE_RTT; N_RTT = 32; Power = -70.0 dBm
CFO Error (Hz)
Fractional delay

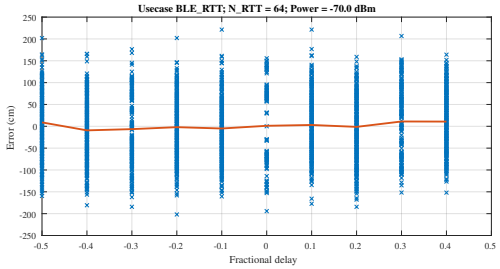

P(CFO error < Frequency)
Frequency (Hz)

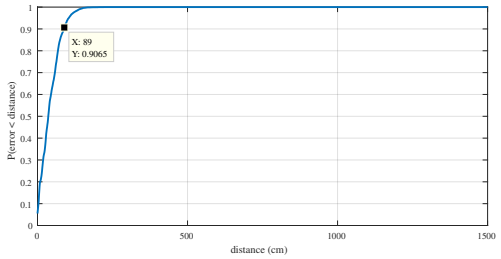

Usecase BLE_RTT-2M; N_RTT = 32; Power = -70.0 dBm
Error (cm)
Fractional delay

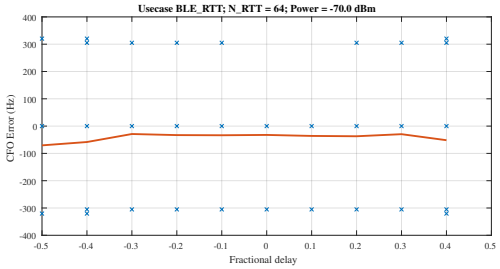

X: 60
Y: 0.9025
P(error < distance)
distance (cm)

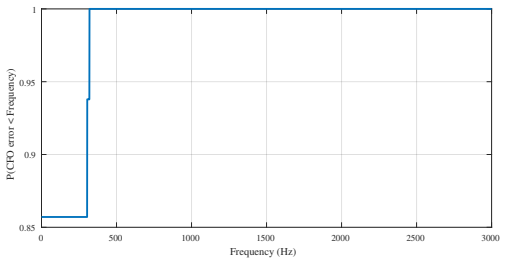

Usecase BLE_RTT-2M; N_RTT = 32; Power = -70.0 dBm
CFO Error (Hz)
Fractional delay

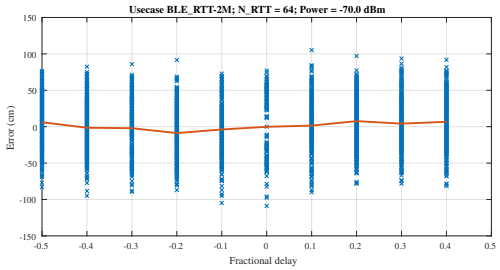

P(CFO error < Frequency)
Frequency (Hz)

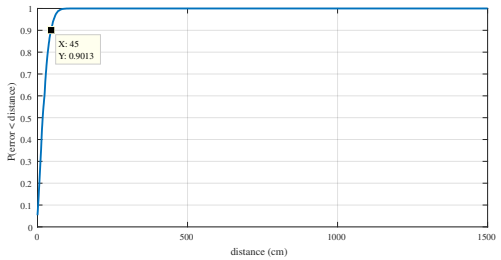


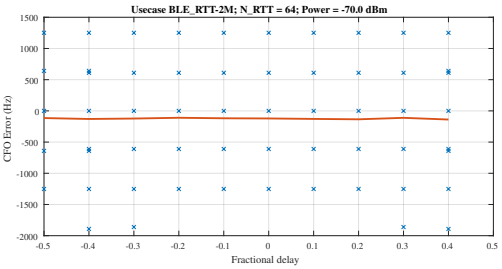


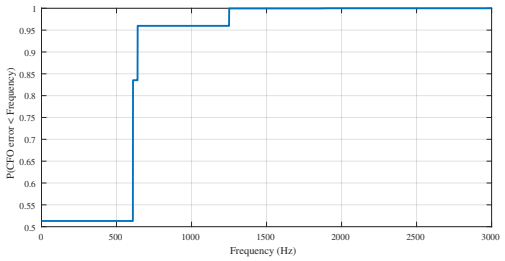




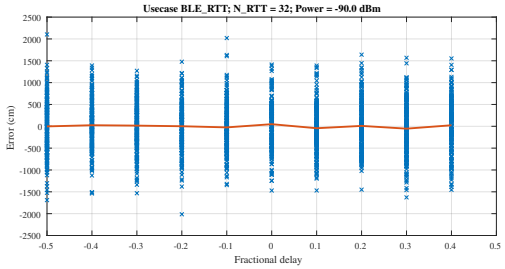


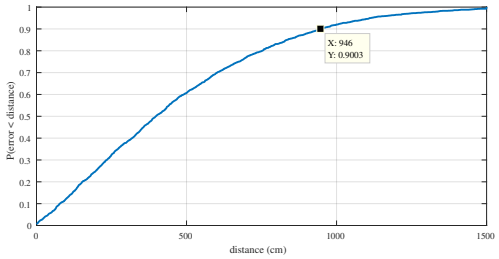


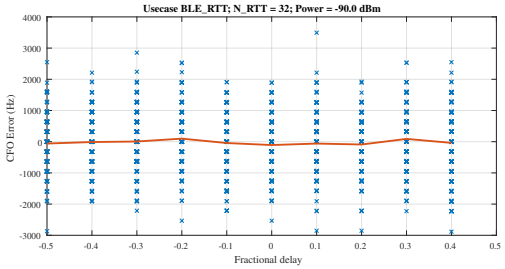


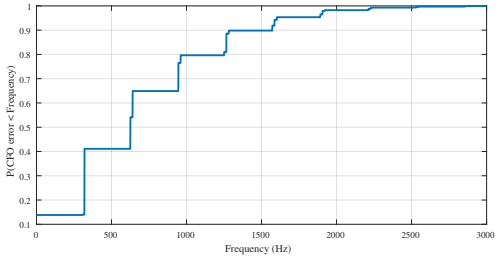


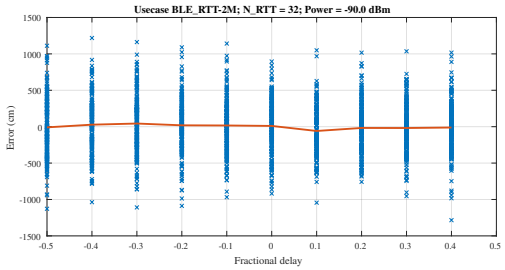
55.4.7.6.4.3.6.3.4 RTT block timing/frequency accuracy at low SNR

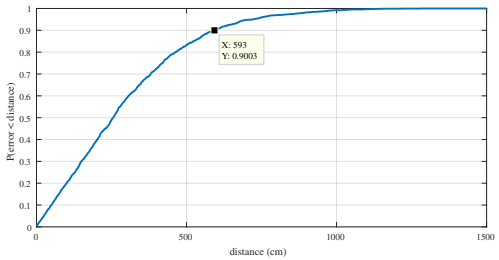

Usecase BLE_RTT; N_RTT = 32; Power = -90.0 dBm
Error (cm)
Fractional delay

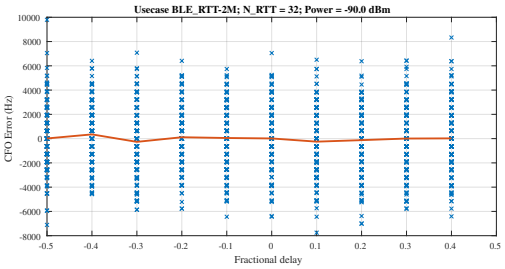

X: 946
Y: 0.9003
P(error < distance)
distance (cm)

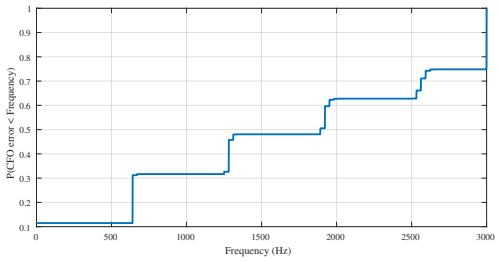

Usecase BLE_RTT; N_RTT = 32; Power = -90.0 dBm
CFO Error (Hz)
Fractional delay

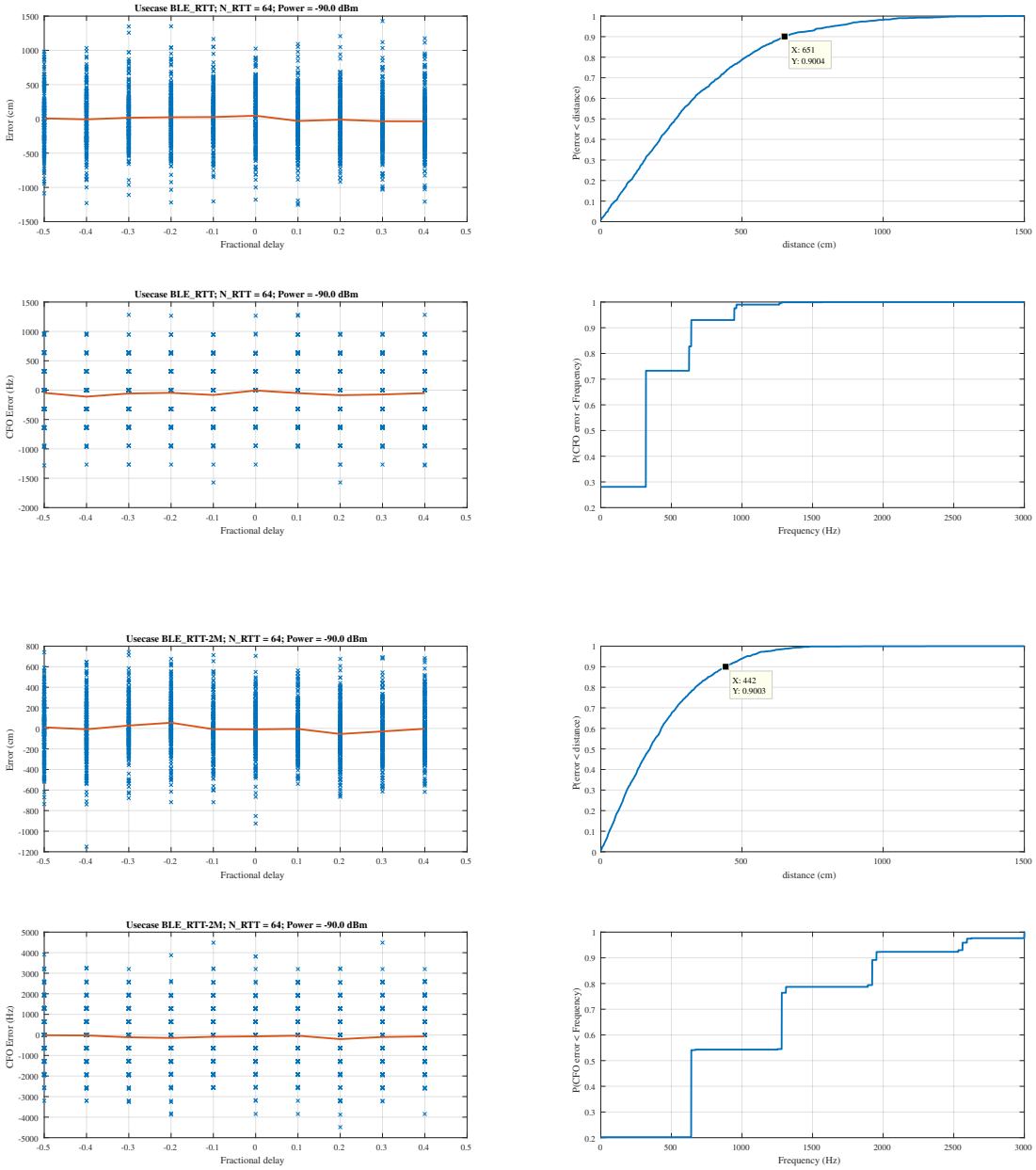

P(CFO error < Frequency)
Frequency (Hz)


Usecase BLE_RTT-2M; N_RTT = 32; Power = -90.0 dBm
Error (cm)
Fractional delay


X: 593
Y: 0.9003
P(error < distance)
distance (cm)


Usecase BLE_RTT-2M; N_RTT = 32; Power = -90.0 dBm
CFO Error (Hz)
Fractional delay


P(CFO error < Frequency)
Frequency (Hz)



55.4.7.6.4.3.6.4 Methodology for time-stamp determination

Following steps are required to determine the time-stamp for each transaction:

1. Determine the integer component of the time-stamp by capturing the "AA detection" event
2. Adjust the integer value using *rtt_int_adj* value provided in a HARTT status register (note that this value is expressed in PHY sampling rate cycles and it needs to be properly adjusted to the clock frequency)
3. Determine the [c_{010}, c_{011}, c_{111}] that correspond to the PN sequence used for the RTT transaction
4. Determine the [k_1, k_2, k_3] coefficients using the matrix transformation **Γ**
5. Compute the fractional delay using relation $p=k_1*\delta+k_2*\delta^2+k_3*\delta^3$ (δ is reported in the *PDelta* result)

6. Compute the final time-stamp combining the integer and the fractional components (note that the fractional value is expressed in PHY sampling rate cycles and it needs to be properly adjusted to the clock frequency)

55.4.7.6.4.3.6.5 Considerations for the choice of AA patterns

In order to get an accurate RTT estimation the AA pattern must have good autocorrelation properties and a controllable ISI at the beginning and at the end of the over-the-air transmitted waveform. In order to achieve that there are a few measures that can be taken:

- Make sure that there is a transition between the bit preceding the AA pattern and the first bit of AA
- Make sure that there is a transition between the last bit of AA and the first bit that follows after
- Use a scoring mechanism to select AA with good correlation properties
- Take into account the flatness of the main correlation lobe

The BLE protocol already imposes a transition between the last bit of the preamble and the first bit of the AA. For HADM RTT packets a rule was put in place to insure a transition at the end of the AA by defining 4 trailing bits that look like the preamble. However, for legacy packets the AA has to be chosen such that its' last bit is the complement of the first bit of the PDU transmitted over the air. It is important to keep in mind that, if FEC is used (e.g. whitening), the first OTA bit may not be the protocol bit. So, when choosing the AA pattern, the application has to have knowledge of the first protocol bit and the FEC operation that this bit may be subject to.

The BLE AA generation rules are not used in HADM because they can mess-up the entropy and reduce security. This is why a scoring mechanism is required, to achieve a tradeoff between security and correlation properties. The score is computed by summing up the bit autocorrelation of the AA pattern in the vicinity of the main lobe. In HADM, at each RTT transaction, two AA patterns are computed and the one with the best score is used. An alternative would use an AA pattern that has a score less or equal than a threshold. The methodology of computing the score is given in the code below:

```
C = zeros(depth,1);
N = length(aa_pattern);
for m = 1:depth
C(m) = 2*sum(mod(aa_pattern(1:N-m)+aa_pattern((1+m):N),2)) - (N-m);
end
score = sum(abs(C));
```

The value for *depth* used by HADM is 3.

As previously shown in the current section it has been observed that the structure of the AA pattern influences the shape of the correlation (which is used to determine the ToF and thus the distance). To correlate the structure with the correlation a set of counts is being used. The flatness of the main lobe is mainly determined by c_{111}, i.e. higher this count flatter the main autocorrelation lobe. It was shown in this section how to compute the polynomial fit coefficients, depending on the counts, in order to get best accuracy. However, if the methodology presented here is not possible to implement, the AA with flat autocorrelation can be avoided by imposing a threshold for c_{111}.

As an example, for $depth$=3, the application can use only AAs for which $depth$<9 and c_{111}<8.

55.4.7.6.4.4 DMAs

This section describes all the DMAs that the 2.4GHz PHY module generates.

Table 456. DMA signal list

Page Name	Page description	Data valid used by DMA capture(as per model)	Signals names in (as per model)	Signal bit-width	data type	location in bus	Signals description
PHY_DMA_P0	Demodulator and frequency synchronization	PHY sample clock (PhySampClk)	Sb	7	sfix7_en4	[31:25]	Soft-bits
			Bits	1	bool	[24]	Demod. hard-bits
			Ferr	9	sfix9_en8	[23:15]	Freq. tracking err. estimate
			PhNb	6	sfix6_En5	[14:9]	Narrowband phase
			CfoEst	9	sifx9_en8	[8:0]	CFO estimate,can byte read [7:0] as sfix8_en8
PHY_DMA_P1	Demodulator and timing synchronization	Symbol valid (SymVld)	Sb	7	sfix7_en4	[31:25]	Soft-bits,wih [24]=1'b0, can byte read as sfix8_en5
			1'b0	1		[24]	reserved
			Terr	9	sfix9_en4	[23:15]	Timing tracking err. estimate
			Bits	1	bool	[14]	Demod. hard-bits
			3'b0	3		[13:11]	reserved
			Magnitude	11	ufix11_en11	[10:0]	RSSI Magnitude
PHY_DMA_P2	Un-coded acquisition	PHY sample clock (PhySampClk)	PhWb	6	sfix6_En5	[31:26]	Wideband phase
			CfoEst	10	sfix10_en9	[25:16]	CFO estimate of uncoded AA0
			Found SFD	1	bool	[15]	AA found
			Signal high	1	bool	[14]	High signal presence detection
			Signal present	1	bool	[13]	Low signal presence detection
			Pp	5	ufix5_en5	[12:8]	AA complex correlation magnitude
			1'b0	1		[7]	reserved
			Hamming Value	7	ufix7	[6:0]	AA Hamming distance,can byte read[7:0] as ufix8
PHY_DMA_P3	LR PD	PHY sample clock (PhySampClk)	mag	8	ufix8_en8	[31:24]	LR preamble correlation magnitude
			cfoC	9	sfix9_en8	[23:15]	LR preamble instantaneous CFO est.
			Cnt	3	ufix3	[14:12]	LR peaks above threshold count
			PDLR	2	ufix2	[11:10]	LR preamble found
			Cfo est.	10	sfix10_en9	[9:0]	LR CFO estimate,can byte read [7:0] as sfix8_en9, or word read [9:0] as sfix10_en9

Table continues on the next page...

Table 456. DMA signal list (continued)

Page Name	Page description	Data valid used by DMA capture(as per model)	Signals names in (as per model)	Signal bit-width	data type	location in bus	Signals description
PHY_DMA_P4	LR AAD	PHY sample clock (PhySampClk)	AACorr	11	sfix11_en1	[31:21]	LR AA correlation magnitude
			2'b0	2		[20:19]	reserved
			AAFndLR	1	bool	18	LR AA found
			AAFracLR[2:1]	2	sfix2_en1	[17:16]	LR fractional estimate
			AAHam	7	ufix7	[15:9]	LR AA Hamming distance
			AASbs	3	sfix3_en1	[8:6]	LR soft-bits used for corr. (from M2n)
			PhNb	6	sfix6_En5	[5:0]	Narrowband phase

55.4.7.6.5 Application information

This section describes applications supported by the 2.4GHz PHY module.

One of the possible uses of the PHY module is the frames time/frequency synchronization and PDU bits demodulation corresponding to different protocols. As an example, the PHY module supports the packet reception for Bluetooth low energy uncoded (1Mbps and 2 Mbps) and long range (125 kbps and 500 kbps).

Another use of the PHY is the high accuracy time of flight estimation which is used for localization.

55.4.7.7 Transceiver Sequence Manager

55.4.7.7.1 Introduction

55.4.7.7.1.1 Overview

The Transceiver Sequence Manager controls the warmup and warmdown processes for all radio sequences. The TSM provides for a TX sequence, and an RX sequence. For both TX and RX, a warmup and a warmdown sequence is supported. The length of each sequence is programmable, from 1 – 254us. The resolution of the TSM is 1us. Controls for all analog and digital transceiver blocks are provided. Each TSM output enables, or otherwise provides control for, a transceiver-related block. Each TSM output has 4 8-bit registers, with which to control the point at which the output asserts during the warmup (1 register for TX, 1 for RX), and the point at which it deasserts (1 register for TX, 1 for RX). Some TSM outputs, which are known in advance to require identical timing, are “ganged together”, to reduce area and required programming. For DFT, and non-mission-mode validation, each TSM output can be placed under direct software control, using register overrides. Any sequence can be temporarily halted, by setting an optional, programmable breakpoint.

55.4.7.7.1.2 Features

The TSM includes the following features:

- TX warmup and warmdown sequence and RX warmup and warmdown sequence sequence for every controllable output.
- Programmable sequence length from 1 to 254 microseconds for each sequence in 1 microsecond increments.
- Multi-protocol
- Ability to place TSM outputs under software control for test and non-mission mode validation.
- Ability to signal link layers upon PLL unlock or RF_NOT_ALLOWED conditions.

- Fast Warmup Capability for TX and RX
- LPPS Support
- Programmable breakpoint capability to halt any sequence temporarily.

55.4.7.7.2 Functional Description

55.4.7.7.2.1 Sequence Counter

The TSM supports 1 TX and 1 RX sequence. Each sequence consists of 3 phases:

1. WARMUP phase
2. ON phase
3. WARMDOWN phase

The central element of the TSM is an 8-bit counter. The counter is held at 0 during the idle state. From idle state, any sequence can be launched, by an initiating event. (See Section “Sequence Initiation”). At an initiating event, the TSM counter counts up from 0 to a programmed stop point during the WARMUP phase, determined by the END_OF_TX_WU or END_OF_RX_WU register, depending on whether the sequence is TX or RX. At the end-of-warmup “stop point”, the TSM counter holds its count, and the TSM sequence enters the ON phase. The TSM counter will remain in the ON phase, and hold its count, until the initiating event deasserts (See Sequence Termination). When either of these conditions occurs during the ON phase, the TSM will resume counting from the point it was holding during the ON phase, and the sequence will enter the WARMDOWN phase. The counter will continue counting until a programmed stop point is reached. This stop point is determined by the END_OF_TX_WD or END_OF_RX_WD register, depending on whether the sequence is TX or RX. Once this point is reached, the sequence returns to idle, and the TSM counter returns to 0.

The 4 8-bit registers which control the duration of the WARMUP phase (END_OF_TX_WU[7:0] and END_OF_RX_WU[7:0]), and the duration of the WARMDOWN phase (END_OF_TX_WD[7:0] and END_OF_RX_WD[7:0]), reside in the END_OF_SEQ register.

All TSM-controlled outputs are active-high. Any TSM-controlled output can be asserted once during a sequence (either TX or RX sequence), and then deasserted once. Or, the output can be held in a deasserted state for the duration of the sequence. The precise timing for the assertion and deassertion of each TSM-controlled output, for both TX and RX sequences, is determined by 4 8-bit registers assigned to that output (see TSM-Controlled Outputs). The TSM resolution is 1us. This is the update rate for the TSM counter, and sets the granularity with which warmup and warmdown processes can be controlled.

55.4.7.7.2.2 TSM-Controlled Outputs

The TSM-controlled outputs are shown in the table below. In general, each output signal is assigned a unique TSM_TIMING register to control its timing, but in cases where output signals will always be asserted and deasserted at the same time, the signals can share a TSM_TIMING register (e.g., seq_ldo_gang_pup).

Output Signal Name	TSM_TIMING #	TSM_TIMING Name
tsm_rf_active	0	tsm_rf_active
tsm_rf_status	1	tsm_rf_status
tsm_rf_priority	2	tsm_rf_priority
tsm_irq0_start_trig	3	tsm_irq0_start_trig
tsm_irq1_stop_trig	4	tsm_irq1_stop_trig
gpio0_trig_en	5	gpio0_trig_en
gpio1_trig_en	6	gpio1_trig_en

Table continues on the next page...

Table continued from the previous page...

Output Signal Name	TSM_TIMING #	TSM_TIMING Name
gpio2_trig_en	7	gpio2_trig_en
gpio3_trig_en	8	gpio3_trig_en
dcoc_gain_cfg_en	9	dcoc_gain_cfg_en
ldo_cal_en	10	ldo_cal_en
pll_dig_en	11	pll_dig_en
sigma_delta_en	12	sigma_delta_en
dcoc_cal_en	13	dcoc_cal_en
tx_dig_en	14	tx_dig_en
freq_targ_ld_en	15	freq_targ_ld_en
rx_init	16	rx_init
rx_dig_en	17	rx_dig_en
rx_phy_en	18	rx_phy_en
seq_bg_pup_ibg_cal	19	seq_bg_pup_ibg_cal
seq_ldotrim_pup	20	seq_ldotrim_pup
seq_ldo_cal_pup	21	seq_ldo_cal_pup
seq_bg_fc	22	seq_bg_fc
seq_ldo_pll_fc	23	seq_ldo_gang_fc
seq_ldo_vco_fc		
seq_ldo_rxtxhf_fc		
seq_ldo_rxtxlf_fc		
seq_ldo_ant_pup	24	seq_ldo_gang_pup
seq_ldo_pll_pup		
seq_ldo_vco_pup		
seq_ldo_xo_dist_pup		
seq_ldo_rxtxhf_pup		
seq_ldo_rxtxlf_pup		
seq_ldo_lv_pup	25	seq_ldo_lv_pup
seq_bg_pup	26	seq_bg_pup
seq_bg_pup_ibg_ant	27	seq_bg_pup_ibg_ant
seq_bg_pup_ibg_xo_dist	28	seq_bg_pup_ibg_xo_dist
seq_bg_pup_ibg_tx	29	seq_bg_pup_ibg_tx
seq_bg_pup_ibg_rx	30	seq_bg_pup_ibg_rx

Table continues on the next page...

Table continued from the previous page...

Output Signal Name	TSM_TIMING #	TSM_TIMING Name
seq_tsm_iso_b_2p4ghz	31	seq_tsm_iso_b_2p4ghz
seq_rccal_pup	32	seq_rccal_pup
seq_pd_en_fcal_bias	33	seq_pd_en_fcal_bias
seq_pd_pup	34	seq_pd_pup
seq_vco_pup	35	seq_vco_pup
seq_xo_dist_en	36	seq_xo_dist_en
seq_xo_dist_en_clk_ref	37	seq_xo_dist_en_clk_ref
seq_xo_en_clk_2p4g	38	seq_xo_en_clk_2p4g
seq_xo_dist_en_clk_adcdac	39	seq_xo_dist_en_clk_adcdac
seq_dac_pup	40	seq_dac_pup
seq_vco_en_hpm	41	seq_vco_en_hpm
seq_lo_pup_vlo_fbk	42	seq_lo_pup_vlo_fbk
seq_lo_pup_vlo_rx	43	seq_lo_pup_vlo_rx
seq_lo_pup_vlo_rxdrv	44	seq_lo_pup_vlo_rxdrv
seq_lo_pup_vlo_tx	45	seq_lo_pup_vlo_tx
seq_lo_pup_vlo_txdrv	46	seq_lo_pup_vlo_txdrv
seq_divn_pup	47	seq_divn_pup
seq_divn_openloop[1]	48	seq_divn_closedloop
seq_pd_en_pd_drv	49	seq_pd_en_pd_drv
seq_cbpf_en_dcoc	50	seq_cbpf_en_dcoc
seq_rx_lna_pup	51	seq_rx_gang_pup
seq_adc_pup		
seq_cbpf_pup		
seq_rx_mix_pup		
seq_spare1[2]		
seq_spare3	52	seq_spare3

1. This signal is inverted with respect to the timing signal
2. This is the pup for the RX peak detector

During the WARMUP and WARMDOWN phases of any sequence, any TSM-controlled output can be programmed to assert once and then deassert once, by programming the timing registers associated with the output. Each output has 4 8-bit timing registers associated with it. The name of the register matches the name of the output:

*OUTPUTNAME*_TX_HI[7:0]

*OUTPUTNAME*_TX_LO[7:0]

*OUTPUTNAME*_RX_HI[7:0]

*OUTPUTNAME*_RX_LO[7:0]

In the case of grouped outputs, the timing registers associated with the signal controlling the group, control timing for the entire group.

During the WARMUP phase of a TX sequence, as the 8-bit TSM counter increments, once the TSM counter equals or exceeds the programmed value of *OUTPUTNAME*_TX_HI[7:0], but is less than the programmed value of *OUTPUTNAME*_TX_LO[7:0], the corresponding TSM output will transition high. During the WARMUP or WARMDOWN phase of the TX sequence, once the TSM counter equals or exceeds the programmed value of *OUTPUTNAME*_TX_LO[7:0], the TSM output will transition low. If the programmed value of *OUTPUTNAME*_TX_HI[7:0], is greater than the length of the TX sequence (set by register END_OF_TX_WD[7:0]), then that signal will never transition high during the TX sequence. A convenient way of ensuring a signal does not assert during a TX sequence, is to set its *OUTPUTNAME*_TX_HI[7:0]=255.

During the WARMUP phase of a RX sequence, as the 8-bit TSM counter increments, once the TSM counter equals or exceeds the programmed value of *OUTPUTNAME*_RX_HI[7:0], but is less than the programmed value of *OUTPUTNAME*_RX_LO[7:0], the corresponding TSM output will transition high. During the WARMUP or WARMDOWN phase of the RX sequence, once the TSM counter equals or exceeds the programmed value of *OUTPUTNAME*_RX_LO[7:0], the TSM output will transition low. If the programmed value of *OUTPUTNAME*_RX_HI[7:0], is greater than the length of the RX sequence (set by register END_OF_RX_WD[7:0]), then that signal will never transition high during the RX sequence. A convenient way of assuring a signal does not assert during a RX sequence, is to set its *OUTPUTNAME*_RX_HI[7:0]=255.

The following diagram depicts the TSM counter, and the control logic used to assert and deassert the TSM-controlled outputs. For clarity, logic for only 2 of the outputs (pll_dig_en, sigma_delta_en) are shown.

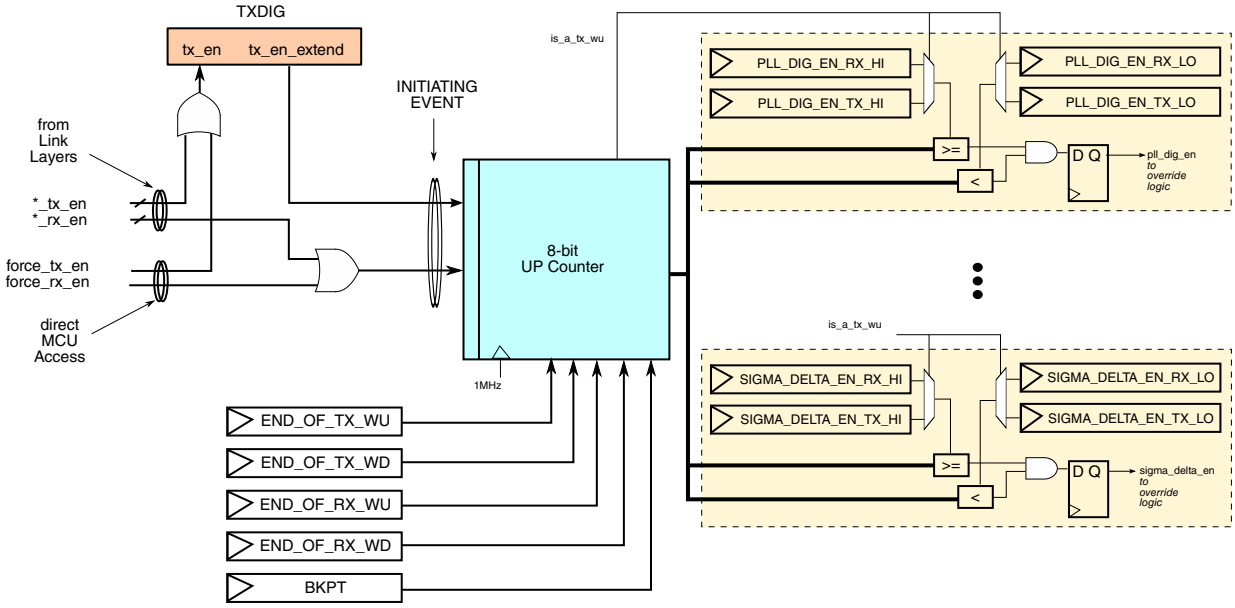


Figure 333. TSM BLOCK DIAGRAM

55.4.7.7.2.3 Timing Registers

There are a set of timing registers for each TSM-controlled output (or, for the signal controlling the group for the “grouped” outputs). Each register set consists of 4 bitfields: one to control the assertion time of the output for TX sequences, one to control the assertion time for RX sequence, one to control the deassertion time for TX sequences, and one to control the deassertion time for RX sequences. For TSM-controlled outputs that are grouped, one set of timing registers controls the timing for all members of the group. For each of these sets, the following table lists the SoC register name, and 4 timing register names associated with each.

SOC REGISTER NAME	BITS [31:24]	BITS [23:16]	BITS [15:8]	BITS [7:0]
TIMINGxx	RX_LO[7:0]	RX_HI[7:0]	TX_LO[7:0]	TX_HI[7:0]

Some of the TSM outputs have relevance for TX sequences only. For those, timing registers are only provided for TX assertion and deassertion times. The RX timings for such outputs are essentially hardwired to 255, meaning there will be no signal assertion during RX sequences.

Some of the TSM outputs have relevance for RX sequences only. For those, timing registers are only provided for RX assertion and deassertion times. The TX timings for such outputs are essentially hardwired to 255, meaning there will be no signal assertion during TX sequences.

55.4.7.7.2.4 Sequence Initiation

The SoC will include one or more protocol engines. Any of the protocol engines, can launch a TSM sequence. Each protocol engine will have an initiating signal for a TX sequence, and another for an RX sequence.

For direct software control, the MCU/Host can initiate a TX sequence warmup directly, by setting the FORCE_TX_EN bit, and later initiate a TX sequence warmdown by clearing the bit. The MCU/Host can initiate a RX sequence warmup directly, by setting the FORCE_RX_EN bit, and later initiate a RX sequence warmdown by clearing the bit. These bits reside in the TSM_CTRL register.

From idle state, the asserting of any initiating event (*source*_tx_en or *source*_rx_en), will begin the TSM WARMUP phase and begin incrementing the TSM counter from 0. The initiating source needs to hold the initiating signal asserted (high) through the course of the WARMUP, and then for the duration of the desired ON phase. Deasserting the initiating signal transitions the TSM to the WARMDOWN phase (see Section “Sequence Termination”).

Needless to say, from idle state, a TX-initiating event (assertion of any TSM *source*_tx_en input) will launch a TX sequence. The TSM output **tx_mode** will go high to indicate a TX sequence is underway. Also, from idle state, an RX-initiating event (assertion of any TSM *source*_rx_en input) will launch an RX sequence. The TSM output **rx_mode** will go high to indicate a RX sequence is underway.

Only 1 initiating source can be allowed to assert at any given time. Under no circumstances should a TX and RX initiating source be asserted simultaneously.

Once a TSM sequence has been launched, the WARMUP phase is entered and TSM counter incrementing will commence. Up-counting will continue until one of the following conditions is met:

1. Deassertion of the initiating event. TSM will transition to WARMDOWN phase
2. tsm_count=END_OF_*SEQ*_WU, where *SEQ*=TX or RX. TSM will transition to ON phase
3. Breakpoint. TSM stay in its current phase and hold its count

A transition from WARMUP to ON phase means TSM will hold its count at END_OF_*SEQ*_WU, where *SEQ*=TX or RX.

A transition from ON to WARMDOWN phase means TSM will cease its hold, and resume counting at END_OF_*SEQ*_WU+1, where *SEQ* = TX or RX. (see Section “Sequence Termination”).

A transition from WARMUP to WARMDOWN phase means the tsm_count will jump to END_OF_*SEQ*_WU+1 (where *SEQ* = TX or RX), and resume counting from there. (see Section “Sequence Termination”).

The 4 8-bit registers which control the duration of the WARMUP phase (END_OF_TX_WU[7:0] and END_OF_RX_WU[7:0]), and the duration of the WARMDOWN phase (END_OF_TX_WD[7:0] and END_OF_RX_WD[7:0]), reside in the END_OF_SEQ register.

55.4.7.7.2.5 Sequence Termination

A sequence may be terminated during the WARMUP on ON phase by deassertion of the initiating event (*source*_tx_en or *source*_rx_en). This will cause a transition to WARMDOWN phase, at which point the TSM counter will resume counting from END_OF_*SEQ*_WU+1 (where *SEQ*=TX or RX).

NOTE

For TX, deassertion of the initiating tx_en event does not cause an immediate transition to TSM WARMDOWN phase. The tx_en deassertion is first recognized by the TXDIG to ramp-down the PA. After the TXDIG completes the PA ramp-down, its tx_en_extend output to the TSM will de-assert causing the TSM to transition to WARMDOWN. See more on PA ramp timing in the next section.

55.4.7.7.2.6 PA Ramp Timing

In previous generation (Gen3.0) of radios, the PA ramp was controlled in the TSM. The PA ramp-up was started near the end of the TSM TX warmup sequence and completed before the end of the TSM TX warmup sequence. The PA ramp-down started at beginning of the TSM TX warmdown sequence and completed before the end of the TSM TX warmdown sequence. In addition to the PA ramp timing, the TSM also output the PA ramp values from a programmable look-up table.

For the latest generation of radio, the PA ramp will be controlled in the TXDIG instead of the TSM. The operation is described as follows.

When a Link Layer asserts *source*_tx_en (or software sets the FORCE_TX_EN bit) to start a TX burst, the TSM TX warmup sequence will begin. Near the end of the TSM TX warmup sequence, a TSM timing signal (tx_dig_en) will signal to the TXDIG to ramp-up the PA and start sending TX data. The TXDIG will also control the PA power-up signal (seq_tx_pa_pup in figure below) to the analog to turn on the PA. After the TXDIG completes the PA ramp-up, it outputs pa_wu_complete to the TSM, and the TSM in turn creates 1us-width pulses tsm_wu_almost_complete and tsm_wu_complete. Note that the TSM count may or may not have reached end_of_tx_wu by the time the TXDIG finishes ramping up the PA; in the example below, the PA ramp-up does not complete until after TSM count has reached end_of_tx_wu. Note also that the figure shows the link layer (or force tx_en signal) is a direct input to the TXDIG, and the TXDIG outputs a tx_en_extend (which asserts at the same time as tx_en) to the TSM; the TXDIG takes no action on assertion of tx_en but the de-assertion is used in ramp-down as described in the next paragraph.

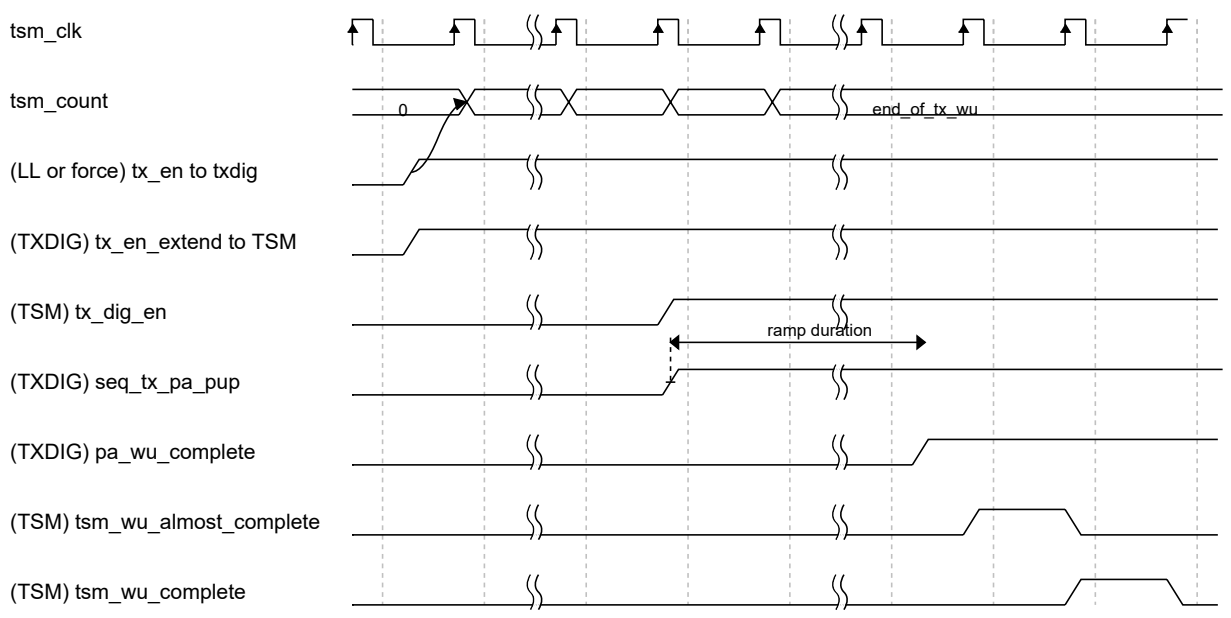


Figure 334. PA Ramp-Up Timing

When the Link Layer deasserts *source*_tx_en (or software clears the FORCE_TX_EN bit) to end a TX burst, the TXDIG will begin to ramp-down the PA. The TSM will not begin its warmdown until after the TXDIG has completed ramp-down of the PA. The TXDIG will provide an output signal (tx_en_extend) to the TSM which asserts at the same time as *source*_tx_en but does not de-assert until after the PA ramp-down has completed by the TXDIG. At that time, the TSM will complete its TX warm-down sequence (deasserting tx_dig_en, and turning off the PLL and other circuits in the analog, etc.) . Since the PA ramp-down is handled in the TXDIG, the TSM TX warm-down timing is expected to be shorter than in previous generation (C90) radios.

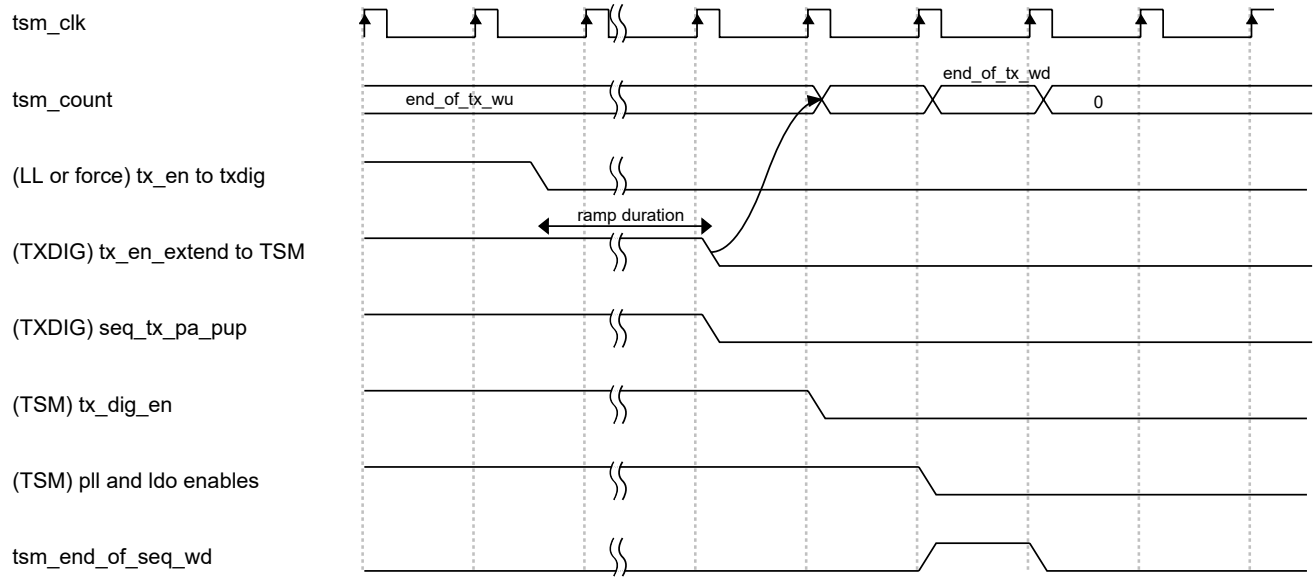


Figure 335. PA Ramp-Down Timing

For all link layers, if a TX PLL unlock or RF_NOT_ALLOWED condition occurs the link layer will need to de-assert the tx_en. Then the PA ramp-down and TSM TX warmdown will proceed as described in the figure above. (See also PLL Unlock and RF_NOT_ALLOWED Conditions)

55.4.7.7.2.7 Overrides

Each of the TSM-controlled outputs has independent override control, except for the 4 GPIO triggers. In addition, two special TSM outputs, tx_mode and rx_mode, also have their own override controls.

Two register bits are assigned to each output, to implement the override function. The bits are named:

*OUTPUTNAME*_OVRD_EN

*OUTPUTNAME*_OVRD

Setting the *OUTPUTNAME*_OVRD_EN=1, allows the *OUTPUTNAME*_OVRD bit to directly control the state of the output. Clearing *OUTPUTNAME*_OVRD_EN=0 returns control of the output to the TSM.

The override bit-pairs reside in the TSM_OVRDyy registers.

Overrides have no effect on TSM operation, TSM counting, sequence initiation, or sequence termination. Overrides may be engaged at any time, during any TSM phase, including during idle state.

55.4.7.7.2.8 Breakpoint

Breakpoint can be used to temporarily suspend a TSM sequence. A breakpoint can be set during the WARMUP or WARMDOWN phase of any TSM sequence. An 8-bit BKPT[7:0] register field resides in the TSM_CTRL register. During a TSM sequence, when the TSM counter matches the BKPT register value, counting stops and the counter holds in its current state. Deassertion of the initiating event will be ignored during the breakpoint. The breakpoint can be lifted by modifying the BKPT[7:0] register. Once the breakpoint is lifted, the TSM proceeds in the phase it was in prior to the breakpoint match, and the TSM counter begins incrementing again from the point at which the breakpoint occurred. Deassertion of the initiaing event will then be recognized and handled as described in Sequence Termination. The default value for BKPT[7:0] is 255, which is greater than the length of any allowed sequence, so a breakpoint will not trigger during a sequence, unless a lower value is programmed.

55.4.7.7.2.9 PLL Unlock and RF_NOT_ALLOWED Conditions

In some situations, it may be desirable to abort an RX or TX sequence. The TSM can detect these situations and alert the link layers. The link layer can then decide to de-assert its rx_en and tx_en outputs to the TSM which causes the sequence to end.

55.4.7.7.2.9.1 PLL Unlock

The TSM can be configured to detect and signal the link layer for any of the following PLL unlock conditions:

1. Coarse Tune Unlock
2. Frequency Target Unlock

The TSM is configured to react to PLL unlock condition via the TSM CTRL register's , ABORT_ON_FREQ_TARG, ABORT_ON_CTUNE,RX_ABORT_DIS and TX_ABORT_DIS bits. Additionally, the table below shows the TSM controlling signal for each the mechanism.

Lock Detect Mechanism	TSM Controlling Signal
Coarse Tune	pll_dig_en (indirect)
Freq. Target	freq_targ_ld_en

55.4.7.7.2.9.2 RF_NOT_ALLOWED

As of the Gen4.5 radio, the RF_NOT_ALLOWED is not qualified by the TSM. The RF_NOT_ALLOWED will signal will be used directly by the link layer(s), which will take appropriate action to de-assert their tx_en or rx_en signal as needed.

55.4.7.7.2.10 TSM RF_ACTIVE Extension

The TSM RF_ACTIVE can be used as the radio's RF_ACTIVE coexistence output. As such, this signal needs to be asserted high a programmable amount of time before a radio event takes place and remains high until either the radio event has been completed or aborted. There should be no glitches/unwanted transitions between successive radio events (e.g., RX->TX or TX->RX). This can be a problem for protocols, such as Bluetooth LE or 802.15.4, which require the transmission of an acknowledge packet shortly after reception and reception of an acknowledge packet shortly after transmission. Bluetooth LE Advertising and Scan states offer other scenarios where closely-spaced (but not adjoining) TX and RX operations are required. The inter-frame spacing between the successive RX and TX operations is long enough such that TSM RF_ACTIVE would normally deassert and reassert between operations, creating the unwanted transitions which must be avoided.

To remedy this, a feature has been added to TSM which allows the TSM RF_ACTIVE output to be optionally extended by a programmable amount, after the nominal TSM sequence completes. The programmable limit is long enough to accommodate the RF_ACTIVE glitch which would otherwise occur using the 802.15.4 192µs turnaround time. The TSM RF_ACTIVE extension will occur for all sequences (TX and RX) for which TSM RF_ACTIVE is programmed to assert, whenever register TSM_SPARE1_EXTEND>0. The hardware mechanism for extending TSM RF_ACTIVE is illustrated in the following diagram:

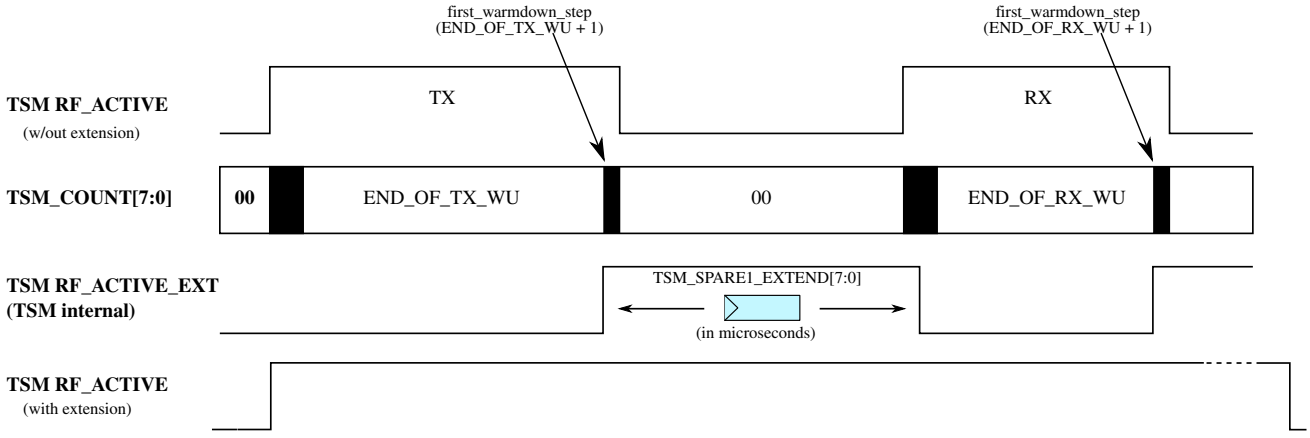


Figure 336. TSM RF_ACTIVE EXTENSION

The trigger for the extension is the TSM's first warmdown step (END_OF_xx_WU + 1), which will be reached whenever the Link Layer deasserts its TX/RX command. If enabled, the extension signal will assert (high) at this point, and will remain high for *TSM_SPARE1_EXTEND* microseconds. The extension signal (TSM RF_ACTIVE_EXT) is logically "OR-ed" with the existing TSM-controlled output TSM RF_ACTIVE, to generate the composite, extended RF_ACTIVE signal. Software can cut short the extension at any time by writing TSM_SPARE1_EXTEND=0. Setting this register to 0 also disables the extension feature from activating on any new TSM sequence. The register TSM_SPARE1_EXTEND[7:0] yields an extension range from 0 to 255µs.

55.4.7.7.2.11 Low Power Preamble Search (LPPS)

To reduce power consumption during 802.15.4 preamble-search receive state, a low-power-preamble-search (LPPS) mode can be optionally enabled. When engaged, the correlators are disabled approximately 50% of the time during preamble search. The same signal used to dynamically enable/disable the correlators, can be used to enable/disable selected analog and RF blocks in tandem, to further reduce power consumption. The LPPS_CTRL register in XCVR space, has one bit to master-enable LPPS mode (LPPS_ENABLE), and several other bits to allow selected analog and RF blocks to be “duty-cycled” in sync with the correlators. For more details on the LPPS algorithm, see Chapter Multi-standard PHY, Section *IEEE Low-power Preamble Search* for more details on the LPPS algorithm.

By default, LPPS mode is not enabled.

During LPPS mode, correlators and RF blocks are duty-cycled with timing controlled by the LPPS signal **lpps_lp_en**, which is a single-wire output from the 802.15.4 demodulator. In general, eligible blocks will be powered off when **lpps_lp_en** is high, and perform normal operation when **lpps_lp_en** is low. Transitions on **lpps_lp_en** will be handled by the TSM in the following ways:

lpps_lp_en 1 → 0:

1. TSM outputs which are programmed to be eligible for LPPS duty-cycling (e.g, those associated with TZA, BBA, ADC) shall be reasserted at this point.
2. TSM shall recycle to a programmable point in the RX warmup. The TSM RECYCLE_COUNT2[7:0] register selects the TSM_COUNT to recycle to.
3. After jumping to the TSM_COUNT indicated in RECYCLE_COUNT2, TSM shall from this point forward execute a Fast RX Warmup (see Section FAST TSM Fast Warmups later in this chapter). For the Fast RX Warmup, the registers LPPS_START_RX[7:0] and LPPS_DEST_RX[7:0] shall be used to program the TSM "jump" start and end points, as defined in the Fast RX Warmup section. The LPPS_START_RX and LPPS_DEST_RX registers reside in the LPPS_CTRL register. These registers are specific to LPPS fast warmups, and do not apply to non-LPPS fast warmups.

lpps_lp_en 0 → 1:

1. TSM outputs which are programmed to be eligible for LPPS duty-cycling (e.g, those associated with TZA, BBA, ADC) shall be deasserted at this point.
2. TSM outputs which are not eligible for LPPS duty-cycling shall remain asserted.

Note that several LPPS timing parameters are configurable (see Chapter Multi-standard PHY, Section *IEEE Low-power Preamble Search*), and LPPS_FILL_COUNT[6:0] is based on the LPPS fast warmup time. Software configuring LPPS parameters, such as LPPS_FILL_COUNT, must calculate the LPPS fast warmup time, in µs, based on the register programming described in the section above. Here is the equation:

LPPS_FAST_WARMUP_DURATION = END_OF_RX_WU - RECYCLE_COUNT2 + 1 - (LPPS_DEST_RX - LPPS_START_RX - 1)

for example,

LPPS_FAST_WARMUP_DURATION = 104 - 28 + 1 - (102 - 40 - 1)

LPPS_FAST_WARMUP_DURATION = 16µs

Note that the penalty for the reduced power consumption in LPPS mode during preamble search, is a slight degradation in receiver sensitivity. Note also that the TSM override bits, also override LPPS functionality. The override bits provide ultimate control over all TSM outputs.

Programming control for LPPS, includes a master-enable register bit for the LPPS state-machine, as well as individual control bits to enable selected analog and digital blocks to be turned on-and-off during LPPS operation. The following table lists the LPPS register bits, and the TSM outputs controlled by each.

Table 457. LPPS Register Controls

LPPS Register Bit	Description
LPPS_ENABLE	Master enable for LPPS mode. Allows correlators to be duty-cycled during Preamble Search, and selected analog and digital blocks to be duty-cycled simultaneously.
LPPS_LNA_MIX_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_rx_lna_pup, seq_rx_mix_pup , seq_spare1 for peak detector
LPPS_CBPF_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_cbpf_pup, seq_cbpf_en_dcoc
LPPS_ADC_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_adc_pup
LPPS_LO_RX_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_lo_pup_vlo_rx
LPPS_LO_RXDRV_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_lo_pup_vlo_rxdrv
LPPS_RX_DIG_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: rx_dig_en
LPPS_RX_PHY_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: rx_phy_en

55.4.7.7.2.12 FAST and POWERSAVE TSM Warmup and FAST RX2TX Transition

The previous sections illustrate the detailed control and timing generated by the TSM, operating in nominal-timing mode. There are situations that allow for the possibility to streamline and shorten the TSM warmup sequences. These situations arise when the transceiver is operating on the same RF channel for many consecutive transmit and/or receive operations. Multiple closely-spaced TX operations on the same channel allow the HPM calibration to be skipped, after such a calibration is performed once on a given RF channel; similarly, multiple closely spaced RX operations on the same channel allow DC offset calibration to be skipped, after such a calibration is performed once on a given RF channel.

Fast and Power-saving TSM warmups have been devised to accomplish this streamlining. Once engaged, during a fast or power-saving warmup, a portion of the nominal, programmed warmup is bypassed, or skipped over, by instructing the TSM

counter to jump, once it reaches a certain count, to a new count that is not just an increment-by-one. The starting point for the “jump” as well as the destination count, or end point for the “jump”, are TSM programmable registers. There are one pair of start/destination registers for TX, and another for RX, allowing one such jump per sequence.

Power-saving warmups can be used in conjuction with any of the link layers. In a power-saving warmup, the PLL sends a pwrsave_tx_wu_ok (or pwrsave_rx_wu_ok) signal to the TSM to let the TSM known that that the PLL is on the same channel as the previous tx (or rx) channel. For a power-saving warmup, the TSM will insert some idle time after the link layer asserts tx_en (or rx_en) so that the overall time from the link layer asserting tx_en (or rx_en) to when the radio is on-air remains constant. The link layer is therefore unaware that a power-saving warmup has occurred. The power savings benefit is that the analog and calibration circuits are on for a shorter during during warmup.

Fast TSM warmup is required by the link layer. Currently, only the Generic LL supports fast TSM warmups. A fast warmup is similar to a power-saving warmup, except that the TSM does not insert any idle time after the link layers tx_en (or rx_en). So the time from the link layer asserting tx_en (or rx_en) to when the radio is on-air is shorter for a fast warmup. A fast warmup request from the link layer has higher priority than a power-saving warmup ok from the PLL. In fact, the TSM does not qualify a fast warmup request with the pwrsave_tx_wu_ok (or pwrsave_rx_wu_ok) signal from the PLL; the Generic LL programmer must ensure that previous warmup calibrations can be skipped before requesting a fast warmup.

Enables bits provide software with fine control over precisely when fast and power-saving warmups are allowed, versus nominal warmups.

In addition, TSM sequences have in the past been discrete TX or RX operations, with a required software command to initiate the sequence, and a return-to-idle after the sequence is complete, before a new sequence-start command can be issued. For some link layers, it is highly desirable to be able to jump directly from the end of a CCA measurement (CCA = Clear Channel Assessment), which is essentially an RX operation, to a packet transmit operation, without returning the TSM to IDLE in between, if CCA deems the channel to be free. A new TSM transition called “Fast RX2TX” accomplishes this. The Generic or 802.15.4 Link Layer will signal to TSM when to perform such a transition, and TSM will respond by jumping from the RX “ON” phase, to a programmable point within the TX warmup. At the very least, this will allow the re-warmup of the required analog regulators to be skipped, since they can stay on during the transition. Also, since the PLL is staying on the same channel, there is no need to cycle the VCO off, and then on again, so the VCO-related TSM controls can be programmed to maintain their states during the transition. Enable bits provide software with fine control over precisely when a “Fast RX2TX” is allowed, versus a nominal RX-to-IDLE-to-TX ordering of events.

Fast RX2TX can also be combined with a Fast TX Warmup to yield an ultrafast TX warmup sequence, if, as is likely, the RF channel has not changed since the previous TX warmup. In this case, the Fast TX warmup is conditioned by the PLL's pwrsave_tx_wu_ok.

Interaction with other blocks is required to support TSM fast warmup capability. The PLL digital block is responsible for selecting the correct frequency for RF operations, and will keep track of whether consecutive warmup operations utilize the same channel or not; it will signal this information to the TSM. The PLL digital block will maintain such tracking separately for TX and RX warmups. The TSM will then signal back to the PLL digital block when a TX warmup is actually “fast” or not; the PLL digital block will use this information to instruct its state machine to skip HPM calibration. For RX warmups, the TSM will signal to the RX digital block, when to skip DC calibration, and instead use stored values.

The new inter-block signaling to support fast TSM warmups is illustrated in the following diagram, followed by a table which describes the new signaling.

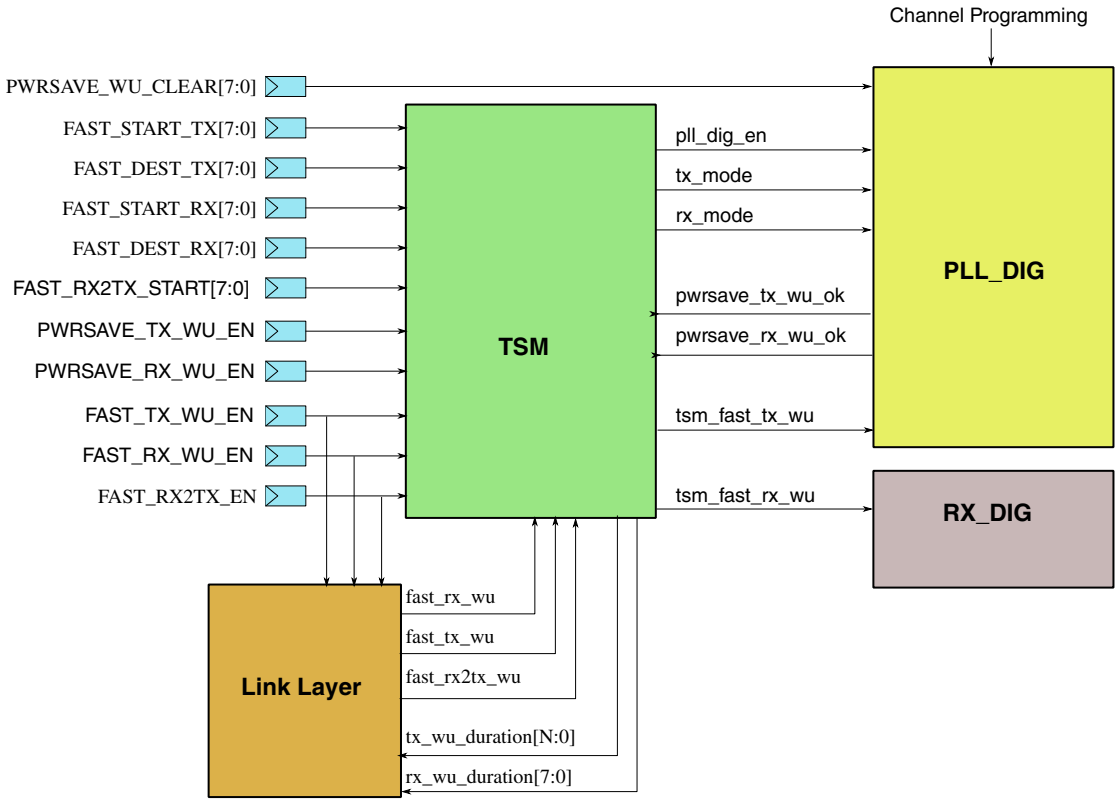


Figure 337. XCVR SIGNALLING TO SUPPORT TSM FAST WARMUPS

"Fast Warmup" Signal	Source	Destination	Description
pll_dig_en	TSM	PLL Digital	Rising edge of this signal used by PLL_DIG to capture the Channel Programming for the current warmup
tx_mode	TSM	PLL Digital	Used in conjunction with pll_dig_en to capture the Channel Programming for TX sequences
rx_mode	TSM	PLL Digital	Used in conjunction with pll_dig_en to capture the Channel Programming for RX sequences
fast_rx_wu	Generic LL	TSM	Link layer requests a Fast RX warmup
fast_tx_wu	Generic LL	TSM	Link layer requests a Fast TX warmup
fast_rx2tx_wu	ZSM (802.15.4) or Generic LL	TSM	The RF channel is free to transmit immediately after

Table continues on the next page...

Table continued from the previous page...

"Fast Warmup" Signal	Source	Destination	Description
			an RX burst. TSM should jump directly from RX "ON" phase to TX "WARMUP" phase, entering at the FAST_RX2TX_START point.
pwrsave_rx_wu_ok	PLL Digital	TSM	Indicates that the Channel Programming on the *current* RX warmup has not changed since the *last* RX warmup
pwrsave_tx_wu_ok	PLL Digital	TSM	Indicates that the Channel Programming on the *current* TX warmup has not changed since the *last* TX warmup
tsm_fast_tx_wu	TSM	PLL Digital	This is a fast or power-saving TX warmup. PLL_DIG should skip calibration.
tsm_fast_rx_wu	TSM	RX Digital	This is a fast or power-saving RX warmup. RX Dig should skip DCOC Cal.
rx_warmup_duration[7:0]	TSM	Link Layer	Indicates the RX warmup time.
tx_warmup_duration[7:0]	TSM	Link Layer	Indicates the TX warmup time, including PA ramp time

TSM Registers to support Fast TSM Warmups and Fast RX2TX are described in the following table.

Field	R/W	Description
PWRSAVE_TX_WU_EN	rw	1: Enable Power-Saving TX Warmup (if pwrsave_tx_wu_ok asserted by PLL_DIG) 0: Disable Power-Saving TX Warmups
PWRSAVE_RX_WU_EN	rw	1: Enable Power-Saving RX Warmup (if pwrsave_rx_wu_ok asserted by PLL_DIG) 0: Disable Power-Saving RX Warmups
PWRSAVE_WU_CLR	rw	This is not used directly by the TSM. It creates a signal to the PLL to clear its wu_ok. It should

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		be set and then cleared by software. 1: Direct the PLL to clear the *wu_ok signals 0: Do nothing
FAST_TX_WU_EN	rw	1: Enable Fast TX Warmup (if fast_tx_wu_ok asserted by PLL_DIG) 0: Disable Fast TX Warmups
FAST_RX_WU_EN	rw	1: Enable Fast RX Warmup (if fast_rx_wu_ok asserted by PLL_DIG) 0: Disable Fast RX Warmups
FAST_RX2TX_EN	rw	1: Enable Fast RX-to-TX transitions (if fast_rx2tx_wu is asserted by ZSM) 0: Disable Fast RX-to-TX transitions
FAST_START_TX[7:0]	rw	During a Fast or Power-Saving TX Warmup, this value represents the starting TSM count for the portion of the warmup that is to be skipped over. If Fast or Power-Saving TX Warmups are enabled, this count will never be reached; its place in the TSM warmup will be replaced by the contents of FAST_DEST_TX[7:0], thereby executing the jump.
FAST_DEST_TX[7:0]	rw	During a Fast or Power-Saving TX Warmup, this value represents the ending TSM count for the portion of the warmup that is to be skipped over. If Fast or Power-Saving TX Warmups are enabled, FAST_START_TX[7:0] will never be reached and will instead be replaced by the contents of this register, thereby executing the jump. Example: FAST_START_TX = 10 FAST_DEST_TX = 15

Table continues on the next page...

Table continued from the previous page...

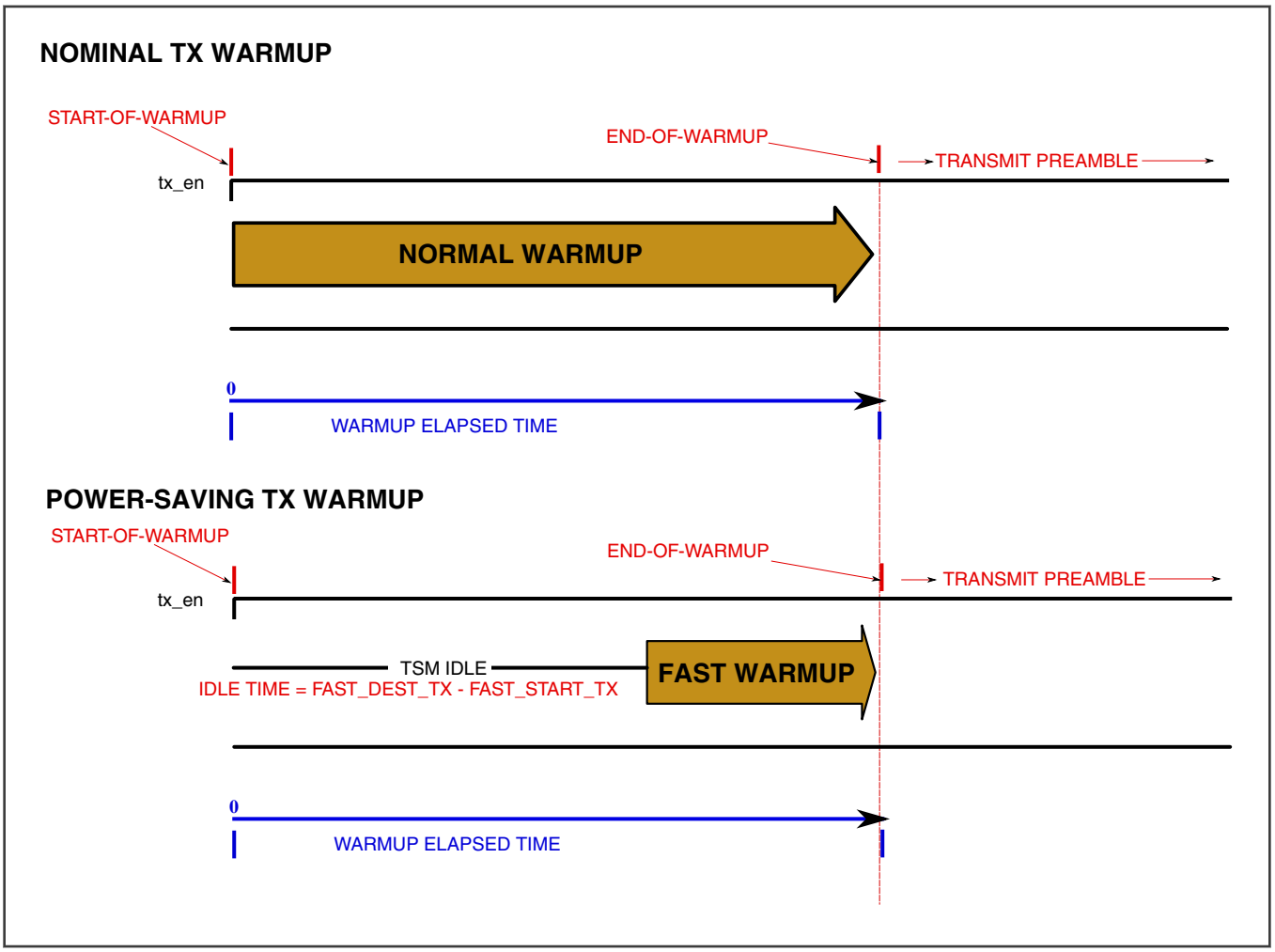
Field	R/W	Description
		The TSM will count as follows: … 7,8,9,15,16,17 …
FAST_START_RX[7:0]	rw	During a Fast or Power-Saving RX Warmup, this value represents the starting TSM count for the portion of the warmup that is to be skipped over. If Fast or Power-Saving RX Warmups are enabled, this count will never be reached; its place in the TSM warmup will be replaced by the contents of FAST_DEST_RX[7:0], thereby executing the jump.
FAST_DEST_RX[7:0]	rw	During a Fast or Power-Saving RX Warmup, this value represents the ending TSM count for the portion of the warmup that is to be skipped over. If Fast or Power-Saving RX Warmups are enabled, FAST_START_RX[7:0] will never be reached and will instead be replaced by the contents of this register, thereby executing the jump. Example: FAST_START_TX = 10 FAST_DEST_TX = 15 The TSM will count as follows: … 7,8,9,15,16,17 …
FAST_RX2TX_START[7:0]	rw	During a Fast RX-to-TX transition (FAST_RX2TX_EN=1) the contents of this register represent the entry point (TSM count) in the TX warmup sequence to which the TSM jumps to when Link Layer asserts fast_rx2tx_wu. The TSM will jump from the RX “ON” phase to the TX “WARMUP” phase, starting at this count.

Software running on the various protocol engines which utilize the TSM to provide access to the channel, need not be aware that the total warmup time has been reduced, when the TSM executes a power-saving warmup, as opposed to a nominal warmup. (With the exception of the Generic LL), Link Layer software expects TSM warmup times to be a fixed constant, actually 2 fixed constants, one for TX warmups and one for RX warmups. Software must account for these TX and RX warmup times when they are scheduling transceiver operations, so that the first symbol of preamble is transmitted at the correct instant for transmit operations, and that the receiver is ready to receive the first symbol of preamble at the correct instant for receive operations. Software views these warmup times as fixed lead times that must be accounted for in advance of a transceiver operation.

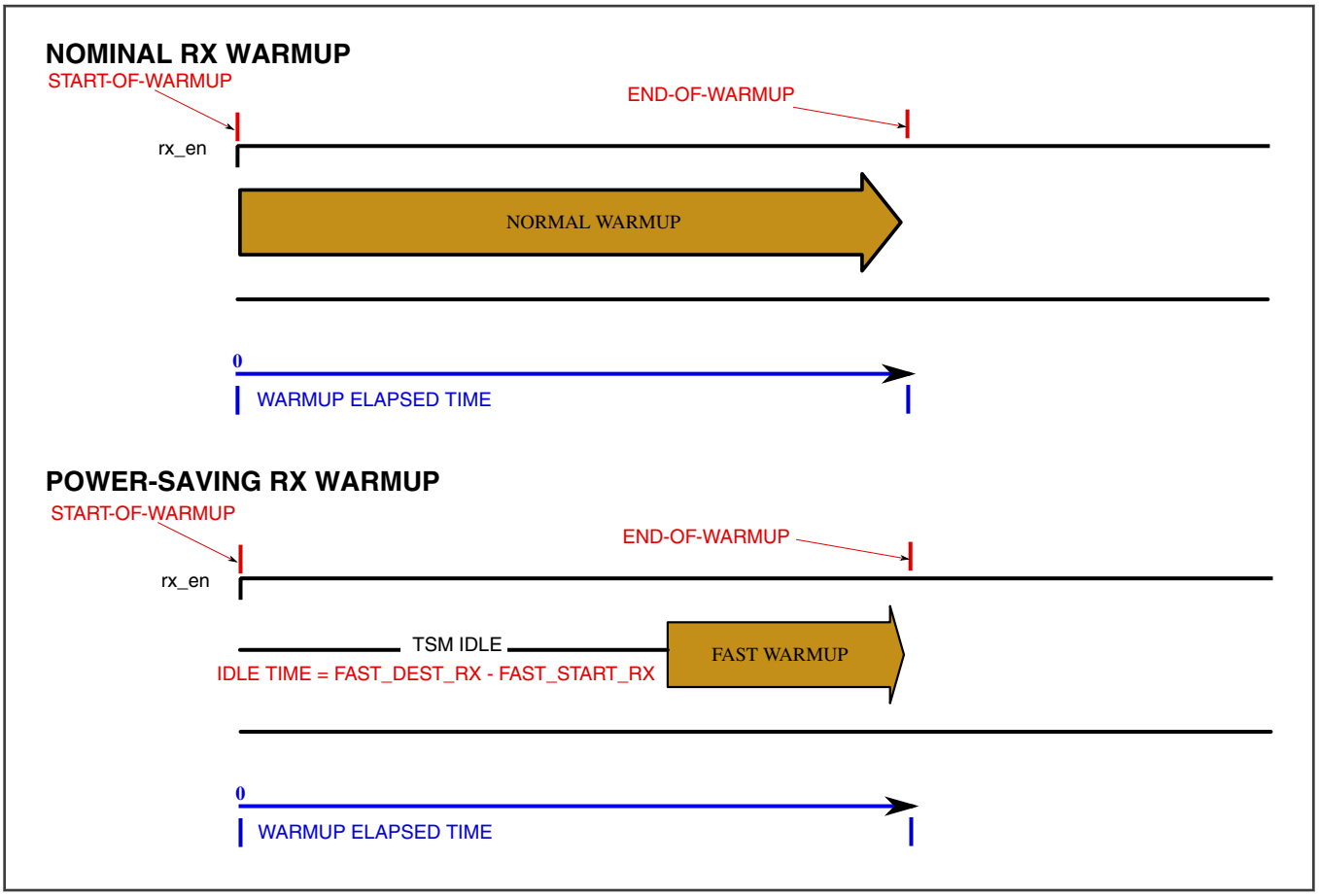
In order to make the power-saving warmup time reduction transparent to software, when a power-saving TSM warmup is executed, the TSM will delay the start of the warmup, by the amount of time skipped by the jump. That keeps the total elapsed time,

from the instant the software launches the TSM sequence, to the point at which the warmup completes, identical for power-saving and nominal warmups.

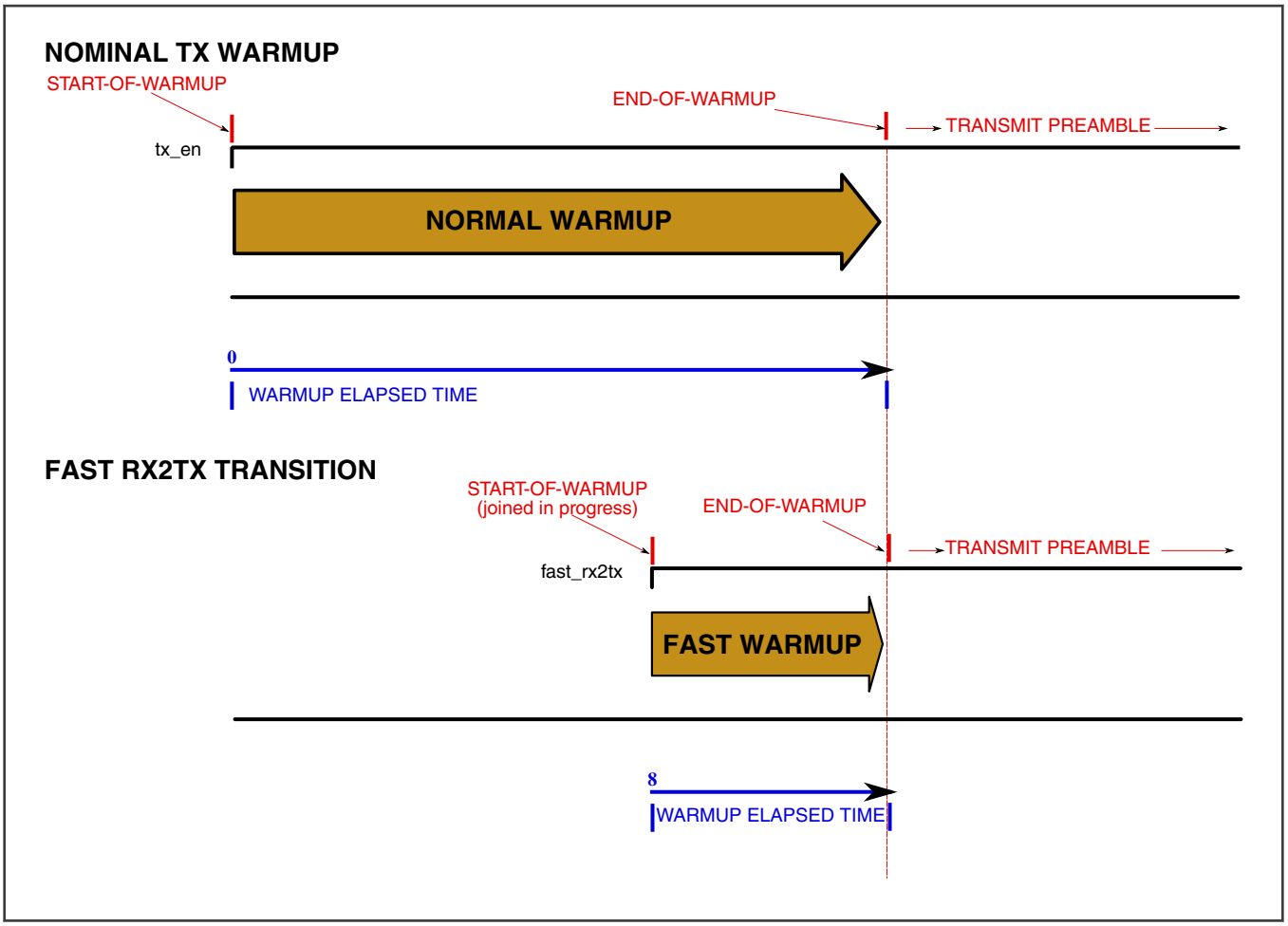
The following diagram illustrates how the TSM delays the start of the Power-saving TX sequence, relative to the nominal TX sequence, so that the delta time between tx_en (TSM launch signal), and the transmission of the first bit of preamble, is the same in either case.



The following diagram illustrates how the TSM delays the start of the Power-saving RX sequence, relative to the nominal RX sequence, so that the delta time between rx_en (TSM launch signal), and the ability to receive first bit of preamble, is the same in either case.



The FAST RX2TX transition is handled differently. The objective in such a transition is to get to the point of transmitting the first bit of preamble ASAP. There is no warmup time that software needs to account for in this case, since the RX-to-TX transition is handled completely by hardware. The following diagram compares a nominal TX warmup, with a foreshortened “Fast RX2TX” warmup.



55.4.7.7.2.13 FAST Transition Support for Ranging Sequence Manager

For the Ranging Sequence Manager (RSM), the TSM supports the following additional features:

- Ability for the RSM to request Fast RX-to-TX transition described in previous section. The RSM outputs a fast_rx2tx_wu signal to the TSM to request this transition
- Fast TX-to-RX feature, analogous to the Fast RX-to-TX feature. The RSM outputs a fast_tx2rx_wu signal to the TSM to request this transition.
- An alternate set of START times for the Fast RX-to-TX and Fast TX-to-RX features. This allows the RSM to use the primary set of START times for its IP (Interlude Period) transitions and the alternate set for its FC (Frequency Change) transitions. The RSM outputs a signal fast_fc_wu to the TSM to select which set to use.
- Ability for the RSM to request FAST RX in conjunction with Fast TX-to-RX transiiton. These use the FAST_START_RX[7:0] and FAST_DEST_RX[7:0] as described in the previous section. The RSM outputs a fast_rx_wu signal to request FAST RX.
- Ability for the RSM to request FAST TX in conjunction with Fast RX-to-TX transiiton. These use the FAST_START_TX[7:0] and FAST_DEST_TX[7:0] as described in the previous section. The RSM outputs a fast_tx_wu signal to request FAST TX.

Refer to Role Swap and Frequency Change Transitions section in the Ranging Sequence Manager topic for more information about RSM use of these TSM features.

The TSM Registers added to support these additional features for the RSM are described in the following table.

Field	R/W	Description
FAST_TX2RX_EN	rw	1: Enable Fast TX-to-RX transitions (if fast_tx2rx_wu is asserted by RSM) 0: Disable Fast TX-to-RX transitions
FAST_TX2RX_START[7:0]	rw	During a Fast TX-to-RX transition (FAST_TX2RX_EN=1) the contents of this register represent the entry point (TSM count) in the RX warmup sequence to which the TSM jumps to when RSM asserts fast_tx2rx_wu with fast_fc_wu=0. The TSM will jump from the TX “ON” phase to the RX “WARMUP” phase, starting at this count.
FAST_RX2TX_START_FC[7:0]	rw	During a Fast RX-to-TX transition (FAST_RX2TX_EN=1) the contents of this register represent the entry point (TSM count) in the TX warmup sequence to which the TSM jumps to when RSM asserts fast_rx2tx_wu with fast_fc_wu=1. (If fast_fc_wu=0, the TSM will use FAST_RX2TX_START[7:0]) The TSM will jump from the RX “ON” phase to the TX “WARMUP” phase, starting at this count.
FAST_TX2RX_START_FC[7:0]	rw	During a Fast TX-to-RX transition (FAST_TX2RX_EN=1) the contents of this register represent the entry point (TSM count) in the RX warmup sequence to which the TSM jumps to when RSM asserts fast_tx2rx_wu with fast_fc_wu=1. (If fast_fc_wu=0, the TSM will use FAST_TX2RX_START[7:0]) The TSM will jump from the TX “ON” phase to the RX “WARMUP” phase, starting at this count.

55.4.7.7.2.14 Programming Restrictions

The following restrictions are imposed on TSM programming. Violating the restrictions may result in unpredictable behavior.

1. The maximum value for END_OF_TX_WU[7:0] is 253
2. The maximum value for END_OF_RX_WU[7:0] is 253
3. The maximum value for END_OF_TX_WD[7:0] is 254
4. The maximum value for END_OF_RX_WD[7:0] is 254
5. END_OF_TX_WD[7:0] – END_OF_TX_WU[7:0] >= 1

6. END_OF_RX_WD[7:0] – END_OF_RX_WU[7:0] >= 1
7. BKPT[7:0] != END_OF_TX_WD[7:0]
8. BKPT[7:0] != END_OF_RX_WD[7:0]
9. For any TSM output: OUTPUTNAME_TX_HI[7:0] <= OUTPUTNAME_TX_LO[7:0]
10. For any TSM output: OUTPUTNAME_RX_HI[7:0] <= OUTPUTNAME_RX_LO[7:0]

Restrictions 9 and 10 above, if violated, will result in the TSM-controlled output held in a deasserted state throughout the sequence.

To guarantee non-assertion of a particular TSM-controlled output throughout the course of a sequence, simply program its OUTPUTNAME_SEQ_HI = OUTPUTNAME_SEQ_LO = 255 (where SEQ = TX or RX).

55.4.7.7.2.15 Interrupts

For debug purposes, TSM has two timing control outputs (tsm_irq0_start_trigger_en and tsm_irq1_stop_trigger_en) which can be used to either generate interrupts to the host processor. Like all other TSM-controlled outputs, the point during the TX or RX warmup at which the interrupt is triggered, is fully programmable.

To use either tsm_irq0_start_trigger_en or tsm_irq1_stop_trigger_en as an interrupt, set the TSM_IRQ0_EN or TSM_IRQ1_EN bit, respectively, in the TSM_CTRL registers. Because the TSM is part of XCVR space, and XCVR space does not have its own interrupt vector, the TSM interrupt is "assigned" to the Link Layer currently in possession of the radio. Each Link Layer controller has in its address space, a TSM_IRQ status bit, as well as a enable bit to allow/disallow the TSM interrupt to assert the Link Layer's interrupt line to the MCU. The TSM_IRQ status bit, as it appears in each Link Layer's address space, is a logical "OR" of the TSM's TSM0_IRQ and TSM1_IRQ. The Link Layer TSM_IRQ is read-only. It can be cleared by clearing both TSM0_IRQ and TSM1_IRQ in the XCVR's XCVR_STATUS register. In the XCVR_STATUS register, TSM0_IRQ and TSM1_IRQ are both write-1-to-clear bits. See the Register Description for each individual Link Layer, for a description of the TSM_IRQ read-only interrupt status bit, and its bit positioning within the Link Layer's address space.

The tsm_irq0_start_trigger_en can also be used as a start trigger source for the Transceiver DMA or Packet RAM Debug mode. The tsm_irq1_stop_trigger_en can also be used as a stop trigger source for the Packet RAM Debug mode.

55.4.7.7.3 Memory Map and Register Definition

55.4.7.7.3.1 XCVR_TSM register descriptions

55.4.7.7.3.1.1 XCVR_TSM_REG memory map

XCVR_TSM base address: 48A0_7800h

Offset	Register	Width (In bits)	Access	Reset value
0h	TSM CONTROL (CTRL)	32	RW	FF00_0400h
4h	TSM CONTROL (LPPS_CTRL)	32	RW	FFFF_0000h
8h	TSM END OF SEQUENCE (END_OF_SEQ)	32	RW	5C5A_7270h
Ch	WARMUP LATENCY (WU_LATENCY)	32	RW	0000_0000h
10h	TSM RECYCLE COUNT (RECYCLE_COUNT)	32	RW	002A_1258h
14h	TSM FAST WARMUP CONTROL 1 (FAST_CTRL1)	32	RW	0000_FF00h
18h	TSM FAST WARMUP CONTROL 2 (FAST_CTRL2)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1Ch	TSM FAST WARMUP CONTROL 3 (FAST_CTRL3)	32	RW	0000_FF00h
20h	TSM_TIMING00 (TIMING00)	32	RW	FFFF_FFFFh
24h	TSM_TIMING01 (TIMING01)	32	RW	FFFF_FFFFh
28h	TSM_TIMING02 (TIMING02)	32	RW	FFFF_FFFFh
2Ch	TSM_TIMING03 (TIMING03)	32	RW	FFFF_FFFFh
30h	TSM_TIMING04 (TIMING04)	32	RW	FFFF_FFFFh
34h	TSM_TIMING05 (TIMING05)	32	RW	FFFF_FFFFh
38h	TSM_TIMING06 (TIMING06)	32	RW	FFFF_FFFFh
3Ch	TSM_TIMING07 (TIMING07)	32	RW	FFFF_FFFFh
40h	TSM_TIMING08 (TIMING08)	32	RW	FFFF_FFFFh
44h	TSM_TIMING09 (TIMING09)	32	RW	5700_FFFFh
48h	TSM_TIMING10 (TIMING10)	32	RW	1108_1108h
4Ch	TSM_TIMING11 (TIMING11)	32	RW	5B13_7113h
50h	TSM_TIMING12 (TIMING12)	32	RW	5B19_714Fh
54h	TSM_TIMING13 (TIMING13)	32	RW	572D_FFFFh
58h	TSM_TIMING14 (TIMING14)	32	RW	0000_716Fh
5Ch	TSM_TIMING15 (TIMING15)	32	RW	5B58_716Fh
60h	TSM_TIMING16 (TIMING16)	32	RW	5958_0000h
64h	TSM_TIMING17 (TIMING17)	32	RW	5B58_0000h
68h	TSM_TIMING18 (TIMING18)	32	RW	5B58_0000h
6Ch	TSM_TIMING19 (TIMING19)	32	RW	1100_1100h
70h	TSM_TIMING20 (TIMING20)	32	RW	1100_1100h
74h	TSM_TIMING21 (TIMING21)	32	RW	1900_1100h
78h	TSM_TIMING22 (TIMING22)	32	RW	5800_6F00h
7Ch	TSM_TIMING23 (TIMING23)	32	RW	5800_6F00h
80h	TSM_TIMING24 (TIMING24)	32	RW	5B00_7100h
84h	TSM_TIMING25 (TIMING25)	32	RW	5C00_7200h
88h	TSM_TIMING26 (TIMING26)	32	RW	5C00_7200h
8Ch	TSM_TIMING27 (TIMING27)	32	RW	5C00_7200h
90h	TSM_TIMING28 (TIMING28)	32	RW	5C00_7200h
94h	TSM_TIMING29 (TIMING29)	32	RW	FFFF_7200h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
98h	TSM_TIMING30 (TIMING30)	32	RW	5C00_FFFFh
9Ch	TSM_TIMING31 (TIMING31)	32	RW	5B08_7108h
A0h	TSM_TIMING32 (TIMING32)	32	RW	1911_FFFFh
A4h	TSM_TIMING33 (TIMING33)	32	RW	2311_5A11h
A8h	TSM_TIMING34 (TIMING34)	32	RW	5B11_7111h
ACh	TSM_TIMING35 (TIMING35)	32	RW	5B11_7111h
B0h	TSM_TIMING36 (TIMING36)	32	RW	5B11_7111h
B4h	TSM_TIMING37 (TIMING37)	32	RW	5B11_7111h
B8h	TSM_TIMING38 (TIMING38)	32	RW	5B11_7111h
BCh	TSM_TIMING39 (TIMING39)	32	RW	5B2B_7111h
C0h	TSM_TIMING40 (TIMING40)	32	RW	FFFF_7111h
C4h	TSM_TIMING41 (TIMING41)	32	RW	FFFF_7111h
C8h	TSM_TIMING42 (TIMING42)	32	RW	5B12_7112h
CCh	TSM_TIMING43 (TIMING43)	32	RW	5B12_FFFFh
D0h	TSM_TIMING44 (TIMING44)	32	RW	5B12_FFFFh
D4h	TSM_TIMING45 (TIMING45)	32	RW	FFFF_7112h
D8h	TSM_TIMING46 (TIMING46)	32	RW	FFFF_7112h
DCh	TSM_TIMING47 (TIMING47)	32	RW	5B13_7157h
E0h	TSM_TIMING48 (TIMING48)	32	RW	5B23_715Ah
E4h	TSM_TIMING49 (TIMING49)	32	RW	5B23_715Ah
E8h	TSM_TIMING50 (TIMING50)	32	RW	5B2B_FFFFh
ECh	TSM_TIMING51 (TIMING51)	32	RW	5B2B_FFFFh
F0h	TSM_TIMING52 (TIMING52)	32	RW	FFFF_FFFFh
F4h	TSM OVERRIDE REGISTER 0 (OVRD0)	32	RW	0000_0000h
F8h	TSM OVERRIDE REGISTER 1 (OVRD1)	32	RW	0000_0000h
FCh	TSM OVERRIDE REGISTER 2 (OVRD2)	32	RW	0000_0000h
100h	TSM OVERRIDE REGISTER 3 (OVRD3)	32	RW	0000_0000h

55.4.7.7.3.1.2 TSM CONTROL (CTRL)

Offset

Register	Offset
CTRL	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BKPT								TSM_SPARE1_EXTEND							
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSM_LL_INHIBIT				PLL_UNL...	PLL_UNL...	TSM_IRQ...	TSM_IRQ...	ABORT_O...	ABORT_O...	RX_ABOR...	TX_ABOR...	FORCE_R...	FORCE_T...	TSM_SOF...	0
W					W1C											
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 BKPT	TSM Breakpoint Temporarily halt a TSM sequence during the warmup or warmdown phase. When the TSM counter matches the value of BKPT[7:0], breakpoint will take effect and the TSM counter will stop and hold its count. Breakpoint will remain in effect as long as BKPT[7:0] matches the TSM counter value. The TSM Breakpoint can be lifted by modifying the contents of this register. The default value of this register, 0xFF, is greater than the length of the longest possible sequence, so a breakpoint will never be triggered unless BKPT[7:0] is programmed to a value less than the length of sequence.
23-16 TSM_SPARE1_EXTEND	TSM RF_ACTIVE Extension Duration When TSM output RF_ACTIVE is enabled to assert during any TX or RX sequence and TSM_SPARE1_EXTEND[7:0] > 0, this register specifies the number of microseconds for which RF_ACTIVE is to remain asserted beyond the end of the TSM warmdown. This extension is intended to close any gap in the coexistence output RF_ACTIVE that may occur between consecutive TX/RX or RX/TX TSM operations.
15-12 TSM_LL_INHIBIT	TSM Per-Link-Layer Inhibit Nominally, any protocol engine can request TSM to start a TX or RX sequence at any time. In a multiprotocol setting, individual protocol engines can (optionally) be selectively blocked from accessing TSM by setting one of the following bits of TSM_LL_INHIBIT[3:0]: xxx1: Bluetooth LE commands to TSM are inhibited xxx0: Bluetooth LE commands to TSM are permitted

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1xxx: GENERIC_FSK commands to TSM are inhibited 0xxx: GENERIC_FSK commands to TSM are permitted
11 PLL_UNLOCK_IRQ	PLL Unlock IRQ PLL Un-lock Interrupt Status bit. A '1' indicates an unlock event has occurred in the PLL. This is write a '1' to clear bit. 0b - A PLL Unlock Interrupt has not occurred 1b - A PLL Unlock Interrupt has occurred
10 PLL_UNLOCK_IRQ_EN	PLL Unlock Interrupt Enable 0b - allows PLL unlock event to generate an interrupt 1b - A PLL unlock event will set the PLL_UNLOCK_IRQ status bit, but an interrupt is not generated
9 TSM_IRQ1_EN	TSM_IRQ1 Enable/Disable bit TSM_IRQ1 Enable/Disable bit. Intended for debug purposes only. 0b - TSM_IRQ1 is disabled 1b - TSM_IRQ1 is enabled
8 TSM_IRQ0_EN	TSM_IRQ0 Enable/Disable bit TSM_IRQ0 Enable/Disable bit. Intended for debug purposes only. 0b - TSM_IRQ0 is disabled 1b - TSM_IRQ0 is enabled
7 ABORT_ON_FREQ_TARG	Abort On Frequency Target Lock Detect Failure 0b - don't allow TSM abort on Frequency Target Unlock Detect 1b - allow TSM abort on Frequency Target Unlock Detect
6 ABORT_ON_CTUNE	Abort On Coarse Tune Lock Detect Failure 0b - don't allow TSM abort on Coarse Tune Unlock Detect 1b - allow TSM abort on Coarse Tune Unlock Detect
5 RX_ABORT_DIS	Receive Abort Disable RX Abort disable. When set, prevents PLL unlock events during RX sequences from aborting the sequence.
4 TX_ABORT_DIS	Transmit Abort Disable TX Abort disable. When set, prevents PLL unlock events during TX sequences from aborting the sequence.
3	Force Receive Enable

Table continues on the next page...

Table continued from the previous page...

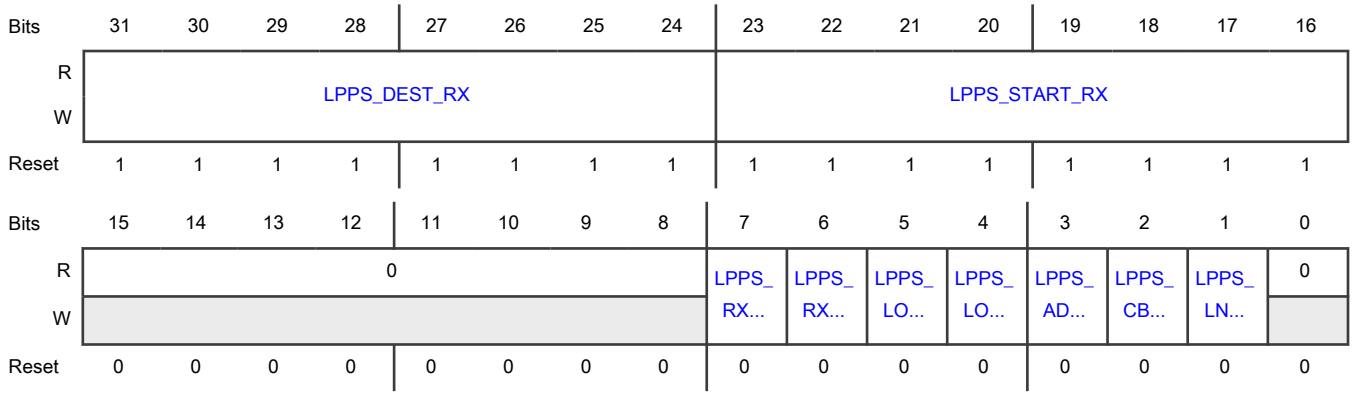
Field	Function
FORCE_RX_EN	Direct software control to launch a RX TSM sequence. Initiates RX Warmup on a 0 to 1 transition and RX Warmdown on a 1 to 0 transition. 0b - TSM Idle 1b - TSM executes a RX sequence
2 FORCE_TX_EN	Force Transmit Enable Direct software control to launch a TX TSM sequence. Initiates a TX Warmup sequence on a 0 to 1 transition and a TX Warmdown sequence on a 1 to 0 transition. 0b - TSM Idle 1b - TSM executes a TX sequence
1 TSM_SOFT_RESET	TSM Soft Reset Writing 1 to this bit returns TSM to its IDLE state, with TSM_COUNT=0. Writing 0 to this bit removes the forced-reset condition and resumes normal operation. The bit is not self-clearing. For contingency purposes only 0b - TSM Soft Reset removed. Normal operation. 1b - TSM Soft Reset engaged. TSM forced to IDLE, and holds there until the bit is cleared.
0 —	Reserved

55.4.7.7.3.1.3 TSM CONTROL (LPPS_CTRL)

Offset

Register	Offset
LPPS_CTRL	4h

Diagram



Fields

Field	Function
31-24 LPPS_DEST_RX	LPPS Fast TSM RX Warmup "Jump-to" Point During a LPPS Fast RX Warmup, this value represents the ending TSM count for the portion of the warmup that is to be skipped over. In an LPPS Fast RX Warmup, LPPS_START_RX[7:0] will never be reached and will instead be replaced by the contents of this register, thereby executing the jump. Example: LPPS_START_RX = 10 LPPS_DEST_RX = 15 The TSM will count as follows: … 7,8,9,15,16,17 …
23-16 LPPS_START_RX	LPPS Fast TSM RX Warmup "Jump-from" Point During a LPPS Fast RX Warmup, this value represents the starting TSM count for the portion of the warmup that is to be skipped over. In an LPPS Fast RX Warmup, this count will never be reached; its place in the TSM warmup will be replaced by the contents of LPPS_DEST_RX[7:0], thereby executing the jump.
15-8 —	Reserved
7 LPPS_RX_PHY_ALLOW	LPPS_RX_PHY_ALLOW When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: rx_phy_en
6 LPPS_RX_DIG_ALLOW	LPPS_RX_DIG_ALLOW When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: rx_dig_en
5 LPPS_LO_RXDRV_ALLOW	LPPS_LO_RXDRV_ALLOW When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_lo_pup_vlo_rxdrv
4 LPPS_LO_RX_ALLOW	LPPS_LO_RX_ALLOW When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_lo_pup_vlo_rx
3 LPPS_ADC_ALLOW	LPPS_ADC_ALLOW When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_adc_pup
2 LPPS_CBPF_ALLOW	LPPS_CBPF_ALLOW When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_adc_pup, seq_cbpf_en_dcoc
1	LPPS_LNA_MIX_ALLOW

Table continues on the next page...

Table continued from the previous page...

Field	Function
LPPS_LNA_MIX_ALLOW	When set to one, allows the following TSM outputs to be controlled by the LPPS state machine: seq_rx_lna_pup, seq_rx_mix_pup, seq_spare1
0 —	Reserved

55.4.7.7.3.1.4 TSM END OF SEQUENCE (END_OF_SEQ)

Offset

Register	Offset
END_OF_SEQ	8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	END_OF_RX_WD								END_OF_RX_WU							
Reset	0	1	0	1	1	1	0	0	0	1	0	1	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	END_OF_TX_WD								END_OF_TX_WU							
Reset	0	1	1	1	0	0	1	0	0	1	1	1	0	0	0	0

Fields

Field	Function
31-24 END_OF_RX_WD	End of RX Warmdown This register defines the point at which the TSM RX sequence warmdown completes, and the TSM returns to idle. The duration of the TSM warmdown phase is determined by: END_OF_RX_WD – END_OF_RX_WU.
23-16 END_OF_RX_WU	End of RX Warmup This register defines the length of the TSM RX warmup sequence. After the assertion of a RX sequence-initiating event, when the TSM counter reaches the count matching this register, it will stop and hold its count, and the TSM will transition from the WARMUP to the ON phase.
15-8 END_OF_TX_WD	End of TX Warmdown This register defines the point at which the TSM TX sequence warmdown completes, and the TSM returns to idle. The duration of the TSM warmdown phase is determined by: END_OF_TX_WD – END_OF_TX_WU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 END_OF_TX_WU	End of TX Warmup This register defines the length of the TSM TX warmup sequence. After the assertion of a TX sequence-initiating event, when the TSM counter reaches the count matching this register, it will stop and hold its count, and the TSM will transition from the WARMUP to the ON phase.

55.4.7.7.3.1.5 WARMUP LATENCY (WU_LATENCY)

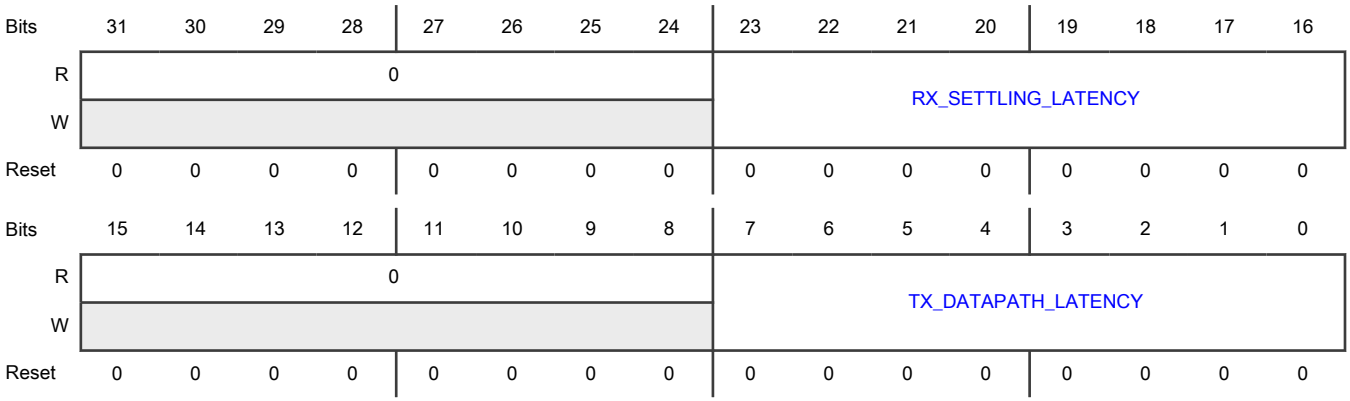
Offset

Register	Offset
WU_LATENCY	Ch

Function

This register contains the RX and TX latency values. These are added to the TSM's warmup duration to provide the Generic link layer a total warmup time.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RX_SETTLING_LATENCY	RX Settling Latency RX_SETTLING_LATENCY allows for a bulk delay value (in microseconds) to be added to the RX TSM warm-up time for link layer scheduling. The programmed value corresponds to the time that the RX data path may need for RX filters and clocks to settle and/or any post-warm-up RX activities that need to be completed before a valid RF signal can be correctly processed.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7-0 TX_DATAPATH_LATENCY	TX Datapath Latency TX_DATAPATH_LATENCY allows for a bulk delay value (in microseconds) to be added to the TX TSM warm-up time for link layer scheduling. This value comprises cumulative TX data path processing latency including contributions by RBME, TX-Dig, PLL and Antenna Interface as per TX configuration.

55.4.7.7.3.1.6 TSM RECYCLE COUNT (RECYCLE_COUNT)

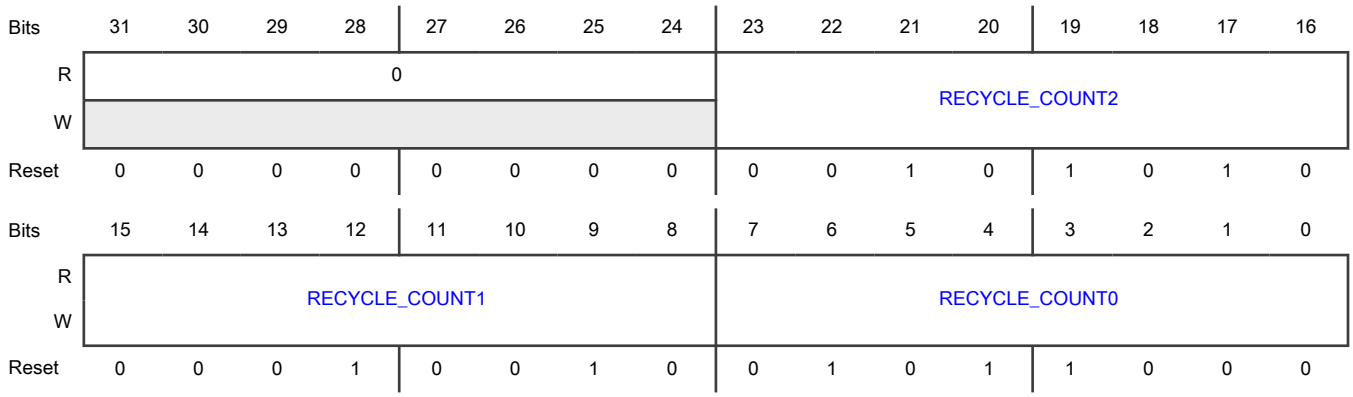
Offset

Register	Offset
RECYCLE_COUNT	10h

Function

This register contains the TSM Recycle "Jump-to" points for the 3 types of TSM Recycle

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RECYCLE_COUNT2	TSM RX Recycle Count 2 The RECYCLE_COUNT2[7:0] register determines the TSM count value to which the TSM "recycles" when, in LPPS mode, an LPPS power save cycle completes and the receiver is switched back on, necessitating a foreshortened RX warmup. The intention is for this register to be programmed to a TSM

Table continues on the next page...

Table continued from the previous page...

Field	Function
	count value that is just prior to the assertion of rx_lna_en, but there are no restrictions on programming this register. An LPPS recycle is a command from the LPPS state machine to TSM, requesting a jump from the TSM ON phase back to a programmable point in the WARMUP phase, and resume counting from there.
15-8 RECYCLE_COUNT1	TSM RX Recycle Count 1 The RECYCLE_COUNT1 register determines the TSM count value to which the TSM "recycles" when a link layer asserts the tsm_recycle[1] signal. The intention is for RECYCLE_COUNT1 to be programmed to a TSM count value such that the TSM de-asserts, and then re-asserts its "pll_dig_en" output, to effectuate a Dual PAN on-the-fly channel change, but there are no restrictions on programming this register. For 802.15.4 Sequence Manager (ZSM), tsm_recycle[1] is associated with the "RX_PAN1" state. For Generic_FSK Link Layer, tsm_recycle[1] is associated with the "RX_PAN1" state.
7-0 RECYCLE_COUNT0	TSM RX Recycle Count 0 The RECYCLE_COUNT0[7:0] register determines the TSM count value to which the TSM "recycles" when the 802.15.4 Sequence Manager (ZSM) state "RX_CYC" is reached and the ZSM asserts tsm_recycle[0] to TSM. This register also determines the TSM count value to which the TSM recycles when the ZSM state RX_CCCA is reached because tsm_recycle[0] is also asserted in this state. This register also determines the TSM count value to which the TSM recycles when the Generic_FSK Link Layer asserts a tsm_recycle to TSM. The intention is for this register to be programmed to a TSM count value such that the TSM re-asserts its "rx_init" output, but there are no restrictions on programming this register. An RX recycle is a command from any protocol engine to TSM, requesting a jump from the TSM ON phase back to a programmable point in the WARMUP phase, and resume counting from there. A recycle will result from the reception of a packet with bad CRC or one which fails packet-filtering rules, or the end of a CCA operation in Continuous CCA mode which results in a channel indicating "busy".

55.4.7.7.3.1.7 TSM FAST WARMUP CONTROL 1 (FAST_CTRL1)

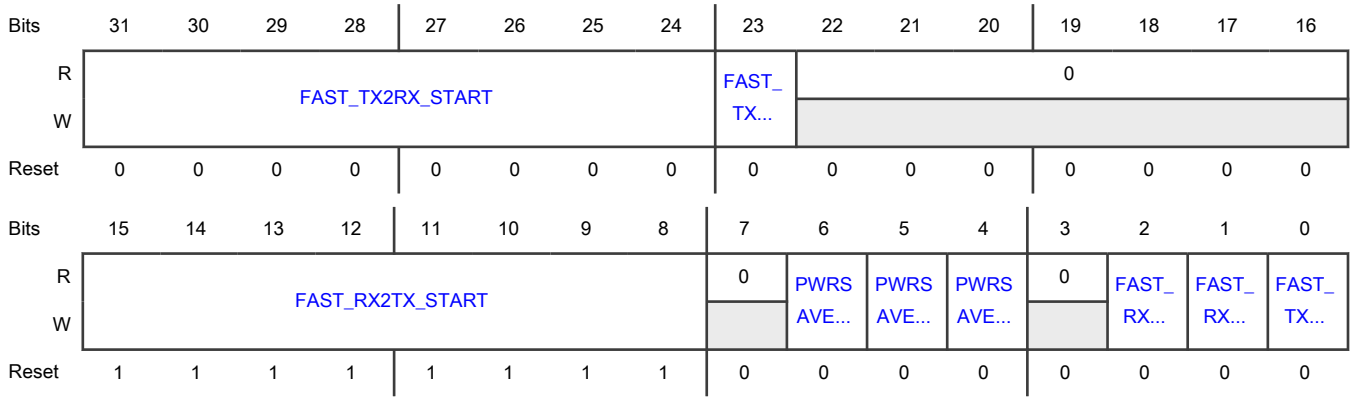
Offset

Register	Offset
FAST_CTRL1	14h

Function

This register provides enabling and control for Fast TSM Mode

Diagram



Fields

Field	Function
31-24 FAST_TX2RX_START	TSM "Jump-to" point for a Fast TSM TX-to-RX Transition. During a Fast TX-to-RX transition (FAST_TX2RX_EN=1), the contents of this register represent the entry point (TSM count) in the RX warmup sequence to which the TSM jumps to when Ranging sequence manager asserts fast_tx2rx_wu. The TSM will jump from the TX “ON” phase to the RX “WARMUP” phase, starting at this count.
23 FAST_TX2RX_EN	Fast TSM TX-to-RX Transition Enable This bit enables a special state transition in the Ranging Sequence Manager, combined with a TSM jump from the TX "ON" phase to the RX "Warmup" phase 0b - Disable Fast TX-to-RX transitions 1b - Enable Fast TX-to-RX transitions (if fast_tx2rx_wu is asserted by Ranging sequence manager)
22-16 —	Reserved
15-8 FAST_RX2TX_START	TSM "Jump-to" point for a Fast TSM RX-to-TX Transition. During a Fast RX-to-TX transition (FAST_RX2TX_EN=1) the contents of this register represent the entry point (TSM count) in the TX warmup sequence to which the TSM jumps to when 802.15.4 ZSM asserts fast_rx2tx_wu. The TSM will jump from the RX “ON” phase to the TX “WARMUP” phase, starting at this count.
7 —	Reserved
6 PWRSAVE_WU_CLEAR	PowerSave TSM Warmup Clear State This bit clears the RF channel memory in the PLL Digital Block, forcing the next TSM TX Warmup to be normal (not powersave), and the next TSM RX Warmup to be normal (not powersave), regardless of whether the RF channel has actually changed or not. This bit is not self-clearing. Write '1' to clear channel memory, then write '0' to proceed with TSM operations.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 PWRSAVE_RX_WU_EN	Power Save TSM RX Warmup Enable 0b - PowerSave TSM RX Warmups are disabled 1b - PowerSave TSM RX Warmups are enabled, if the RF channel has not changed since the last RX warmup.
4 PWRSAVE_TX_WU_EN	Power Save TSM TX Warmup Enable 0b - PowerSave TSM TX Warmups are disabled 1b - PowerSave TSM TX Warmups are enabled, if the RF channel has not changed since the last TX warmup.
3 —	Reserved
2 FAST_RX2TX_EN	Fast TSM RX-to-TX Transition Enable This bit enables a special state transition in the Link Layer, combined with a TSM jump from the RX "ON" phase to the TX "Warmup" phase, to accelerate the CCA-to-TX transition in the Link Layer autosequences 0b - Disable Fast RX-to-TX transitions 1b - Enable Fast RX-to-TX transitions (if fast_rx2tx_wu is asserted by the Link Layer)
1 FAST_RX_WU_EN	Fast TSM RX Warmup Enable 0b - Fast TSM RX Warmups are disabled 1b - Fast TSM RX Warmups are enabled, if the RF channel has not changed since the last RX warmup, and for Bluetooth LE mode, the RF channel is not an advertising channel.
0 FAST_TX_WU_EN	Fast TSM TX Warmup Enable 0b - Fast TSM TX Warmups are disabled 1b - Fast TSM TX Warmups are enabled, if the RF channel has not changed since the last TX warmup, and for Bluetooth LE mode, the RF channel is not an advertising channel.

55.4.7.7.3.1.8 TSM FAST WARMUP CONTROL 2 (FAST_CTRL2)

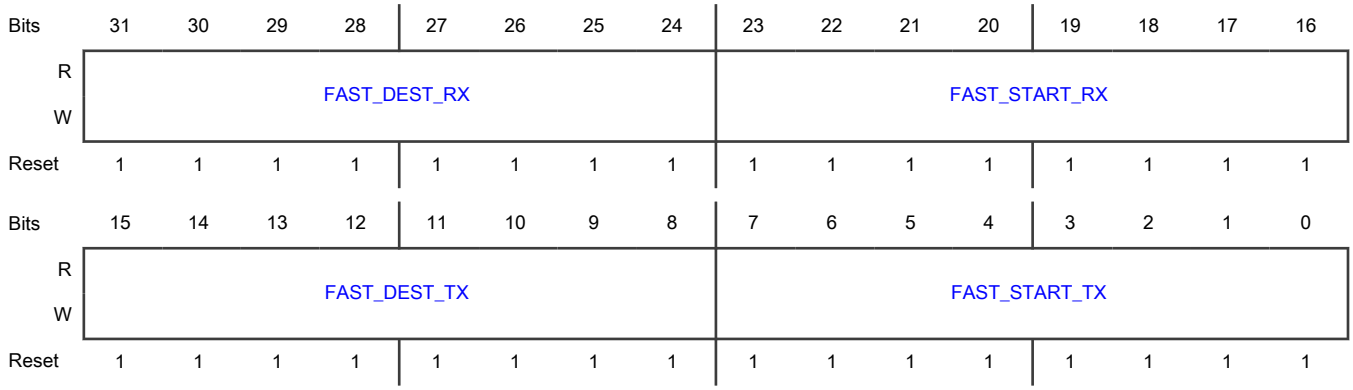
Offset

Register	Offset
FAST_CTRL2	18h

Function

This register provides configuration for Fast TSM Mode

Diagram



Fields

Field	Function
31-24 FAST_DEST_RX	Fast TSM RX "Jump-to" Point During a Fast RX Warmup, this value represents the ending TSM count for the portion of the warmup that is to be skipped over. If Fast RX Warmups are enabled, FAST_START_RX[7:0] will never be reached and will instead be replaced by the contents of this register, thereby executing the jump. Example: FAST_START_RX = 10 FAST_DEST_RX = 15 The TSM will count as follows: … 7,8,9,15,16,17 …
23-16 FAST_START_RX	Fast TSM RX "Jump-from" Point During a Fast RX Warmup, this value represents the starting TSM count for the portion of the warmup that is to be skipped over. If Fast RX Warmups are enabled, this count will never be reached; its place in the TSM warmup will be replaced by the contents of FAST_DEST_RX[7:0], thereby executing the jump.
15-8 FAST_DEST_TX	Fast TSM TX "Jump-to" Point During a Fast TX Warmup, this value represents the ending TSM count for the portion of the warmup that is to be skipped over. If Fast TX Warmups are enabled, FAST_START_TX[7:0] will never be reached and will instead be replaced by the contents of this register, thereby executing the jump. Example: FAST_START_TX = 10 FAST_DEST_TX = 15 The TSM will count as follows: … 7,8,9,15,16,17 …
7-0 FAST_START_TX	Fast TSM TX "Jump-from" Point During a Fast TX Warmup, this value represents the starting TSM count for the portion of the warmup that is to be skipped over. If Fast TX Warmups are enabled, this count will never be reached; its place in the TSM warmup will be replaced by the contents of FAST_DEST_TX[7:0], thereby executing the jump.

55.4.7.7.3.1.9 TSM FAST WARMUP CONTROL 3 (FAST_CTRL3)

Offset

Register	Offset
FAST_CTRL3	1Ch

Function

This register provides enabling and control for Fast TSM Mode

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FAST_TX2RX_START_FC								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FAST_RX2TX_START_FC								0							
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 FAST_TX2RX_START_FC	TSM "Jump-to" point for RSM's FC TX-to-RX Transition During a Fast TX-to-RX transition (FAST_TX2RX_EN=1), the contents of this register represent the entry point (TSM count) in the RX warmup sequence to which the TSM jumps to when Ranging sequence manager asserts fast_tx2rx_wu and fast_fc_wu. The TSM will jump from the TX “ON” phase to the RX “WARMUP” phase, starting at this count.
23-16 —	Reserved
15-8 FAST_RX2TX_START_FC	TSM "Jump-to" point for RSM's FC RX-to-TX Transition During a Fast RX-to-TX transition (FAST_RX2TX_EN=1) the contents of this register represent the entry point (TSM count) in the TX warmup sequence to which the TSM jumps to when the Ranging Sequence Manager asserts fast_rx2tx_wu and fast_fc_wu. The TSM will jump from the RX “ON” phase to the TX “WARMUP” phase, starting at this count.
7-0 —	Reserved

55.4.7.7.3.1.10 TSM_TIMING00 (TIMING00)

Offset

Register	Offset
TIMING00	20h

Function

This register contains the timing values to control the assertion and deassertion times for both TX and RX sequences for the RF_ACTIVE TSM signal or signal group.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RF_ACTIVE_RX_LO								RF_ACTIVE_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RF_ACTIVE_TX_LO								RF_ACTIVE_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 RF_ACTIVE_RX_LO	De-assertion time setting for RF_ACTIVE (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RF_ACTIVE signal or group will transition from HI to LO.
23-16 RF_ACTIVE_RX_HI	Assertion time setting for RF_ACTIVE_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RF_ACTIVE signal or group will transition from LO to HI.
15-8 RF_ACTIVE_TX_LO	De-assertion time setting for RF_ACTIVE (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the RF_ACTIVE signal or group will transition from HI to LO.
7-0 RF_ACTIVE_TX_HI	Assertion time setting for RF_ACTIVE (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the RF_ACTIVE signal or group will transition from LO to HI.

55.4.7.7.3.1.11 TSM_TIMING01 (TIMING01)

Offset

Register	Offset
TIMING01	24h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RF_STATUS_RX_LO								RF_STATUS_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RF_STATUS_TX_LO								RF_STATUS_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

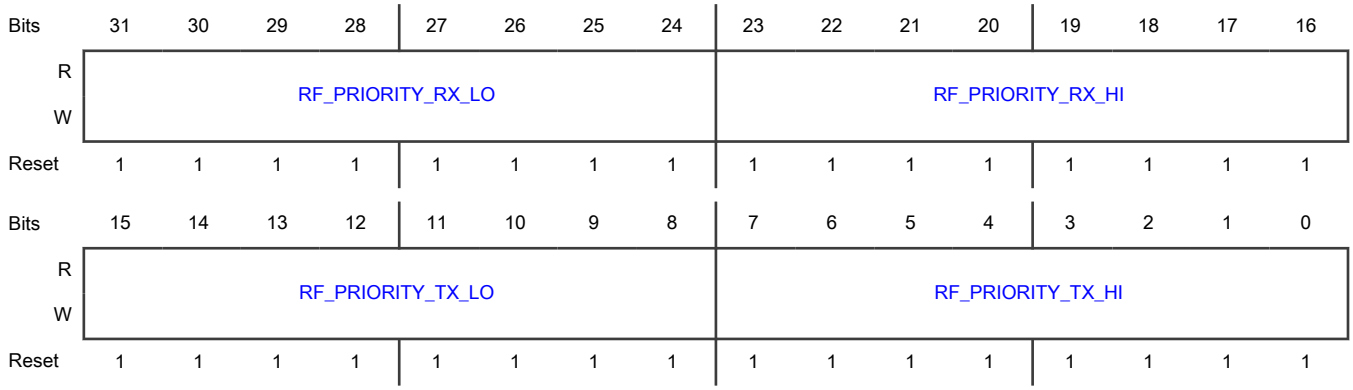
Field	Function
31-24 RF_STATUS_RX_LO	De-assertion time setting for RF_STATUS (RX) This field feedback status during a TSM RX sequence (the tsm_count[7:0] value) at which the RF_STATUS signal or group will transition from HI to LO.
23-16 RF_STATUS_RX_HI	Assertion time setting for RF_STATUS (RX) This field feedback status during a TSM RX sequence (the tsm_count[7:0] value) at which the RF_STATUS signal or group will transition from LO to HI.
15-8 RF_STATUS_TX_LO	De-assertion time setting for RF_STATUS (TX) This field feedback status during a TSM TX sequence (the tsm_count[7:0] value) at which the RF_STATUS signal or group will transition from HI to LO.
7-0 RF_STATUS_TX_HI	Assertion time setting for RF_STATUS (TX) This field feedback status during a TSM TX sequence (the tsm_count[7:0] value) at which the RF_STATUS signal or group will transition from LO to HI.

55.4.7.7.3.1.12 TSM_TIMING02 (TIMING02)

Offset

Register	Offset
TIMING02	28h

Diagram



Fields

Field	Function
31-24 RF_PRIORITY_RX_LO	De-assertion time setting for RF_PRIORITY (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RF_PRIORITY signal or group will transition from HI to LO.
23-16 RF_PRIORITY_RX_HI	Assertion time setting for RF_PRIORITY (RX) This field sets the point during a TSM RF_PRIORITY RX sequence (the tsm_count[7:0] value) at which the RF_PRIORITY signal or group will transition from LO to HI.
15-8 RF_PRIORITY_TX_LO	De-assertion time setting for RF_PRIORITY (TX) This field feedback priority during a TSM TX sequence (the tsm_count[7:0] value) at which the RF_PRIORITY signal or group will transition from HI to LO.
7-0 RF_PRIORITY_TX_HI	Assertion time setting for RF_PRIORITY (TX) This field feedback priority during a TSM TX sequence (the tsm_count[7:0] value) at which the RF_PRIORITY signal or group will transition from LO to HI.

55.4.7.7.3.1.13 TSM_TIMING03 (TIMING03)

Offset

Register	Offset
TIMING03	2Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IRQ0_START_TRIG_RX_LO								IRQ0_START_TRIG_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IRQ0_START_TRIG_TX_LO								IRQ0_START_TRIG_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 IRQ0_START_TRIG_RX_LO	De-assertion time setting for IRQ0_START_TRIG (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the IRQ0_START_TRIG signal or group will transition from HI to LO.
23-16 IRQ0_START_TRIG_RX_HI	Assertion time setting for IRQ0_START_TRIG (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the IRQ0_START_TRIG signal or group will transition from LO to HI.
15-8 IRQ0_START_TRIG_TX_LO	De-assertion time setting for IRQ0_START_TRIG (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the IRQ0_START_TRIG signal or group will transition from HI to LO.
7-0 IRQ0_START_TRIG_TX_HI	Assertion time setting for IRQ0_START_TRIG (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the IRQ0_START_TRIG signal or group will transition from LO to HI.

55.4.7.7.3.1.14 TSM_TIMING04 (TIMING04)

Offset

Register	Offset
TIMING04	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IRQ1_STOP_TRIG_RX_LO								IRQ1_STOP_TRIG_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IRQ1_STOP_TRIG_TX_LO								IRQ1_STOP_TRIG_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

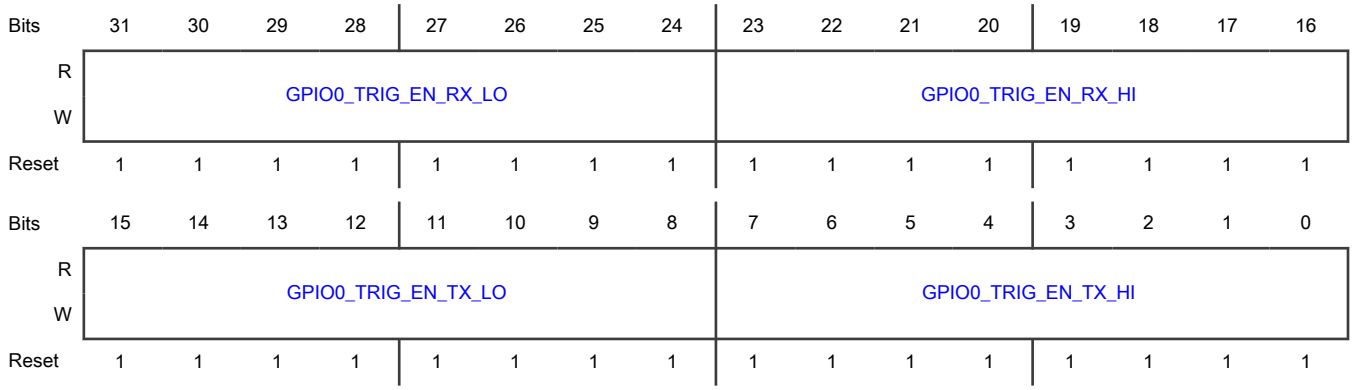
Field	Function
31-24 IRQ1_STOP_TRIG_RX_LO	De-assertion time setting for IRQ1_STOP_TRIG (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the IRQ1_STOP_TRIG signal or group will transition from HI to LO.
23-16 IRQ1_STOP_TRIG_RX_HI	Assertion time setting for IRQ1_STOP_TRIG (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the IRQ1_STOP_TRIG signal or group will transition from LO to HI.
15-8 IRQ1_STOP_TRIG_TX_LO	De-assertion time setting for IRQ1_STOP_TRIG (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the IRQ1_STOP_TRIG signal or group will transition from HI to LO.
7-0 IRQ1_STOP_TRIG_TX_HI	Assertion time setting for IRQ1_STOP_TRIG (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the IRQ1_STOP_TRIG signal or group will transition from LO to HI.

55.4.7.7.3.1.15 TSM_TIMING05 (TIMING05)

Offset

Register	Offset
TIMING05	34h

Diagram



Fields

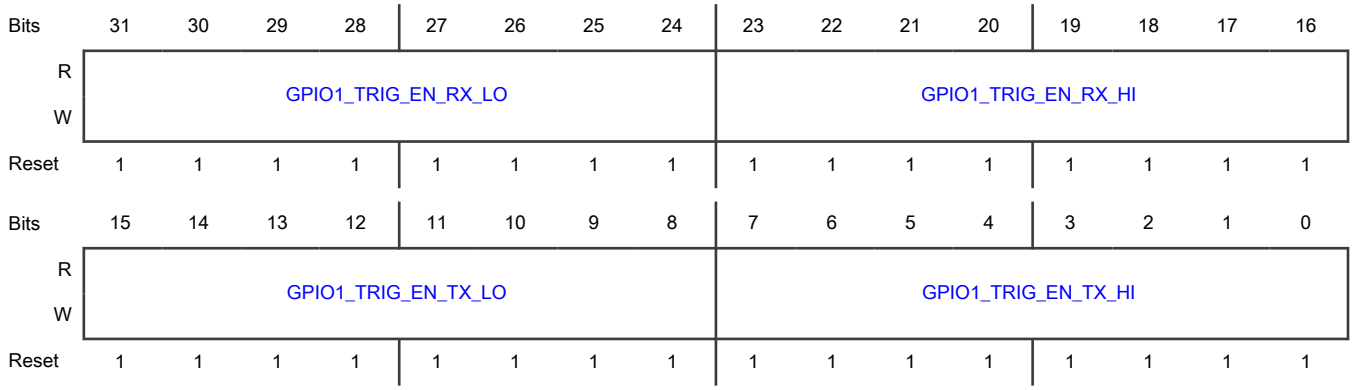
Field	Function
31-24 GPIO0_TRIG_EN_RX_LO	De-assertion time setting for GPIO0_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO0_TRIG_EN signal or group will transition from HI to LO.
23-16 GPIO0_TRIG_EN_RX_HI	Assertion time setting for GPIO0_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO0_TRIG_EN signal or group will transition from LO to HI.
15-8 GPIO0_TRIG_EN_TX_LO	De-assertion time setting for GPIO0_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO0_TRIG_EN signal or group will transition from HI to LO.
7-0 GPIO0_TRIG_EN_TX_HI	Assertion time setting for GPIO0_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO0_TRIG_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.16 TSM_TIMING06 (TIMING06)

Offset

Register	Offset
TIMING06	38h

Diagram



Fields

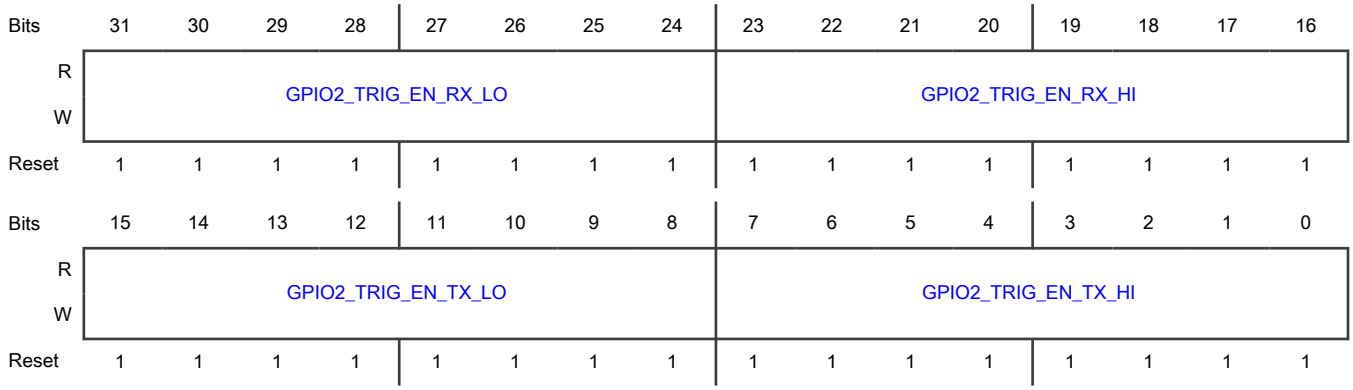
Field	Function
31-24 GPIO1_TRIG_EN_RX_LO	De-assertion time setting for GPIO1_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO1_TRIG_EN signal or group will transition from HI to LO.
23-16 GPIO1_TRIG_EN_RX_HI	Assertion time setting for GPIO1_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO1_TRIG_EN signal or group will transition from LO to HI.
15-8 GPIO1_TRIG_EN_TX_LO	De-assertion time setting for GPIO1_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO1_TRIG_EN signal or group will transition from HI to LO.
7-0 GPIO1_TRIG_EN_TX_HI	Assertion time setting for GPIO1_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO1_TRIG_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.17 TSM_TIMING07 (TIMING07)

Offset

Register	Offset
TIMING07	3Ch

Diagram



Fields

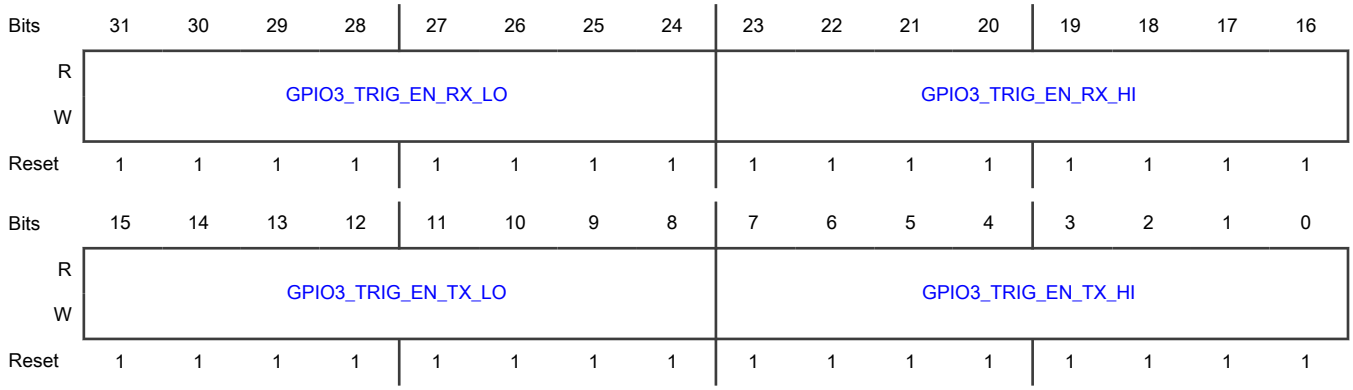
Field	Function
31-24 GPIO2_TRIG_EN_RX_LO	De-assertion time setting for GPIO2_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO2_TRIG_EN signal or group will transition from HI to LO.
23-16 GPIO2_TRIG_EN_RX_HI	Assertion time setting for GPIO2_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO2_TRIG_EN signal or group will transition from LO to HI.
15-8 GPIO2_TRIG_EN_TX_LO	De-assertion time setting for GPIO2_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO2_TRIG_EN signal or group will transition from HI to LO.
7-0 GPIO2_TRIG_EN_TX_HI	Assertion time setting for GPIO2_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO2_TRIG_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.18 TSM_TIMING08 (TIMING08)

Offset

Register	Offset
TIMING08	40h

Diagram



Fields

Field	Function
31-24 GPIO3_TRIG_EN_RX_LO	De-assertion time setting for GPIO3_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO3_TRIG_EN signal or group will transition from HI to LO.
23-16 GPIO3_TRIG_EN_RX_HI	Assertion time setting for GPIO3_TRIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the GPIO3_TRIG_EN signal or group will transition from LO to HI.
15-8 GPIO3_TRIG_EN_TX_LO	De-assertion time setting for GPIO3_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO3_TRIG_EN signal or group will transition from HI to LO.
7-0 GPIO3_TRIG_EN_TX_HI	Assertion time setting for GPIO3_TRIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the GPIO3_TRIG_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.19 TSM_TIMING09 (TIMING09)

Offset

Register	Offset
TIMING09	44h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DCOC_GAIN_CFG_EN_RX_LO								DCOC_GAIN_CFG_EN_RX_HI							
Reset	0	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DCOC_GAIN_CFG_EN_TX_LO								DCOC_GAIN_CFG_EN_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 DCOC_GAIN_CFG_EN_RX_LO	De-assertion time setting for DCOC_GAIN_CFG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the DCOC_GAIN_CFG_EN signal or group will transition from HI to LO.
23-16 DCOC_GAIN_CFG_EN_RX_HI	Assertion time setting for DCOC_GAIN_CFG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the DCOC_GAIN_CFG_EN signal or group will transition from LO to HI.
15-8 DCOC_GAIN_CFG_EN_TX_LO	De-assertion time setting for DCOC_GAIN_CFG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the DCOC_GAIN_CFG_EN signal or group will transition from HI to LO. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.
7-0 DCOC_GAIN_CFG_EN_TX_HI	Assertion time setting for DCOC_GAIN_CFG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the DCOC_GAIN_CFG_EN signal or group will transition from LO to HI. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.

55.4.7.7.3.1.20 TSM_TIMING10 (TIMING10)

Offset

Register	Offset
TIMING10	48h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LDO_CAL_EN_RX_LO								LDO_CAL_EN_RX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LDO_CAL_EN_TX_LO								LDO_CAL_EN_TX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0

Fields

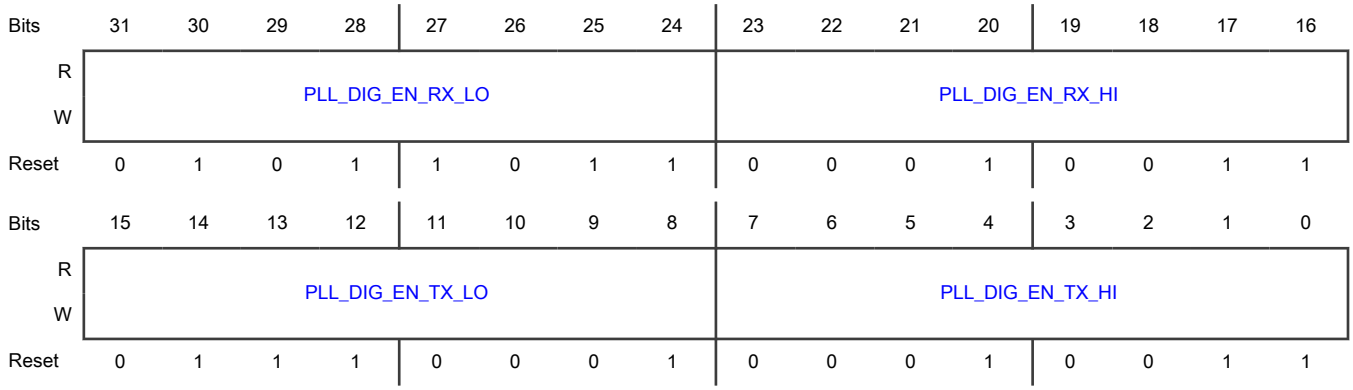
Field	Function
31-24 LDO_CAL_EN_RX_LO	De-assertion time setting for LDO_CAL_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the LDO_CAL_EN signal or group will transition from HI to LO.
23-16 LDO_CAL_EN_RX_HI	Assertion time setting for LDO_CAL_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the LDO_CAL_EN signal or group will transition from LO to HI.
15-8 LDO_CAL_EN_TX_LO	De-assertion time setting for LDO_CAL_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the LDO_CAL_EN signal or group will transition from HI to LO.
7-0 LDO_CAL_EN_TX_HI	Assertion time setting for LDO_CAL_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the LDO_CAL_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.21 TSM_TIMING11 (TIMING11)

Offset

Register	Offset
TIMING11	4Ch

Diagram



Fields

Field	Function
31-24 PLL_DIG_EN_RX_LO	De-assertion time setting for PLL_DIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the PLL_DIG_EN signal or group will transition from HI to LO.
23-16 PLL_DIG_EN_RX_HI	Assertion time setting for PLL_DIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the PLL_DIG_EN signal or group will transition from LO to HI.
15-8 PLL_DIG_EN_TX_LO	De-assertion time setting for PLL_DIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the PLL_DIG_EN signal or group will transition from HI to LO.
7-0 PLL_DIG_EN_TX_HI	Assertion time setting for PLL_DIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the PLL_DIG_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.22 TSM_TIMING12 (TIMING12)

Offset

Register	Offset
TIMING12	50h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIGMA_DELTA_EN_RX_LO								SIGMA_DELTA_EN_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIGMA_DELTA_EN_TX_LO								SIGMA_DELTA_EN_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	1	0	0	1	1	1	1

Fields

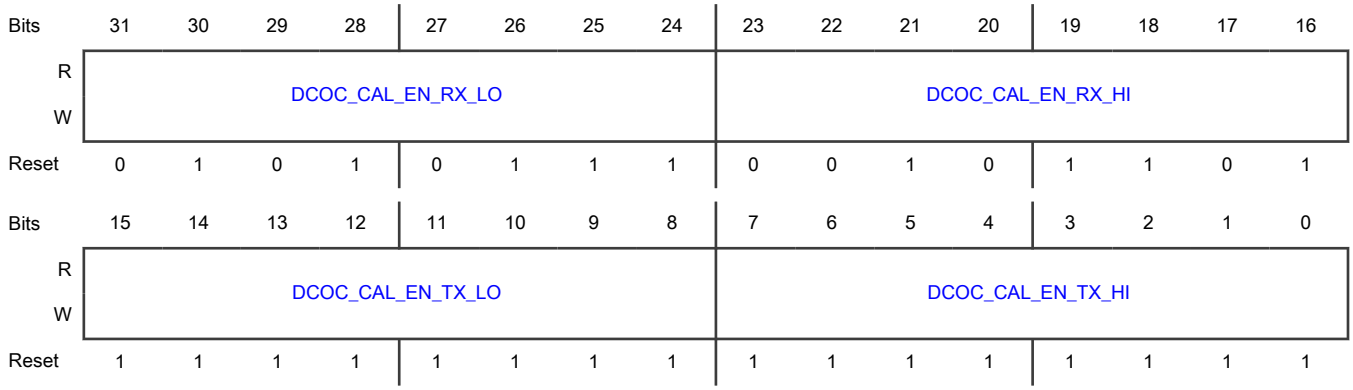
Field	Function
31-24 SIGMA_DELTA_EN_RX_LO	De-assertion time setting for SIGMA_DELTA_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SIGMA_DELTA_EN signal or group will transition from HI to LO.
23-16 SIGMA_DELTA_EN_RX_HI	Assertion time setting for SIGMA_DELTA_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SIGMA_DELTA_EN signal or group will transition from LO to HI.
15-8 SIGMA_DELTA_EN_TX_LO	De-assertion time setting for SIGMA_DELTA_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SIGMA_DELTA_EN signal or group will transition from HI to LO.
7-0 SIGMA_DELTA_EN_TX_HI	Assertion time setting for SIGMA_DELTA_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SIGMA_DELTA_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.23 TSM_TIMING13 (TIMING13)

Offset

Register	Offset
TIMING13	54h

Diagram



Fields

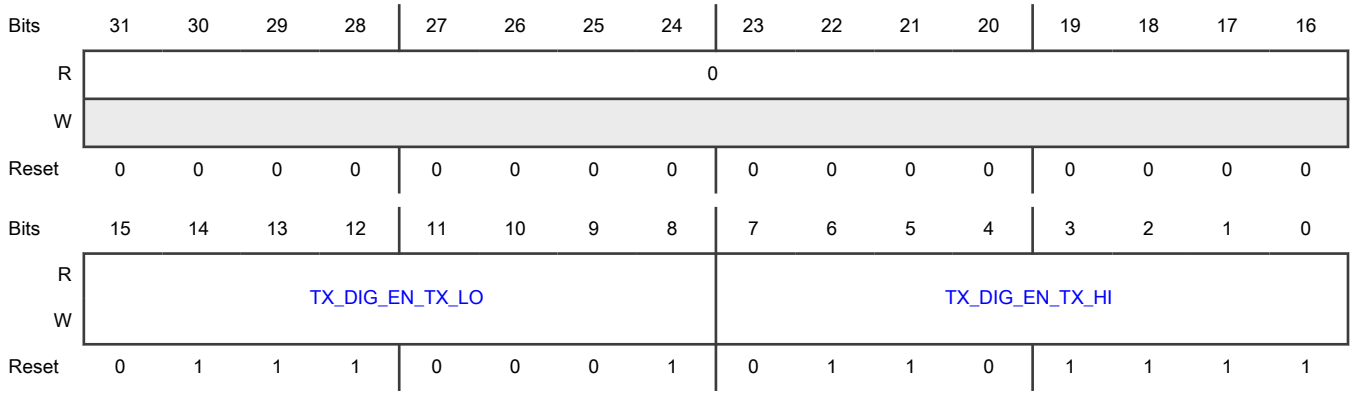
Field	Function
31-24 DCOC_CAL_EN_RX_LO	De-assertion time setting for DCOC_CAL_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the DCOC_CAL_EN signal or group will transition from HI to LO.
23-16 DCOC_CAL_EN_RX_HI	Assertion time setting for DCOC_CAL_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the DCOC_CAL_EN signal or group will transition from LO to HI.
15-8 DCOC_CAL_EN_TX_LO	De-assertion time setting for DCOC_CAL_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the DCOC_CAL_EN signal or group will transition from HI to LO. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.
7-0 DCOC_CAL_EN_TX_HI	Assertion time setting for DCOC_CAL_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the DCOC_CAL_EN signal or group will transition from LO to HI. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.

55.4.7.7.3.1.24 TSM_TIMING14 (TIMING14)

Offset

Register	Offset
TIMING14	58h

Diagram



Fields

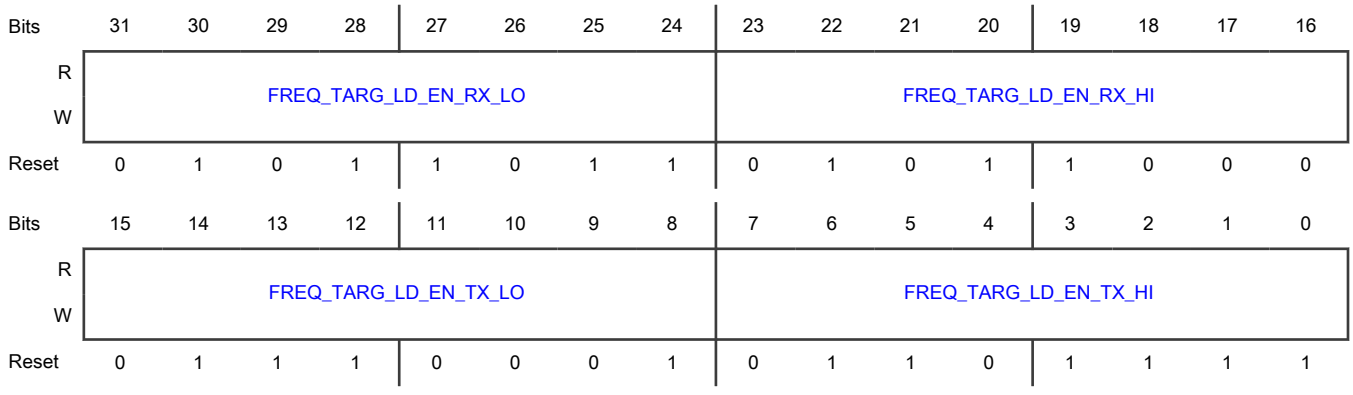
Field	Function
31-16 —	Reserved
15-8 TX_DIG_EN_TX_LO	De-assertion time setting for TX_DIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the TX_DIG_EN signal or group will transition from HI to LO.
7-0 TX_DIG_EN_TX_HI	Assertion time setting for TX_DIG_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the TX_DIG_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.25 TSM_TIMING15 (TIMING15)

Offset

Register	Offset
TIMING15	5Ch

Diagram



Fields

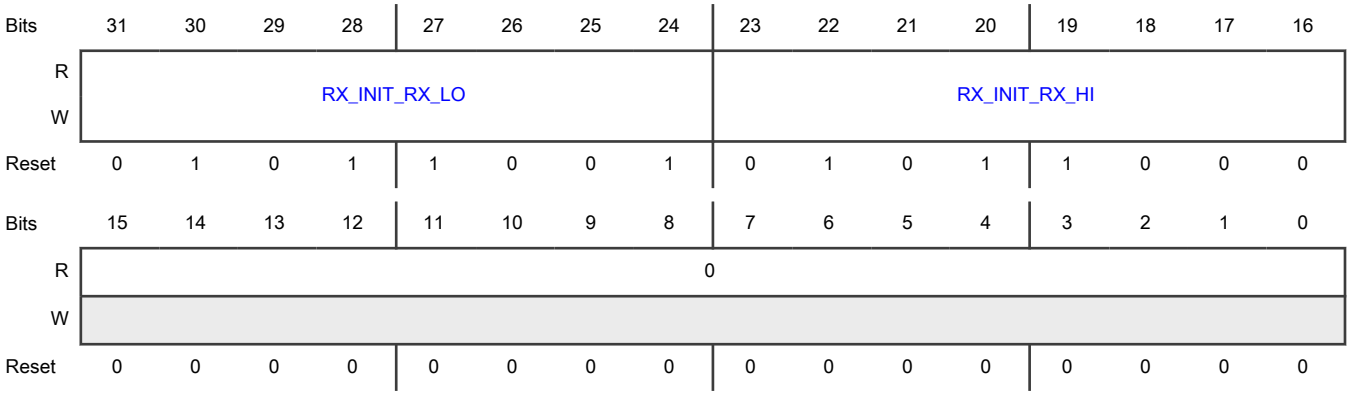
Field	Function
31-24 FREQ_TARG_LD_EN_RX_LO	De-assertion time setting for FREQ_TARG_LD_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the FREQ_TARG_LD_EN signal or group will transition from HI to LO.
23-16 FREQ_TARG_LD_EN_RX_HI	Assertion time setting for FREQ_TARG_LD_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the FREQ_TARG_LD_EN signal or group will transition from LO to HI.
15-8 FREQ_TARG_LD_EN_TX_LO	De-assertion time setting for FREQ_TARG_LD_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the FREQ_TARG_LD_EN signal or group will transition from HI to LO.
7-0 FREQ_TARG_LD_EN_TX_HI	Assertion time setting for FREQ_TARG_LD_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the FREQ_TARG_LD_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.26 TSM_TIMING16 (TIMING16)

Offset

Register	Offset
TIMING16	60h

Diagram



Fields

Field	Function
31-24	De-assertion time setting for RX_INIT (RX)

Table continues on the next page...

Table continued from the previous page...

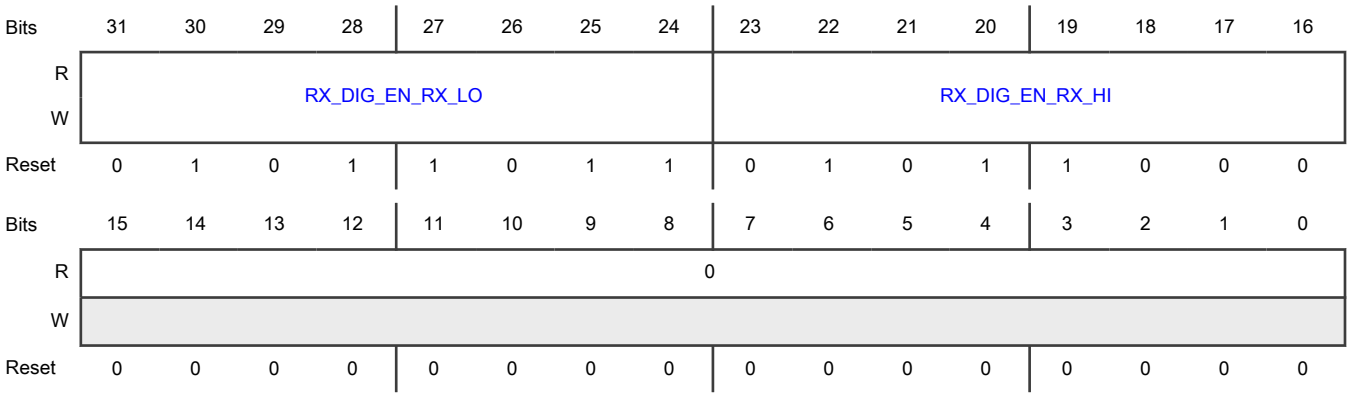
Field	Function
RX_INIT_RX_LO	This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RX_INIT signal or group will transition from HI to LO.
23-16 RX_INIT_RX_HI	Assertion time setting for RX_INIT (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RX_INIT signal or group will transition from LO to HI.
15-0 —	Reserved

55.4.7.7.3.1.27 TSM_TIMING17 (TIMING17)

Offset

Register	Offset
TIMING17	64h

Diagram



Fields

Field	Function
31-24 RX_DIG_EN_RX_LO	De-assertion time setting for RX_DIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RX_DIG_EN signal or group will transition from HI to LO.
23-16 RX_DIG_EN_RX_HI	Assertion time setting for RX_DIG_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RX_DIG_EN signal or group will transition from LO to HI.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 —	Reserved

55.4.7.7.3.1.28 TSM_TIMING18 (TIMING18)

Offset

Register	Offset
TIMING18	68h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RX_PHY_EN_RX_LO								RX_PHY_EN_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	1	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 RX_PHY_EN_RX_LO	De-assertion time setting for RX_PHY_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RX_PHY_EN signal or group will transition from HI to LO.
23-16 RX_PHY_EN_RX_HI	Assertion time setting for RX_PHY_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the RX_PHY_EN signal or group will transition from LO to HI.
15-0 —	Reserved

55.4.7.7.3.1.29 TSM_TIMING19 (TIMING19)

Offset

Register	Offset
TIMING19	6Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_PUP_IBG_CAL_RX_LO								SEQ_BG_PUP_IBG_CAL_RX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_PUP_IBG_CAL_TX_LO								SEQ_BG_PUP_IBG_CAL_TX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_BG_PUP_IBG_CAL_RX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_CAL (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_CAL signal or group will transition from HI to LO.
23-16 SEQ_BG_PUP_IBG_CAL_RX_HI	Assertion time setting for SEQ_BG_PUP_IBG_CAL (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_CAL signal or group will transition from LO to HI.
15-8 SEQ_BG_PUP_IBG_CAL_TX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_CAL (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_CAL signal or group will transition from HI to LO.
7-0 SEQ_BG_PUP_IBG_CAL_TX_HI	Assertion time setting for SEQ_BG_PUP_IBG_CAL (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_CAL signal or group will transition from LO to HI.

55.4.7.7.3.1.30 TSM_TIMING20 (TIMING20)

Offset

Register	Offset
TIMING20	70h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LDOTRIM_PUP_RX_LO								SEQ_LDOTRIM_PUP_RX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LDOTRIM_PUP_TX_LO								SEQ_LDOTRIM_PUP_TX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_LDOTRIM_PUP_RX_LO	De-assertion time setting for SEQ_LDOTRIM_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDOTRIM_PUP signal or group will transition from HI to LO.
23-16 SEQ_LDOTRIM_PUP_RX_HI	Assertion time setting for SEQ_LDOTRIM_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDOTRIM_PUP signal or group will transition from LO to HI.
15-8 SEQ_LDOTRIM_PUP_TX_LO	De-assertion time setting for SEQ_LDOTRIM_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDOTRIM_PUP signal or group will transition from HI to LO.
7-0 SEQ_LDOTRIM_PUP_TX_HI	Assertion time setting for SEQ_LDOTRIM_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDOTRIM_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.31 TSM_TIMING21 (TIMING21)

Offset

Register	Offset
TIMING21	74h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LDO_CAL_PUP_RX_LO								SEQ_LDO_CAL_PUP_RX_HI							
Reset	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LDO_CAL_PUP_TX_LO								SEQ_LDO_CAL_PUP_TX_HI							
Reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_LDO_CAL_PUP_RX_LO	De-assertion time setting for SEQ_LDO_CAL_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_CAL_PUP signal or group will transition from HI to LO.
23-16 SEQ_LDO_CAL_PUP_RX_HI	Assertion time setting for SEQ_LDO_CAL_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_CAL_PUP signal or group will transition from LO to HI.
15-8 SEQ_LDO_CAL_PUP_TX_LO	De-assertion time setting for SEQ_LDO_CAL_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_CAL_PUP signal or group will transition from HI to LO.
7-0 SEQ_LDO_CAL_PUP_TX_HI	Assertion time setting for SEQ_LDO_CAL_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_CAL_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.32 TSM_TIMING22 (TIMING22)

Offset

Register	Offset
TIMING22	78h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_FC_RX_LO								SEQ_BG_FC_RX_HI							
Reset	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_FC_TX_LO								SEQ_BG_FC_TX_HI							
Reset	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_BG_FC_RX_LO	De-assertion time setting for SEQ_BG_FC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_FC signal or group will transition from HI to LO.
23-16 SEQ_BG_FC_RX_HI	Assertion time setting for SEQ_BG_FC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_FC signal or group will transition from LO to HI.
15-8 SEQ_BG_FC_TX_LO	De-assertion time setting for SEQ_BG_FC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_FC signal or group will transition from HI to LO.
7-0 SEQ_BG_FC_TX_HI	Assertion time setting for SEQ_BG_FC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_FC signal or group will transition from LO to HI.

55.4.7.7.3.1.33 TSM_TIMING23 (TIMING23)

Offset

Register	Offset
TIMING23	7Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LDO_GANG_FC_RX_LO								SEQ_LDO_GANG_FC_RX_HI							
Reset	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LDO_GANG_FC_TX_LO								SEQ_LDO_GANG_FC_TX_HI							
Reset	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_LDO_GANG_FC_RX_LO	De-assertion time setting for SEQ_LDO_(PLL/RXTXHF/RXTXLF/VCO)_FC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(PLL/RXTXHF/RXTXLF/VCO)_FC signal or group will transition from HI to LO.
23-16 SEQ_LDO_GANG_FC_RX_HI	Assertion time setting for SEQ_LDO_(PLL/RXTXHF/RXTXLF/VCO)_FC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(PLL/RXTXHF/RXTXLF/VCO)_FC signal or group will transition from LO to HI.
15-8 SEQ_LDO_GANG_FC_TX_LO	De-assertion time setting for SEQ_LDO_(PLL/RXTXHF/RXTXLF/VCO)_FC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(PLL/RXTXHF/RXTXLF/VCO)_FC signal or group will transition from HI to LO.
7-0 SEQ_LDO_GANG_FC_TX_HI	Assertion time setting for SEQ_LDO_GANG_FC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_GANG_FC signal or group will transition from LO to HI.

55.4.7.7.3.1.34 TSM_TIMING24 (TIMING24)

Offset

Register	Offset
TIMING24	80h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LDO_GANG_PUP_RX_LO								SEQ_LDO_GANG_PUP_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LDO_GANG_PUP_TX_LO								SEQ_LDO_GANG_PUP_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_LDO_GANG_PUP_RX_LO	De-assertion time setting for SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP signal or group will transition from HI to LO.
23-16 SEQ_LDO_GANG_PUP_RX_HI	Assertion time setting for SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP signal or group will transition from LO to HI.
15-8 SEQ_LDO_GANG_PUP_TX_LO	De-assertion time setting for SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP signal or group will transition from HI to LO.
7-0 SEQ_LDO_GANG_PUP_TX_HI	Assertion time setting for SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_(ANT/PLL/RXTXHF/RXTXLF/VCO/XO_DIST)_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.35 TSM_TIMING25 (TIMING25)

Offset

Register	Offset
TIMING25	84h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LDO_LV_PUP_RX_LO								SEQ_LDO_LV_PUP_RX_HI							
Reset	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LDO_LV_PUP_TX_LO								SEQ_LDO_LV_PUP_TX_HI							
Reset	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_LDO_LV_PUP_RX_LO	De-assertion time setting for SEQ_LDO_LV_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_LV_PUP signal or group will transition from HI to LO.
23-16 SEQ_LDO_LV_PUP_RX_HI	Assertion time setting for SEQ_LDO_LV_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_LV_PUP signal or group will transition from LO to HI.
15-8 SEQ_LDO_LV_PUP_TX_LO	De-assertion time setting for SEQ_LDO_LV_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_LV_PUP signal or group will transition from HI to LO.
7-0 SEQ_LDO_LV_PUP_TX_HI	Assertion time setting for SEQ_LDO_LV_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LDO_LV_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.36 TSM_TIMING26 (TIMING26)

Offset

Register	Offset
TIMING26	88h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_PUP_RX_LO								SEQ_BG_PUP_RX_HI							
Reset	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_PUP_TX_LO								SEQ_BG_PUP_TX_HI							
Reset	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_BG_PUP_RX_LO	De-assertion time setting for SEQ_BG_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP signal or group will transition from HI to LO.
23-16 SEQ_BG_PUP_RX_HI	Assertion time setting for SEQ_BG_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP signal or group will transition from LO to HI.
15-8 SEQ_BG_PUP_TX_LO	De-assertion time setting for SEQ_BG_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP signal or group will transition from HI to LO.
7-0 SEQ_BG_PUP_TX_HI	Assertion time setting for SEQ_BG_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.37 TSM_TIMING27 (TIMING27)

Offset

Register	Offset
TIMING27	8Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_PUP_IBG_ANT_RX_LO								SEQ_BG_PUP_IBG_ANT_RX_HI							
Reset	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_PUP_IBG_ANT_TX_LO								SEQ_BG_PUP_IBG_ANT_TX_HI							
Reset	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_BG_PUP_IBG_ANT_RX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_ANT (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_ANT signal or group will transition from HI to LO.
23-16 SEQ_BG_PUP_IBG_ANT_RX_HI	Assertion time setting for SEQ_BG_PUP_IBG_ANT (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_ANT signal or group will transition from LO to HI.
15-8 SEQ_BG_PUP_IBG_ANT_TX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_ANT (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_ANT signal or group will transition from HI to LO.
7-0 SEQ_BG_PUP_IBG_ANT_TX_HI	Assertion time setting for SEQ_BG_PUP_IBG_ANT (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_ANT signal or group will transition from LO to HI.

55.4.7.7.3.1.38 TSM_TIMING28 (TIMING28)

Offset

Register	Offset
TIMING28	90h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_PUP_IBG_XO_DIST_RX_LO								SEQ_BG_PUP_IBG_XO_DIST_RX_HI							
Reset	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_PUP_IBG_XO_DIST_TX_LO								SEQ_BG_PUP_IBG_XO_DIST_TX_HI							
Reset	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_BG_PUP_IBG_XO_DIST_RX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_XO_DIST (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_XO_DIST signal or group will transition from HI to LO.
23-16 SEQ_BG_PUP_IBG_XO_DIST_RX_HI	Assertion time setting for SEQ_BG_PUP_IBG_XO_DIST (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_XO_DIST signal or group will transition from LO to HI.
15-8 SEQ_BG_PUP_IBG_XO_DIST_TX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_XO_DIST (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_XO_DIST signal or group will transition from HI to LO.
7-0 SEQ_BG_PUP_IBG_XO_DIST_TX_HI	Assertion time setting for SEQ_BG_PUP_IBG_XO_DIST (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_XO_DIST signal or group will transition from LO to HI.

55.4.7.7.3.1.39 TSM_TIMING29 (TIMING29)

Offset

Register	Offset
TIMING29	94h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_PUP_IBG_TX_RX_LO								SEQ_BG_PUP_IBG_TX_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_PUP_IBG_TX_TX_LO								SEQ_BG_PUP_IBG_TX_TX_HI							
Reset	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SEQ_BG_PUP_IBG_TX_RX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_TX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_TX signal or group will transition from HI to LO.
23-16 SEQ_BG_PUP_IBG_TX_RX_HI	Assertion time setting for SEQ_BG_PUP_IBG_TX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_TX signal or group will transition from LO to HI.
15-8 SEQ_BG_PUP_IBG_TX_TX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_TX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_TX signal or group will transition from HI to LO.
7-0 SEQ_BG_PUP_IBG_TX_TX_HI	Assertion time setting for SEQ_BG_PUP_IBG_TX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_TX signal or group will transition from LO to HI.

55.4.7.7.3.1.40 TSM_TIMING30 (TIMING30)

Offset

Register	Offset
TIMING30	98h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_PUP_IBG_RX_RX_LO								SEQ_BG_PUP_IBG_RX_RX_HI							
Reset	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_BG_PUP_IBG_RX_TX_LO								SEQ_BG_PUP_IBG_RX_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_BG_PUP_IBG_RX_RX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_RX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_RX signal or group will transition from HI to LO.
23-16 SEQ_BG_PUP_IBG_RX_RX_HI	Assertion time setting for SEQ_BG_PUP_IBG_RX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_RX signal or group will transition from LO to HI.
15-8 SEQ_BG_PUP_IBG_RX_TX_LO	De-assertion time setting for SEQ_BG_PUP_IBG_RX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_RX signal or group will transition from HI to LO. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.
7-0 SEQ_BG_PUP_IBG_RX_TX_HI	Assertion time setting for SEQ_BG_PUP_IBG_RX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_BG_PUP_IBG_RX signal or group will transition from LO to HI. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.

55.4.7.7.3.1.41 TSM_TIMING31 (TIMING31)

Offset

Register	Offset
TIMING31	9Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_TSM_ISO_B_2P4GHZ_RX_LO								SEQ_TSM_ISO_B_2P4GHZ_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_TSM_ISO_B_2P4GHZ_TX_LO								SEQ_TSM_ISO_B_2P4GHZ_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	0	1	0	0	0

Fields

Field	Function
31-24 SEQ_TSM_ISO_B_2P4GHZ_RX_LO	De-assertion time setting for SEQ_TSM_ISO_B_2P4GHZ (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_TSM_ISO_B_2P4GHZ signal or group will transition from HI to LO.
23-16 SEQ_TSM_ISO_B_2P4GHZ_RX_HI	Assertion time setting for SEQ_TSM_ISO_B_2P4GHZ (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_TSM_ISO_B_2P4GHZ signal or group will transition from LO to HI.
15-8 SEQ_TSM_ISO_B_2P4GHZ_TX_LO	De-assertion time setting for SEQ_TSM_ISO_B_2P4GHZ (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_TSM_ISO_B_2P4GHZ signal or group will transition from HI to LO.
7-0 SEQ_TSM_ISO_B_2P4GHZ_TX_HI	Assertion time setting for SEQ_TSM_ISO_B_2P4GHZ (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_TSM_ISO_B_2P4GHZ signal or group will transition from LO to HI.

55.4.7.7.3.1.42 TSM_TIMING32 (TIMING32)

Offset

Register	Offset
TIMING32	A0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_RCCAL_PUP_RX_LO								SEQ_RCCAL_PUP_RX_HI							
Reset	0	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_RCCAL_PUP_TX_LO								SEQ_RCCAL_PUP_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_RCCAL_PUP_RX_LO	De-assertion time setting for SEQ_RCCAL_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_RCCAL_PUP signal or group will transition from HI to LO.
23-16 SEQ_RCCAL_PUP_RX_HI	Assertion time setting for SEQ_RCCAL_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_RCCAL_PUP signal or group will transition from LO to HI.
15-8 SEQ_RCCAL_PUP_TX_LO	De-assertion time setting for SEQ_RCCAL_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0]value) at which the SEQ_RCCAL_PUP signal or group will transition from HI toLO. **NOTE** This signal can be asserted only during WU or FC transtions.
7-0 SEQ_RCCAL_PUP_TX_HI	Assertion time setting for SEQ_RCCAL_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0]value) at which the SEQ_RCCAL_PUP signal or group will transition from LO toHI. **NOTE** This signal can be asserted only during WU or FC transtions.

55.4.7.7.3.1.43 TSM_TIMING33 (TIMING33)

Offset

Register	Offset
TIMING33	A4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_PD_EN_FCAL_BIAS_RX_LO								SEQ_PD_EN_FCAL_BIAS_RX_HI							
Reset	0	0	1	0	0	0	1	1	0	0	0	1	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_PD_EN_FCAL_BIAS_TX_LO								SEQ_PD_EN_FCAL_BIAS_TX_HI							
Reset	0	1	0	1	1	0	1	0	0	0	0	1	0	0	0	1

Fields

Field	Function
31-24 SEQ_PD_EN_FCAL_BIAS_RX_LO	De-assertion time setting for SEQ_PD_EN_FCAL_BIAS (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_FCAL_BIAS signal or group will transition from HI to LO.
23-16 SEQ_PD_EN_FCAL_BIAS_RX_HI	Assertion time setting for SEQ_PD_EN_FCAL_BIAS (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_FCAL_BIAS signal or group will transition from LO to HI.
15-8 SEQ_PD_EN_FCAL_BIAS_TX_LO	De-assertion time setting for SEQ_PD_EN_FCAL_BIAS (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_FCAL_BIAS signal or group will transition from HI to LO.
7-0 SEQ_PD_EN_FCAL_BIAS_TX_HI	Assertion time setting for SEQ_PD_EN_FCAL_BIAS (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_FCAL_BIAS signal or group will transition from LO to HI.

55.4.7.7.3.1.44 TSM_TIMING34 (TIMING34)

Offset

Register	Offset
TIMING34	A8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_PD_PUP_RX_LO								SEQ_PD_PUP_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_PD_PUP_TX_LO								SEQ_PD_PUP_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	1

Fields

Field	Function
31-24 SEQ_PD_PUP_RX_LO	De-assertion time setting for SEQ_PD_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_PD_PUP signal or group will transition from HI to LO.
23-16 SEQ_PD_PUP_RX_HI	Assertion time setting for SEQ_PD_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_PD_PUP signal or group will transition from LO to HI.
15-8 SEQ_PD_PUP_TX_LO	De-assertion time setting for SEQ_PD_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_PD_PUP signal or group will transition from HI to LO.
7-0 SEQ_PD_PUP_TX_HI	Assertion time setting for SEQ_PD_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_PD_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.45 TSM_TIMING35 (TIMING35)

Offset

Register	Offset
TIMING35	ACh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_VCO_PUP_RX_LO								SEQ_VCO_PUP_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_VCO_PUP_TX_LO								SEQ_VCO_PUP_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	1

Fields

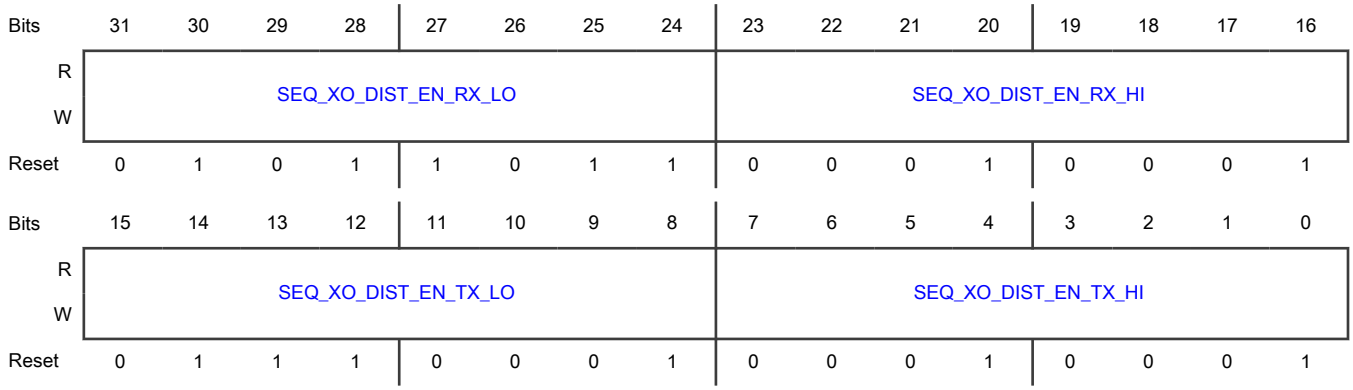
Field	Function
31-24 SEQ_VCO_PUP_RX_LO	De-assertion time setting for SEQ_VCO_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_PUP signal or group will transition from HI to LO.
23-16 SEQ_VCO_PUP_RX_HI	Assertion time setting for SEQ_VCO_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_PUP signal or group will transition from LO to HI.
15-8 SEQ_VCO_PUP_TX_LO	De-assertion time setting for SEQ_VCO_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_PUP signal or group will transition from HI to LO.
7-0 SEQ_VCO_PUP_TX_HI	Assertion time setting for SEQ_VCO_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.46 TSM_TIMING36 (TIMING36)

Offset

Register	Offset
TIMING36	B0h

Diagram



Fields

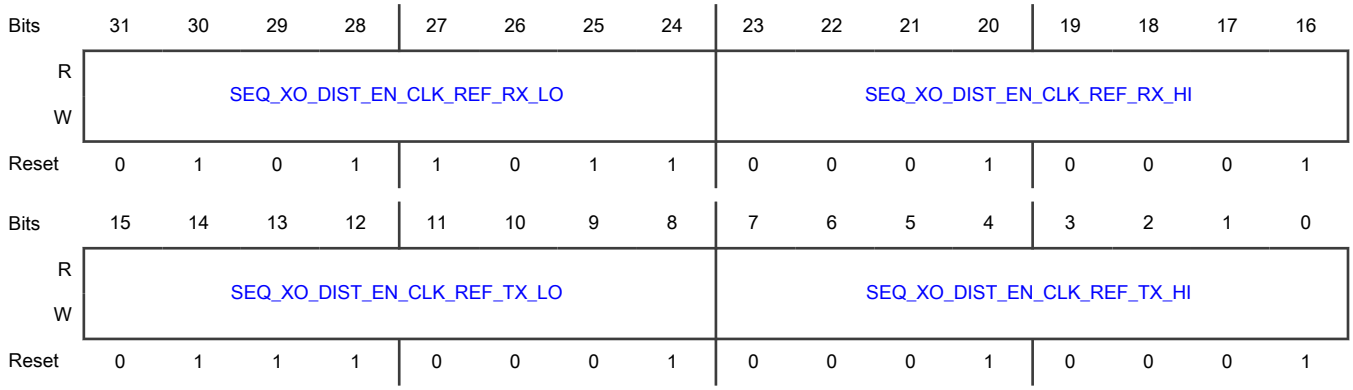
Field	Function
31-24 SEQ_XO_DIST_EN_RX_LO	De-assertion time setting for SEQ_XO_DIST_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN signal or group will transition from HI to LO.
23-16 SEQ_XO_DIST_EN_RX_HI	Assertion time setting for SEQ_XO_DIST_EN (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN signal or group will transition from LO to HI.
15-8 SEQ_XO_DIST_EN_TX_LO	De-assertion time setting for SEQ_XO_DIST_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN signal or group will transition from HI to LO.
7-0 SEQ_XO_DIST_EN_TX_HI	Assertion time setting for SEQ_XO_DIST_EN (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN signal or group will transition from LO to HI.

55.4.7.7.3.1.47 TSM_TIMING37 (TIMING37)

Offset

Register	Offset
TIMING37	B4h

Diagram



Fields

Field	Function
31-24 SEQ_XO_DIST_EN_CLK_REF_RX_LO	De-assertion time setting for SEQ_XO_DIST_EN_CLK_REF (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_REF signal or group will transition from HI to LO.
23-16 SEQ_XO_DIST_EN_CLK_REF_RX_HI	Assertion time setting for SEQ_XO_DIST_EN_CLK_REF (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_REF signal or group will transition from LO to HI.
15-8 SEQ_XO_DIST_EN_CLK_REF_TX_LO	De-assertion time setting for SEQ_XO_DIST_EN_CLK_REF (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_REF signal or group will transition from HI to LO.
7-0 SEQ_XO_DIST_EN_CLK_REF_TX_HI	Assertion time setting for SEQ_XO_DIST_EN_CLK_REF (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_REF signal or group will transition from LO to HI.

55.4.7.7.3.1.48 TSM_TIMING38 (TIMING38)

Offset

Register	Offset
TIMING38	B8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_XO_EN_CLK_2P4G_RX_LO								SEQ_XO_EN_CLK_2P4G_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_XO_EN_CLK_2P4G_TX_LO								SEQ_XO_EN_CLK_2P4G_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	1

Fields

Field	Function
31-24 SEQ_XO_EN_CLK_2P4G_RX_LO	De-assertion time setting for SEQ_XO_EN_CLK_2P4G (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_EN_CLK_2P4G signal or group will transition from HI to LO.
23-16 SEQ_XO_EN_CLK_2P4G_RX_HI	Assertion time setting for SEQ_XO_EN_CLK_2P4G (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_EN_CLK_2P4G signal or group will transition from LO to HI.
15-8 SEQ_XO_EN_CLK_2P4G_TX_LO	De-assertion time setting for SEQ_XO_EN_CLK_2P4G (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_EN_CLK_2P4G signal or group will transition from HI to LO.
7-0 SEQ_XO_EN_CLK_2P4G_TX_HI	Assertion time setting for SEQ_XO_EN_CLK_2P4G (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_EN_CLK_2P4G signal or group will transition from LO to HI.

55.4.7.7.3.1.49 TSM_TIMING39 (TIMING39)

Offset

Register	Offset
TIMING39	BCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_XO_DIST_EN_CLK_ADCDAC_RX_LO								SEQ_XO_DIST_EN_CLK_ADCDAC_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	1	0	1	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_XO_DIST_EN_CLK_ADCDAC_TX_LO								SEQ_XO_DIST_EN_CLK_ADCDAC_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	1

Fields

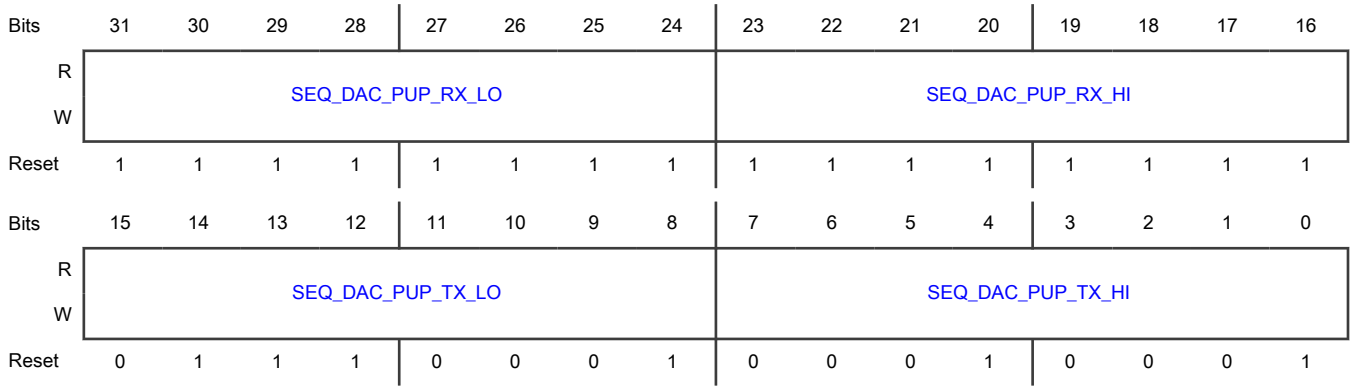
Field	Function
31-24 SEQ_XO_DIST_EN_CLK_ADCDAC_RX_LO	De-assertion time setting for SEQ_XO_DIST_EN_CLK_ADCDAC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_ADCDAC signal or group will transition from HI to LO.
23-16 SEQ_XO_DIST_EN_CLK_ADCDAC_RX_HI	Assertion time setting for SEQ_XO_DIST_EN_CLK_ADCDAC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_ADCDAC signal or group will transition from LO to HI.
15-8 SEQ_XO_DIST_EN_CLK_ADCDAC_TX_LO	De-assertion time setting for SEQ_XO_DIST_EN_CLK_ADCDAC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_ADCDAC signal or group will transition from HI to LO.
7-0 SEQ_XO_DIST_EN_CLK_ADCDAC_TX_HI	Assertion time setting for SEQ_XO_DIST_EN_CLK_ADCDAC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_XO_DIST_EN_CLK_ADCDAC signal or group will transition from LO to HI.

55.4.7.7.3.1.50 TSM_TIMING40 (TIMING40)

Offset

Register	Offset
TIMING40	C0h

Diagram



Fields

Field	Function
31-24 SEQ_DAC_PUP_RX_LO	De-assertion time setting for SEQ_DAC_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_DAC_PUP signal or group will transition from HI to LO.
23-16 SEQ_DAC_PUP_RX_HI	Assertion time setting for SEQ_DAC_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_DAC_PUP signal or group will transition from LO to HI.
15-8 SEQ_DAC_PUP_TX_LO	De-assertion time setting for SEQ_DAC_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_DAC_PUP signal or group will transition from HI to LO.
7-0 SEQ_DAC_PUP_TX_HI	Assertion time setting for SEQ_DAC_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_DAC_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.51 TSM_TIMING41 (TIMING41)

Offset

Register	Offset
TIMING41	C4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_VCO_EN_HPM_RX_LO								SEQ_VCO_EN_HPM_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_VCO_EN_HPM_TX_LO								SEQ_VCO_EN_HPM_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	1

Fields

Field	Function
31-24 SEQ_VCO_EN_HPM_RX_LO	De-assertion time setting for SEQ_VCO_EN_HPM (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_EN_HPM signal or group will transition from HI to LO.
23-16 SEQ_VCO_EN_HPM_RX_HI	Assertion time setting for SEQ_VCO_EN_HPM (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_EN_HPM signal or group will transition from LO to HI.
15-8 SEQ_VCO_EN_HPM_TX_LO	De-assertion time setting for SEQ_VCO_EN_HPM (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_EN_HPM signal or group will transition from HI to LO.
7-0 SEQ_VCO_EN_HPM_TX_HI	Assertion time setting for SEQ_VCO_EN_HPM (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_VCO_EN_HPM signal or group will transition from LO to HI.

55.4.7.7.3.1.52 TSM_TIMING42 (TIMING42)

Offset

Register	Offset
TIMING42	C8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LO_PUP_VLO_FBK_RX_LO								SEQ_LO_PUP_VLO_FBK_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LO_PUP_VLO_FBK_TX_LO								SEQ_LO_PUP_VLO_FBK_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	1	0

Fields

Field	Function
31-24 SEQ_LO_PUP_VLO_FBK_RX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_FBK (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_FBK signal or group will transition from HI to LO.
23-16 SEQ_LO_PUP_VLO_FBK_RX_HI	Assertion time setting for SEQ_LO_PUP_VLO_FBK (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_FBK signal or group will transition from LO to HI.
15-8 SEQ_LO_PUP_VLO_FBK_TX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_FBK (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_FBK signal or group will transition from HI to LO.
7-0 SEQ_LO_PUP_VLO_FBK_TX_HI	Assertion time setting for SEQ_LO_PUP_VLO_FBK (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_FBK signal or group will transition from LO to HI.

55.4.7.7.3.1.53 TSM_TIMING43 (TIMING43)

Offset

Register	Offset
TIMING43	CCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LO_PUP_VLO_RX_RX_LO								SEQ_LO_PUP_VLO_RX_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LO_PUP_VLO_RX_TX_LO								SEQ_LO_PUP_VLO_RX_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_LO_PUP_VLO_RX_RX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_RX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RX signal or group will transition from HI to LO.
23-16 SEQ_LO_PUP_VLO_RX_RX_HI	Assertion time setting for SEQ_LO_PUP_VLO_RX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RX signal or group will transition from LO to HI.
15-8 SEQ_LO_PUP_VLO_RX_TX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_RX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RX signal or group will transition from HI to LO.
7-0 SEQ_LO_PUP_VLO_RX_TX_HI	Assertion time setting for SEQ_LO_PUP_VLO_RX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RX signal or group will transition from LO to HI.

55.4.7.7.3.1.54 TSM_TIMING44 (TIMING44)

Offset

Register	Offset
TIMING44	D0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LO_PUP_VLO_RXDRV_RX_LO								SEQ_LO_PUP_VLO_RXDRV_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LO_PUP_VLO_RXDRV_TX_LO								SEQ_LO_PUP_VLO_RXDRV_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_LO_PUP_VLO_RXDRV_RX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_RXDRV (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RXDRV signal or group will transition from HI to LO.
23-16 SEQ_LO_PUP_VLO_RXDRV_RX_HI	Assertion time setting for SEQ_LO_PUP_VLO_RXDRV (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RXDRV signal or group will transition from LO to HI.
15-8 SEQ_LO_PUP_VLO_RXDRV_TX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_RXDRV (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RXDRV signal or group will transition from HI to LO.
7-0 SEQ_LO_PUP_VLO_RXDRV_TX_HI	Assertion time setting for SEQ_LO_PUP_VLO_RXDRV (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_RXDRV signal or group will transition from LO to HI.

55.4.7.7.3.1.55 TSM_TIMING45 (TIMING45)

Offset

Register	Offset
TIMING45	D4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LO_PUP_VLO_TX_RX_LO								SEQ_LO_PUP_VLO_TX_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LO_PUP_VLO_TX_TX_LO								SEQ_LO_PUP_VLO_TX_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	1	0

Fields

Field	Function
31-24 SEQ_LO_PUP_VLO_TX_RX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_TX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TX signal or group will transition from HI to LO.
23-16 SEQ_LO_PUP_VLO_TX_RX_HI	Assertion time setting for SEQ_LO_PUP_VLO_TX (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TX signal or group will transition from LO to HI.
15-8 SEQ_LO_PUP_VLO_TX_TX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_TX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TX signal or group will transition from HI to LO.
7-0 SEQ_LO_PUP_VLO_TX_TX_HI	Assertion time setting for SEQ_LO_PUP_VLO_TX (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TX signal or group will transition from LO to HI.

55.4.7.7.3.1.56 TSM_TIMING46 (TIMING46)

Offset

Register	Offset
TIMING46	D8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_LO_PUP_VLO_TXDRV_RX_LO								SEQ_LO_PUP_VLO_TXDRV_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LO_PUP_VLO_TXDRV_TX_LO								SEQ_LO_PUP_VLO_TXDRV_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	0	0	1	0	0	1	0

Fields

Field	Function
31-24 SEQ_LO_PUP_VLO_TXDRV_RX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_TXDRV (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TXDRV signal or group will transition from HI to LO.
23-16 SEQ_LO_PUP_VLO_TXDRV_RX_HI	Assertion time setting for SEQ_LO_PUP_VLO_TXDRV (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TXDRV signal or group will transition from LO to HI.
15-8 SEQ_LO_PUP_VLO_TXDRV_TX_LO	De-assertion time setting for SEQ_LO_PUP_VLO_TXDRV (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TXDRV signal or group will transition from HI to LO.
7-0 SEQ_LO_PUP_VLO_TXDRV_TX_HI	Assertion time setting for SEQ_LO_PUP_VLO_TXDRV (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_LO_PUP_VLO_TXDRV signal or group will transition from LO to HI.

55.4.7.7.3.1.57 TSM_TIMING47 (TIMING47)

Offset

Register	Offset
TIMING47	DCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_DIVN_PUP_RX_LO								SEQ_DIVN_PUP_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	0	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_DIVN_PUP_TX_LO								SEQ_DIVN_PUP_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	1	0	1	0	1	1	1

Fields

Field	Function
31-24 SEQ_DIVN_PUP_RX_LO	De-assertion time setting for SEQ_DIVN_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_PUP signal or group will transition from HI to LO.
23-16 SEQ_DIVN_PUP_RX_HI	Assertion time setting for SEQ_DIVN_PUP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_PUP signal or group will transition from LO to HI.
15-8 SEQ_DIVN_PUP_TX_LO	De-assertion time setting for SEQ_DIVN_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_PUP signal or group will transition from HI to LO.
7-0 SEQ_DIVN_PUP_TX_HI	Assertion time setting for SEQ_DIVN_PUP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_PUP signal or group will transition from LO to HI.

55.4.7.7.3.1.58 TSM_TIMING48 (TIMING48)

Offset

Register	Offset
TIMING48	E0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_DIVN_CLOSEDLOOP_RX_LO								SEQ_DIVN_CLOSEDLOOP_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	1	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_DIVN_CLOSEDLOOP_TX_LO								SEQ_DIVN_CLOSEDLOOP_TX_HI							
Reset	0	1	1	1	0	0	0	1	0	1	0	1	1	0	1	0

Fields

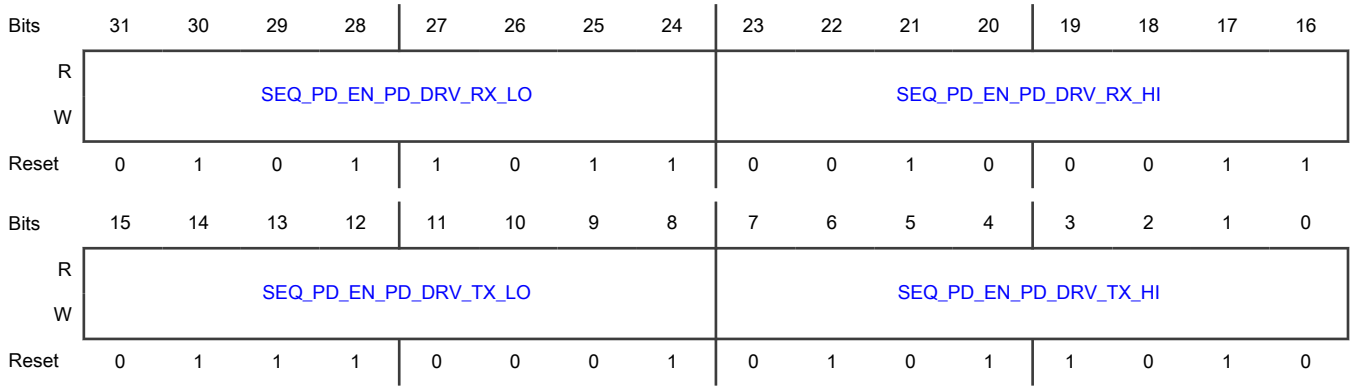
Field	Function
31-24 SEQ_DIVN_CLOSEDLOOP_RX_LO	De-assertion time setting for SEQ_DIVN_CLOSEDLOOP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_CLOSEDLOOP signal or group will transition from HI to LO.
23-16 SEQ_DIVN_CLOSEDLOOP_RX_HI	Assertion time setting for SEQ_DIVN_CLOSEDLOOP (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_CLOSEDLOOP signal or group will transition from LO to HI.
15-8 SEQ_DIVN_CLOSEDLOOP_TX_LO	De-assertion time setting for SEQ_DIVN_CLOSEDLOOP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_CLOSEDLOOP signal or group will transition from HI to LO.
7-0 SEQ_DIVN_CLOSEDLOOP_TX_HI	Assertion time setting for SEQ_DIVN_CLOSEDLOOP (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_DIVN_CLOSEDLOOP signal or group will transition from LO to HI.

55.4.7.7.3.1.59 TSM_TIMING49 (TIMING49)

Offset

Register	Offset
TIMING49	E4h

Diagram



Fields

Field	Function
31-24 SEQ_PD_EN_PD_DRV_RX_LO	De-assertion time setting for SEQ_PD_EN_PD_DRV (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_PD_DRV signal or group will transition from HI to LO.
23-16 SEQ_PD_EN_PD_DRV_RX_HI	Assertion time setting for SEQ_PD_EN_PD_DRV (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_PD_DRV signal or group will transition from LO to HI.
15-8 SEQ_PD_EN_PD_DRV_TX_LO	De-assertion time setting for SEQ_PD_EN_PD_DRV (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_PD_DRV signal or group will transition from HI to LO.
7-0 SEQ_PD_EN_PD_DRV_TX_HI	Assertion time setting for SEQ_PD_EN_PD_DRV (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_PD_EN_PD_DRV signal or group will transition from LO to HI.

55.4.7.7.3.1.60 TSM_TIMING50 (TIMING50)

Offset

Register	Offset
TIMING50	E8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_CBPF_EN_DCOC_RX_LO								SEQ_CBPF_EN_DCOC_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	1	0	1	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_CBPF_EN_DCOC_TX_LO								SEQ_CBPF_EN_DCOC_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_CBPF_EN_DCOC_RX_LO	De-assertion time setting for SEQ_CBPF_EN_DCOC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_CBPF_EN_DCOC signal or group will transition from HI to LO.
23-16 SEQ_CBPF_EN_DCOC_RX_HI	Assertion time setting for SEQ_CBPF_EN_DCOC (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_CBPF_EN_DCOC signal or group will transition from LO to HI.
15-8 SEQ_CBPF_EN_DCOC_TX_LO	De-assertion time setting for SEQ_CBPF_EN_DCOC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_CBPF_EN_DCOC signal or group will transition from HI to LO. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.
7-0 SEQ_CBPF_EN_DCOC_TX_HI	Assertion time setting for SEQ_CBPF_EN_DCOC (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_CBPF_EN_DCOC signal or group will transition from LO to HI. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.

55.4.7.7.3.1.61 TSM_TIMING51 (TIMING51)

Offset

Register	Offset
TIMING51	ECh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_RX_GANG_PUP_RX_LO								SEQ_RX_GANG_PUP_RX_HI							
Reset	0	1	0	1	1	0	1	1	0	0	1	0	1	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_RX_GANG_PUP_TX_LO								SEQ_RX_GANG_PUP_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_RX_GANG_PUP_RX_LO	De-assertion time setting for SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 signal or group will transition from HI to LO.
23-16 SEQ_RX_GANG_PUP_RX_HI	Assertion time setting for SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 signal or group will transition from LO to HI.
15-8 SEQ_RX_GANG_PUP_TX_LO	De-assertion time setting for SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 signal or group will transition from HI to LO. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.
7-0 SEQ_RX_GANG_PUP_TX_HI	Assertion time setting for SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_(RX_LAN/RX_MIX/CBPF/ADC_PUP) and SEQ_SPARE1 signal or group will transition from LO to HI. **NOTE** This signal can be asserted in most of the time except for IP RX2TX.

55.4.7.7.3.1.62 TSM_TIMING52 (TIMING52)

Offset

Register	Offset
TIMING52	F0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_SPARE3_RX_LO								SEQ_SPARE3_RX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_SPARE3_TX_LO								SEQ_SPARE3_TX_HI							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24 SEQ_SPARE3_RX_LO	De-assertion time setting for SEQ_SPARE3 (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_SPARE3 signal or group will transition from HI to LO.
23-16 SEQ_SPARE3_RX_HI	Assertion time setting for SEQ_SPARE3 (RX) This field sets the point during a TSM RX sequence (the tsm_count[7:0] value) at which the SEQ_SPARE3 signal or group will transition from LO to HI.
15-8 SEQ_SPARE3_TX_LO	De-assertion time setting for SEQ_SPARE3 (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_SPARE3 signal or group will transition from HI to LO.
7-0 SEQ_SPARE3_TX_HI	Assertion time setting for SEQ_SPARE3 (TX) This field sets the point during a TSM TX sequence (the tsm_count[7:0] value) at which the SEQ_SPARE3 signal or group will transition from LO to HI.

55.4.7.7.3.1.63 TSM OVERRIDE REGISTER 0 (OVRD0)

Offset

Register	Offset
OVRD0	F4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEQ_BG_...	SEQ_BG_...	RX_PHY_...	RX_PHY_...	RX_DIG_...	RX_DIG_...	RX_INIT...	RX_INIT...	FREQ_TA...	FREQ_TA...	TX_DIG_...	TX_DIG_...	DCOC_CA...	DCOC_CA...	SIGMA_D...	SIGMA_D...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PLL_DIG...	PLL_DIG...	LDO_CAL...	LDO_CAL...	DCOC_GA...	DCOC_GA...	TSM_IRQ...	TSM_IRQ...	TSM_IRQ...	TSM_IRQ...	TSM_RF_...	TSM_RF_...	TSM_RF_...	TSM_RF_...	TSM_RF_...	TSM_RF_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SEQ_BG_PUP_IBG_CAL_OVRD	Override value for SEQ_BG_PUP_IBG_CAL When SEQ_BG_PUP_IBG_CAL_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_pup_ibg_cal". This bit is ignored when SEQ_BG_PUP_IBG_CAL_OVRD_EN==0.
30 SEQ_BG_PUP_IBG_CAL_OVRD_EN	Override control for SEQ_BG_PUP_IBG_CAL 0b - Normal operation. 1b - Use the state of SEQ_BG_PUP_IBG_CAL_OVRD to override the signal "seq_bg_pup_ibg_cal".
29 RX_PHY_EN_OVRD	Override value for RX_PHY_EN When RX_PHY_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "rx_phy_en". This bit is ignored when RX_PHY_EN_OVRD_EN==0.
28 RX_PHY_EN_OVRD_EN	Override control for RX_PHY_EN 0b - Normal operation. 1b - Use the state of RX_PHY_EN_OVRD to override the signal "rx_phy_en".
27 RX_DIG_EN_OVRD	Override value for RX_DIG_EN When RX_DIG_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "rx_dig_en". This bit is ignored when RX_DIG_EN_OVRD_EN==0.
26 RX_DIG_EN_OVRD_EN	Override control for RX_DIG_EN 0b - Normal operation. 1b - Use the state of RX_DIG_EN_OVRD to override the signal "rx_dig_en".
25 RX_INIT_EN_OVRD	Override value for RX_INIT_EN When RX_INIT_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "rx_init_en". This bit is ignored when RX_INIT_EN_OVRD_EN==0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 RX_INIT_EN_OVRD_EN	Override control for RX_INIT_EN 0b - Normal operation. 1b - Use the state of RX_INIT_EN_OVRD to override the signal "rx_init_en".
23 FREQ_TARG_LD_EN_OVRD	Override value for FREQ_TARG_LD_EN When FREQ_TARG_LD_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "freq_targ_ld_en". This bit is ignored when FREQ_TARG_LD_EN_OVRD_EN==0.
22 FREQ_TARG_LD_EN_OVRD_EN	Override control for FREQ_TARG_LD_EN 0b - Normal operation. 1b - Use the state of FREQ_TARG_LD_EN_OVRD to override the signal "freq_targ_ld_en".
21 TX_DIG_EN_OVRD	Override value for TX_DIG_EN When TX_DIG_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "tx_dig_en". This bit is ignored when TX_DIG_EN_OVRD_EN==0.
20 TX_DIG_EN_OVRD_EN	Override control for TX_DIG_EN 0b - Normal operation. 1b - Use the state of TX_DIG_EN_OVRD to override the signal "tx_dig_en".
19 DCOC_CAL_EN_OVRD	Override value for DCOC_CAL_EN When DCOC_CAL_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "dcoc_cal_en". This bit is ignored when DCOC_CAL_EN_OVRD_EN==0.
18 DCOC_CAL_EN_OVRD_EN	Override control for DCOC_CAL_EN 0b - Normal operation. 1b - Use the state of DCOC_CAL_EN_OVRD to override the signal "dcoc_cal_en".
17 SIGMA_DELTA_EN_OVRD	Override value for SIGMA_DELTA_EN When SIGMA_DELTA_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "sigma_delta_en". This bit is ignored when SIGMA_DELTA_EN_OVRD_EN==0.
16 SIGMA_DELTA_EN_OVRD_EN	Override control for SIGMA_DELTA_EN 0b - Normal operation. 1b - Use the state of SIGMA_DELTA_EN_OVRD to override the signal "sigma_delta_en".
15 PLL_DIG_EN_OVRD	Override value for PLL_DIG_EN When PLL_DIG_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "pll_dig_en". This bit is ignored when PLL_DIG_EN_OVRD_EN==0.
14	Override control for PLL_DIG_EN 0b - Normal operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
PLL_DIG_EN_OVRD_EN	1b - Use the state of PLL_DIG_EN_OVRD to override the signal "pll_dig_en".
13 LDO_CAL_EN_OVRD	Override value for LDO_CAL_EN When LDO_CAL_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "ldo_cal_en". This bit is ignored when LDO_CAL_EN_OVRD_EN==0.
12 LDO_CAL_EN_OVRD_EN	Override control for LDO_CAL_EN_ 0b - Normal operation. 1b - Use the state of LDO_CAL_EN_OVRD to override the signal "ldo_cal_en".
11 DCOC_GAIN_CFG_EN_OVRD	Override value for DCOC_GAIN_CFG_EN When DCOC_GAIN_CFG_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "dcoc_gain_cfg_en". This bit is ignored when DCOC_GAIN_CFG_EN_OVRD_EN==0.
10 DCOC_GAIN_CFG_EN_OVRD_EN	Override control for DCOC_GAIN_CFG_EN 0b - Normal operation. 1b - Use the state of DCOC_GAIN_CFG_EN_OVRD to override the signal "dcoc_gain_cfg_en".
9 TSM_IRQ1_STOP_TRIG_OVRD	Override value for TSM_IRQ1_STOP_TRIG When TSM_IRQ1_STOP_TRIG_OVRD_EN=1, this value overrides the mission mode state of the signal "tsm_irq1_stop_trig". This bit is ignored when TSM_IRQ1_STOP_TRIG_OVRD_EN==0.
8 TSM_IRQ1_STOP_TRIG_OVRD_EN	Override control for TSM_IRQ1_STOP_TRIG 0b - Normal operation. 1b - Use the state of TSM_IRQ1_STOP_TRIG_OVRD to override the signal "tsm_irq1_stop_trig".
7 TSM_IRQ0_START_TRIG_OVRD	Override value for TSM_IRQ0_START_TRIG When TSM_IRQ0_START_TRIG_OVRD=1, this value overrides the mission mode state of the signal "tsm_irq0_start_trig". This bit is ignored when TSM_IRQ0_START_TRIG_OVRD==0.
6 TSM_IRQ0_START_TRIG_OVRD_EN	Override control for TSM_IRQ0_START_TRIG_EN 0b - Normal operation. 1b - Use the state of TSM_IRQ0_START_TRIG_OVRD to override the signal "tsm_irq0_start_trig".
5 TSM_RF_PRIORITY_OVRD	Override value for tsm_rf_priority When TSM_RF_PRIORITY_OVRD_EN=1, this value overrides the mission mode state of the signal "tsm_rf_priority". This bit is ignored when TSM_RF_PRIORITY_OVRD_EN==0.
4	Override control for TSM_RF_PRIORITY_EN

Table continues on the next page...

Table continued from the previous page...

Field	Function
TSM_RF_PRIORITY_OVRD_EN	0b - Normal operation. 1b - Use the state of TSM_RF_PRIORITY_OVRD to override the signal "tsm_rf_priority".
3 TSM_RF_STATUS_OVRD	Override value for TSM_RF_STATUS When TSM_RF_STATUS_OVRD_EN=1, this value overrides the mission mode state of the signal "tsm_rf_status". This bit is ignored when TSM_RF_STATUS_OVRD_EN==0.
2 TSM_RF_STATUS_OVRD_EN	Override control for TSM_RF_STATUS_EN 0b - Normal operation. 1b - Use the state of TSM_RF_STATUS_OVRD to override the signal "tsm_rf_status".
1 TSM_RF_ACTIVE_OVRD	Override value for tsm_rf_active When TSM_RF_ACTIVE_OVRD_EN=1, this value overrides the mission mode state of the signal "tsm_rf_active". This bit is ignored when TSM_RF_ACTIVE_OVRD_EN==0.
0 TSM_RF_ACTIVE_OVRD_EN	Override control for TSM_RF_ACTIVE 0b - Normal operation. 1b - Use the state of TSM_RF_ACTIVE_OVRD to override the signal "tsm_rf_active".

55.4.7.7.3.1.64 TSM OVERRIDE REGISTER 1 (OVRD1)

Offset

Register	Offset
OVRD1	F8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SEQ_BG_...	SEQ_BG_...	SEQ_BG_...	SEQ_BG_...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_BG_...	SEQ_BG_...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...	SEQ_LDO...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SEQ_BG_PUP_IBG_ANT_OVRD	Override value for SEQ_BG_PUP_IBG_ANT When SEQ_BG_PUP_IBG_ANT_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_pup_ibg_ant". This bit is ignored when SEQ_BG_PUP_IBG_ANT_OVRD_EN==0.
30 SEQ_BG_PUP_IBG_ANT_OVRD_EN	Override control for SEQ_BG_PUP_IBG_ANT 0b - Normal operation. 1b - Use the state of SEQ_BG_PUP_IBG_ANT_OVRD to override the signal "seq_bg_pup_ibg_ant".
29 SEQ_BG_PUP_OVRD	Override value for SEQ_BG_PUP When SEQ_BG_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_pup". This bit is ignored when SEQ_BG_PUP_OVRD_EN==0.
28 SEQ_BG_PUP_OVRD_EN	Override control for SEQ_BG_PUP_OVRD_EN 0b - Normal operation. 1b - Use the state of SEQ_BG_PUP_OVRD to override the signal "seq_bg_pup".
27 SEQ_LDO_LV_PUP_OVRD	Override value for SEQ_LDO_LV_PUP When SEQ_LDO_LV_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_lv_pup". This bit is ignored when SEQ_LDO_LV_PUP_OVRD_EN==0.
26 SEQ_LDO_LV_PUP_OVRD_EN	Override control for SEQ_LDO_LV_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDO_LV_PUP_OVRD to override the signal "seq_ldo_lv_pup".
25 SEQ_LDO_RXTXLF_PUP_OVRD	Override value for SEQ_LDO_RXTXLF_PUP When SEQ_LDO_RXTXLF_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_rxtxlf_pup". This bit is ignored when SEQ_LDO_RXTXLF_PUP_OVRD_EN==0.
24 SEQ_LDO_RXTXLF_PUP_OVRD_EN	Override control for SEQ_LDO_RXTXLF_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDO_RXTXLF_PUP_OVRD to override the signal "seq_ldo_rxtxlf_pup".
23 SEQ_LDO_RXTXHF_PUP_OVRD	Override value for SEQ_LDO_RXTXHF_PUP When SEQ_LDO_RXTXHF_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_rxtxhf_pup". This bit is ignored when SEQ_LDO_RXTXHF_PUP_OVRD_EN==0.
22	Override control for SEQ_LDO_RXTXHF_PUP

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEQ_LDO_RXTXHF_PUP_OVRD_EN	0b - Normal operation. 1b - Use the state of SEQ_LDO_RXTXHF_PUP_OVRD to override the signal "seq_ldo_rxtxhf_pup".
21 SEQ_LDO_XO_DIST_PUP_OVRD	Override value for SEQ_LDO_XO_DIST_PUP When SEQ_LDO_XO_DIST_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_xo_dist_pup". This bit is ignored when SEQ_LDO_XO_DIST_PUP_OVRD_EN==0.
20 SEQ_LDO_XO_DIST_PUP_OVRD_EN	Override control for SEQ_LDO_XO_DIST_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDO_XO_DIST_PUP_OVRD to override the signal "seq_ldo_xo_dist_pup".
19 SEQ_LDO_VCO_PUP_OVRD	Override value for SEQ_LDO_VCO_PUP When SEQ_LDO_VCO_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_vco_pup". This bit is ignored when SEQ_LDO_VCO_PUP_OVRD_EN==0.
18 SEQ_LDO_VCO_PUP_OVRD_EN	Override control for SEQ_LDO_VCO_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDO_VCO_PUP_OVRD to override the signal "seq_ldo_vco_pup".
17 SEQ_LDO_PLL_PUP_OVRD	Override value for SEQ_LDO_PLL_PUP When SEQ_LDO_PLL_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_pll_pup". This bit is ignored when SEQ_LDO_PLL_PUP_OVRD_EN==0.
16 SEQ_LDO_PLL_PUP_OVRD_EN	Override control for SEQ_LDO_PLL_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDO_PLL_PUP_OVRD to override the signal "seq_ldo_pll_pup".
15 SEQ_LDO_ANT_PUP_OVRD	Override value for SEQ_LDO_ANT_PUP When SEQ_LDO_ANT_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_ant_pup". This bit is ignored when SEQ_LDO_ANT_PUP_OVRD_EN==0.
14 SEQ_LDO_ANT_PUP_OVRD_EN	Override control for SEQ_LDO_ANT_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDO_ANT_PUP_OVRD to override the signal "seq_ldo_ant_pup".
13 SEQ_LDO_RXTXLF_FC_OVRD	Override value for SEQ_LDO_RXTXLF_FC When SEQ_LDO_RXTXLF_FC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_rxtxlf_fc". This bit is ignored when SEQ_LDO_RXTXLF_FC_OVRD_EN==0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 SEQ_LDO_RXTXLF_FC_OVRD_EN	Override control for SEQ_LDO_RXTXLF_FC 0b - Normal operation. 1b - Use the state of SEQ_LDO_RXTXLF_FC_OVRD to override the signal "seq_ldo_rxtxlf_fc".
11 SEQ_LDO_RXTXHF_FC_OVRD	Override value for SEQ_LDO_RXTXHF_FC When SEQ_LDO_RXTXHF_FC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_rxtxhf_fc". This bit is ignored when SEQ_LDO_RXTXHF_FC_OVRD_EN==0.
10 SEQ_LDO_RXTXHF_FC_OVRD_EN	Override control for SEQ_LDO_RXTXHF_FC 0b - Normal operation. 1b - Use the state of SEQ_LDO_RXTXHF_FC_OVRD to override the signal "seq_ldo_rxtxhf_fc".
9 SEQ_LDO_VCO_FC_OVRD	Override value for SEQ_LDO_VCO_FC When SEQ_LDO_VCO_FC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_vco_fc". This bit is ignored when SEQ_LDO_VCO_FC_OVRD_EN==0.
8 SEQ_LDO_VCO_FC_OVRD_EN	Override control for SEQ_LDO_VCO_FC 0b - Normal operation. 1b - Use the state of SEQ_LDO_VCO_FC_OVRD to override the signal "seq_ldo_vco_fc".
7 SEQ_LDO_PLL_FC_OVRD	Override value for SEQ_LDO_PLL_FC When SEQ_LDO_PLL_FC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_pll_fc". This bit is ignored when SEQ_LDO_PLL_FC_OVRD_EN==0.
6 SEQ_LDO_PLL_FC_OVRD_EN	Override control for SEQ_LDO_PLL_FC 0b - Normal operation. 1b - Use the state of SEQ_LDO_PLL_FC_OVRD to override the signal "seq_ldo_pll_fc".
5 SEQ_BG_FC_OVRD	Override value for SEQ_BG_FC When SEQ_BG_FC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_fc". This bit is ignored when SEQ_BG_FC_OVRD_EN==0.
4 SEQ_BG_FC_OVRD_EN	Override control for SEQ_BG_FC 0b - Normal operation. 1b - Use the state of SEQ_BG_FC_OVRD to override the signal "seq_bg_fc".
3 SEQ_LDO_CAL_PUP_OVRD	Override value for SEQ_LDO_CAL_PUP When SEQ_LDO_CAL_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldo_cal_pup". This bit is ignored when SEQ_LDO_CAL_PUP_OVRD_EN==0.
2	Override control for SEQ_LDO_CAL_PUP

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEQ_LDO_CAL_PUP_OVRD_EN	0b - Normal operation. 1b - Use the state of SEQ_LDO_CAL_PUP_OVRD to override the signal "seq_ldo_cal_pup".
1 SEQ_LDOTRIM_PUP_OVRD	Override value for SEQ_LDOTRIM_PUP When SEQ_LDOTRIM_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_ldotrim_pup". This bit is ignored when SEQ_LDOTRIM_PUP_OVRD_EN==0.
0 SEQ_LDOTRIM_PUP_OVRD_EN	Override control for SEQ_LDOTRIM_PUP 0b - Normal operation. 1b - Use the state of SEQ_LDOTRIM_PUP_OVRD to override the signal "seq_ldotrim_pup".

55.4.7.7.3.1.65 TSM OVERRIDE REGISTER 2 (OVRD2)

Offset

Register	Offset
OVRD2	FCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEQ_LO_...	SEQ_LO_...	SEQ_LO_...	SEQ_LO_...	SEQ_VCO...	SEQ_VCO...	SEQ_DAC...	SEQ_DAC...	SEQ_XO_...	SEQ_XO_...	SEQ_XO_...	SEQ_XO_...	SEQ_XO_...	SEQ_XO_...	SEQ_XO_...	SEQ_XO_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEQ_VCO...	SEQ_VCO...	SEQ_PD_...	SEQ_PD_...	SEQ_PD_...	SEQ_PD_...	SEQ_RCC...	SEQ_RCC...	SEQ_TSM...	SEQ_TSM...	SEQ_BG_...	SEQ_BG_...	SEQ_BG_...	SEQ_BG_...	SEQ_BG_...	SEQ_BG_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SEQ_LO_PUP_VLO_RXDRV_OVRD	Override value for SEQ_LO_PUP_VLO_RXDRV When SEQ_LO_PUP_VLO_RXDRV_OVRD=1, this value overrides the mission mode state of the signal "seq_lo_pup_vlo_rxdrv". This bit is ignored when SEQ_LO_PUP_VLO_RXDRV_OVRD==0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 SEQ_LO_PUP_VLO_RXDRV_OVRD_EN	Override control for SEQ_LO_PUP_VLO_RXDRV 0b - Normal operation. 1b - Use the state of SEQ_LO_PUP_VLO_RXDRV_OVRD to override the signal "seq_lo_pup_vlo_rxdrv".
29 SEQ_LO_PUP_VLO_FBK_OVRD	Override value for SEQ_LO_PUP_VLO_FBK When SEQ_LO_PUP_VLO_FBK_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_lo_pup_vlo_fbk". This bit is ignored when SEQ_LO_PUP_VLO_FBK_OVRD_EN==0.
28 SEQ_LO_PUP_VLO_FBK_OVRD_EN	Override control for SEQ_LO_PUP_VLO_FBK 0b - Normal operation. 1b - Use the state of SEQ_LO_PUP_VLO_FBK_OVRD to override the signal "seq_lo_pup_vlo_fbk".
27 SEQ_VCO_EN_HPM_OVRD	Override value for SEQ_VCO_EN_HPM When SEQ_VCO_EN_HPM_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_vco_en_hpm". This bit is ignored when SEQ_VCO_EN_HPM_OVRD_EN==0.
26 SEQ_VCO_EN_HPM_OVRD_EN	Override control for SEQ_VCO_EN_HPM 0b - Normal operation. 1b - Use the state of SEQ_VCO_EN_HPM_OVRD to override the signal "seq_vco_en_hpm".
25 SEQ_DAC_PUP_OVRD	Override value for SEQ_DAC_PUP When SEQ_DAC_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_dac_pup". This bit is ignored when SEQ_DAC_PUP_OVRD_EN==0.
24 SEQ_DAC_PUP_OVRD_EN	Override control for SEQ_DAC_PUP 0b - Normal operation. 1b - Use the state of SEQ_DAC_PUP_OVRD to override the signal "seq_dac_pup".
23 SEQ_XO_DIST_EN_CLK_ADCDAC_OVRD	Override value for SEQ_XO_DIST_EN_CLK_ADCDAC When SEQ_XO_DIST_EN_CLK_ADCDAC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_xo_dist_en_clk_adcdac". This bit is ignored when SEQ_XO_DIST_EN_CLK_ADCDAC_OVRD_EN==0.
22 SEQ_XO_DIST_EN_CLK_ADCDAC_OVRD_EN	Override control for SEQ_XO_DIST_EN_CLK_ADCDAC 0b - Normal operation. 1b - Use the state of SEQ_XO_DIST_EN_CLK_ADCDAC_OVRD_EN to override the signal "seq_xo_dist_en_clk_adcdac".
21	Override value for SEQ_XO_EN_CLK_2P4G_OVRD_EN

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEQ_XO_EN_CLK_2P4G_OVRD	When SEQ_XO_EN_CLK_2P4G_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_xo_en_clk_2p4g". This bit is ignored when SEQ_XO_EN_CLK_2P4G_OVRD_EN==0.
20 SEQ_XO_EN_CLK_2P4G_OVRD_EN	Override control for SEQ_XO_EN_CLK_2P4G 0b - Normal operation. 1b - Use the state of SEQ_XO_EN_CLK_2P4G_OVRD to override the signal "seq_xo_en_clk_2p4g".
19 SEQ_XO_DIST_EN_CLK_REF_OVRD	Override value for SEQ_XO_DIST_EN_CLK_REF When SEQ_XO_DIST_EN_CLK_REF_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_xo_dist_en_clk_ref". This bit is ignored when SEQ_XO_DIST_EN_CLK_REF_OVRD_EN==0.
18 SEQ_XO_DIST_EN_CLK_REF_OVRD_EN	Override control for SEQ_XO_DIST_EN_CLK_REF 0b - Normal operation. 1b - Use the state of SEQ_XO_DIST_EN_CLK_REF_OVRD to override the signal "seq_xo_dist_en_clk_ref".
17 SEQ_XO_DIST_EN_OVRD	Override value for SEQ_XO_DIST_EN When SEQ_XO_DIST_EN_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_xo_dist_en". This bit is ignored when SEQ_XO_DIST_EN_OVRD_EN==0.
16 SEQ_XO_DIST_EN_OVRD_EN	Override control for SEQ_XO_DIST_EN 0b - Normal operation. 1b - Use the state of SEQ_XO_DIST_EN_OVRD to override the signal "seq_xo_dist_en".
15 SEQ_VCO_PUP_OVRD	Override value for SEQ_VCO_PUP When SEQ_VCO_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_vco_pup". This bit is ignored when SEQ_VCO_PUP_OVRD_EN==0.
14 SEQ_VCO_PUP_OVRD_EN	Override control for SEQ_VCO_PUP 0b - Normal operation. 1b - Use the state of SEQ_VCO_PUP_OVRD to override the signal "seq_vco_pup".
13 SEQ_PD_PUP_OVRD	Override value for SEQ_PD_PUP When SEQ_PD_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_pd_pup". This bit is ignored when SEQ_PD_PUP_OVRD_EN==0.
12 SEQ_PD_PUP_OVRD_EN	Override control for SEQ_PD_PUP 0b - Normal operation. 1b - Use the state of SEQ_PD_PUP_OVRD to override the signal "seq_pd_pup".
11	Override value for SEQ_PD_EN_FCAL_BIAS

Table continues on the next page...

Table continued from the previous page...

Field	Function
SEQ_PD_EN_FCAL_BIAS_OVRD	When SEQ_PD_EN_FCAL_BIAS_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_pd_en_fcal_bias". This bit is ignored when SEQ_PD_EN_FCAL_BIAS_OVRD_EN==0.
10 SEQ_PD_EN_FCAL_BIAS_OVRD_EN	Override control for SEQ_PD_EN_FCAL_BIAS 0b - Normal operation. 1b - Use the state of SEQ_PD_EN_FCAL_BIAS_OVRD to override the signal "seq_pd_en_fcal_bias".
9 SEQ_RCCAL_PUP_OVRD	Override value for SEQ_RCCAL_PUP When SEQ_RCCAL_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "rx_rccal_pup". This bit is ignored when SEQ_RCCAL_PUP_OVRD_EN==0.
8 SEQ_RCCAL_PUP_OVRD_EN	Override control for SEQ_RCCAL_PUP 0b - Normal operation. 1b - Use the state of SEQ_RCCAL_PUP_OVRD to override the signal "rx_rccal_pup".
7 SEQ_TSM_ISO_B_2P4GHZ_OVRD	Override value for SEQ_TSM_ISO_B_2P4GHZ When SEQ_TSM_ISO_B_2P4GHZ_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_tsm_iso_b_2p4ghz". This bit is ignored when SEQ_TSM_ISO_B_2P4GHZ_OVRD_EN==0.
6 SEQ_TSM_ISO_B_2P4GHZ_OVRD_EN	Override control for SEQ_TSM_ISO_B_2P4GHZ 0b - Normal operation. 1b - Use the state of SEQ_TSM_ISO_B_2P4GHZ_OVRD to override the signal "seq_tsm_iso_b_2p4ghz".
5 SEQ_BG_PUP_IBG_RX_OVRD	Override value for SEQ_BG_PUP_IBG_RX When SEQ_BG_PUP_IBG_RX_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_pup_ibg_rx". This bit is ignored when SEQ_BG_PUP_IBG_RX_OVRD_EN==0.
4 SEQ_BG_PUP_IBG_RX_OVRD_EN	Override control for SEQ_BG_PUP_IBG_RX 0b - Normal operation. 1b - Use the state of SEQ_BG_PUP_IBG_RX_OVRD to override the signal "seq_bg_pup_ibg_rx".
3 SEQ_BG_PUP_IBG_TX_OVRD	Override value for SEQ_BG_PUP_IBG_TX When SEQ_BG_PUP_IBG_TX_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_pup_ibg_tx". This bit is ignored when SEQ_BG_PUP_IBG_TX_OVRD_EN==0.
2 SEQ_BG_PUP_IBG_TX_OVRD_EN	Override control for SEQ_BG_PUP_IBG_TX 0b - Normal operation. 1b - Use the state of SEQ_BG_PUP_IBG_TX_OVRD to override the signal "seq_bg_pup_ibg_tx".

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 SEQ_BG_PUP_IBG_XO_DIST_OVRD	Override value for SEQ_BG_PUP_IBG_XO_DIST When SEQ_BG_PUP_IBG_XO_DIST_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_bg_pup_ibg_xo_dist". This bit is ignored when SEQ_BG_PUP_IBG_XO_DIST_OVRD_EN==0.
0 SEQ_BG_PUP_IBG_XO_DIST_OVRD_EN	Override control for SEQ_BG_PUP_IBG_XO_DIST 0b - Normal operation. 1b - Use the state of SEQ_BG_PUP_IBG_XO_DIST_OVRD to override the signal "seq_bg_pup_ibg_xo_dist".

55.4.7.7.3.1.66 TSM OVERRIDE REGISTER 3 (OVRD3)

Offset

Register	Offset
OVRD3	100h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	RX_MODE...	RX_MODE...	TX_MODE...	TX_MODE...	SEQ_SPA...	SEQ_SPA...	SEQ_SPA...	SEQ_SPA...	SEQ_RX_...	SEQ_RX_...	SEQ_CBP...	SEQ_CBP...	SEQ_ADC...	SEQ_ADC...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEQ_RX_...	SEQ_RX_...	SEQ_CBP...	SEQ_CBP...	SEQ_PD_...	SEQ_PD_...	SEQ_DIV...	SEQ_DIV...	SEQ_DIV...	SEQ_DIV...	SEQ_LO_...	SEQ_LO_...	SEQ_LO_...	SEQ_LO_...	SEQ_LO_...	SEQ_LO_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29	Override value for RX_MODE

Table continues on the next page...

Table continued from the previous page...

Field	Function
RX_MODE_OVRD	When RX_MODE_OVRD_EN=1, this value overrides the mission mode state of the signal "rx_mode". This bit is ignored when RX_MODE_OVRD_EN==0.
28 RX_MODE_OVRD_EN	Override control for RX_MODE 0b - Normal operation. 1b - Use the state of RX_MODE_OVRD to override the signal "rx_mode".
27 TX_MODE_OVRD	Override value for TX_MODE When TX_MODE_OVRD_EN=1, this value overrides the mission mode state of the signal "tx_mode". This bit is ignored when TX_MODE_OVRD_EN==0.
26 TX_MODE_OVRD_EN	Override control for TX_MODE_OVRD 0b - Normal operation. 1b - Use the state of TX_MODE_OVRD to override the signal "tx_mode".
25 SEQ_SPARE3_OVRD	Override value for SEQ_SPARE3 When SEQ_SPARE3_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_spare3". This bit is ignored when SEQ_SPARE3_OVRD_EN==0.
24 SEQ_SPARE3_OVRD_EN	Override control for SEQ_SPARE3 0b - Normal operation. 1b - Use the state of SEQ_SPARE3_OVRD to override the signal "seq_spare3".
23 SEQ_SPARE1_OVRD	Override value for SEQ_SPARE1 When SEQ_SPARE1_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_spare1". This bit is ignored when SEQ_SPARE1_OVRD_EN==0.
22 SEQ_SPARE1_OVRD_EN	Override control for SEQ_SPARE1 0b - Normal operation. 1b - Use the state of SEQ_SPARE1_OVRD to override the signal "seq_spare1".
21 SEQ_RX_MIX_PUP_OVRD	Override control for SEQ_RX_MIX_PUP 0b - Normal operation. 1b - Use the state of SEQ_RX_MIX_PUP_OVRD to override the signal "seq_rx_mix_pup".
20 SEQ_RX_MIX_PUP_OVRD_EN	Override control for SEQ_RX_MIX_PUP 0b - Normal operation. 1b - Use the state of SEQ_RX_MIX_PUP_OVRD to override the signal "seq_rx_mix_pup".
19 SEQ_CBPF_PUP_OVRD	Override value for SEQ_CBPF_PUP When SEQ_CBPF_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_cbpf_pup". This bit is ignored when SEQ_CBPF_PUP_OVRD_EN==0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 SEQ_CBPF_PUP_OVRD_EN	Override control for SEQ_CBPF_PUP 0b - Normal operation. 1b - Use the state of SEQ_CBPF_PUP_OVRD to override the signal "seq_cbpf_pup".
17 SEQ_ADC_PUP_OVRD	Override value for RX_DIG_EN When SEQ_ADC_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_adc_pup". This bit is ignored when SEQ_ADC_PUP_OVRD_EN==0.
16 SEQ_ADC_PUP_OVRD_EN	Override control for SEQ_ADC_PUP 0b - Normal operation. 1b - Use the state of SEQ_ADC_PUP_OVRD to override the signal "seq_adc_pup".
15 SEQ_RX_LNA_PUP_OVRD	Override value for SEQ_RX_LNA_PUP When SEQ_RX_LNA_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_rx_lna_pup". This bit is ignored when SEQ_RX_LNA_PUP_OVRD_EN==0.
14 SEQ_RX_LNA_PUP_OVRD_EN	Override control for SEQ_RX_LNA_PUP 0b - Normal operation. 1b - Use the state of SEQ_RX_LNA_PUP_OVRD to override the signal "seq_rx_lna_pup".
13 SEQ_CBPF_EN_DCOC_OVRD	Override value for SEQ_CBPF_EN_DCOC When SEQ_CBPF_EN_DCOC_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_cbpf_en_dcoc". This bit is ignored when SEQ_CBPF_EN_DCOC_OVRD_EN==0.
12 SEQ_CBPF_EN_DCOC_OVRD_EN	Override control for SEQ_CBPF_EN_DCOC 0b - Normal operation. 1b - Use the state of SEQ_CBPF_EN_DCOC_OVRD to override the signal "seq_cbpf_en_dcoc".
11 SEQ_PD_EN_PD_DRV_OVRD	Override value for SEQ_PD_EN_PD_DRV When SEQ_PD_EN_PD_DRV_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_pd_en_pd_drv". This bit is ignored when SEQ_PD_EN_PD_DRV_OVRD_EN==0.
10 SEQ_PD_EN_PD_DRV_OVRD_EN	Override control for SEQ_PD_EN_PD_DRV 0b - Normal operation. 1b - Use the state of SEQ_PD_EN_PD_DRV_OVRD to override the signal "seq_pd_en_pd_drv".
9 SEQ_DIVN_OPENLOOP_OVRD	Override value for SEQ_DIVN_OPENLOOP When SEQ_DIVN_OPENLOOP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_divn_openloop". This bit is ignored when SEQ_DIVN_OPENLOOP_OVRD_EN==0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 SEQ_DIVN_OPENLOOP_OVRD_EN	Override control for SEQ_DIVN_OPENLOOP 0b - Normal operation. 1b - Use the state of SEQ_DIVN_OPENLOOP_OVRD to override the signal "seq_divn_openloop".
7 SEQ_DIVN_PUP_OVRD	Override value for SEQ_DIVN_PUP When SEQ_DIVN_PUP_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_divn_pup". This bit is ignored when SEQ_DIVN_PUP_OVRD_EN==0.
6 SEQ_DIVN_PUP_OVRD_EN	Override control for SEQ_DIVN_PUP 0b - Normal operation. 1b - Use the state of SEQ_DIVN_PUP_OVRD to override the signal "seq_divn_pup".
5 SEQ_LO_PUP_VLO_TXDRV_OVRD	Override value for SEQ_LO_PUP_VLO_TXDRV When SEQ_LO_PUP_VLO_TXDRV_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_lo_pup_vlo_txdrv". This bit is ignored when SEQ_LO_PUP_VLO_TXDRV_OVRD_EN==0.
4 SEQ_LO_PUP_VLO_TXDRV_OVRD_EN	Override control for SEQ_LO_PUP_VLO_TXDRV 0b - Normal operation. 1b - Use the state of SEQ_LO_PUP_VLO_TXDRV_OVRD to override the signal "seq_lo_pup_vlo_txdrv".
3 SEQ_LO_PUP_VLO_TX_OVRD	Override value for SEQ_LO_PUP_VLO_TX When SEQ_LO_PUP_VLO_TX_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_lo_pup_vlo_tx". This bit is ignored when SEQ_LO_PUP_VLO_TX_OVRD_EN==0.
2 SEQ_LO_PUP_VLO_TX_OVRD_EN	Override control for SEQ_LO_PUP_VLO_TX 0b - Normal operation. 1b - Use the state of SEQ_LO_PUP_VLO_TX_OVRD to override the signal "seq_lo_pup_vlo_tx".
1 SEQ_LO_PUP_VLO_RX_OVRD	Override value for SEQ_LO_PUP_VLO_RX When SEQ_LO_PUP_VLO_RX_OVRD_EN=1, this value overrides the mission mode state of the signal "seq_lo_pup_vlo_rx". This bit is ignored when SEQ_LO_PUP_VLO_RX_OVRD_EN==0.
0 SEQ_LO_PUP_VLO_RX_OVRD_EN	Override control for SEQ_LO_PUP_VLO_RX 0b - Normal operation. 1b - Use the state of SEQ_LO_PUP_VLO_RX_OVRD to override the signal "seq_lo_pup_vlo_rx".

55.4.7.8 Transceiver DMA and Packet RAM Debug Modes

55.4.7.8.1 Introduction

For debug and evaluation purposes, the Radio provides the capability to capture a variety of internal data during the reception process, and store it to either system memory (using Transceiver DMA mode) or radio packet ram (using Packet RAM Debug mode), for analysis and post-processing

55.4.7.8.1.1 Overview

During silicon debug and evaluation, it is often desirable to gain access to data internal to the digital receiver, in order to, for example:

- troubleshoot packet loss issues
- troubleshoot unexpectedly high Bit Error Rate
- fine-tune receiver initialization and configuration parameters

The radio offers 3 methods of accessing such internal data:

1. Transceiver DMA mode
2. Packet RAM Debug mode
3. Digital Test Mux (DTEST)

This section describes the first 2 methods.

Examples of internal data desirable for capture are raw ADC samples, RX digital I and Q outputs, and PHY soft decision data. Transceiver DMA and Packet RAM Debug modes allow a source of internal data to be selected, automate the acquisition of the data, pack the data into a format suitable and optimized for its destination (system or radio memory), implement the interfaces required to access those memories, store the data to memory under program control, and subsequently allow host MCU access to the acquired data for analysis and post-processing. For DMA mode, the SoC's RAM is used via the SoC DMA. For RAM debug mode, the radio packet RAM is made available for this debug mode.

55.4.7.8.1.2 Features

- Captures any of several sources from RX_DIG or PHY
- DMA engine outputs data to an Async FIFO in the SoC which communicates with SoC DMA controller to access system memory
- Packet RAM Debug engine automates transfers to radio memory
- Efficient packing of source data optimized to destination memory dimensions
- Efficient packing of source data optimized to make best use of available bandwidth
- Samples can be decimated by x1, x2, x4, or x8 independently for DMA and Packet RAM
- DMA mask feature allows RSM and LCL_CTRL to specify when samples should be captured
- DMA and Packet RAM engine clocking disabled in mission mode to reduce power
- Simultaneous DMA and Packet RAM Debug operation possible
- Hardware start-triggering capability with several hardware sources
- Start trigger can be delayed by programmable amount up to 2ms (11bits @ 1us resolution)
- Stop-on-trigger capability for Packet RAM Debug

55.4.7.8.1.3 Block Diagram

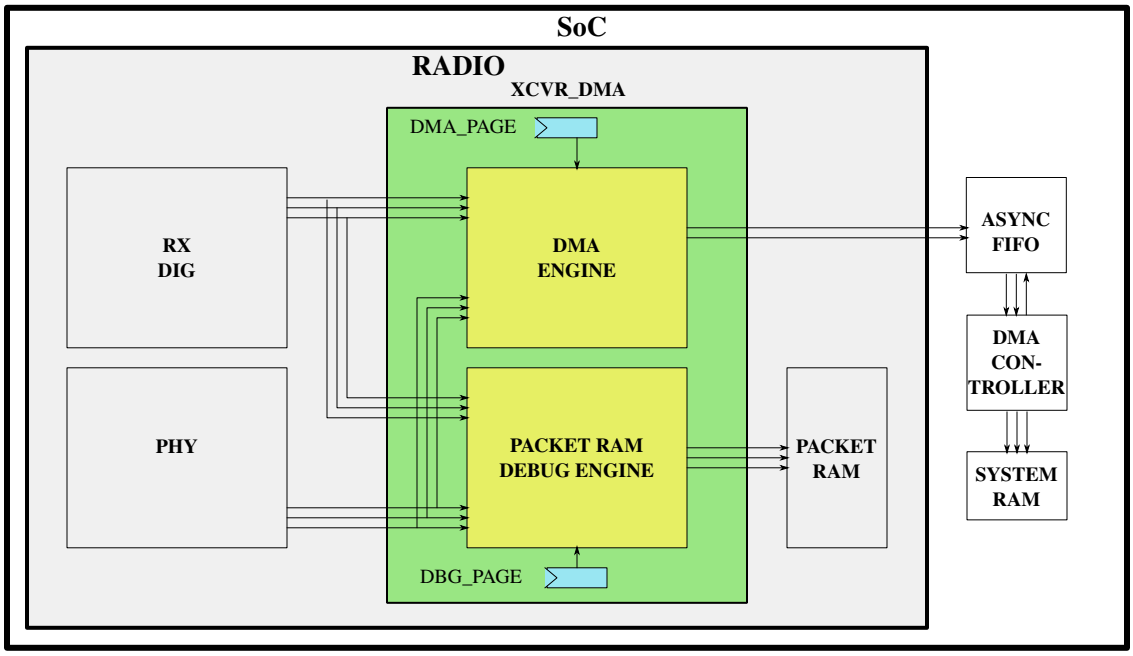


Figure 338. Transceiver DMA and Packet RAM Debug

55.4.7.8.2 Functional Description

55.4.7.8.2.1 Transceiver DMA Mode

Tranceiver DMA mode takes internal receive data from the RX_DIG or PHY, packs the data into an optimal format for bandwidth or RAM dimensions, and hands off the data to an external (SoC) Async FIFO which in turn communicates with the SoC DMA controller for ultimate transfer to system memory. Control and configuration of transceiver DMA operation is governed by the bit fields in the DMA_CTRL register of XCVR address space. These bit fields are referenced in the description that follows. In a typical transceiver DMA debug session, the transceiver DMA hardware performs the following steps:

1. Selects a source of internal data from RX_DIG or PHY for capture based on DMA_PAGE[3:0] setting
2. Waits for a hardware start-trigger, if so configured
3. Waits for a programmable delay (up to 2ms @ 1us resolution), if so configured
4. Uses a defined capture signal from RX_DIG or PHY to latch the internal data into a holding register. Depending on DMA_DEC[1:0], every sample, every other sample, every 4th sample, or every 8th sample will be captured.
5. If capture size is less than 32-bits, packs 2 or more captures of internal data into a 32-bit register until the packing requirements for that DMA_PAGE setting are met
6. Drives 32-bit data along with a valid strobe signal to an SoC Asynchronous FIFO which communites with the SoC DMA controller
7. Repeats steps 3-5 until the debug session is complete (DMA_PAGE set to 0).

If a hardware start-trigger is selected (DMA_START_TRG > 0), the dma engine will wait until the selected triggering event occurs before acquiring any samples; otherwise, acquisition begins as soon as DMA_PAGE is set to a non-zero value.

The maximum DMA transfer rate in this mode is not limited by the radio: the transceiver DMA is capable of sustained 32bit transfers at the reference clock rate, so any limitation would be due to the Async FIFO, DMA, or SoC clocking.

55.4.7.8.2.1.1 PAGE TABLE for 2.4GHz Transceiver Gen4.5

The following table describes the available DMA_PAGE settings. The internal source data selected, the capture signal employed, the data packing method, the OSR setting, and the data rate, are provided for each DMA_PAGE setting.

MNEMONIC	PACKING	CAPTURE SIGNAL	REMARKS
DMAIDLE	---	---	No DMA Activity. All clocks to transceiver DMA hardware gated off.
RXDIG-IQ	{rx_dft_iq_out_q[10:0], rx_mixer_idx[9:5], rx_dft_iq_out_i[10:0], rx_mixer_idx[4:0]}	rx_dft_iq_out_vld	Q and I data captured at various locations in RXDIG. Data is MSB aligned in each 16bit half-word. Check XCVR_RX_DIG->CTRL1[RX_MIXER_IDX_OUT_MODE] for rx_mixer_idx[9:0] mapping.
RXDIG-IQ-ALT	{rx_dft_iq_out_q[10:0], 3'b0, antx_out, crc_fail, rx_dft_iq_out_i[10:0], 3'b0, rx_pd_fnd, aa_sfd_matched}	rx_dft_iq_out_vld	Q and I data captured at various locations in RXDIG plus additional info in the LSBs. I/Q data is MSB aligned in each 16bit half-word.
ADC-IQ	{adc_code_q[10:0], 5'b0, adc_code_i[10:0], 5'b0}	rx_dig_en	Raw ADC capture of I and Q channel. Data is MSB aligned in each 16bit half-word. Bandwidth requirement is 1 word per reference clock cycle.
PHASE	{rx_dft_ph_out[5:0], 2'b0,rx_dft_ph_out[5:0], 2'b0,rx_dft_ph_out[5:0], 2'b0,rx_dft_ph_out[5:0], 2'b0}	rx_dft_ph_out_vld	Sync phase or demod phase (as selected in RXDIG). Newest sample in MSB. Each byte contains a MSB-aligned phase sample.
RSSI-PHASE	{rx_dft_rssi_out[7:0], rx_dig_ph_hi_res[7:0], rx_dft_rssi_out[7:0], rx_dig_ph_hi_res[7:0]}	rx_dft_rssi_ph_hi_res_vld	8bit Narrowband RSSI (rssi, rssi raw, lqi, snr, noise) or Wideband RSSI (rss, rssi raw) plus 8bit high-resolution phase. Newest sample in MS bytes.
MAG-PHASE	{rx_dig_ph_hi_res[7:0], 8'b0, rx_dft_mag_out[11:0], 4'b0}	rx_dft_mag_out_vld	RSSI magnitude + 8bit high-resolution phase.
2P4GHz-PHY	{phy_dma_mux[31:0]}	phy_dma_vld	PHY DMA mux output as selected in PHY
DETERMINISTIC	{counter_q[10:0], 5'b0, counter_i[10:0], 5'b0}	rx_dig_iq_vld	Mimics the RXDIGIQ page, but uses deterministic data for software debug purposes. The Q up-counter start from

Table continues on the next page...

Table continued from the previous page...

MNEMONIC	PACKING	CAPTURE SIGNAL	REMARKS
			0x400, the I up-counter starts from 0x000. The counters reset when dma_page=0, and start counting when the selected trigger fires

55.4.7.8.2.1.2 Example Procedure

The following steps outline the setup and execution of a Transceiver DMA debug session. This example acquires RX DIG I/Q data:

1. Program Receiver for desired protocol
2. If needed, configure RXDIG or PHY to enable the desired DMA output signals from those modules.
3. Enable clocking to the DMA engine by setting DMA_CTRL[DMA_EN]=1
4. Select a DMA start-trigger source, if desired, by setting DMA_CTRL[DMA_START_TRIG] to a non-zero value
5. Select a start-trigger delay value, if desired, by setting DMA_CTRL[DMA_START_DLY] to a non-zero value in microseconds.
6. Select a decimation rate for samples captured, if desired, by setting DMA_CTRL[DMA_DEC] to a non-zero value
7. Program DMA_CTRL[DMA_PAGE]=RXDIGIQ
8. Program the SOC DMA to capture data
9. Start the transceiver's RX sequence by e.g. setting TSM_CTRL[FORCE_RX_EN]=1
10. Begin transmitting RF to the DUT
11. Wait for sufficient RX data to be collected to satisfy the requirements of the debug session. E.g., wait for SOC DMA interrupt
12. Program DMA_CTRL[DMA_PAGE]=0 to terminate the acquisition
13. Download the captured data from system memory

Note: Step 7 (select the DMA_PAGE to enable the DMA engine) should be performed as a sequential, discrete write to the DMA_CTRL register, after steps 3-6 (enabling the clock, and configuring start delay, decimation, and start trigger) have been completed. A single write to DMA_CTRL is permitted for for steps 3-6.

55.4.7.8.2.2 Packet RAM Debug Mode

Packet RAM Debug mode takes internal receive data from the RX_DIG or PHY, packs the data into an format suitable for storage in the radio's Packet RAM, and then directly fills the RAMswith source data, under program control. Control and configuration of Packet RAM Debug operation is governed by the bit fields in the DBG_RAM* register of XCVR address space. These bit fields are referenced in the description that follows. In a typical Packet RAM debug session, the debug hardware performs the following steps:

1. Selects a source of internal data from RX_DIG or PHY for capture based on DBG_PAGE[3:0] setting
2. Waits for a hardware start-trigger, if so configured
3. Waits for a programmable delay (up to 2ms @ 1us resolution), if so configured
4. Uses a defined capture signal from RX_DIG or PHY to latch the internal data into a holding register. Depending on DBG_DEC[1:0], every sample, every other sample, every 4th sample, or every 8th sample will be captured.
5. If capture size is less than 32-bits, packs 2 or more captures of internal data into a 32-bit register until the packing requirements for that DBG_PAGE setting are met

6. Directly accesses the Packet RAM to store the receive data (addressing, write enabling, chip enabling, etc.). The first and last address to use in the ram can be programmed.
7. Repeats steps 3-5 until the debug session is complete: DBG_PAGE set to 0 ; a stop-on-trigger event occurs if so configure; or packet RAM (last address above reached) is full.

The bandwidth to Packet RAM is expected to be higher than Transceiver DMA, since the overhead of DMA-related synchronization and multi-master arbitration can be avoided. The tradeoff is that the total Packet RAM space is smaller than the space available in system memory, restricting the size of the block of data that can be saved.

55.4.7.8.2.2.1 Acquisition

In a typical debug session, the selection of a receive data source is made by setting DBG_PAGE to a non-zero value (see the table below for a description of the debug pages). Setting DBG_PAGE to a non-zero value activates the Packet RAM Debug Engine. The engine has write-only access to the RAM while receive data is being collected. (XCVR host IPS access is not blocked, but should be avoided during acquisition). If a hardware start-trigger is selected (DBG_START_TRG > 0), the debug engine will wait until the selected triggering event occurs before acquiring any samples; otherwise, acquisition begins as soon as DBG_PAGE is set to a non-zero value. Starting at the address indicated by DBG_RAM_FIRST, the RAM will fill with receive data determined by the DBG_PAGE setting. The address for the RAM is always reset to DBG_RAM_FIRST when DBG_PAGE=0.

If a stop-trigger is not selected (DBG_STOP_TRG = 0): When the RAM reaches DBG_RAM_LAST (programmable last address), further write attempts by the debug engine to the RAM are halted, and the DBG_RAM_FULL status bit become set. When sufficient receive data has been acquired to satisfy the requirements of the debug session, the acquisition should be terminated by setting DBG_PAGE=0. The DBG_RAM_FULL and DBG_RAM_STOP status bits are "sticky"; they will remain valid even after DBG_PAGE is set to 0 to end the session. The bits will self-clear when a new acquisition is started.

If a stop-trigger is selected (DBG_STOP_TRG > 0): When the RAM reaches DBG_RAM_LAST address, further write attempts by the debug engine to the RAM are not halted, but the DBG_RAM_FULL status bits do become set. The debug engine will wrap back to the starting address (DBG_RAM_FIRST) and continue filling, until the selected stop-trigger event occurs. When the stop-trigger event occurs, the debug engine will stop filling the RAM immediately, the DBG_STOP_TRIGGERED status bit will become set, and the register DBG_RAM_STOP will hold the address of the last RAM word filled with a valid sample before the trigger. At this point, the acquisition should be terminated by setting DBG_PAGE=0. The DBG_STOP_TRIGGERED and STOP_ADDRESS are "sticky"; they will hold their state even after DBG_PAGE is set to 0 to end the session. DBG_TRIGGERED will self-clear, and DBG_RAM_STOP will reset to 0, when a new acquisition is started by setting DBG_PAGE to a non-zero value.

55.4.7.8.2.2.2 Downloading

As soon as sufficient data has been collected to satisy the requirements of the debug session, DBG_PAGE should be set to 0 to end the acquisition. The receive data stored in RAM may now be downloaded from the Packet RAM for analysis and post-processing.

55.4.7.8.2.2.3 PAGE TABLE for 2.4GHz Transceiver Gen4.5

The DBG_PAGE table settings are the same as the DMA_PAGE table except that the DETERMINISTIC page is not supported for DBG_PAGE

55.4.7.8.2.2.4 Example Procedure

The following steps outline the setup and execution of a Packet RAM debug session. This example acquires RX DIG I/Q data

1. Program Receiver for desired protocol
2. If needed, configure RXDIG or PHY to enable the desired DMA output signals from those modules.
3. Enable clocking to the Packet RAM Debug engine by setting DBG_RAM_CTRL[DBG_EN]=1
4. Select a DBG start-trigger source, if desired, by setting DBG_RAM_CTRL[DBG_START_TRG] to a non-zero value
5. Select a start-trigger delay value, if desired, by setting DBG_RAM_CTRL[DBG_START_DLY] to a non-zero value in microseconds.
6. Select a decimation rate for samples captured, if desired, by setting DBG_RAM_CTRL[DBG_DEC] to a non-zero value

7. Select a DBG stop-trigger source, if desired, by setting DBG_RAM_CTRL[DBG_STOP_TRG] to a non-zero value
8. Program DBG_RAM_ADDR to select the desired starting address (DBG_RAM_FIRST) and ending address (DBG_RAM_LAST) in the RAM
9. Program DBG_RAM_CTRL[DBG_PAGE]=RXDIGIQ
10. Start the transceiver's RX sequence by e.g. setting TSM_CTRL[FORCE_RX_EN]=1
11. Begin transmitting RF to the DUT
12. **If a stop-trigger is selected (DBG_STOP_TRG > 0)** wait for DBG_STOP_TRIGGERED=1, indicating a stop-trigger occurred
13. **If a stop-trigger is not selected (DBG_STOP_TRG = 0)** wait for sufficient RX data to be collected, or poll DBG_RAM_FULL, to satisfy the requirements of the debug session
14. Program DBG_RAM_CTRL[DBG_PAGE]=0 to terminate the acquisition
15. Download the captured data from Packet RAM

Note: Step 9 (select the DBG_PAGE to enable the DBG engine) should be performed as a sequential, discrete write to the DBG_RAM_CTRL register, after steps 3-8 (enabling the clock, and configuring start delay, decimation, start/stop triggers and ram starting address) have been completed. A single write to DBG_RAM_CTRL is permitted for steps 3-7.

55.4.7.8.2.3 DMA Mask

The DMA mask feature allows the XCVR_DMA to remain enabled but only be active when the DMA mask input is asserted high, indicating that there are desirable samples to be captured. The DMA mask input to the Transceiver DMA module is generated by an OR of the DMA mask outputs from the following modules:

- LCL_CTRL:
  - for CTE use case: the DMA mask output ensures that samples are captured during the reference period and during sample slots, but not during switching slots or other times
  - when use with RSM: the DMA mask output is used to avoid capture of samples after the antenna is switched during RSM's PM_RX state
- RSM: the DMA mask output is used to identify samples during RSM FM_RX state, and optionally during PM_RX states (if LCL_CTRL is not enabled during RSM's PM_RX states)

Also note that, for RSM use cases, the Transceiver DMA can remain enabled through an entire RSM event which consists of multiple RX (and TX) bursts; that it, it is not necessary to re-enable the Transceiver DMA for each individual RX burst within an RSM event.

The DMA mask feature in XCVR_DMA is only enabled if the DBG_SIGNAL_VALID_MASK_EN bit (Packet RAM Debug Mode) or DMA_SIGNAL_VALID_MASK_EN bit (Transceiver DMA Mode) bit is set. When enabled, the data valid strobe input selected by DBG_PAGE (Packet RAM Debug Mode) or DMA_PAGE(Transceiver DMA Mode) is AND'ed with the DMA mask input signal so that samples are only captured while the DMA mask signal is high and when there is valid data.

NOTE

The RXDIG also supports a feature to mask its data valid strobe outputs to the Transceiver DMA using the DMA mask signal (OR of RSM and LCL_CTRL DMA mask signals, as indicated above). This RXDIG feature (enabled by setting RXDIG's RX_IQ_PH_OUTPUT_COND bit), can also be combined with RXDIG's IQ phase output average windowing feature to reduce the number of samples which need to be stored to RAM, so this is method is generally preferred over enabling the DMA mask in the Transceiver DMA module. If using the RXDIG feature (RX_IQ_PH_OUTPUT_COND bit set), then the DBG_SIGNAL_VALID_MASK_EN (Packet RAM Debug Mode) and DMA_SIGNAL_VALID_MASK_EN (Transceiver DMA Mode) bits should not be set.

55.4.7.8.2.4 Simultaneous Transceiver DMA and Packet RAM Debug

Transceiver DMA mode and Packet RAM Debug mode are described in the previous sections. The DMA and Debug engines are separate, independent entities, which theoretically allows both to be employed simultaneously. The most practical case for simul operation would be to select an RX_DIG source for DMA_PAGE and a PHY source for DBG_PAGE. During such a simultaneous debug session, RX_DIG data would go to system memory and PHY data to Packet RAM. After acquisition, data would be downloaded from both memories. Post-processing software could consolidate the data and take advantage of the synergy that results from combining data captured simultaneously from multiple sources. (DTEST allows a possible third simultaneous source, beyond the scope of this Block Guide).

In addition, it is also theoretically possible to combine either or both debug modes with mission-mode packet reception, with the caveat that the Packet RAM would not be available to receive mission-mode packet data in such a scenario. (Packet processing, filtering, CRC checking, and Link Layer interrupts don't rely on packet RAM and so would not be impacted by the debug modes). Such a combination has been demonstrated successfully in radio-level simulation, with the following setup:

1. The GEN_FSK Link Layer is engaged in RX mode
2. Transceiver DMA mode is engaged with DMA_PAGE set to PHYPHASE (PHY phase capture)
3. Packet RAM Debug mode is engaged with DBG_PAGE set to DMDSOFT (PHY soft decision data)
4. A GEN_FSK standard packet is received, CRC passes, and GEN_FSK RX_IRQ and SEQ_END_IRQ are asserted (GEN_FSK packet data not stored)
5. RX phase data is downloaded from system memory and verified
6. Demod soft decision data is downloaded from Packet RAM and verified

55.4.7.8.2.5 Start Trigger Sources

Registers DMA_CTRL[DMA_START_TRG] and DBG_RAM_CTRL[DBG_START_TRG], enable a hardware trigger that must assert before acquisition begins, for the transceiver DMA and Packet RAM debug engines, respectively. The same description applies to both registers (i.e., trigger sources are the same), as indicated in the following table:

xxx_START_TRG[3:0]	HARDWARE TRIGGER for 2.4GHz
0	no trigger
1	PHY: pd_fnd_to_ll
2	PHY: aa_fnd_to_ll
3	ZBDEMOD: rx_preamble_det
4	ZBDEMOD: rx_sfd_detect
5	RXDIG: agc_gain_chg
6	TSM: rx_dig_en
7	TSM: tsm_irq0_start_trig
8	CRC pass
9	CRC done (Not used for 15.4 LL)
10	Localization control: pattern match
11	GenericLL: cte_present
12	Ranging sequence manager: dma_trigger

55.4.7.8.2.6 Stop Trigger Sources

Register DBG_RAM_CTRL[DBG_STOP_TRG], enables a hardware trigger such that, when it asserts, acquisition terminates immediately, in Packet RAM debug mode, as indicated in the following table (some of these table entries need discussion):

DBG_STOP_TRG[3:0]	HARDWARE TRIGGER for 2.4GHz
0	no trigger
1	PHY: pd_fnd_to_ll
2	PHY: aa_fnd_to_ll
3	ZBDEMOD: rx_preamble_det
4	ZBDEMOD: rx_sfd_detect
5	RXDIG: agc_gain_chg
6	TSM: rx_dig_en
7	TSM: tsm_irq1_stop_trig
8	CRC fail
9	CRC done (Not used for 15.4 LL)
10	RBME ERROR
11	GenLL: HEADER FAIL
12	TSM: pll unlock

Note: For DBG_STOP_TRIG=11 (HEADER FAIL), register GEN_WHITEN_SZ_THR[REC_BAD_PKT] must be set to 0 (default) to cause an RX recycle on a header violation.

55.4.7.9 Transceiver Antenna and Localization Control

55.4.7.9.1 Introduction

The transceiver supports two antenna control mechanisms: Fast Antenna Diversity (FAD), and Localization Control. FAD allows an optimal choice between two antennaes during the PHY's preamble search. Localization Control refers to direction finding (angle of arrival / angle of departure) and ranging. Only one of these is enabled at any given time.

A high level figure illustrating the transceiver modules involved is shown below. The FAD algorithm is in the PHY, which provides an output to control the antenna through the FAD Common pin interface. The Localization control outputs "lant_lut_gpio[3:0]" (to control an external antenna switch) plus a start trigger and DMA mask signal to the XCVR DMA to facilitate capture of receiver samples. For Bluetooth LE direction finding, the Localization Control module accepts CTE control signals from the Bluetooth LE link layer. For ranging, a Ranging Sequence Manager sequences the transceiver between RX and TX operations across a set of programmable frequencies for the purpose of the capturing phase samples (when tones are transmitted) and RTT measurements (when packets are transmitted) in the receiving device, and also provides a trigger to the Localization Control module for optional antenna switching. In addition, the Localization Control module provides a trigger to the SOC (to be connected to a timer) for time-of-flight timestamping; this trigger is a mux of several signals in the transceiver as shown in the figure. The four FAD outputs and LCL_CTRL's lant_lut_gpio[3:0] outputs go to SOC pins via the RF_GPO muxes described in the Coexistence/FEM/LANT connections section of the Radio Control chapter.

The Localization Control , Ranging Sequence Manager and FAD Common Pin Interface are described in the sections below. For more info on the PHY's FAD algorithm refer to the PHY chapter.

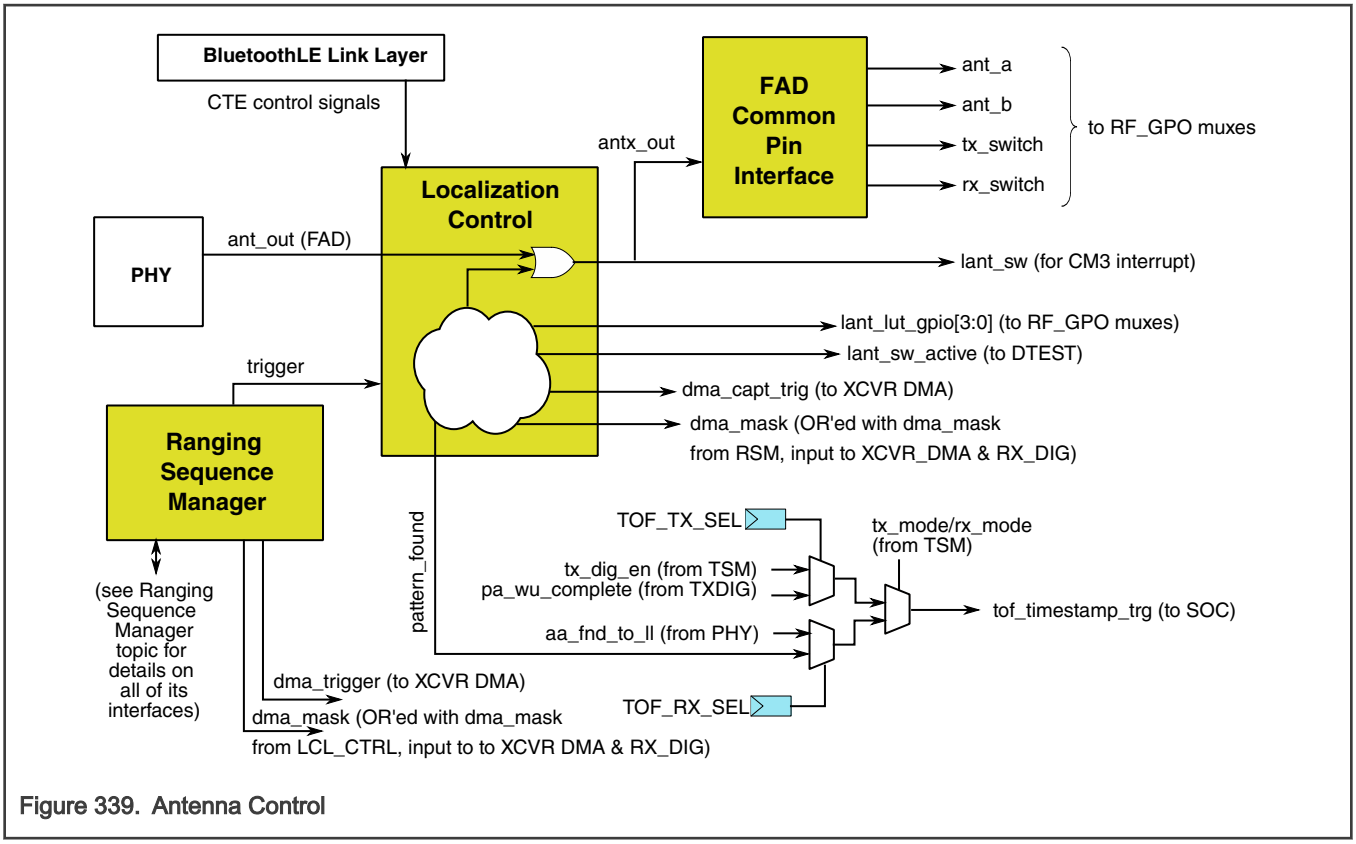


Figure 339. Antenna Control

55.4.7.9.2 Localization Control Module

The LCL_CTRL module implements various controls intended to assist with antenna control and sample capture.

There are 3 significant functions of this module:

1. Scan the bit stream for a pattern
2. Control the switching of the antenna output (lant_lut_gpio[3:0])
3. Provide a data valid mask signal for the XCVR DMA

Within the LCL_CTRL there is pattern matching logic that will assert a pattern found signal when the incoming bit stream matches the pre-programmed pattern. The pattern can be 4, 5, 6, or 8 bytes (setting PM_NUM_BYTES). The pattern must be programmed in the LCL_PM_MSB and LCL_PM_LSB registers and pattern matching must be enabled. If COMP_EN is set, then pattern matching will operate during RX. For pattern matching in TX, both COMP_EN and COMP_TX_EN must be set. The pattern_match output asserts when a pattern match has occurred. For RX, when the pattern_match output asserts the dma_capt_trig to the XCVR_DMA also asserts.

For antenna control, the LCL_CTRL module can be configured using the MODE bit to operate with or without use of the Bluetooth LE Link Layer's Constant Tone Extension (CTE) control signals.

55.4.7.9.2.1 Antenna Control Without Bluetooth LE CTE Signals

For antenna control without using the Bluetooth LE CTE control signals, various settings must be programmed prior to operation. The significant fields are:

- CTE_DUR : Duration of switching activity in doublets, where a doublet = RX/TX_HI_PER + RX/TX_LO_PER
- LANT_INV : When set to 1, the lant_sw signal is inverted
- RX/TX_LCL_EN : Enables antenna switching control in RX and/or TX
- LCL_EN : Enable for the entire LCL_CTRL module

- RX/TX_DELAY_OFF : Extra offset (in samples) to fine tune the RX/TX_DELAY
- RX/TX_DELAY : Delay, in intervals, before antenna switching is to begin
- RX/TX_LO_PER : Number of intervals for which lant_sw should be 0, when switching is active. For RX, RX_LO_PER also indicates the number of intervals that the dma_mask should be 0.
- RX/TX_HI_PER : Number of intervals for which lant_sw should be 1, when switching is active. For RX, RX_HI_PER also indicates the number of intervals that the dma_mask should be 1.
- RX/TX_ANT_TRIG_SEL : Selects the trigger to be observed to begin counting DELAY
- RX/TX_SPINT : Defines the number of samples per interval. This is realized using the sample clock associated with either RX or TX, (typically 4MHz or 8MHz).
- M_ON_DELAY/N_ON_DELAY: The rx/tx_lco_on output will de-assert M_ON_DELAY intervals + N_ON_DELAY doublets earlier than the antenna switching stops.
- LUT_0 to LUT_31: state of lant_lut_gpio[3:0] for the switching activity doublet from 0 to 31
- LUT_WRAP_PTR: Wrap point for the LUT_0 to LUT_31. If programmed to value "N", output will be LUT_0, ..., LUT_N, LUT_0, ...

When enabled, the LCL_CTRL control module begins monitoring the selected trigger source as indicated by RX/TX_ANT_TRIG_SEL. When the trigger occurs, a timer begins. When the timer reaches DELAY+DELAY_OFF, the system will align to the next available sample clock and the antenna switching begins. The switching is described by asserting as 1 for HI_PER, then de-asserting as 0 for LO_PER. One of these cycles is termed a 'doublet'. This initial process is depicted in the figure below.

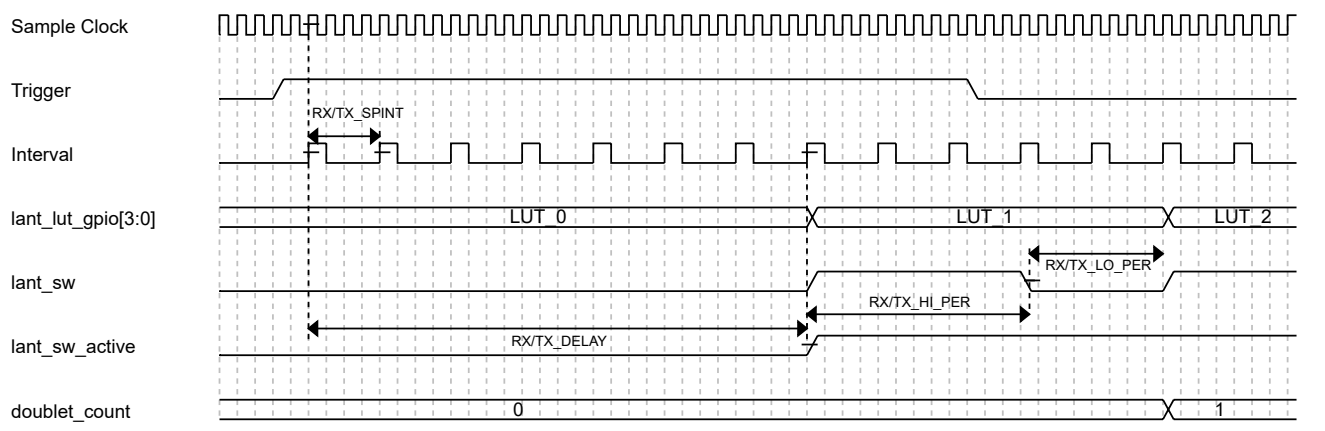


Figure 340. LCL_CTRL startup

After antenna switching has begun, another counter will count the doublets. When the CTE_DUR is reached, switching will stop as depicted in the following figure.

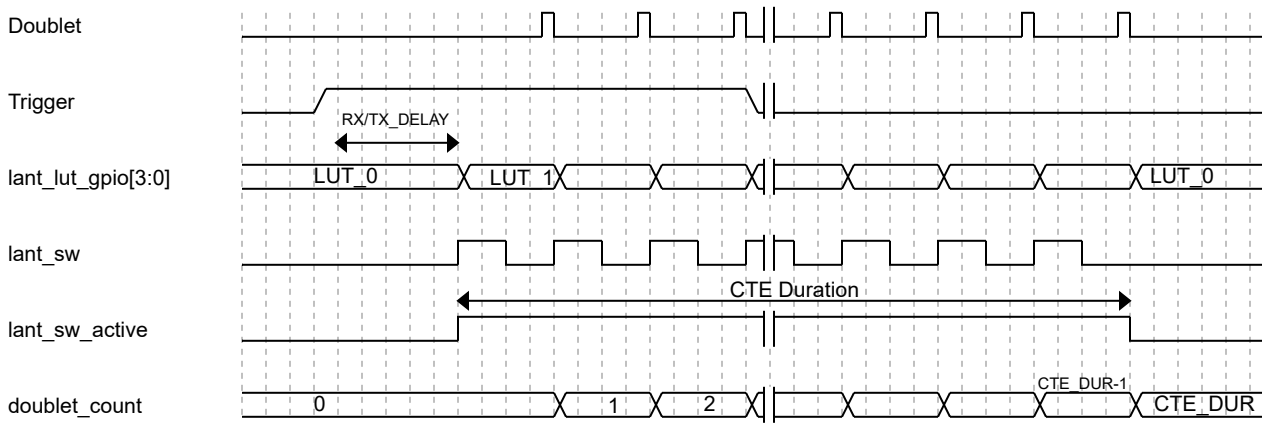


Figure 341. LCL_CTRL shutdown

The LCL_CTLR provides a data valid mask signal for the XCVR DMA so the DMA can avoid capturing unwanted data in the Guard period, and immediately after antenna is switched ("switching slots"). For this feature, there are several additional control settings:

- DMA_MASK_DELAY_OFF: Extra offset (in samples) to fine tune the DMA_MASK_DELAY
- DMA_MASK_DELAY: Delay, in intervals, from RX trigger to the time at which DMA mask asserts
- DMA_MASK_REF_PER: Duration, in intervals, of the reference period after DMA_MASK_DELAY expires during which the dma_mask remains asserted

The following figure depicts the dma_mask signal behavior in a CTE example. As can be seen, the dma_mask allows data to be captured during the reference period and sampling slots while avoid the guard period and switching slots.

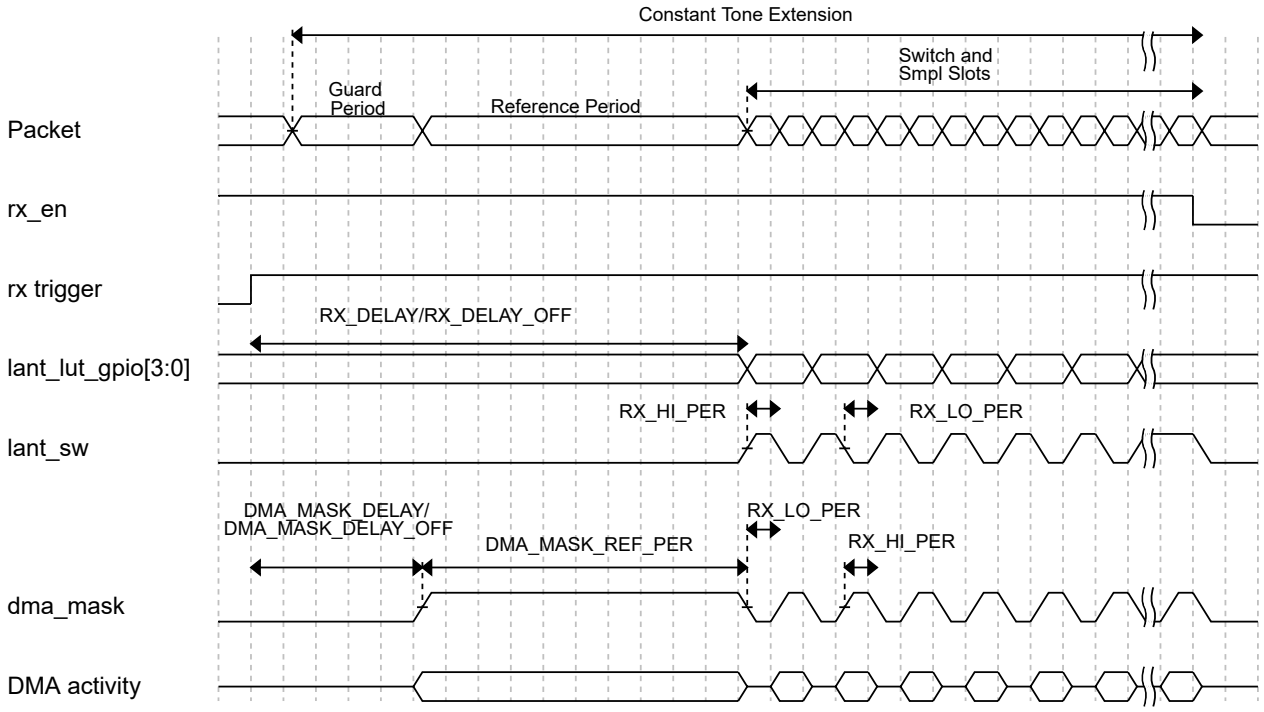


Figure 342. LCL_CTRL dma_mask

55.4.7.9.2.1.1 Antenna Control Used With RSM

Using LCL_CTRL with the RSM is considered a special case of "Antenna Control Without Bluetooth LE CTE Signals."

The RSM's rx and tx trigger output to the LCL_CTRL are expected to be used in this configuration. The RSM will trigger the LCL during it's T_PM (tone exchange) states. Since the RSM supports dynamically switching between 4 different T_PM durations while it is active, the LCL_CTRL include the following registers which are used only in conjunction with RSM:

- RX/TX_LO_PER1-3 : Alternate number of intervals for which lant_sw should be 0, when switching is active. For RX, RX_LO_PER also indicates the number of intervals that the dma_mask should be 0. These are used when RSM signals the LCL_CTRL that it is using T_PM1-3 durations instead of T_PM0 duration. (When T_PM0 duration is used, LCL_CTRL's RX/TX_LO_PER are used.) Refer to Ranging Sequence Manager for more information on T_PM durations.
- RX/TX_HI_PER1-3 : Alternate number of intervals for which lant_sw should be 1, when switching is active. For RX, RX_HI_PER also indicates the number of intervals that the dma_mask should be 1. These are used when RSM signals the LCL_CTRL that it is using T_PM1-3 durations instead of T_PM0 duration. (When T_PM0 duration is used, LCL_CTRL's RX/TX_LO_PER are used.) Refer to Ranging Sequence Manager for more information on T_PM durations.

The figures below illustrate the LCL_CTRL behavior when used with the RSM. Note that the delay from the trigger to the start of antenna switching is different when the RSM is used (as signalled by RSM's "active" output being high) : it includes the LO_PER and HI_PER durations in addition to DELAY/DELAY_OFF. Note that there are two figures for RX: the first one doesn't use DMA_MASK_REF_PER, and the second does.

For Figure 344, the DMA_MASK_REF_PER is not used (programmed to 0). Since the trigger asserts immediately on entry to RSM's T_PM states, the DELAY/DELAY_OFF and DMA_DELAY/DMA_DELAY_OFF values are expected to be programmed to 0 (or very small values), and the DMA mask will be asserted at the end of each LUT_x window.

For Figure 345, the samples captured during the first antenna switching period correspond to DMA_MASK_REF_PER (which would be expected to be programmed to the same value as RX_HI_PER). As compared to the other figure, the DMA mask is centered within the antenna LUT_x window. However, since there is only one DMA_MASK_REF_PER, the logic doesn't support the use of different RX_HI_PER durations / different T_PM durations.

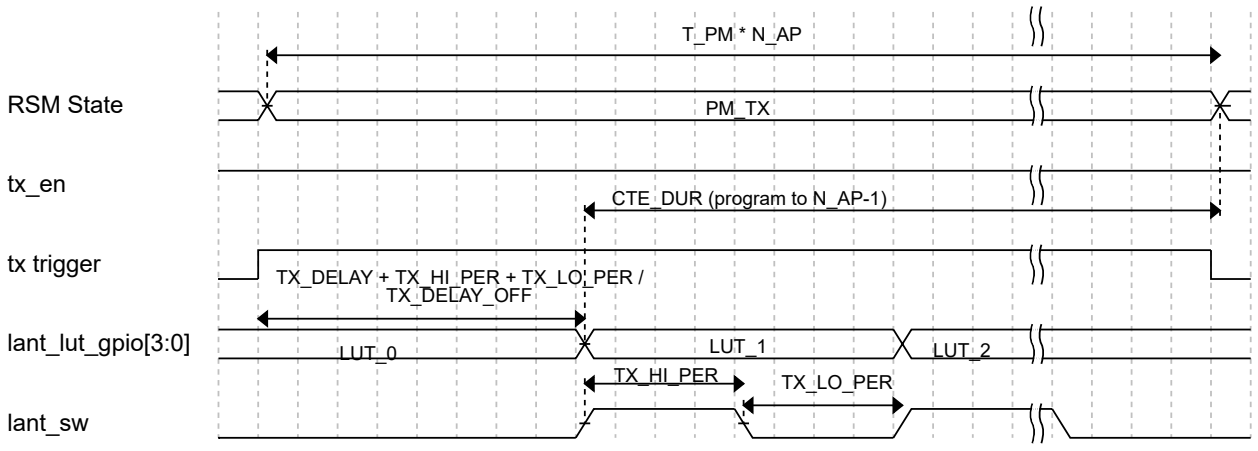


Figure 343. LCL_CTRL Operation with RSM - TX

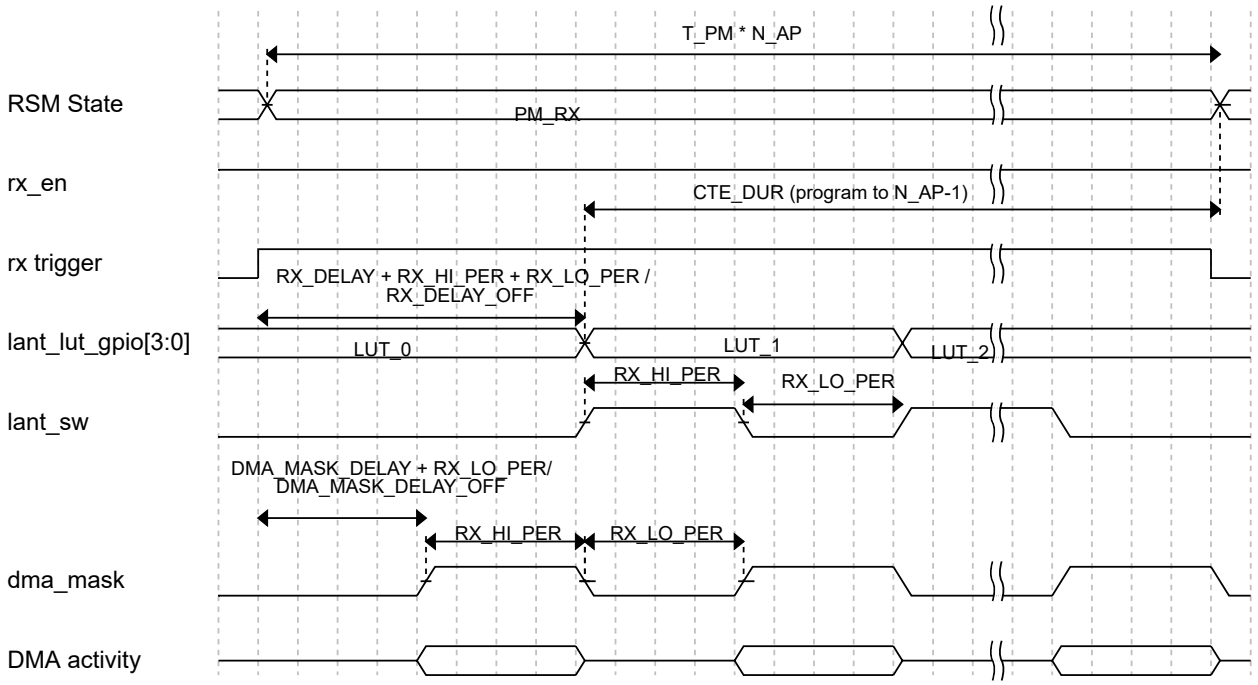


Figure 344. LCL_CTRL Operation with RSM - RX

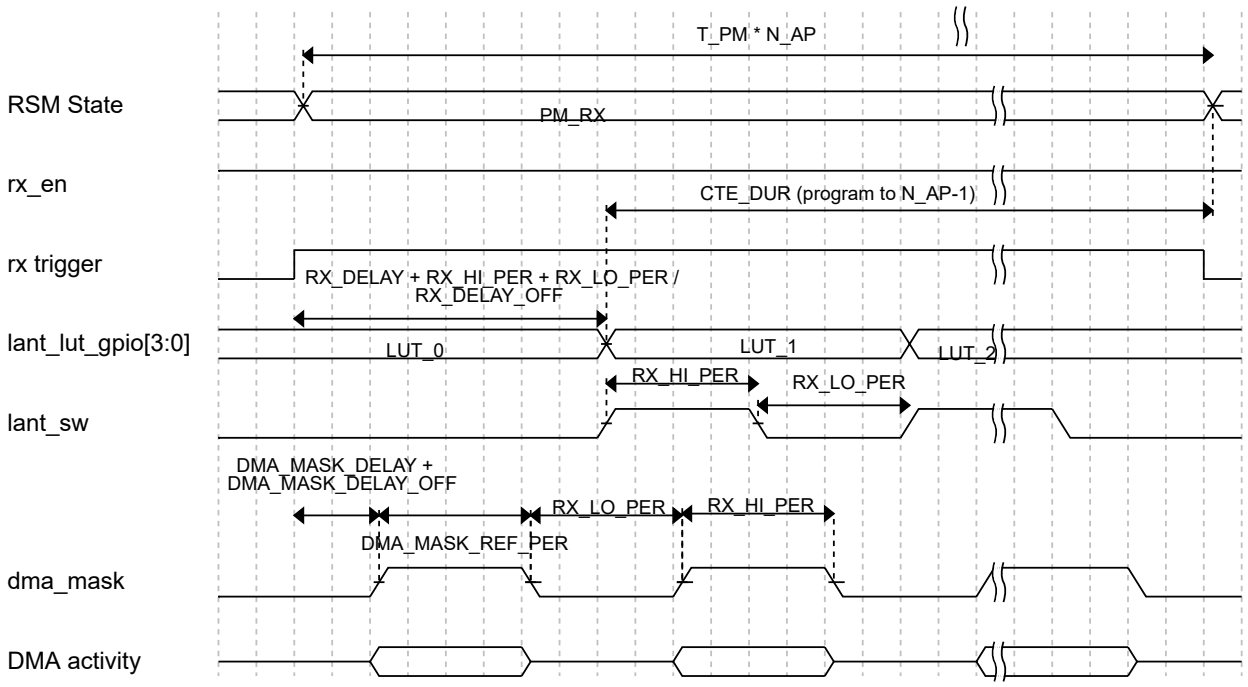


Figure 345. LCL_CTRL Operation with RSM - RX (Alternate Usage)

55.4.7.9.2.2 Antenna Control With Bluetooth LE CTE Signals

In this mode, cte_type[2:0] and cte_sample_dur[2:0] signals from the Bluetooth LE link layer are used to provide configuration of the LCL_CTRL module. The definition for cte_type[2:0] and cte_sample_dur[2:0] are shown in the table below

Table 458. CTE Signal cte_type[2:0]

cte_type[2:0]	Description
0x0	AoA Constant Tone Extension
0x1	AoD Constant Tone Extension with 1us slots
0x2	AoD Constant Tone Extension with 2us slots
0x3	Reserved
0x4	Reserved
0x5	AoD Constant Tone Extension with 4us slots
0x6	AoD Constant Tone Extension with 8us slots
0x7	Reserved

Table 459. CTE Signal cte_sample_dur[2:0]

cte_sample_dur[2:0]	Description
0x0	Reserved

Table continues on the next page...

Table 459. CTE Signal cte_sample_dur[2:0] (continued)

cte_sample_dur[2:0]	Description
0x1	For AoA receive, sampling duration is 1us
0x2	For AoA receive, sampling duration is 2us
0x3	Reserved
0x4	Reserved
0x5	For AoA receive, sampling duration is 4us
0x6	For AoA receive, sampling duration is 8us
0x7	Reserved

For antenna control using the Bluetooth LE CTE control signals, various settings must be programmed prior to operation. The significant fields are:

- LANT_INV : When set to 1, the lant_sw signal is inverted
- RX/TX_LCL_EN : Enables antenna switching control in RX and/or TX. For Bluetooth LE CTE configuration, both bits should be set.
- LCL_EN : Enable for the entire LCL_CTRL module
- RX/TX_DELAY_OFF : Extra offset (in samples) to fine tune the RX/TX_DELAY
- RX/TX_DELAY : Delay, in intervals, before antenna switching is to begin
- RX/TX_ANT_TRIG_SEL : Selects the trigger to be observed to begin counting DELAY
- RX/TX_SPINT : Defines the number of samples per interval. This is realized using the sample clock associated with either RX or TX, (typically 4MHz or 8MHz). For proper operation, RX/TX_SPINT need to be programmed so that the interval is 1us for RX and TX.
- LUT_0 to LUT_31: state of lant_lut_gpio[3:0] for the switching activity doublet from 0 to 31
- LUT_WRAP_PTR: Wrap point for the LUT_0 to LUT_31
- DMA_MASK_REF_PER: Duration, in intervals, of the reference period after DMA_MASK_DELAY expires during which the dma_mask remains asserted. For CTE the reference period is expected to be programmed for 8us.
- DMA_MASK_DELAY_OFF: Extra offset (in samples) to fine tune the DMA_MASK_DELAY
- DMA_MASK_DELAY: Delay, in intervals, from RX trigger to the time at which DMA mask asserts

Note that the following LCL_CTRL fields are not used in this configuration:

- CTE_DUR : In this configuration, the duration is controlled by the Bluetooth LE link layer
- RX/TX_LO_PER : In this configuration, the low period is determined by the CTE control signals
- RX/TX_HI_PER : In this configuration, the high period is determined by the CTE control signals
- M_ON_DELAY/N_ON_DELAY: In this configuration, switching activity stops at the end of the RX or TX burst.
- LANT_BLOCK_RX/TX: In this configuration, for RX lant output is blocked for AoD, and for TX lant output is blocked for AoA

The figures below illustrate the antenna switching operation for Bluetooth LE CTE during Transmit and Receive operations, and an addition figure illustrating the dma_mask behavior for Receive. In the Receive and Transmit figures, three examples of lant_sw / lant_lut_gpio[3:0] behavior are shown for different cte_type (and cte_sample_dur for Receive) input values. In the dma_mask figure, dma_mask and DMA activity behaviors are shown for two example cte_type and cte_sample_dur inputs. Note that these figures show example tx_trigger and rx_trigger signals which assert at the start of the guard period; the actual trigger used in the application (such as "cte_en" from the Bluetooth LE) will likely have different timing so the delays will need to be programmed accordingly.

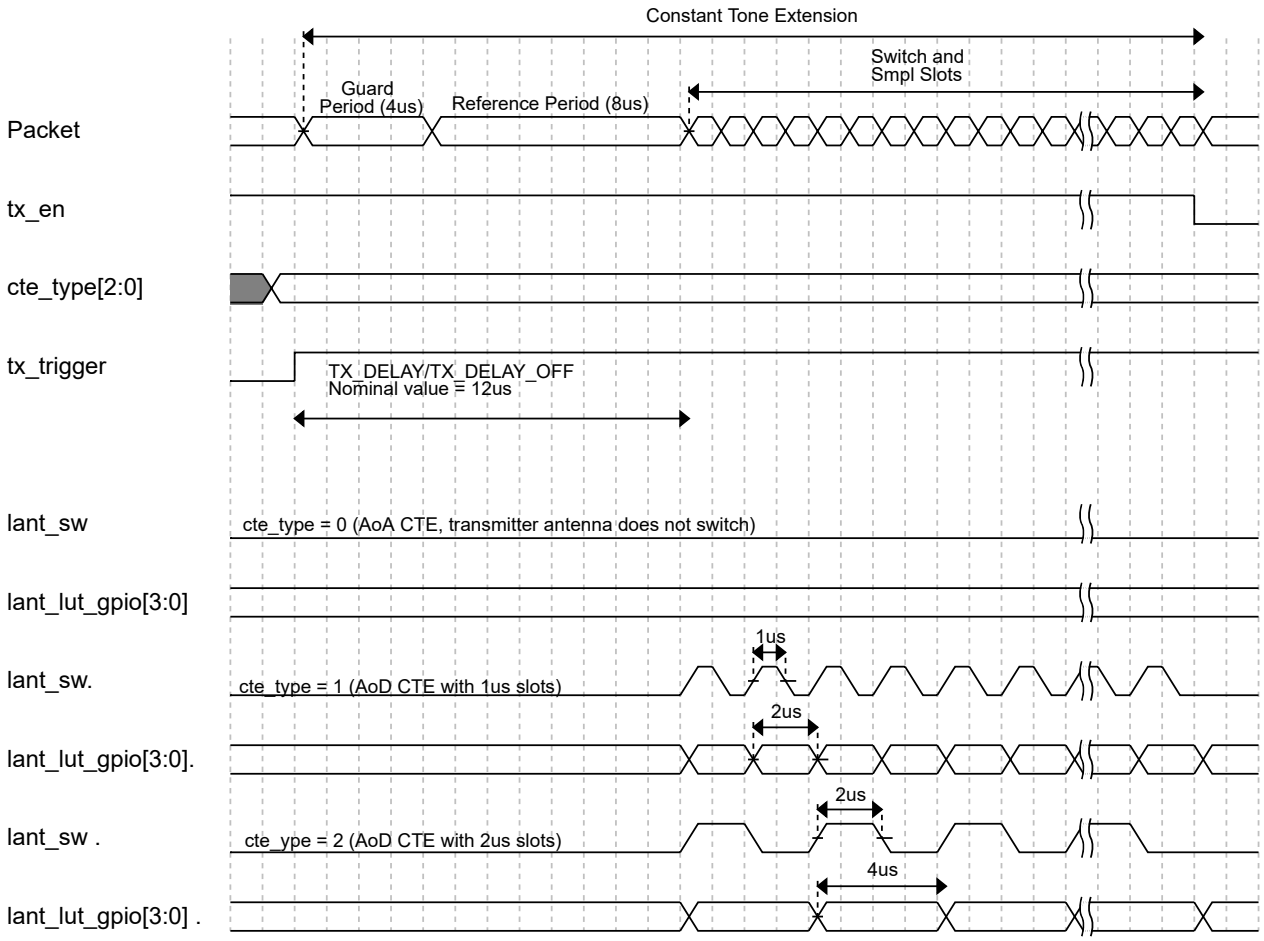


Figure 346. LCL_CTRL CTE - Transmit Antenna Switching

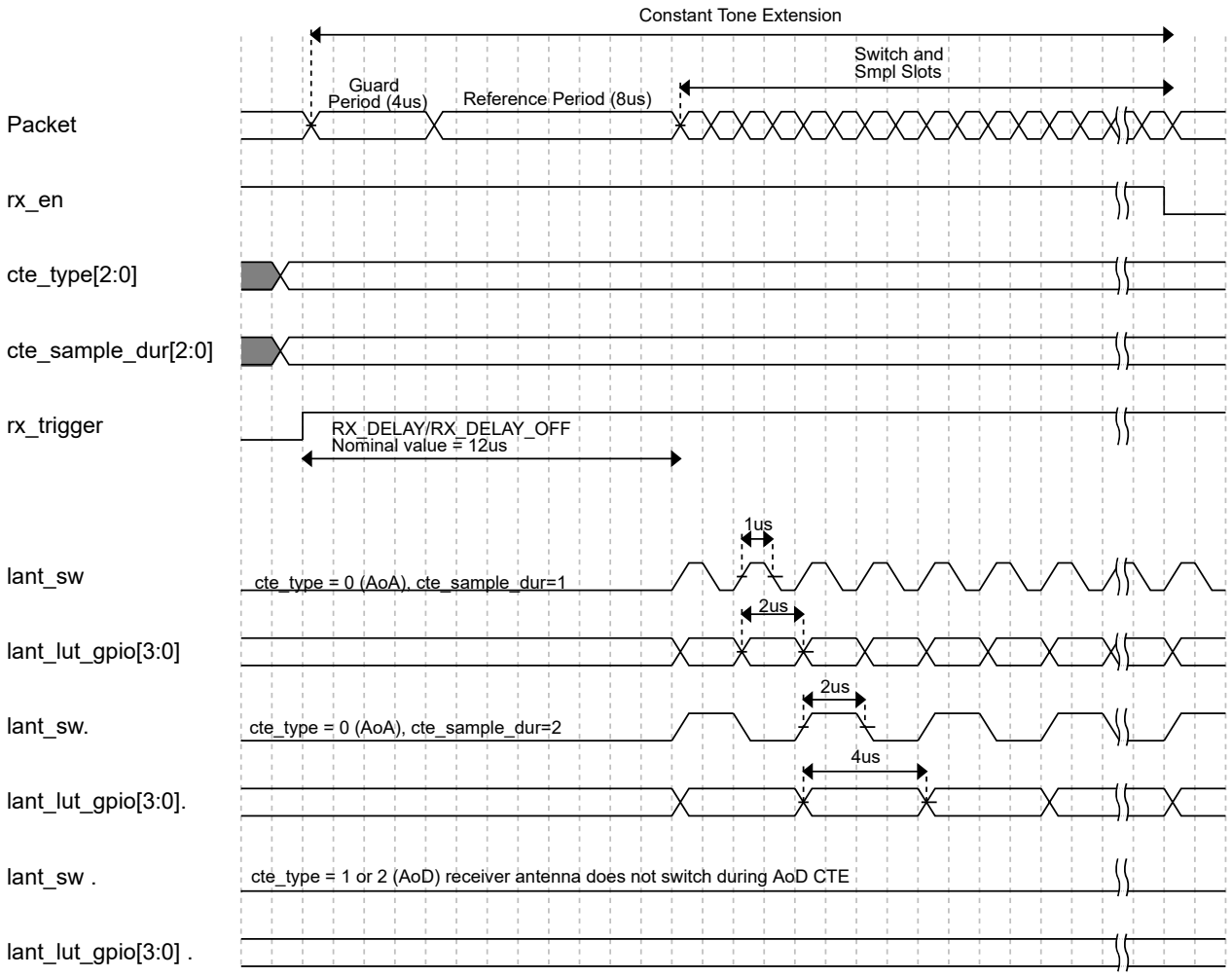


Figure 347. LCL_CTRL CTE - Receive Antenna Switching

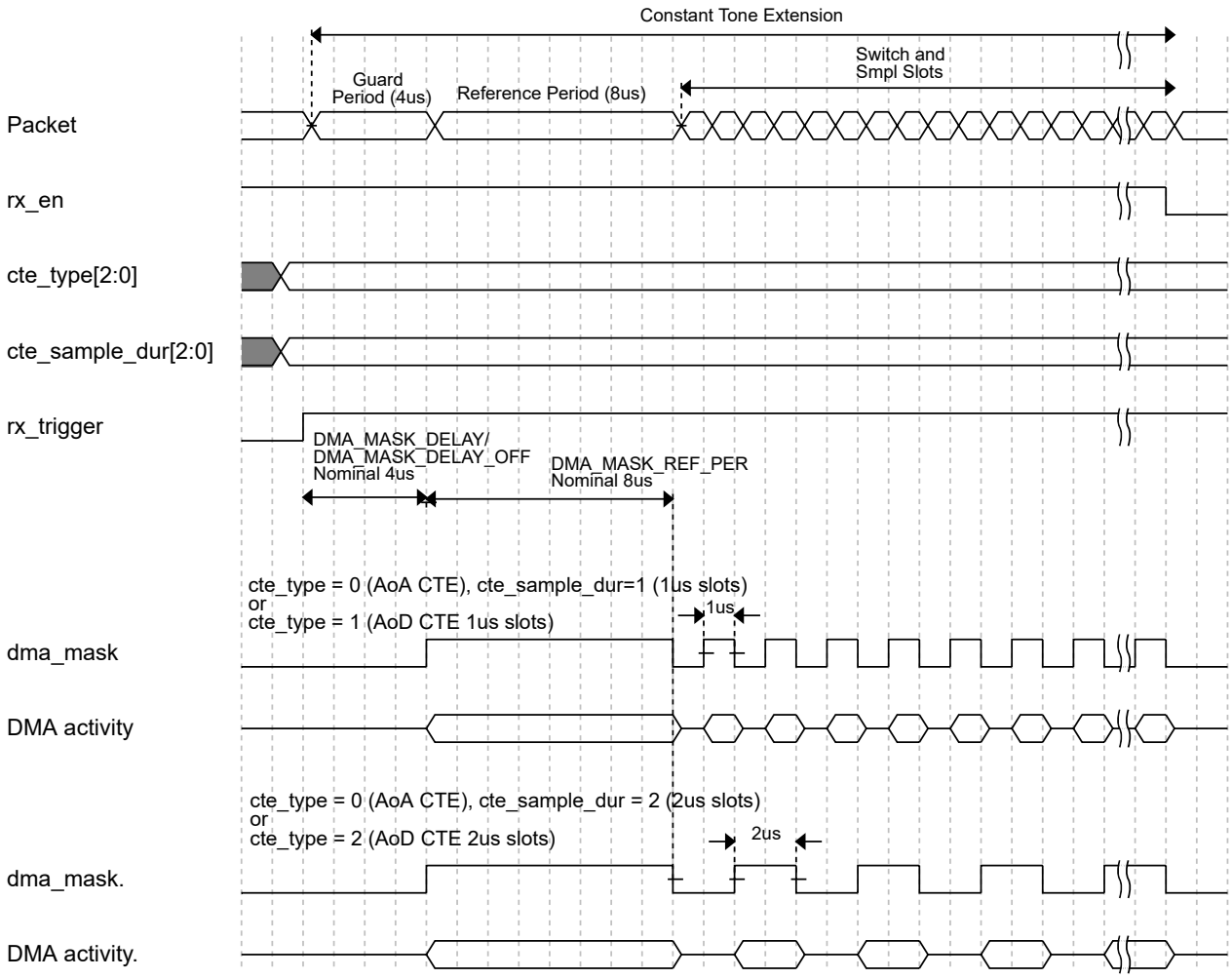


Figure 348. LCL_CTRL CTE - Receive dma_mask

55.4.7.9.3 Ranging Sequence Manager

The Ranging Sequence Manager (RSM) automates RX and TX operations across a set of frequencies in order to allow two devices to capture samples and other data for distance measurement calculations performed by software. To perform this task, the RSM acts like a small link layer, directing the transceiver to perform RX and TX as needed. The figure and list below shows the modules to which the RSM interfaces:

- To control the RSM's behavior in each frequency step, the RSM reads information stored in the Packet RAM. This consists of the desired frequency (ext_channel_num_ovrd in the figure), frequency calibration information (ext_ctune and ext_hpm_cal_factor in the figure) used by the PLL, and whether the frequency step consists of tone exchanges, packet exchanges, or both. If the ext_ctune info is not used and the PLL performs ctune calibration, then on each frequency change the ctune_best_diff value is saved to Packet RAM by the RSM.
- The RSM provides control signals to the TSM to start receive or transmit operations, to switch between receive and transmit operations, and to end receive and transmit operations.
- For transmit, the RSM uses the tx_mod_disable signal to tell the TXDIG whether to modulate data (packet transmit) or not (tone transmit). For packet transmit, the RSM reads pseudo-noise (PN) info from the Packet RAM and provides the data to transmit to the TXDIG. Note that the RSM does not control the transmit power; that should be programmed in the TXDIG

- When receiving packets, the RSM reads PN info from the Packet RAM which is used by the PHY as the access address (AA) and is also provied as "RTT bits" to the PHY. It also directs the RTT (round trip time) results from the PHY to the Packet RAM
- When receiving tones, the RSM provides a dma_trigger and a dma_mask output for use by the Transceiver DMA module. The dma_trigger asserts whenever RSM's RX_EN or TX_EN bitfields are set. The dma_mask asserts at a programmable delay (DMA_DLY or DMA_DLY0 bitfields) from the start of receive, and remains asserted for a programmable duration (DMA_DUR or DMA_DUR0 bitfields). The delay is programmed to allow the receiver analog and digital time to settle, and the duration programmed for the desired sample capture duration. The RSM also sends a signal to the PLL (ext_rx_lpm_mod_dis in the figure) which indicates that for tone reception the PLL should disable low port modulation and instead use a constant value (programmed in PLL) for high port modulation.
- For ranging operations which use antenna switching, the RSM provides triggers to the Localization Control module; lcl_rx_trigger is used in PM_RX states and lcl_tx_trigger is used in PM_TX states. Since the PM duration can take four different values (T_PM0-3), the RSM also outputs a signal (lcl_alt_reg_sel[1:0]) to the Localization Control module to indicate which T_PM duration is currently being used. Note that if the Localization Control module is used in conjunction with RSM, in PM_RX states the Localization Control module will provide the DMA mask control to the Transceiver DMA and the RSM's DMA mask feature should be disabled. Refer to Antenna Control Used With RSM for more information on LCL_CTRL usage with RSM.
- The RSM "active" output is used by RXDIG and TXDIG in order to keep the Rx/Tx clocks running while the RSM is active. This is required for timestamping continuity for RTT (i.e., the relationship of fine time-stamp to the corresponding integer reference clock timestamp is maintained) as well as tone exchange (i.e., digital downmixing of the IF frequency & CIC clocking randomness do not contribute additional phase jitter). This doesn't require any software configuration except for the RXDIG control bit CIC_CNTR_FREE_RUN_EN which resets to 1, and should remain set to ensure the clocks remain running.

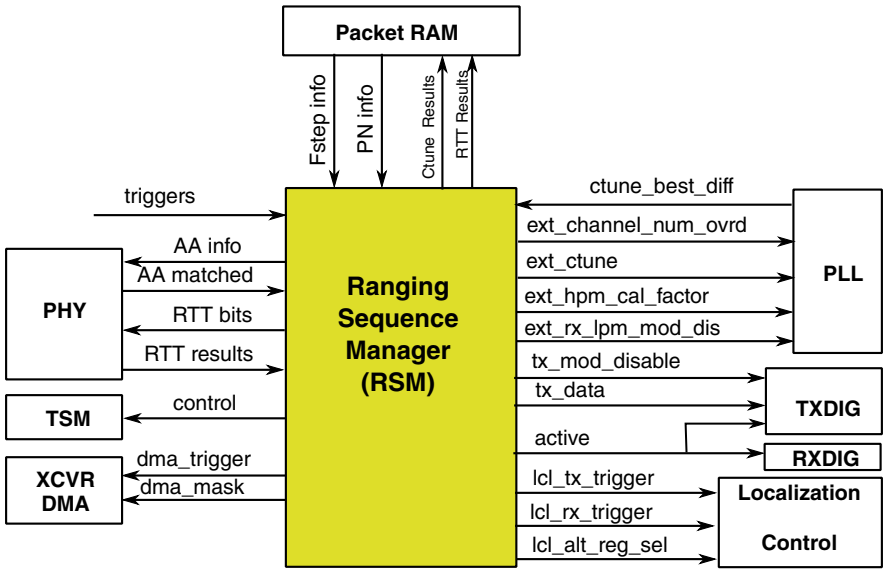


Figure 349. RSM Interfaces

The RSM supports two modes of operation: SQTE (Secure Quick Tone Exchange) and PDE (Phase-based Distance Estimation). These modes, which are described in detail in the following sections, have a good deal in common, including the following:

- The RSM can be configured to begin with a transmit operation (TX_EN bitfield programmed to 1) or receive operation (RX_EN bitfield programmed to 1). These correspond to the Initator and Reflector roles for SQTE, and to Measuring Device (MD) and Reflector Device (RD) roles for PDE, respectively.
- The number of frequency steps is configurable (FREQUENCY_STEP bitfield).
- The durations of the states associated with receive, transmit, frequency change transition and role swap transition (RX to TX or TX to RX) are configurable (T_PM, T_FC, and T_IP2 bitfields), though SQTE has additional configurability (T_IP1 and T_S bitfields).

- The format used for each frequency step is configurable, though for PDE this only includes tone exchanges, while for SQTE this includes tone and/or packet exchanges.

55.4.7.9.3.1 Secure Quick Tone Exchage (SQTE) Mode

The MODE=0 setting is for SQTE, which is shown in the figure below from the Bluetooth LE standards body. In this mode, two devices (one assuming the role of "Initiator" and the other the role of "Reflector") take turns exchanging packets and/or tones. The first set of exchanges are referred to as the Frequency Compensation Step (FCS), which consists of a bidirectional packet exchange and a unidirectional Frequency Measurement (FM) step which is a tone transfer from Reflector to Initiator. The main purpose of FCS is to allow the Initiator to determine the frequency offset of the Reflector with respect to the frequency of the Initiator.

There are a programmable number of SQTE Steps, each of which are individually configured to take one of four formats, as shown in the figure: the FCS step, a Pk-Pk step which only has packet exchange; a Tn-Tn step which only has tone exchange; or a Pk-Tn-Tn-Pk which has both packet and tone exchanges. Usually, the FCS is only used in the first step, but the RSM can be programmed so that it is also used in other steps.

Data is collected during packet reception and tone reception for each "SQTE Step" by both Initiator and Reflector for later post-processing by software. Note that the packets exchanged during SQTE are not normal Bluetooth LE packet; they are comprised of an 8bit (or 16bit for 2Mbps operation) preamble, a 32bit or 64bit pseudo-noise (PN) sequence which is known apriori by both Initiator and Reflector, and a 4bit trailer.

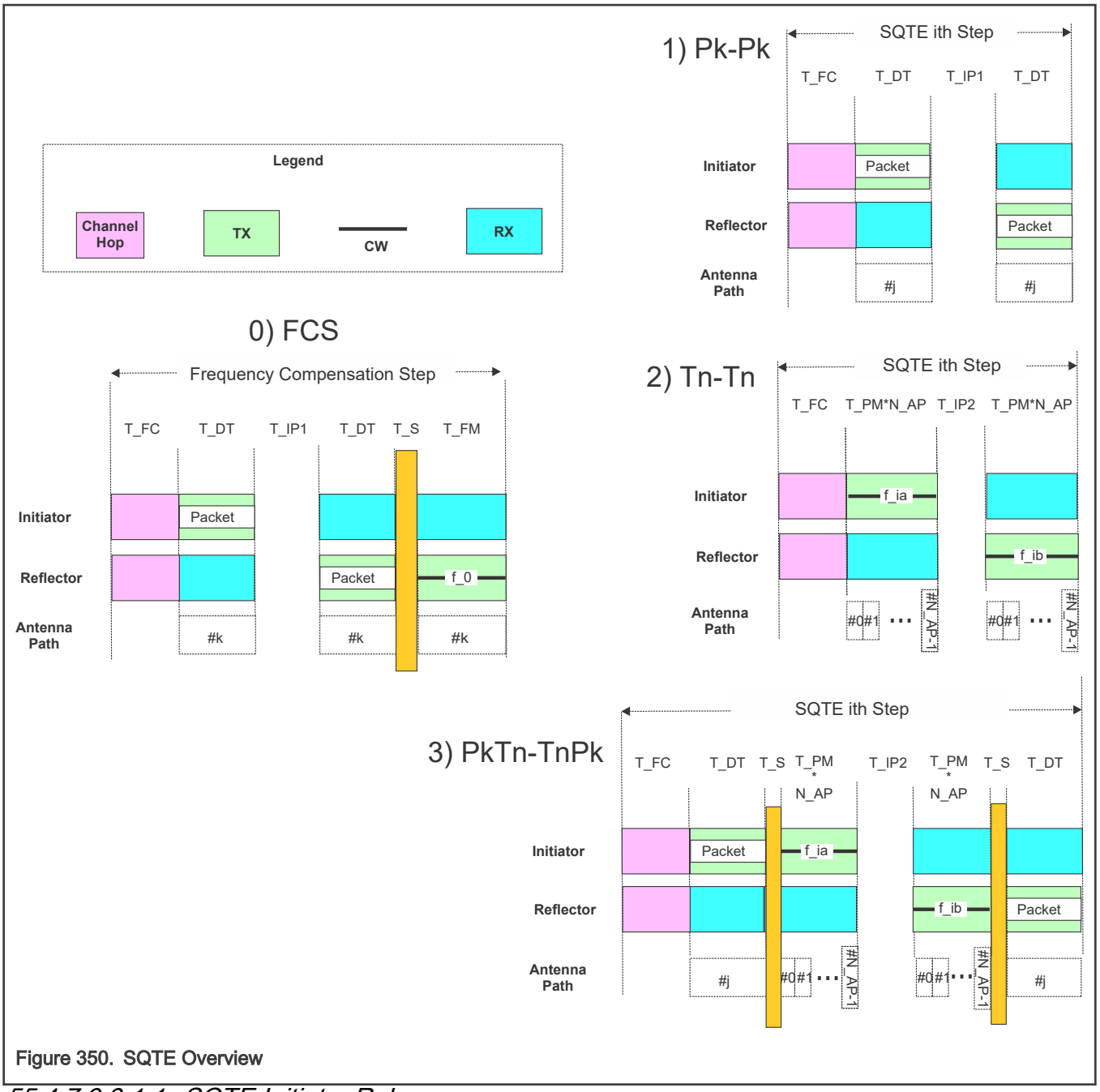


Figure 350. SQTE Overview

55.4.7.9.3.1.1 SQTE Initiator Role

The RSM state diagram below illustrates the RSM states used for SQTE in the Initiator role (TX_EN=1). Some of the states are duplicated in the figure to more clearly show the state transition sequence for the four different step format types (FCS, Pk-Pk, Tn-Tn, and PkTn-Tn-Pk). The states are described as follows:

- IDLE. This is the state the RSM is in after a reset and after a sequence has completed (or been aborted).
- DELAY. If a trigger delay is used, after the selected trigger is recognized, the RSM will stay in the DELAY state for TRIG_DLY microseconds.
- WU (Warmup). The RSM asserts its tx_en output to the TSM and then waits for the TSM's wu_complete signal before leaving this state. The figure shows that WU always transitions to the DT_TX state of the FCS step format but this assumes that the FCS step format is selected for the first frequency step (which is the expected behavior).

- FC_RX2TX (Freqency Change RX-to-TX transition). In this state the RSM directs the TSM to switch from RX to TX mode, change the radio frequency, and start a new step. The nominal duration of this state is T_FC
- DT_TX (Packet Transmit). In this state the RSM directs the transceiver to transmit a "packet" with a preamble, PN data, and a trailer. The duration of this state depends on the RATE and RTT_SEQ_LEN bitfields.
- DT_RX (Packet Receive). In this state the RSM directs the transceiver to receive a "packet" with a preamble, PN data, and a trailer. The nominal duration of this state depends on the RATE and RTT_SEQ_LEN bitfields.
- FM_RX (Tone Receive for FCS). In this state, the RSM directs the transceiver to receive a "tone". The nominal duration of this state is T_FM0 or T_FM1 depending on the t_pm_sel[0] bitfield in the frequency step data stored in Packet RAM.
- PM_RX (Tone Transmit). In this state the TSM directs the transceiver to receive a "tone". The nominal duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- PM_TX (Tone Transmit). In this state the TSM directs the transceiver to transmit a tone. The duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- IP1_TX2RX (Role swap TX-to-RX transition). In this state, the RSM directs the TSM to switch from TX to RX mode. The nominal duration of this state is T_IP1
- IP2_TX2RX (Role swap TX-to-RX transition). In this state, the RSM directs the TSM to switch from TX to RX mode. The nominal duration of this state is T_IP2
- S_TX2TX (Short TX-to-TX transition). In this state, the RSM transitions between packet and tone transmit. The duration of this state is T_S
- S_RX2RX (Short RX-to-RX transition). In this state, the RSM transitions between packet and tone receive. The nominal duration of this state is T_S
- WD (Warmdown). The RSM de-asserts its rx_en or tx_en (depending on what state it was in prior to warmdown) directing the TSM to warm down the transceiver

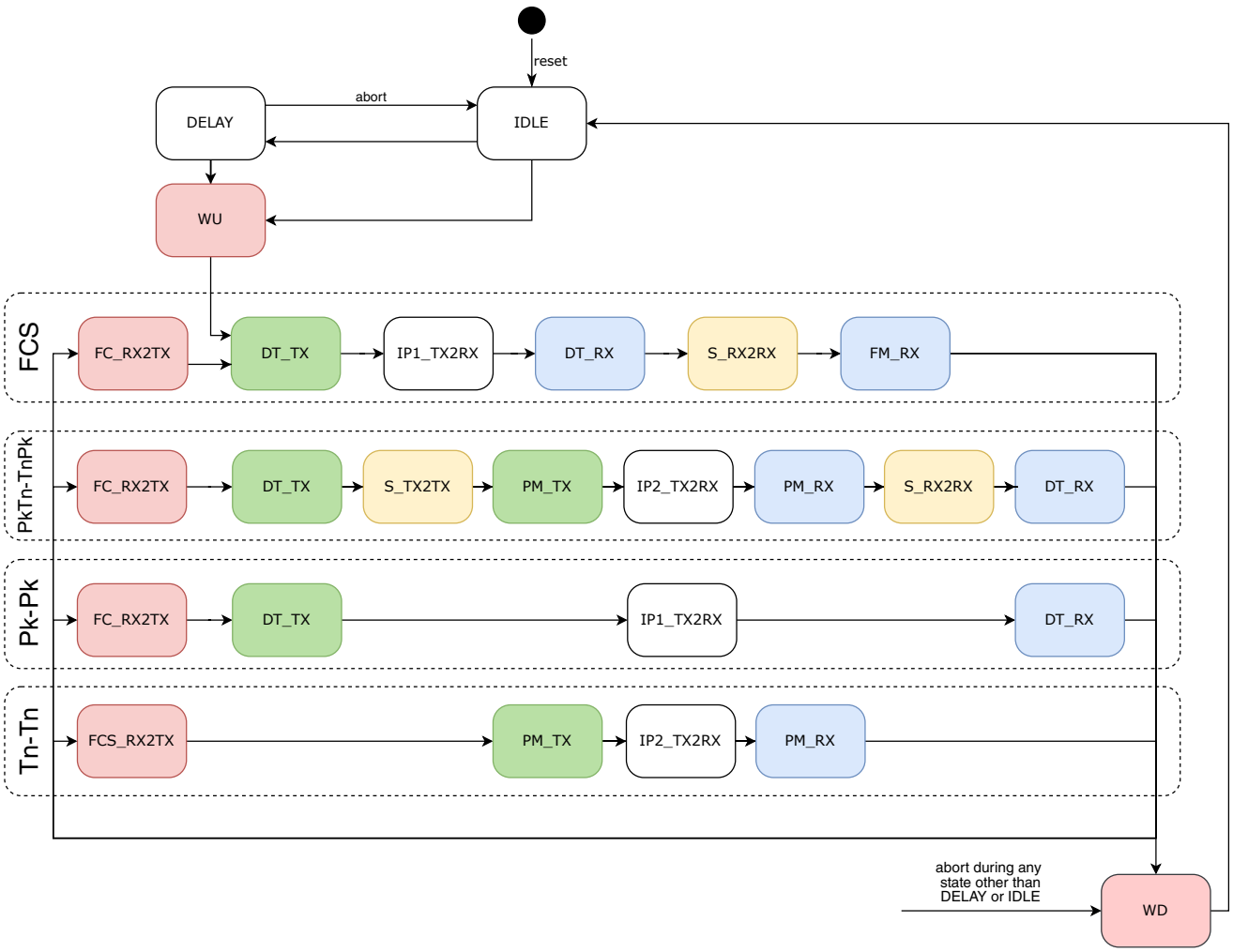


Figure 351. RSM State Transition Diagram for SQTE TX_EN=1 (Initiator role)

Any one of three methods can be used to start RSM operation when the module is configured for SQTE in the Initiator role (TX_EN=1).

1. The RSM is initiated by a hardware trigger (TRIG_SEL>0) which asserts during a previous RX or TX operation controlled by the Bluetooth or Generic LL
2. The RSM is initiated by a software trigger (TRIG_SEL=0)
3. The RSM is initiated immediately (TRIG_DLY=0) by a hardware trigger (TRIG_SEL=seq_spare3) which asserts during a dummy TX operation controlled by the Bluetooth or Generic LL

The figure below shows the timing corresponding to method #1 above. During a previous RX or TX operation controlled by the Bluetooth or Generic LL, a trigger to the is RSM asserted, and then the LL completes it RX or TX operation as usual. Meanwhile, upon recognizing the trigger, the RSM's trigger delay counter starts. After a programmable TRIG_DLY, the RSM will direct the TSM to start a transmit operation with a fast warmup if FAST_FC_TX_WU is set or a normal warmup if not. The state transitions associated with the four frequency step formats when TX_EN=1 are as descibed in the previous state transition diagram figure; note that the timing figure below only shows the Frequency Compensation Step and Pk-Tn-Tn-Pk step formats.

After the last frequency step, the RSM transitions to the WD state where the TSM is warmed down, and then to the IDLE state.

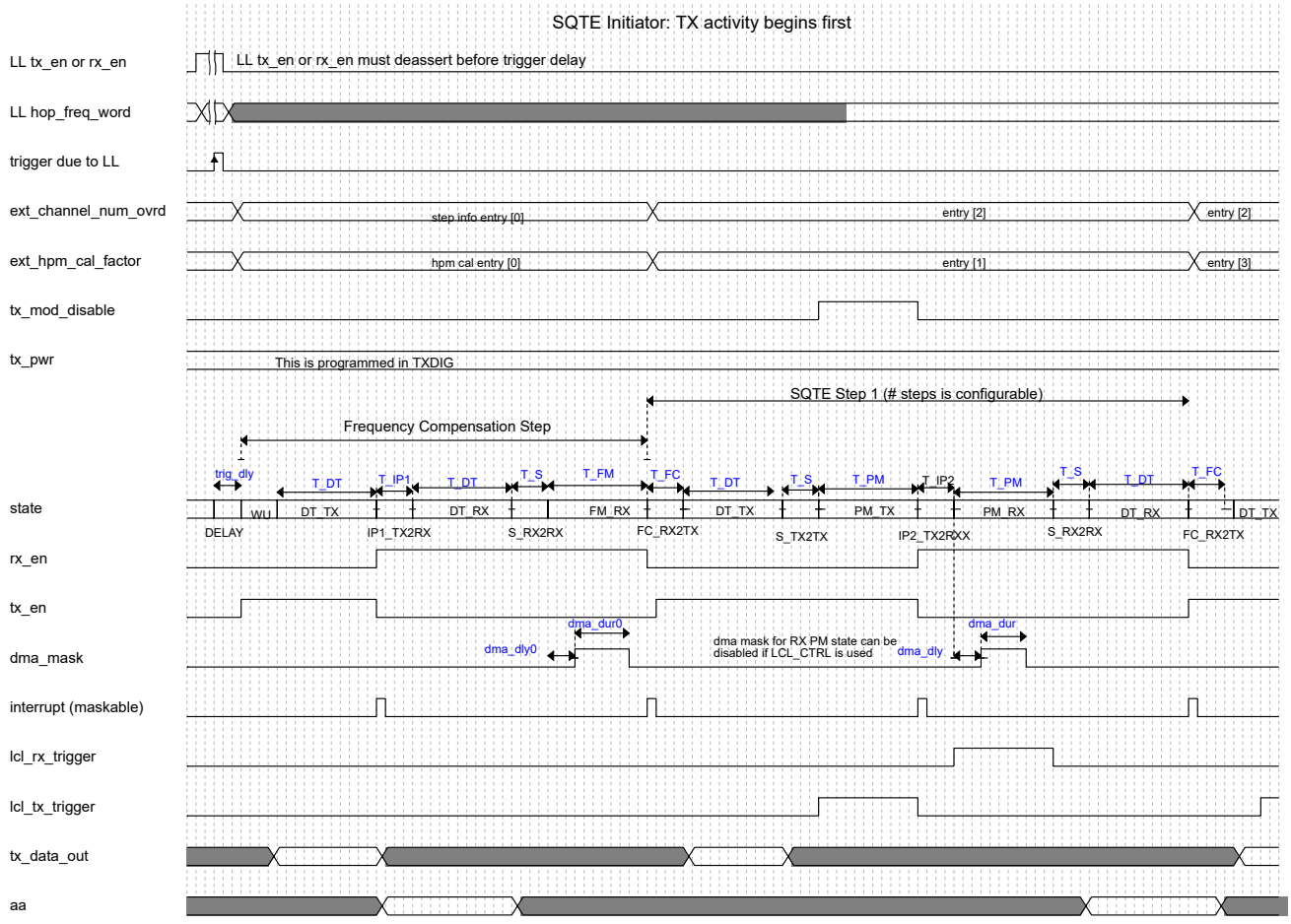


Figure 352. SQTE Mode Timing with TX_EN=1 (Initiator role)

The timing for method #2 (software trigger) is similar to above, but no link layer operation is required. Instead, if TRIG_SEL=0, then the TRIG_DLY counter begins as soon as RSM's TX_EN is programmed to 1.

The timing for method #3 (hardware trigger with dummy LL event) is also similar to above, but differs in the link layer and trigger operation. In this case, the link layer begins a "dummy" TX operation which is only used to trigger the RSM. The RSM trigger for seq_spare3 should be used and the RSM's TRIG_DLY programmed to 0. The TSM should be programmed so that seq_spare3 asserts at time 0. (Seq_spare3 can be programmed to de-assert at any time after 0.) The link layer only needs to assert its "tx_en" signal to the TSM long enough to generate the seq_spare3 trigger to the RSM; after that the link layer can "abort" its TX operation. (Any duration for link layer tx_en from 4us to ~40us should be okay.) The trigger timing for method #2 is shown in the figure below.

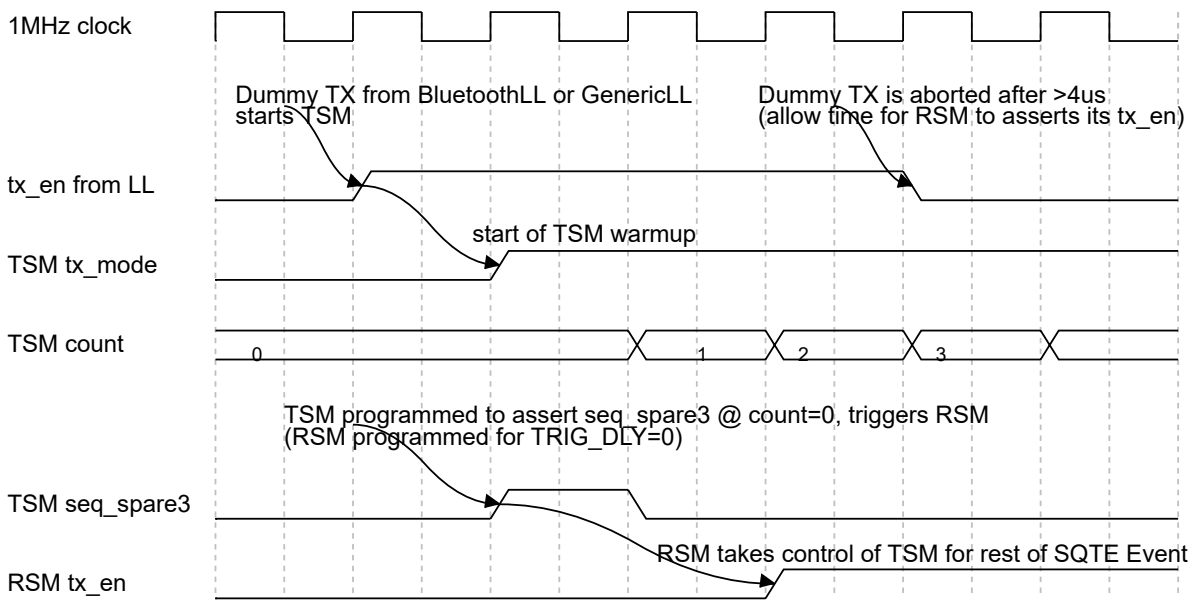


Figure 353. SQTE TX trigger method #3: Dummy LL TX Event for TX_EN=1 (Initiator role)

55.4.7.9.3.1.2 SQTE Reflector Role

The RSM state diagram below illustrates the RSM states used for SQTE in the Reflector role (RX_EN=1). Some of the states are duplicated in the figure to more clearly show the state transition sequence for the four different step format types (FCS, Pk-Pk, Tn-Tn, and PkTn-Tn-Pk). The states are described as follows:

- IDLE. This is the state the RSM is in after a reset and after a sequence has completed (or been aborted).
- DELAY. If a trigger delay is used, after the selected trigger is recognized, the RSM will stay in the DELAY state for TRIG_DLY microseconds.
- WU (Warmup). The RSM asserts its rx_en output to the TSM and then waits for the TSM's wu_complete signal before leaving this state. The figure shows that WU always transitions to the DT_RX state of the FCS step format but this assumes that the FCS step format is selected for the first frequency step (which is the expected behavior).
- FC_TX2RX (Freqency Change TX-to-RX transition). In this state the RSM directs the TSM to switch from TX to RX mode, change the radio frequency, and start a new step. The nominal duration of this state is T_FC
- DT_TX (Packet Transmit). In this state the RSM directs the transceiver to transmit a "packet" with a preamble, PN data, and a trailer. The duration of this state depends on the RATE and RTT_SEQ_LEN bitfields.
- DT_RX (Packet Receive). In this state the RSM directs the transceiver to receive a "packet" with a preamble, PN data, and a trailer. The nominal duration of this state depends on the RATE and RTT_SEQ_LEN bitfields.
- FM_TX (Tone Transmit for FCS). In this state, the RSM directs the transceiver to transmit a "tone". The duration of this state is T_FM0 or T_FM1 depending on the t_pm_sel[0] bitfield in the frequency step data stored in Packet RAM.
- PM_RX (Tone Transmit). In this state the TSM directs the transceiver to receive a "tone". The nominal duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- PM_TX (Tone Transmit). In this state the TSM directs the transceiver to transmit a tone. The duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- IP1_RX2TX (Role swap RX-to-TX transition). In this state, the RSM directs the TSM to switch from RX to TX mode. The nominal duration of this state is T_IP1

- IP2_RX2TX (Role swap RX-to-TX transition). In this state, the RSM directs the TSM to switch from RX to TX mode. The nominal duration of this state is T_IP2
- S_TX2TX (Short TX-to-TX transition). In this state, the RSM transitions between packet and tone transmit. The duration of this state is T_S
- S_RX2RX (Short RX-to-RX transition). In this state, the RSM transitions between packet and tone receive. The nominal duration of this state is T_S
- WD (Warmdown). The RSM de-asserts its rx_en or tx_en (depending on what state it was in prior to warmdown) directing the TSM to warm down the transceiver

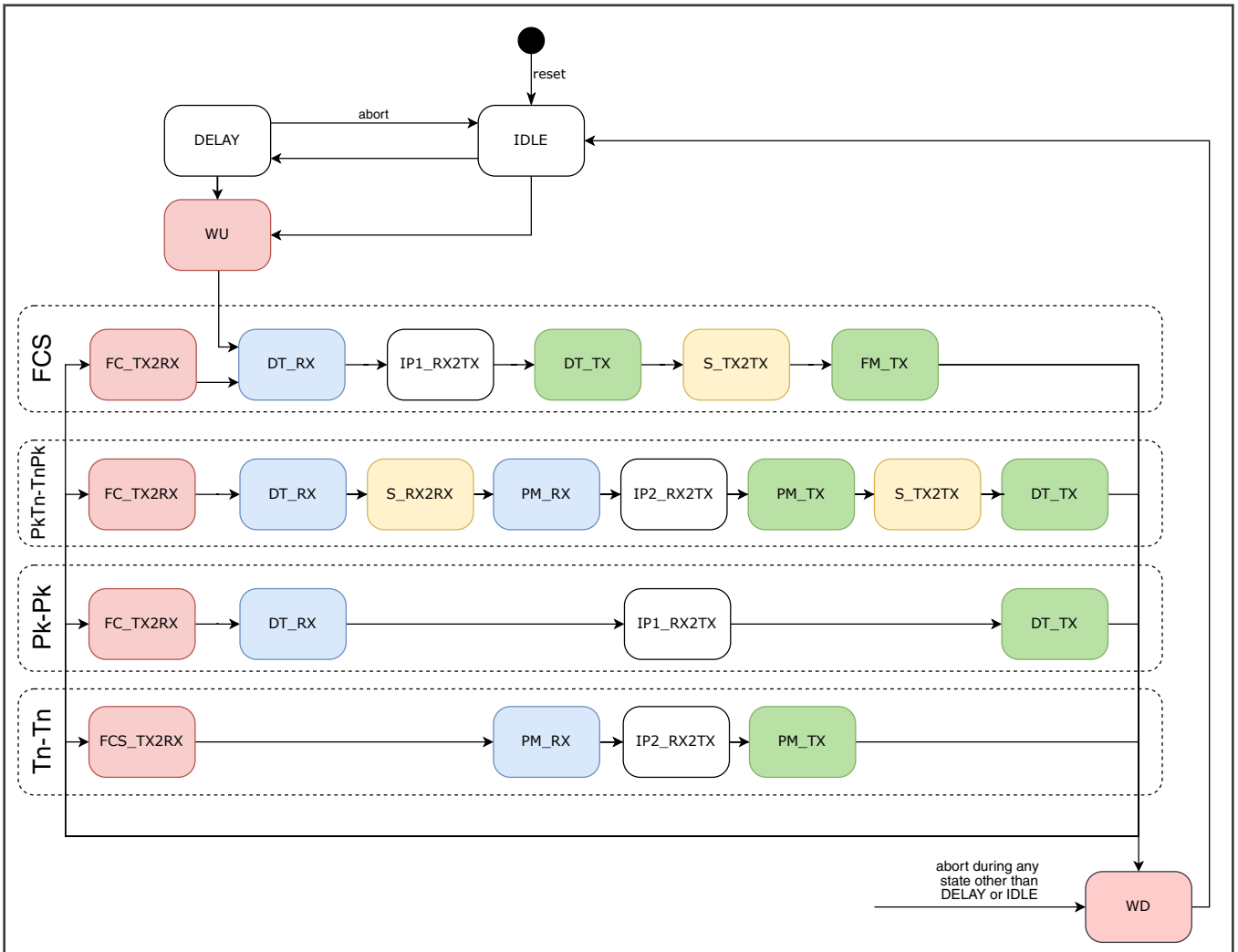


Figure 354. RSM State Transition Diagram for SQTE RX_EN=1 (Reflector role)

Any one of three methods can be used to start RSM operation when the module is configured for SQTE in the Reflector role (RX_EN=1).

1. The RSM is initiated by a hardware trigger (TRIG_SEL>0) which asserts during a previous RX or TX operation controlled by the Bluetooth or Generic LL
2. The RSM is initiated by a software trigger (TRIG_SEL=0)
3. The RSM is initiated immediately (TRIG_DLY=0) by a hardware trigger (TRIG_SEL=seq_spare3) which asserts during a dummy RX operation controlled by the Bluetooth or Generic LL

The figure below shows the timing corresponding to method #1 above. During a previous RX or TX operation controlled by the Bluetooth or Generic LL, a trigger to the is RSM asserted, and then the LL completes it RX or TX operation as usual. Meanwhile, upon recognizing the trigger, the RSM's trigger delay counter starts. After a programmable TRIG_DLY, the RSM will direct the TSM to start a receive operation with a fast warmup if FAST_FC_RX_WU is set or a normal warmup if not. Unless the DT_RX_SYNC_DIS bit is set, then during the FCS RX packet, the RSM will sync the timing using the AA matched signal from the PHY. When SQTE begins with receive, the RSM also utilizes the DMA_DLY0 and DMA_DUR0 to define when and how long dma_mask asserts during the Frequency Error Measurement (T_FM) Step. The state transitions associated with the four frequency step formats when RX_EN=1 are as descibed in the previous state transition diagram figure; note that the timing figure below only shows the Frequency Compensation Step and Pk-Tn-Tn-Pk step formats.

After the last frequency step, the RSM transitions to the WD state where the TSM is warmed down, and then to the IDLE state.

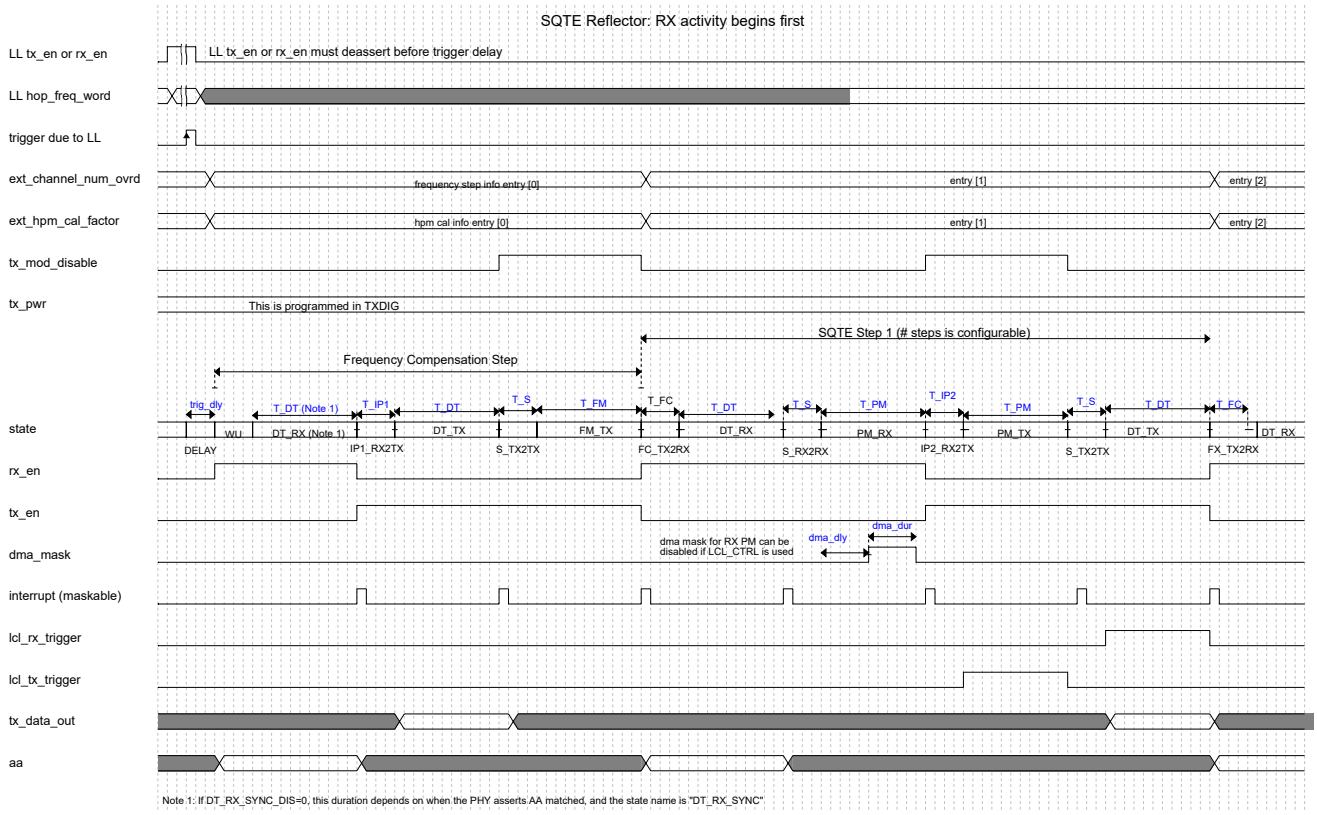


Figure 355. SQTE Timing with RX_EN=1 (Reflector role)

The timing for method #2 (software trigger) is similar to above, but no link layer operation is required. Instead, if TRIG_SEL=0, then the TRIG_DLY counter begins as soon as RSM's RX_EN is programmed to 1.

The timing for method #3 (hardware trigger with dummy LL event) is also similar to above, but differs in the link layer and trigger operation. In this case, the link layer begins a "dummy" RX operation which is only used to trigger the RSM. The RSM trigger for seq_spare3 should be used and the RSM's TRIG_DLY programmed to 0. The TSM should be programmed so that seq_spare3 asserts at time 0. (Seq_spare3 can be programmed to de-assert at any time after 0.) The link layer only needs to assert its "rx_en" signal to the TSM long enough to generate the seq_spare3 trigger to the RSM; after that the link layer can "abort" its RX operation. (Any duration for link layer rx_en from 4us to ~40us should be oky.) The trigger timing for method #2 is shown in the figure below.

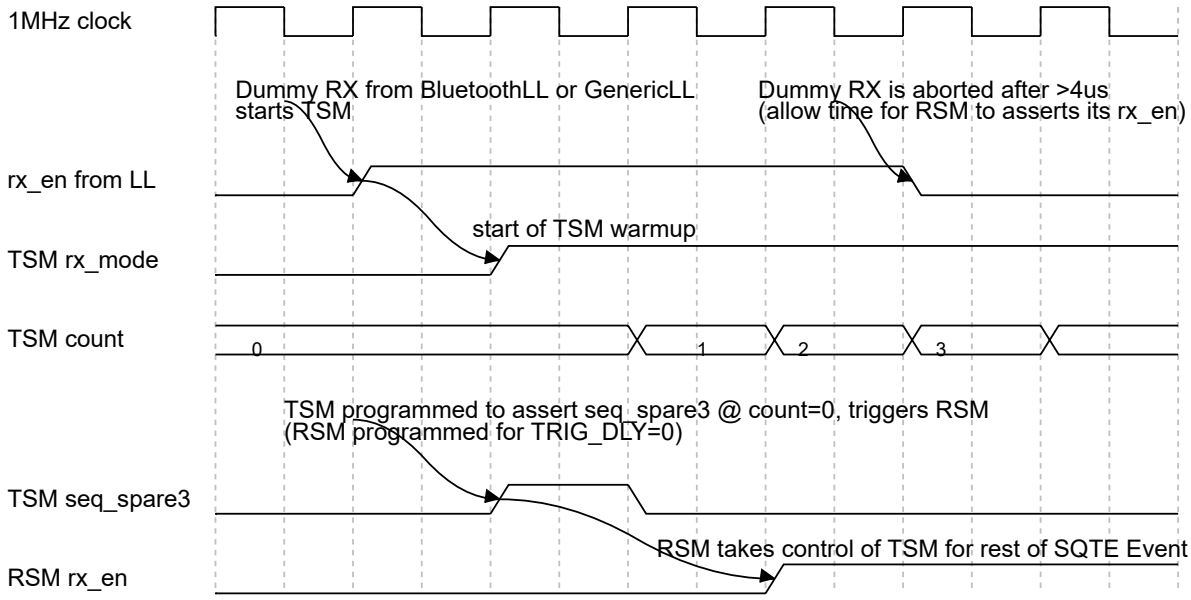


Figure 356. SQTE RX trigger method #3: Dummy LL RX Event for RX_EN=1 (Reflector role)

55.4.7.9.3.2 Phase-based Distance Estimation (PDE) Mode

The MODE=1 setting is for PDE mode. In this mode the transceiver is not warmed down from the LL receive or transmit operation prior to the RSM taking control. Instead, the the RSM is allowed to extend the receive or transmit operation initiated by the LL. For this to work correctly, the frequency used by the LL must match the frequency information in the first set of frequency step information read by the RSM from the Packet RAM (seeConfiguration Stored in Packet RAM).

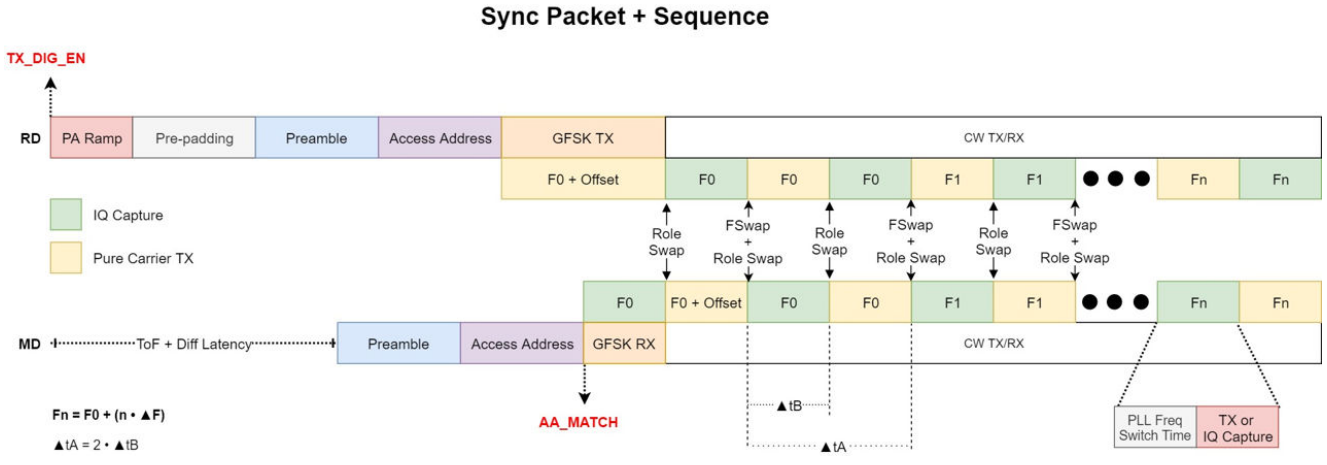


Figure 357. PDE Overview

55.4.7.9.3.2.1 PDE TX_EN=1 (RD Role)

The RSM state diagram below illustrates the RSM states used for PDE with TX_EN=1 (RD Role). Note that unlike SQTE, only one step format (Tn-Tn) is used, so the state diagram is greatly simplified. The states are described as follows:

- IDLE. This is the state the RSM is in after a reset and after a sequence has completed (or been aborted).

- EXT_TX. After the trigger is recognized, the RSM will enter and stay in this state for TRIG_DLY microseconds. During this state, the RSM will assert its tx_en output to the TSM. The link layer continues to control the transceiver at entry to this state, but is expected to de-assert its tx_en output by the time the TRIG_DLY timer expires.
- FC_RX2TX (Freqency Change RX-to-TX transition). In this state the RSM directs the TSM to switch from RX to TX mode, change the radio frequency, and start a new step. The nominal duration of this state is T_FC
- PM_RX (Tone Transmit). In this state the TSM directs the transceiver to receive a "tone". The nominal duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- PM_TX (Tone Transmit). In this state the TSM directs the transceiver to transmit a tone. The duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- IP2_TX2RX (Role swap TX-to-RX transition). In this state, the RSM directs the TSM to switch from TX to RX mode. The nominal duration of this state is T_IP2
- WD (Warmdown). The RSM de-asserts its rx_en or tx_en (depending on what state it was in prior to warmdown) directing the TSM to warm down the transceiver

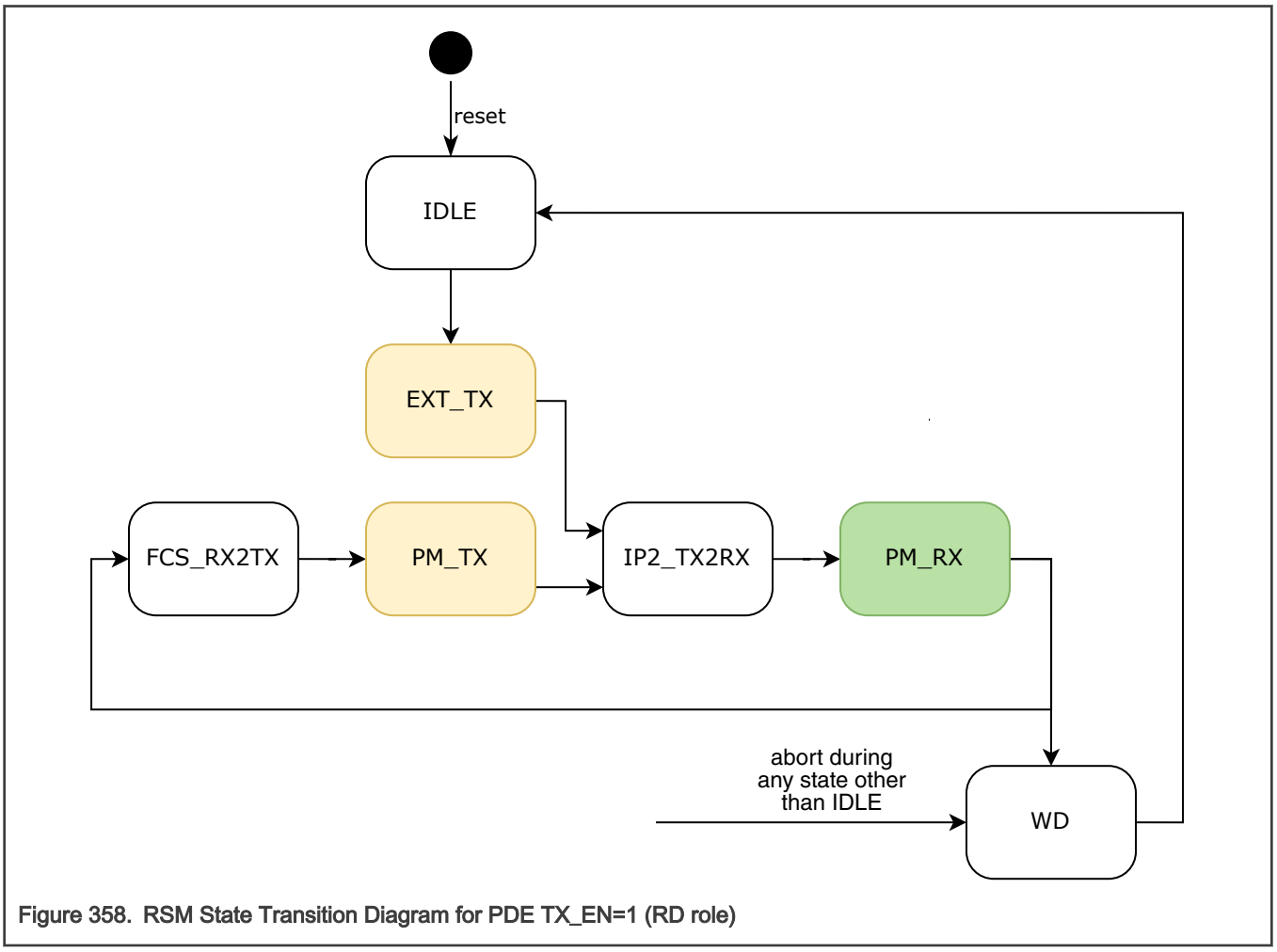


Figure 358. RSM State Transition Diagram for PDE TX_EN=1 (RD role)

The figure below shows PDE operation when the RSM is configured to begin with transmit operation. On assertion of the trigger (for example, TX_DIG_EN from the TSM), the RSM will assert its tx_en signal to keep the transmit operation active until TRIG_DLY expires. By that time, the LL's tx_en should be de-asserted so that the RSM can take full control of the transceiver. When the RSM is configured to begin with transmit, frequency steps occur on RX2TX transitions. Role swaps without frequency changes occur on TX2RX transitions.

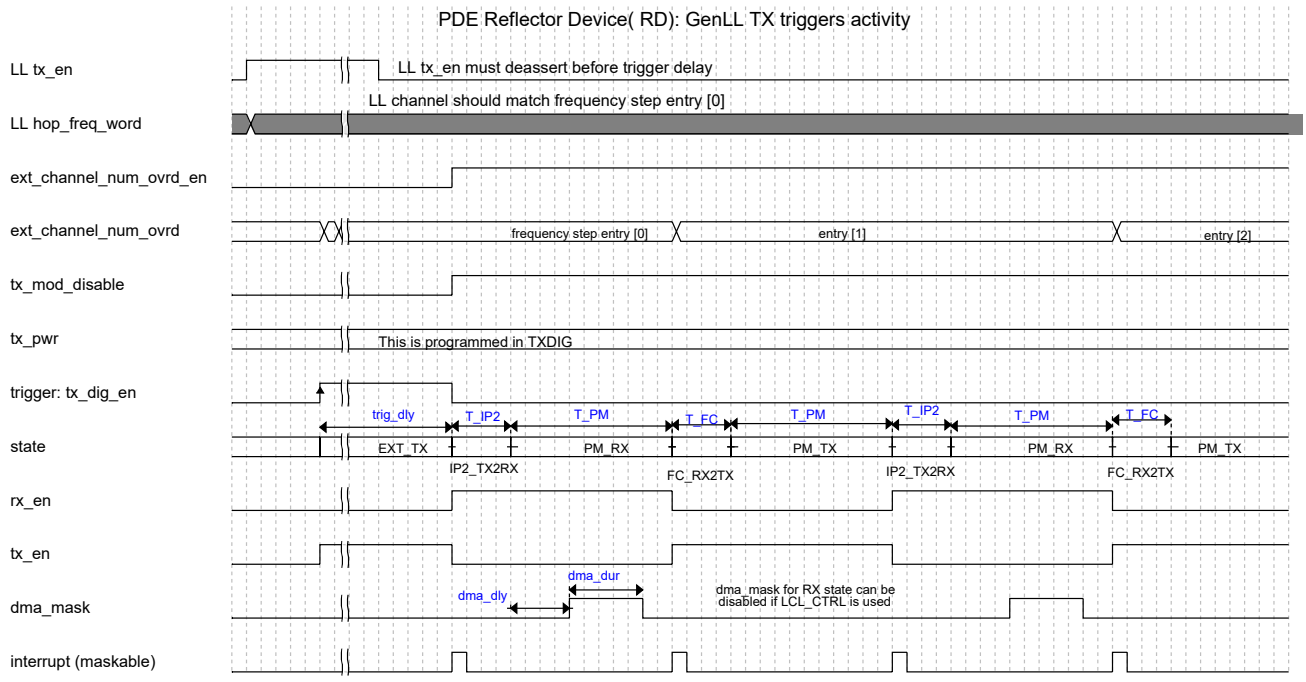


Figure 359. PDE Timing with TX_EN=1 (RD Role)

55.4.7.9.3.2.2 PDE RX_EN=1 (MD Role)

The RSM state diagram below illustrates the RSM states used for PDE with RX_EN=1 (MD Role). Note that unlike SQTE, only one step format (Tn-Tn) is used, so the state diagram is greatly simplified. The states are described as follows:

- IDLE. This is the state the RSM is in after a reset and after a sequence has completed (or been aborted).
- EXT_RX. After the trigger is recognized, the RSM will enter and stay in this state for TRIG_DLY microseconds. During this state, the RSM will assert its rx_en output to the TSM. The link layer continues to control the transceiver at entry to this state, but is expected to de-assert its rx_en output by the time the TRIG_DLY timer expires.
- FC_TX2RX (Freqency Change TX-to-RX transition). In this state the RSM directs the TSM to switch from TX to RX mode, change the radio frequency, and start a new step. The nominal duration of this state is T_FC
- PM_RX (Tone Transmit). In this state the TSM directs the transceiver to receive a "tone". The nominal duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- PM_TX (Tone Transmit). In this state the TSM directs the transceiver to transmit a tone. The duration of this state is T_PM0, T_PM1, T_PM2, or T_PM3 dependening on the t_p_sel[1:0] bitfield in the frequency step data stored in Packet RAM
- IP2_RX2TX (Role swap RX-to-TX transition). In this state, the RSM directs the TSM to switch from RX to TX mode. The nominal duration of this state is T_IP2
- WD (Warmdown). The RSM de-asserts its rx_en or tx_en (depending on what state it was in prior to warmdown) directing the TSM to warm down the transceiver

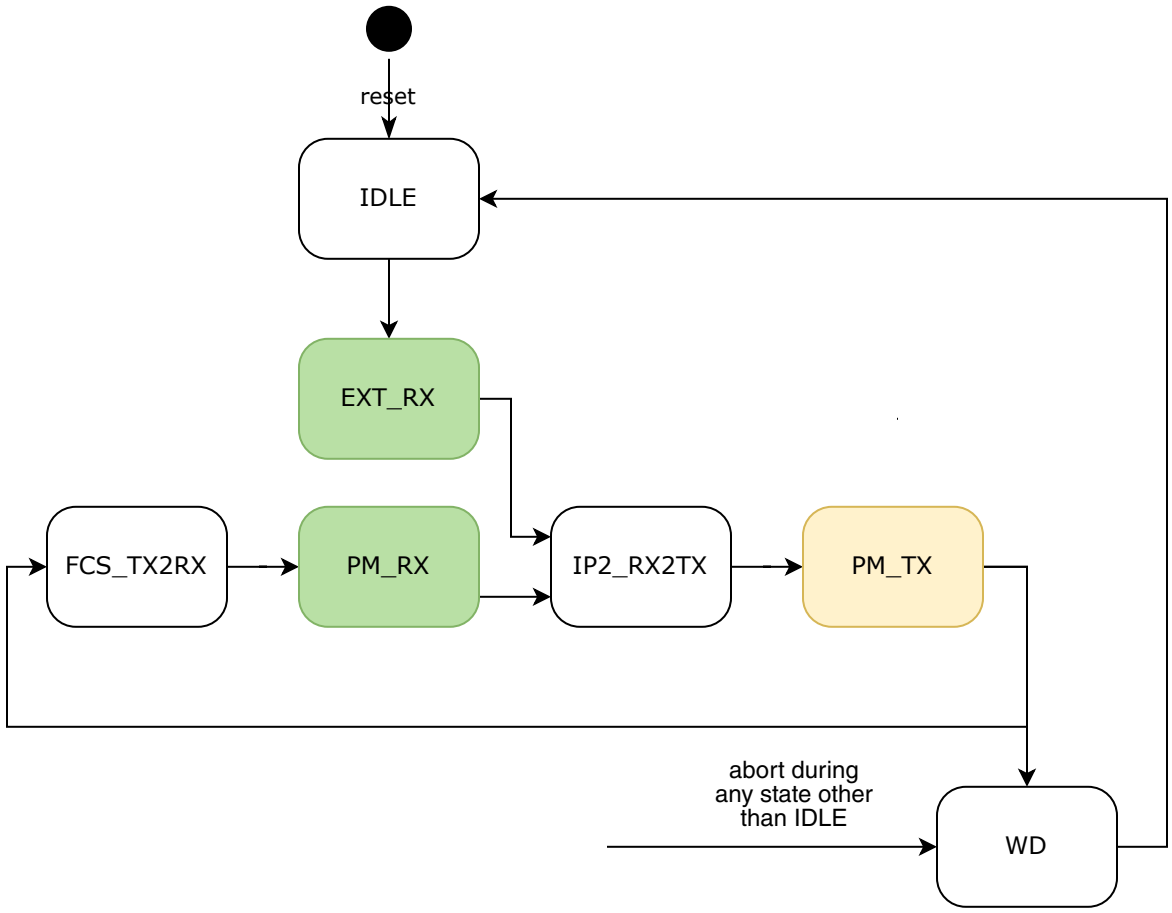


Figure 360. RSM State Transition Diagram for PDE RX_EN=1 (MD role)

The figure below show PDE operation when the RSM is configured to begin with receive operation. On assertion of the trigger (for example, crc_valid), the RSM will assert its rx_en signal to keep the receive operation active until TRIG_DLY expires. By that time, the LL's rx_en should be de-asserted so that the RSM can take full control of the transceiver. When the RSM is configured to begin with receive, frequency steps occur on TX2RX transitions; role swaps without frequency changes occur on RX2TX transitions. When PDE begins with receive, the trigger is also a timing reference for DMA_DLY0 and DMA_DUR0, which define when and how long the dma_mask will be asserted for the initial receive operation.

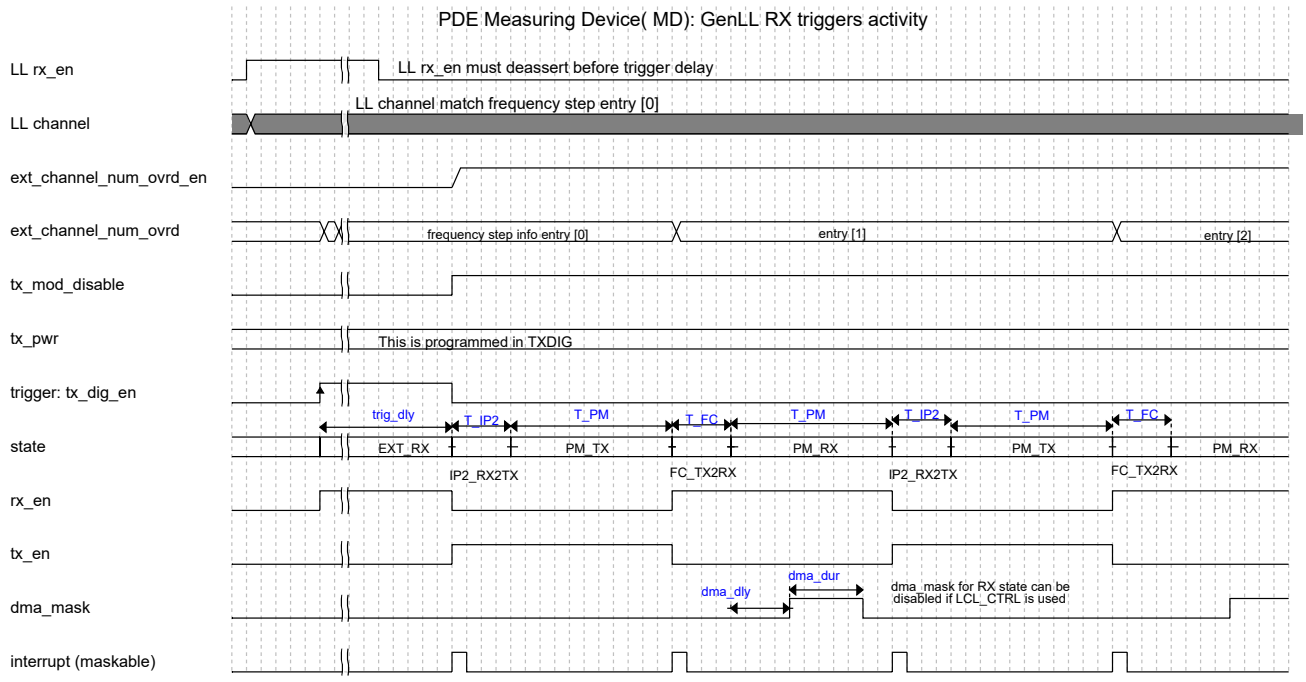


Figure 361. PDE Timing With RX_EN=1 (MD Role)

55.4.7.9.3.3 Programmable Configuration

55.4.7.9.3.3.1 Register Bitfields

The following table describes the control bits used in software configuration of the RSM for PDE and SQTE operation

Table 460. Register Bitfields

Field	Access	Description
MODE	RW	Mode of Operation- 0 - SQTE 1 - PDE
RX_EN	RW	Enables and arms the module for operation starting with Receive. This corresponds to the SQTE "Reflector" role, and the PDE "RD" role. Once set, this bit should remain set until the RSM returns to the IDLE state at the end of the sequence.
TX_EN	RW	Enables and arms the module for operation starting with Transmit. This corresponds to the SQTE "Initiator" role, and the PDE "MD" role. Once set, this bit should remain set until the RSM returns to the IDLE state at the end of the sequence.
RATE	RW	Only used for SQTE. Indicates how the transceiver is configured, and the transfer rate for packets 0 - 1Mbps 1 - 2Mbps

Table continues on the next page...

Table 460. Register Bitfields (continued)

Field	Access	Description
RTT_SEQ_LEN	RW	Only used for SQTE. NOTE that this is programmed in the 2.4GHz PHY. Indicates the length of PN sequence used in packets 0 - 32bits 1 - 64bits
STEPS[7:0]	RW	Number of frequency steps. (For SQTE this also includes the FCS.) The maximum number supported is 128, limited by RX Packet RAM used by RSM. A value of 0 should not be used
T_PM0[5:0], T_PM1[5:0], T_PM2[5:0], and T_PM3[5:0]	RW	PM (phase measurement) durations, units of 10us. 0=10us, ... 0x3F=640us. For SQTE, this is the T_PM*N_AP duration shown in the figures. The T_PM used (T_PM0, T_PM1, T_PM2, or T_PM3) depends on t_pm_sel[1:0] field (see Configuration Stored in Packet RAM)
T_FM0[4:0], T_FM1[4:0]	RW	FM (frequency measurement) duration, units of 10us. 0=10us, ... 0x1F=320us. The T_FM used (T_FM0 or T_FM1) depends on t_pm_sel[0] field (see Configuration Stored in Packet RAM)
T_FC[4:0]	RW	FC (frequency change) duration, units of 5us. 1=5us, ... 0x1F=155us. A value of 0 should not be used
T_IP1[4:0]	RW	Only applies to SQTE. IP1 (interlude period 1) duration, units of 5us. 1=5us, ..., 0x1F=155us. A value of 0 should not be used
T_S[1:0]	RW	Only applies to SQTE. S (short interlude period) duration, units of 5us. 1=5us, 2=10us, 3=15us. A value of 0 should not be used
T_IP2[4:0]	RW	IP2 (interlude period 2) duration, units of 5us. 1=5us, ..., 0x1F=155us. A value of 0 should not be used
FAST_IP_RX_WU	RW	If set, forces TSM to perform a fast RX warmup during IP1_TX2RX and IP2_TX2RX transitions
FAST_IP_TX_WU	RW	If set, forces TSM to perform a fast TX warmup during IP1_RX2TX and IP2_TX2RX transitions
FAST_FC_RX_WU	RW	If set, forces TSM to perform a fast RX warmup during FC_TX2RX transitions (and for SQTE with ipr_rx_en=1 during WU state)
FAST_FC_TX_WU	RW	If set, forces TSM to perform a fast TX warmup during FC_RX2TX transitions (and for SQTE with ipr_tx_en=1 during WU state)
RX_SETTLING_LATENCY	RW	Receiver latency. (From TSM register)
TX_DATA_FLUSH_DLY	RW	TX data flush delay / TX latency. (From TXDIG register)
AA_HAMM[2:0]	RW	Only applies to SQTE. Indicates the Access Address Hamming distance to be used by the PHY when RSM is active
CTUNE	RW	If set, RSM will send coarse tune values read from the Packet RAM to the PLL. If clear, the PLL will perform coarse tune.
HPM_CAL	RW	If set, RSM will send hpm calibration values read from the Packet RAM to the PLL. If clear, the PLL will perform hpm calibration.
TRIG_SEL[2:0]	RW	Selects which trigger to use. If 0, the module is triggered immediately.

Table continues on the next page...

Table 460. Register Bitfields (continued)

Field	Access	Description
TRIG_DLY[10:0]	RW	SQTE mode: delay from trigger to when rx_en or tx_en is asserted, in microseconds PDE mode: delay from trigger to when the first RX2TX or TX2RX transition occurs, in microseconds. For PDE mode, this should be programmed to a value greater than 0.
DMA_DLY[7:0]	RW	Delay from end of RX warmup to assertion of dma_mask signal, in microseconds
DMA_DLY0[7:0]	RW	Same as DMA_DLY[7:0] but used for the SQTE Frequency Measurement (FM) step (in FCS), and for PDE in the handoff from the LL
DMA_DUR[6:0]	RW	Duration of the dma_mask signal, in microseconds. Used in the PM_RX state. NOTE: If used in conjunction with the RXDIG's window averager, to ensure that the same number of samples are captured during each time the dma_mask asserts, the duration should be a multiple of the window averager's duration.
DMA_DUR0[6:0]	RW	Same as DMA_DUR[6:0] but used for the SQTE Frequency Error Measurement Step, and for PDE in the handoff from the LL
DMA_RX_EN	RW	Enable dma_mask to assert in PM_RX State. (Note that the DMA mask is always allowed to assert in the SQTE Frequency Measurement (FM) step in FCS.) This bit is expected to be set in SQTE events where the device does not perform antenna switching (LCL_CTRL is not used).
SW_ABORT	RW	Used to gracefully shut down the module prior to a normal end of sequence. Once set, this bit should remain set until the RSM returns to the IDLE state at the end of the sequence.
DT_RX_SYNC_DIS	RW	Only applies to SQTE, and only used by the Reflector device (RX_EN=1) . Setting this bit disables the auto-sync feature for the Reflector device in the FCS RX packet
DT_RX_SYNC_DLY[3:0]	RW	Only applies to SQTE, and only used by the Reflector device (RX_EN=1) when DT_RX_SYNC_DIS=0. In this configuration the RSM uses the PHY's AA_MATCHED output to sync the Reflector device to the Initiator device's timing. When RTT_SEQ_LEN=0, this corresponds to the delay in microseconds from the PHY's AA_MATCHED to the end of DT_RX duration (the packet duration including preamble and trailer, plus RX_SETTLING_DELAY). The value used for RTT_SEQ_LEN=0 can also be used when RTT_SEQ_LEN=1, though in the latter configuration the RSM internally offsets this value to compensate for the longer packets (since AA_MATCHED uses only the first 32bits of the PN sequence regardless of the PN sequence length).
RX_PHY_EN_MASK_DIS	RW	Only applies to SQTE. If set, the rx_phy_en_mask output is always asserted
SEQ_RCCAL_PUP_MASK_DIS	RW	If set, TSM signal seq_rccal_pup is not modified by RSM. If clear, seq_rccal_pup is masked to 0 for all RSM states except IDLE, WU, and FC
RX_SIGNALS_MASK_DIS	RW	If set, TSM signals dcoc_gain_cfg_en, dcoc_cal_en, seq_bg_pup_ibg_rx, seq_cbpf_en_dcoc, and rx_gang_pup are not modified by RSM. If clear, these signals are masked to 0 during RSM's IP1/IP2_RX2TX states
IRQ_IP1_EN	RW	If set, requests an interrupt at the beginning of the IP1_RX2TX or IP1_TX2RX state
IRQ_IP1	W1C	The bit is set on entry to IP1_RX2TX or IP1_TX2RX states. Write 1 to clear the bit[1].

Table continues on the next page...

Table 460. Register Bitfields (continued)

Field	Access	Description
IRQ_IP2_EN	RW	If set, requests an interrupt at the beginning of the IP2_RX2TX or IP2_TX2RX state
IRQ_IP2	W1C	The bit is set on entry to IP2_TX2RX or IP2_RX2TX states. Write 1 to clear the bit[1].
IRQ_FC_EN	RW	If set, requests an interrupt at the beginning of the FC_RX2TX or FC_TX2RX state
IRQ_FC	W1C	The bit is set on entry to FC_RX2TX or FC_TX2RX states. Write 1 to clear the bit[1].
IRQ_EOS_EN	RW	If set, requests an interrupt at the end of the sequence
IRQ_EOS	W1C	The bit is set to indicate that the sequence is complete. It sets when the RSM transitions from the WD to the IDLE state. Write 1 to clear the bit[1].
IRQ_ABORT_EN	RW	If set, requests an interrupt if the sequence is interrupted due to a PLL abort
IRQ_ABORT	W1C	The bit is set when the sequence is aborted due to pll_abort. This condition can occur during any RSM state except for the IDLE or WD states. Write 1 to clear the bit[1].

1. To clear RSM's W1C bits, RX_EN or TX_EN must be set. It is therefore recommended that software ensures that all the RSM W1C bits are cleared at the end of each RSM sequence prior to clearing RX_EN or TX_EN to disable the module.

55.4.7.9.3.3.2 Configuration Stored in Packet RAM

In addition to the configuration information stored in registers as described in the previous section, the RSM also relies on configuration information stored in the Packet RAM to control the behavior in each individual frequency step. There are 2 separate areas of the Packet RAM used for this configuration information.

First, there is information which is needed for every frequency step. This requires 5bytes per frequency step. It is located in the RX Packet RAM beginning at offset RSM_FSTEP_RAM, and is allocated 640bytes, which limits the maximum STEPS to 128. The frequency step information consists of the following bitfields:

- ext_channel_num_ovrd[15:0]. This defines the frequency to be used for this step. This is decoded by the PLL using the format defined in the PLL's CHAN_MAP[HOP_TBL_CFG_OVRD] bitfield. NOTE: to ensure proper PLL operation when using the RSM, the PLL's CHAN_MAP[CHANNEL_NUM_OVRD] bitfield should be programmed to 0.
- ext_ctune[7:0]. This defines the PLL coarse tune value for this frequency step.
- ext_hpm_cal_factor[12:1]. This defines the 12 MSBs of the PLL high port modulation calibration factor. The LSB, ext_hpm_cal_factor[0], is assigned to 0. Note that this has the same format as PLL's HPM_CAL_FACTOR_MANUAL bitfield
- step_format[1:0]. This 2bit field defines the step format. The values of this are as follows:
  - 00 - Frequency Compensation Step (FCS) format
  - 01 - Pk-Pk format
  - 10 - Tn-Tn format
  - 11 - Pk-Tn-Tn-Pk format.
- t_pm_sel[1:0]. For Tn-Tn and PkTn-Tn-Pk formats, this selects whether T_PM0, T_PM1, T_PM2, or T_PM3 are used for the tone duration in this frequency step : 2'b00 is associated with T_PM0, 2'b01 with T_PM1, 2'b10 with T_PM2, and 2'b11 with T_PM3. For FCS format, t_pm_sel[0] selects whether T_FM0 or T_FM1 is used for this frequency step.

Allowing the coarse tune value and hpm calibration factor for each step to be read from RAM reduces the T_FC duration since the PLL does not have to perform these calibration. Note that even if RSM's CTUNE and HPM_CAL bits are cleared (indicating PLL will perform these calibrations), the step format still consists of 5 bytes.

Table 461. Frequency Step Configuration

Frequency Step	Address Offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	0	ext_channel_num_ovrd[7:0]							
	1	ext_channel_num_ovrd[15:8]							
	2	ext_ctune[7:0]							
	3	ext_hpm_cal_factor[8:1]							
	4	t_pm_sel[1:0]		step_format[1:0]		ext_hpm_cal_factor[12:9]			
1	5	ext_channel_num_ovrd[7:0]							
	6	ext_channel_num_ovrd[15:8]							
	7	ext_ctune[7:0]							
	8	ext_hpm_cal_factor[8:1]							
	9	t_pm_sel[1:0]		step_format[1:0]		ext_hpm_cal_factor[12:9]			
...	...	...							
n	5n	ext_channel_num_ovrd[7:0]							
	5n+1	ext_channel_num_ovrd[15:8]							
	5n+2	ext_ctune[7:0]							
	5n+3	ext_hpm_cal_factor[8:1]							
	5n+4	t_pm_sel[1:0]		step_format[1:0]		ext_hpm_cal_factor[12:9]			

Second, for SQTE steps which have packet exchanges, there is additional information regarding the packet contents. This consists of either 32bits or 64bits, depending on the configured size (RTT_SEQ_LEN) . It is located in the RX Packet RAM beginning at offset RSM_FN_RAM, and is allocated 848bytes, which limits the maximum number of steps with packet exchanges to 106 (32bit PN sequences) or 53 (64bit PN sequences). Each packet exchange consists of the Initiator sending a packet to the Reflector, and the Reflector sending a packet to the Initiator. So, the first word or words stored in Packet RAM are for the Initiator-to-Reflector packet, and the next word or words are for the Reflector-to-Initiator packet. The packet format also includes a 8bit (or 16bit in the case of 2Mbps) preamble and a 4bit trailer. The bit sequence in the preamble and trailer is not read from the Packet RAM; it is determined by hardware based on the PN sequence.

Table 462. PN Configuration for 32bit packets

FCS, Pk-Pk or Pk-Tn-Tn-Pk Step	Address Offset	bits [31:0]
0	0	pn[31:0] for Initiator-to-Reflector packet
	4	pn[31:0] for Reflector-to-Initiator packet
1	8	pn[31:0] for Initiator-to-Reflector packet
	12	pn[31:0] for Reflector-to-Initiator packet
...	...	...
n	8n	pn[31:0] for Initiator-to-Reflector packet
	8n+4	pn[31:0] for Reflector-to-Initiator packet

Table 463. PN Configuration for 64bit packets

FCS, Pk-Pk or Pk-Tn-Tn-Pk Step	Address Offset	bits [31:0]
0	0	pn[31:0] for Initiator-to-Reflector packet
	4	pn[63:32] for Initiator-to-Reflector packet
	8	pn[31:0] for Reflector-to-Initiator packet
	12	pn[63:32] for Reflector-to-Initiator packet
1	16	pn[31:0] for Initiator-to-Reflector packet
	20	pn[63:32] for Initiator-to-Reflector packet
	24	pn[31:0] for Reflector-to-Initiator packet
	28	pn[63:32] for Reflector-to-Initiator packet
...	...	...
n	16n	pn[31:0] for Initiator-to-Reflector packet
	16n+4	pn[63:32] for Initiator-to-Reflector packet
	16n+8	pn[31:0] for Reflector-to-Initiator packet
	16n+12	pn[63:32] for Reflector-to-Initiator packet

55.4.7.9.3.4 Results written to Packet RAM

In the configuration where the PLL performs coarse tune during the RSM operation (CTUNE=0), the RSM will capture the coarse tune best diff value (one byte) and save this to Packet RAM. This will occur on every frequency change operation, and in the case of SQTE during the initial WU state. These results are located in the RX Packet RAM beginning at offset RSM_PCBD_RAM, and is allocated 128bytes, which limits the maximum STEPS to 128.

Table 464. CTUNE Results

Frequency Step	Address Offset	bits [7:0]
0	0	ctune_best_diff[7:0]
1	1	ctune_best_diff[7:0]
...	...	...
n	n	ctune_best_diff[7:0]

For SQTE, during every packet reception, the RSM will save the RTT (round trip time) info from the PHY to Packet RAM. This is a four byte value, which includes 31bits from the PHY (rtt_data[30:0] = {rtt_p_delta[9:0], rtt_ham_dist_sat[1:0], rtt_int_adj[1:0], ha_rtt_cfo[15:0], ha_rtt_found}) and 1bit (rtt_vld) which is 1 if the PHY indicates that the rtt_data[30:0] is valid; otherwise it is 0. These results are saved in the RX Packet RAM beginning at offset RSM_RTT_RAM, and is allocated 424bytes, which limits the maximum number of packet exchanges to 106. Refer to the PHY chapter for more information on rtt_p_delta[9:0], rtt_ham_dist_sat[1:0], rtt_int_adj[1:0], ha_rtt_cfo[15:0], and ha_rtt_found.

Table 465. RTT Results

FCS, Pk-Pk or Pk-Tn-Tn-Pk Step	Address Offset	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	0	rtt_cfo[5:0]						rtt_found	rtt_vld
	1	rtt_cfo[13:6]							
	2	rtt_p_delta[1:0]		rtt_ham_dist_sat[1:0]		rtt_int_adj[1:0]		rtt_cfo[15:14]	
	3	rtt_p_delta[9:2]							
1	4	rtt_cfo[5:0]						rtt_found	rtt_vld
	5	rtt_cfo[13:6]							
	6	rtt_p_delta[1:0]		rtt_ham_dist_sat[1:0]		rtt_int_adj[1:0]		rtt_cfo[15:14]	
	7	rtt_p_delta[9:2]							
...	...	...							
n	4n	rtt_cfo[5:0]						rtt_found	rtt_vld
	4n+1	rtt_cfo[13:6]							
	4n+2	rtt_p_delta[1:0]		rtt_ham_dist_sat[1:0]		rtt_int_adj[1:0]		rtt_cfo[15:14]	
	4n+3	rtt_p_delta[9:2]							

55.4.7.9.3.5 Role Swap and Frequency Change Transitions

This section provides information on the role swap (changing from receive to transmit or vice versa, from receive packet to receive tone or vice versa, or from transmit packet to transmit tone or vice versa) and frequency change transitions (with or without role swap) and their timings in more detail. These transitions are all intended to be handled by the RSM logic without software involvement; however, as a fallback, it is also possible to trigger interrupts at the beginning of transition states so that software can perform any additional tasks. Note that fewer transitions are supported for PDE than for SQTE, and that the transitions used depends also on the role (whether RX_EN or TX_EN bit is set).

55.4.7.9.3.5.1 TSM Features Utilized

The RSM utilizes several TSM features to control these transitions and to minimize the transition durations: fast RX2TX and fast TX2RX; recycle events; and fast RX and TX warmups. RSM usage of these features is described in the next three paragraphs.

First, for role swap the RSM always directs the Transceiver Sequence Manager (TSM) to perform fast RX2TX and fast TX2RX transitions. In these transitions, the TSM jumps from a recieve (transmit) sequence already warmed up to a programmable starting point in the transmit (receive) sequence. The starting points are programmed in the TSM: FAST_TX2RX_START for IP1_TX2RX or IP2_TX2RX, FAST_TX2RX_START_FC for FC_TX2RX, FAST_RX2TX_START for IP1_RX2TX or IP2_RX2TX, and FAST_RX2TX_START_FC for FC_RX2TX. For the FC_RX2TX and FC_TX2RX, the start value is expected to be programmed such that the TSM's pll_dig_en is de-asserted for 1us and then re-asserted, coincident with new frequency information provided to the PLL from the RSM; for the other transitions a frequency change is not performed, so the start time value should be such that the TSM's pll_dig_en does not de-assert.

Second, for S_RX2RX transitions (which are only used in SQTE), the RSM can direct the TSM using a "recycle0" event. In these transitions, the TSM jumps from a receive sequence already warmed up to a programmable starting point in the receive sequence. The starting count is programmed in the TSM's RECYCLE_COUNT0 bitfield such that rx_dig_en cycles and rx_init pulses to reset the receiver. (Note that S_TX2TX does not require a recycle event; the transmitter is not reset during this transition)

Third, to further minimize the duration of the RX2TX and TX2RX transitions, the RSM can direct the TSM to perform a fast TX or fast RX warmup. Fast RX or TX warmups allow the TSM to skip some TSM counts in the middle of the TSM warmup sequence (after the RX2TX and TX2RX start times described above). For example, during a fast RX warmup the DCOC calibration

time can be skipped. The skip times are programmed in the TSM: FAST_TX_START and FAST_TX_DEST for transmit, and FAST_RX_START and FAST_RX_DEST for receive. The use of fast RX or TX warmups during RSM transitions is an optional feature. It is enabled through the RSM's FAST_IP_TX_WU (for IP1_RX2TX and IP2_RX2TX transitions), FAST_IP_RX_WU (for IP1_TX2RX and IP2_TX2RX transitions), FAST_FC_RX_WU (for FC_TX2RX transitions), and FAST_FC_TX_WU (for FC_RX2TX transitions) bitfields.

In below discussion of the transition timings, the following assumptions are made concerning the configuration of the TXDIG ramp:

- TXDIG is programmed for 1us ramp (PA_RAMP_SEL=2'b01).

NOTE

The 1us ramp setting is a requirement for proper operation of the RSM with the TXDIG

- For PA ramp-up: PAD_DLY (delay from tx_dig_en to PA enable) is assumed to be programmed to 0

The RX2TX transition states (IP1_RX2TX, IP2_RX2TX, and FC_RX2TX) are described as follows:

- The RSM signals to the TSM to initiate a fast RX2TX transition. In response the TSM will change from rx_mode to tx_mode and change its count to FAST_RX2TX_START (IP1_RX2TX, IP2_RX2TX) or FAST_RX2TX_START_FC (FC_RX2TX). FAST_RX2TX_START and FAST_RX2TX_START_FC are programmed in the TSM
- If RSM's FAST_IP_TX_WU bit is set (IP1_RX2TX, IP2_RX2TX) or FAST_FC_TX_WU bit is set (FC_RX2TX), the RSM will also signal to the TSM to perform a fast warmup during the TX warmup. As the TSM count increments it will skip from FAST_TX_START to FAST_TX_DEST (which are programmed in the TSM)
- The TSM reaches the end of warmup when its count reaches END_OF_TX_WU and after the TXDIG has asserted the pa_wu_complete signal.

The time needed for the RX2TX transitions are shown in the table below, where the term "pa_ru" accounts for any additional time after END_OF_TX_WU before the PA ramp-up completes:

- if (RAMP_UP_DLY + 2us >= END_OF_TX_WU - TX_DIG_EN_TX_HI) pa_ru = (RAMP_UP_DLY + 2us) - (END_OF_TX_WU - TX_DIG_EN_TX_HI)
- else pa_ru = 0us

Transition	Time needed for transition	Roles
IP1_RX2TX	(END_OF_TX_WU - FAST_RX2TX_START) - (FAST_DEST_TX - FAST_START_TX)*FAST_IP_TX_WU + pa_ru	SQTE Reflector
IP2_RX2TX	(END_OF_TX_WU - FAST_RX2TX_START) - (FAST_DEST_TX - FAST_START_TX)*FAST_IP_TX_WU + pa_ru	SQTE Reflector, PDE MD
FC_RX2TX	(END_OF_TX_WU - FAST_RX2TX_START_FC) - (FAST_DEST_TX - FAST_START_TX_FC)*FAST_FC_TX_WU + pa_ru	SQTE Initiator, PDE RD

The TX2RX transition states (IP1_TX2RX, IP2_TX2RX, and FC_TX2RX) are described as follows:

- The RSM directs the TXDIG to ramp-down the PA
- The RSM signals to the TSM to initiate a fast TX2RX transition. In response the TSM will change from tx_mode to rx_mode and change its count to FAST_TX2RX_START (IP1_TX2RX, IP2_TX2RX) or FAST_TX2RX_START_FC (FC_TX2RX). FAST_TX2RX_START and FAST_TX2RX_START_FC are programmed in the TSM
- If RSM's FAST_IP_RX_WU bit is set (IP1_TX2RX, IP2_TX2RX) or FAST_FC_RX_WU bit is set (FC_TX2RX), the RSM will also signal to the TSM to perform a fast warmup during the RX warmup. As the TSM count increments it will skip from FAST_RX_START to FAST_RX_DEST (which are programmed in the TSM)
- The TSM reaches the end of warmup when its count reaches END_OF_RX_WU.

The time needed for the TX2RX transitions are shown below. The 2us + TX_DATA_FLUSH_DLY is related to PA ramp-down at the beginning of the transition .

Transition	Time needed for transition	Roles
IP1_TX2RX	(END_OF_RX_WU - FAST_TX2RX_START) - (FAST_DEST_RX - FAST_START_RX)*FAST_IP_RX_WU + 2us + TX_DATA_FLUSH_DLY	SQTE Initiator
IP2_TX2RX	(END_OF_RX_WU - FAST_TX2RX_START) - (FAST_DEST_RX - FAST_START_RX)*FAST_IP_RX_WU + 2us + TX_DATA_FLUSH_DLY	SQTE Inititator, PDE RD
FC_TX2RX	(END_OF_RX_WU - FAST_TX2RX_START_FC) - (FAST_DEST_RX - FAST_START_RX)*FAST_FC_RX_WU + 2us + TX_DATA_FLUSH_DLY	SQTE Reflector, PDE MD

The RX2RX transition state (S_RX2RX) is described as follows.

- The RSM signals to the TSM using recycle0. In response the TSM will change its count to RECYCLE_COUNT0
- The TSM reaches the end of warmup when its count reaches END_OF_RX_WU.

The TX2TX transition state (S_TX2TX), doesn't require a recycle event; the TX_DATA_FLUSH_DLY accounts for the flushing delay in the transition. The time needed for the RX2RX and TX2TX transitions are shown below.

Transition	Time needed for transition	Roles
S_RX2RX	(END_OF_RX_WU - RECYCLE_COUNT0)	SQTE Initiator, SQTE Reflector
S_TX2TX	TX_DATA_FLUSH_DLY	SQTE Initiator, SQTE Reflector

For SQTE, the RSM has to warmup the transceiver from the TSM IDLE state. For SQTE Initiator role, if RSM's FAST_FC_TX_WU bit is set, the RSM will signal to the TSM to perform a fast warmup during the TX warmup. For SQTE Reflector role, if RSM's FAST_FC_RX_WU bit is set, the RSM will signal to the TSM to perform a fast warmup during the RX warmup.

Transition	Time needed for transition	Roles
WU	END_OF_TX_WU - (FAST_DEST_TX - FAST_START_TX)*FAST_FC_TX_WU + pa_ru + 3us	SQTE Initiator (TX warmup)
WU	END_OF_RX_WU - (FAST_DEST_RX - FAST_START_RX)*FAST_FC_RX_WU + 1us	SQTE Reflector (RX warmup)

55.4.7.9.3.5.2 RX and TX Latency Handling

When transmitting, in order to be "on air" beginning at the desired time, it is necessary to account for the latency through the TXDIG and PLL. To account for this, the RSM uses the TXDIG programmable value TX_DATA_FLUSH_DLY which is specified in terms of microseconds. The RSM will shift TX states (DT_TX, FM_TX, PM_TX) to begin earlier in time by TX_DATA_FLUSH_DLY to ensure that transmit information is "on air" at the correct time.

When receiving is it necessary to account for latency through the receiver and PHY. The RSM uses the TSM programmable value RX_SETTLING_LATENCY (which is specified in terms of microseconds) for this. The RSM will extend RX states (DT_RX, FM_RX, PM_RX) by RX_SETTLING_LATENCY to ensure that the last "on air" information during T_DT, T_FM, or T_PM can be processed by the receiver.

These concepts are illustrated in the example figure below for a SQTE step of type Pk-Tn-Tn-Pk. In both Initator and Reflector roles, the TX (DT_TX, FM_TX, and PM_TX) states are shifted early by TX_DATA_FLUSH_DLY with respect to the "nominal time", while the RX (DT_RX, FM_RX, and PM_RX) states are extended by RX_SETTLING_LATENCY.

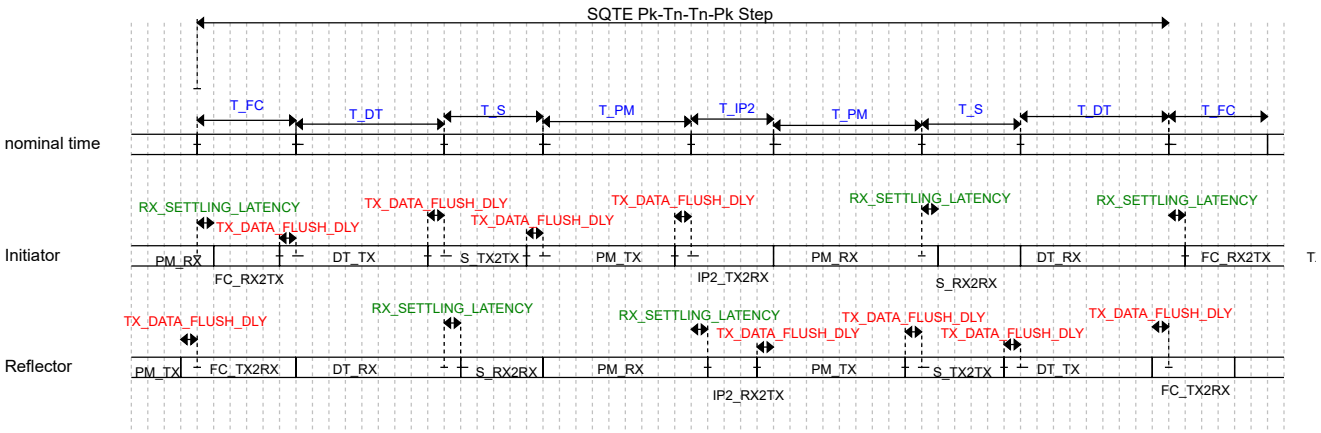


Figure 362. RX and TX Latency

As shown in the timing figure above, after taking into account the RX and TX latencies some of RSM state's durations are different than the "nominal" durations. This is summarized in the table below

Table 466. RSM Actual State Durations

State	Nominal Duration	Actual Duration
FC_RX2TX	T_FC	T_FC - RX_SETTLING_LATENCY - TX_DATA_FLUSH_DLY
IP1_RX2TX	T_IP1	T_IP1 - RX_SETTLING_LATENCY - TX_DATA_FLUSH_DLY
IP2_RX2TX	T_IP2	T_IP2 - RX_SETTLING_LATENCY - TX_DATA_FLUSH_DLY
FC_TX2RX	T_FC	T_FC + TX_DATA_FLUSH_DLY
IP1_TX2RX	T_IP1	T_IP1 + TX_DATA_FLUSH_DLY
IP2_TX2RX	T_IP2	T_IP2 + TX_DATA_FLUSH_DLY
S_RX2RX	T_S	T_S - RX_SETTLING_LATENCY
S_TX2TX	T_S	
PM_RX	T_PM[1]	T_PM[1] + RX_SETTLING_LATENCY
FM_RX	T_FM[2]	T_FM[2] + RX_SETTLING_LATENCY
DT_RX	T_DT[3]	T_DT[3] + RX_SETTLING_LATENCY
PM_TX	T_PM[1]	
FM_TX	T_FM[2]	
DT_TX	T_DT[3]	

1. T_PM0, T_PM1, T_PM2, or T_PM3 depending on t_pm_sel[1:0] for the current step format
2. T_FM0 or T_FM1 depending on t_pm_sel[0] for the current step format
3. T_DT is a function of RATE and RTT_SEQ_LEN programmable bitfields.
   - 1Mbps rate, 32bit rtt_seq_len: 8bit preamble + 32bit data + 4bit trailer = 44us
   - 1Mbps rate, 64bit rtt_seq_len: 8bit preamble + 64bit data + 4bit trailer = 76us
   - 2Mbps rate, 32bit rtt_seq_len: 16bit preamble + 32bit data + 4bit trailer = 26us

- 2Mbps rate, 64bit rtt_seq_len: 16bit preamble + 64bit data + 4bit trailer = 42us

55.4.7.9.3.5.3 Transition Time Summary

The following tables show the transition duration requirements, taking into account the TSM and PA ramp timing, and the TX and RX latency, for different SQTE and PDE roles.

Table 467. SQTE Initiator

Transition Time	Must be programmed to a value no less than
WU[1]	END_OF_TX_WU - (FAST_DEST_TX - FAST_START_TX)*FAST_FC_TX_WU + pa_ru + 3us
T_IP1	IP1_TX2RX - TX_DATA_FLUSH_DLY = (END_OF_RX_WU - FAST_TX2RX_START) - (FAST_DEST_RX - FAST_START_RX)*FAST_IP_RX_WU + 2us
T_S	Maximum of .. S_RX2RX + RX_SETTLING_LATENCY = (END_OF_RX_WU - RECYCLE_COUNT0) + RX_SETTLING_LATENCY S_TX2TX = TX_DATA_FLUSH_DLY
T_IP2	IP2_TX2RX - TX_DATA_FLUSH_DLY = (END_OF_RX_WU - FAST_TX2RX_START) - (FAST_DEST_RX - FAST_START_RX)*FAST_IP_RX_WU + 2us
T_FC	FC_RX2TX + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY = (END_OF_TX_WU - FAST_RX2TX_START_FC) - (FAST_DEST_TX - FAST_START_TX)*FAST_FC_TX_WU + pa_ru + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY

1. The Warmup transition time is shown here for reference, but there is no programmable "WU" bitfield for this in the RSM programming model

Table 468. SQTE Reflector

Transition Time	Must be programmed to a value no less than
WU[1]	END_OF_RX_WU - (FAST_DEST_RX - FAST_START_RX)*FAST_FC_RX_WU + 1us
T_IP1	IP1_RX2TX + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY = (END_OF_TX_WU - FAST_RX2TX_START) - (FAST_DEST_TX - FAST_START_TX)*FAST_IP_TX_WU + pa_ru + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY
T_S	Maximum of .. S_RX2RX + RX_SETTLING_LATENCY = (END_OF_RX_WU - RECYCLE_COUNT0) + RX_SETTLING_LATENCY S_TX2TX = TX_DATA_FLUSH_DLY
T_IP2	IP2_RX2TX + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY = (END_OF_TX_WU - FAST_RX2TX_START) - (FAST_DEST_TX - FAST_START_TX)*FAST_IP_TX_WU + pa_ru + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY

Table continues on the next page...

Table 468. SQTE Reflector (continued)

Transition Time	Must be programmed to a value no less than
T_FC	FC_TX2RX - TX_DATA_FLUSH_DLY = (END_OF_RX_WU - FAST_TX2RX_START_FC) - (FAST_DEST_RX - FAST_START_RX)*FAST_FC_RX_WU + 2us

1. The Warmup transition time is shown here for reference, but there is no programmable "WU" bitfield for this in the RSM programming model

Table 469. PDE MD

Transition Time	Must be programmed to a value no less than
T_IP1	(Not used)
T_S	(Not used)
T_IP2	IP2_RX2TX + RX_SETTLING_LATENCY = (END_OF_TX_WU - FAST_RX2TX_START) - (FAST_DEST_TX - FAST_START_TX)*FAST_IP_TX_WU + pa_ru + RX_SETTLING_LATENCY
T_FC	FC_TX2RX - TX_DATA_FLUSH_DLY = (END_OF_RX_WU - FAST_TX2RX_START_FC) - (FAST_DEST_RX - FAST_START_RX)*FAST_FC_RX_WU + 2us

Table 470. PDE RD

Transition Time	Must be programmed to a value no less than
T_IP1	(Not used)
T_S	(Not used)
T_IP2	IPS_TX2RX = (END_OF_RX_WU - FAST_TX2RX_START) - (FAST_DEST_RX - FAST_START_RX)*FAST_IP_RX_WU + 2us
T_FC	FC_RX2TX + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY= (END_OF_TX_WU - FAST_RX2TX_START_FC) - (FAST_DEST_TX - FAST_START_TX)*FAST_FC_TX_WU + pa_ru + RX_SETTLING_LATENCY + TX_DATA_FLUSH_DLY

55.4.7.9.4 Fast Antenna Diversity (FAD) Common Pin Interface

55.4.7.9.4.1 Introduction

The Fast Antenna Diversity (FAD) mode, when enabled, allows the optimal choice between two antennae to be selected during the PHY's preamble search. The output of the PHY's FAD algorithm is a 1-bit signal which indicates the selected antenna. This output is provided to the FAD Common Pin Interface logic. Refer to the PHY chapter(s) for more information on the FAD algorithm.

The localization control output (lant_sw) can also be used as the input to the FAD Common Pin Interface instead of the PHY signal, but the primary use case is for FAD.

55.4.7.9.4.2 Common Pin Interface

The FAD Common Pin Interface consists of 4 pins: ANT_A, ANT_B, TX_SWITCH, and RX_SWITCH. All pins are outputs from the device. The switch outputs convey timing information to the external module directly interfacing to the antennae, and they also can be used to control an external LNA or PA, via use of a Front End Module, or FEM. The FAD Common Pin Interface offers multiple configurations that allow for a wide range of FEM control schemes. Each of the 4 FAD-related outputs can be configured to be in FAD or TSM mode; in FAD mode, ANT_A and ANT_B are controlled by the selected FAD algorithm through signal antx_out; in TSM mode, ANT_A and ANT_B are pure timing outputs with programmable timing configurable in the TSM. The 4-bit register FAD_CTRL[FAD_NOT_GPIO] determines whether each FAD-related output is in FAD or TSM mode:

In FAD mode (FAD_NOT_GPIO[0]=1), ANT_A is controlled by the FAD algorithm, modified by the register settings {ANTX_CTRLMODE, ANTX_EN, and ANTX_POL}.

In FAD mode (FAD_NOT_GPIO[1]=1), ANT_B is controlled by the FAD algorithm, modified by the register settings {ANTX_CTRLMODE, ANTX_EN, and ANTX_POL}.

In FAD mode (FAD_NOT_GPIO[2]=1), TX_SWITCH output is derived from TSM output **gpio2_trig_en**, modified by the register settings {ANTX_CTRLMODE, ANTX_EN, and ANTX_POL}.

In FAD mode (FAD_NOT_GPIO[3]=1), RX_SWITCH output is derived from TSM output **gpio3_trig_en**, modified by the register settings {ANTX_CTRLMODE, ANTX_EN, and ANTX_POL}.

In TSM mode, ANT_A and ANT_B become pure TSM timing-controlled outputs, similar to TX_SWITCH and RX_SWITCH, and with independently-controlled timing and capable of providing signalling in TX mode, RX mode, or both; for pins in TSM mode, the ANTX_CTRLMODE, ANTX_EN, and ANTX_POL register settings have no effect.

In TSM mode (FAD_NOT_GPIO[0]=0), ANT_A output is a pure timing signal, derived from TSM output **gpio0_trig_en**.

In TSM mode (FAD_NOT_GPIO[1]=0), ANT_B output is a pure timing signal, derived from TSM output **gpio1_trig_en**.

In TSM mode (FAD_NOT_GPIO[2]=0), TX_SWITCH output is derived from TSM output **gpio2_trig_en**.

In TSM mode (FAD_NOT_GPIO[3]=0), RX_SWITCH output is derived from TSM output **gpio3_trig_en**.

The control register XCVR_CTRL[DEMOD_SEL] determines which FAD algorithm drives the FAD Common Pin Interface: 0b10 = 802.15.4 FAD, 0b01 = FSK FAD.

The state of the antenna selection can be ascertained at any time by reading the FAD_CTRL[ANTX] bit: 0 = ANT_A, 1 = ANT_B.

Antenna selection can be placed under direct software control: Setting the FAD_CTRL[ANTX_OVRD_EN]=1 allows FAD_CTRL[ANTX_OVRD] to substitute for the algorithm-driven antenna selection; in this case, ANTX_OVRD=0 asserts ANT_A, and ANTX_OVRD=1 asserts ANT_B. Note that the algorithm-driven state of the antenna is not lost while ANTX_OVRD_EN=1. The override is temporary, for as long as the override enable is set, and control over antenna state reverts to the antx_out when ANTX_OVRD_EN is cleared to 0. The FAD_CTRL register resides in XCVR space.

The following diagram illustrates the FAD pinout, register controls, and FAD/TSM-mode multiplexing.

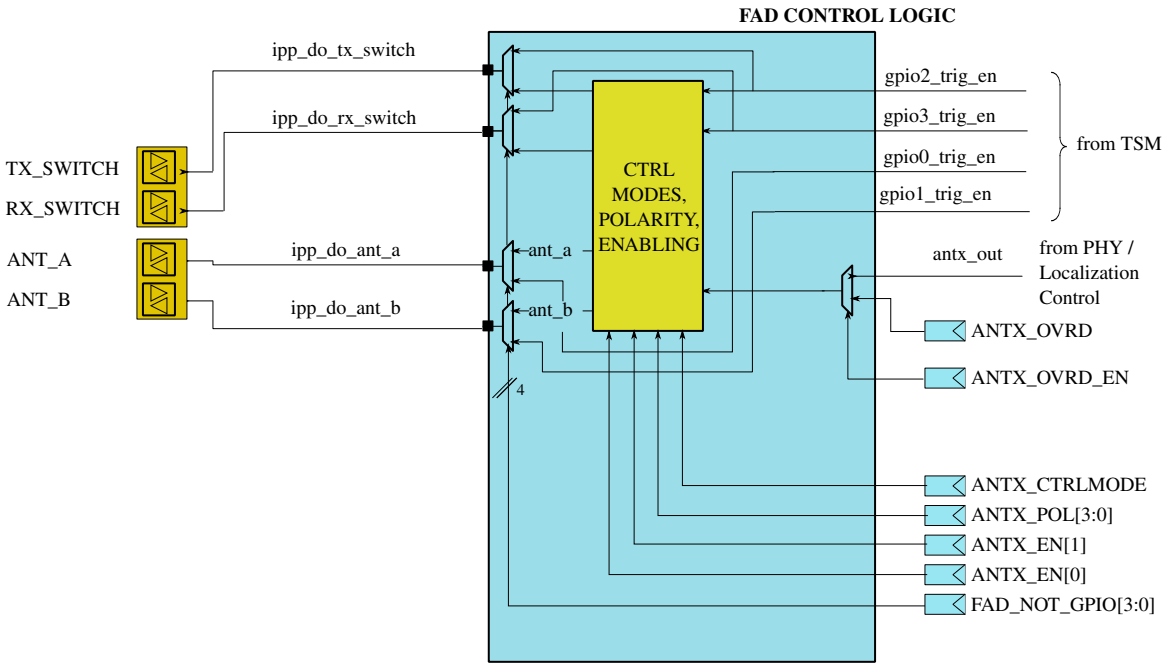


Figure 363. FAD Common Pin Interface

55.4.7.9.4.3 Control Bits

The following table describes the programmable registers which control FAD Common Pin Interface:

Table 471. FAD Common Pin Interface Control Bits

Field	R/W	Description
ANTX_CTRLMODE	rw	**FAD control mode-** selects the single or dual mode **0 : single mode.** ANT_A = NOT(antx) AND(gpio2_trig_en OR gpio3_trig_en) TXSWITCH = gpio2_trig_en RXSWITCH= (gpio2_trig_en OR gpio3_trig_en) **1:dual mode:** ANT_A = NOT(antx) AND(gpio2_trig_en OR gpio3_trig_en) ANT_B = antx AND(gpio2_trig_en OR gpio3_trig_en) TX_SWITCH = gpio2_trig_en RX_SWITCH = gpio3_trig_en
ANTX_EN[1:0]	rw	**FAD Pin Enable-** 00: all disabled

Table continues on the next page...

Table 471. FAD Common Pin Interface Control Bits (continued)

Field	R/W	Description
		01:only RX_SWITCH/TX_SWITCH enabled 10:only ANT_A/ANT_B enabled 11: all enabled
ANTX_POL[3:0]	rw	**Antenna controls mode-** Control the polarity of the FAD pins: ANTX_pol<0>=1 : invert the ANT_A output ANTX_pol<1>=1 : invert the ANT_B output ANTX_pol<2>=1 : invert the TX_SWITCH output ANTX_pol<3>=1 : invert the RX_SWITCH output
FAD_NOT_GPIO[3:0]	rw	**xxx1**: The ANT_A pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **xxx0**: The ANT_A pad is controlled directly by the TSM output gpio0_trig_en **xx1x**: The ANT_B pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **xx0x**: The ANT_B pad is controlled directly by the TSM output gpio1_trig_en **x1xx**: The TX_SWITCH pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **x0xx**: The TX_SWITCH pad is controlled directly by the TSM output gpio2_trig_en **1xxx**: The RX_SWITCH pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **0xxx**: The RX_SWITCH pad is controlled directly by the TSM output gpio3_trig_en
ANTX	ro	**Antenna Selection State-** The ANTX bit reflects the state of the antx_out signal.
DEMOD_SEL[1:0]	rw	**These bits select the demodulator used during reception. Also select which FAD engine is used to select the antenna-** • 0b00: No demodulator selected

Table continues on the next page...

Table 471. FAD Common Pin Interface Control Bits (continued)

Field	R/W	Description
		• 0b01: Use NXP Multi-standard PHY demodulator, and its FAD engine • 0b10: Use Legacy 802.15.4 demodulator, and the 802.15.4 FAD engine • 0b11: Reserved
ANTX_OVRD_EN	rw	**FAD ANTX Override Enable-** • 0: antx_out controls the antenna selection (mission mode) • 1: ANTX_OVRD controls the antenna selection. 0 = ANT_A, 1 = ANT_B.
ANTX_OVRD	rw	**FAD ANTX Override -** If ANTX_OVRD_EN=1, this bit is used instead of antx_out

55.4.7.9.4.4 FEM Example Use Cases

In 2.4GHz systems, the Radio Frequency (RF) Front-End Module is one of the most critical parts. Acting as an interface between the antenna and RF transceiver, the RF front-end module, or FEM, includes sensitive components such as an antenna Low Noise Amplifier (LNA) , power amplifiers (PA), antenna tuning switches, and a power management unit.

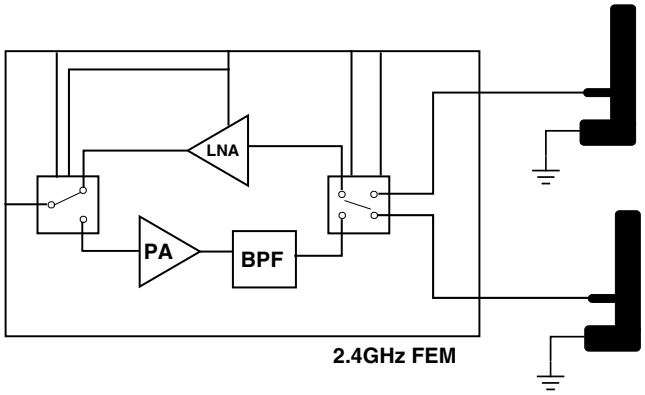


Figure 364. FEM Block Diagram

To accommodate multiple FEM interfaces and control schemes, Fast Antenna Diversity incorporates 2 Antenna Pin Control Modes, which are governed by the register control bit ANTX_CTRLMODE

Some FEM devices use a single input pin to switch the antenna selection. This is called single-mode operation. Operate Fast Antenna Diversity in this mode by programming ANTX_CTRLMODE=0. A timing diagram of single-mode operation is shown below (note: FAD output ANT_B is not used). In this example, one packet is received on Antenna A, followed by another packet received on Antenna B. Both packets require an auto-Acknowledge packet to be transmitted immmediately following reception.

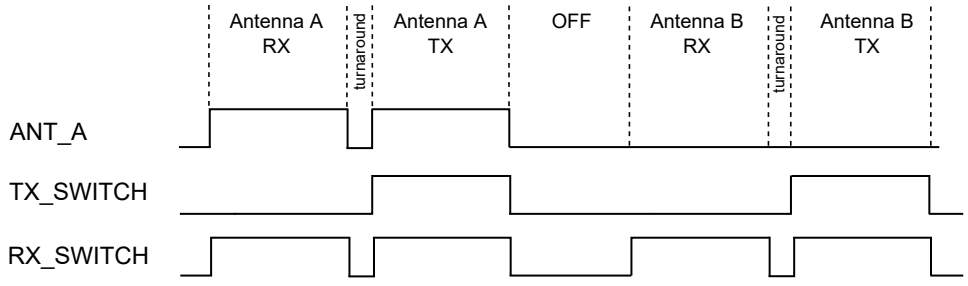


Figure 365. FAD Single Mode Timing

An example FEM Interface diagram, with FAD operated in single mode, is shown below.

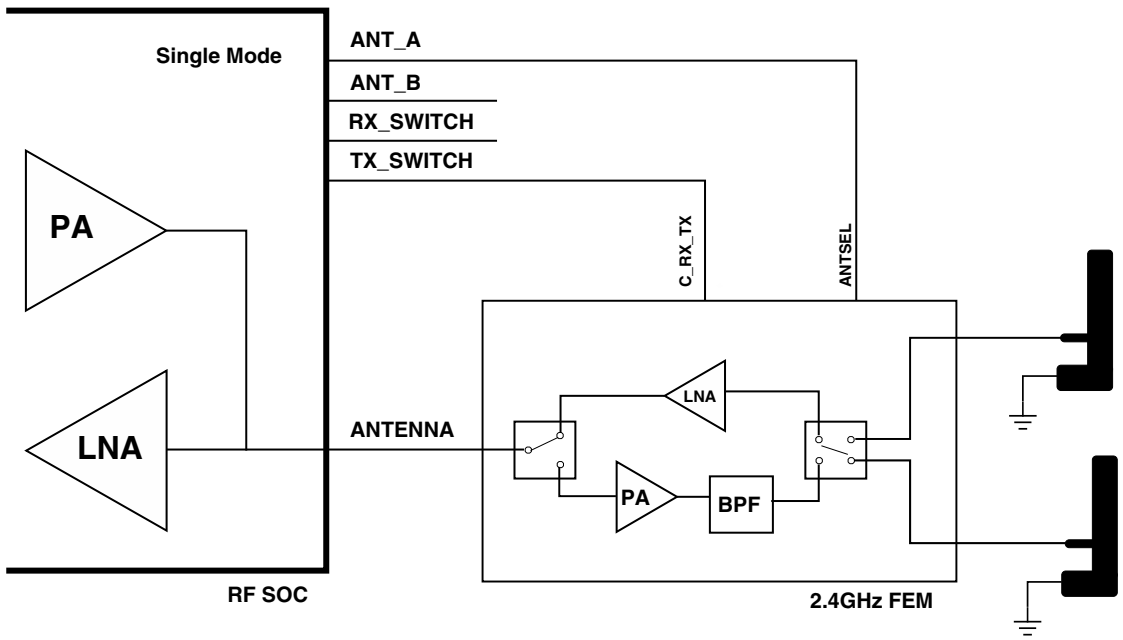


Figure 366. FEM Interface Single Mode

Other FEM devices use 2 input pins to switch the antenna selection. This is called dual-mode operation. Operate Fast Antenna Diversity in this mode by programming ANTX_CTRLMODE=1. A timing diagram of dual-mode operation is shown below. In this example, one packet is received on Antenna A, followed by another packet received on Antenna B. Both packets require an auto-Acknowledge packet to be transmitted immmediately following reception.

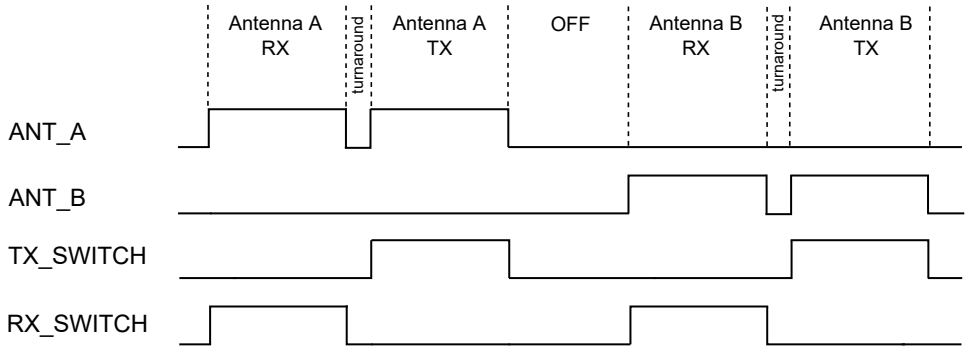


Figure 367. FAD Dual Mode Timing

An example FEM Interface diagram, with FAD operated in dual mode, is shown below.

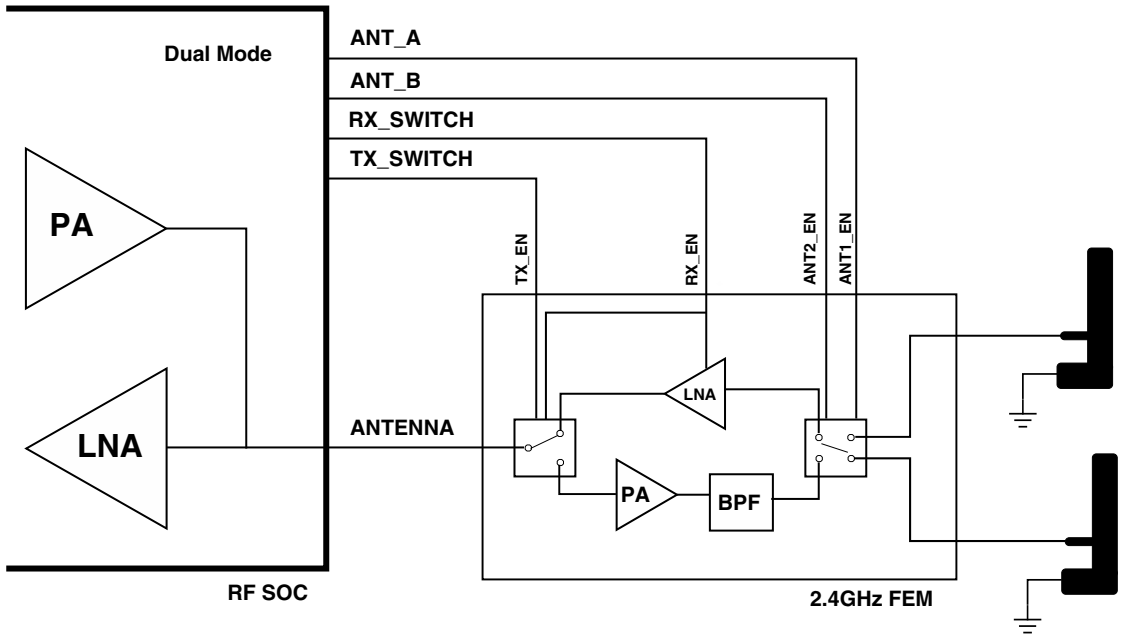


Figure 368. FEM Interface Dual Mode

Other FEM use cases, include external PA only, external LNA only, or external PA plus LNA. In such cases, ANT_A and ANT_B are not required, FAD can be disabled, and TX_SWITCH and RX_SWITCH pins can be operated in TSM mode (pure timing signals). This is called Single Antenna configuration. An example FEM interface diagram illustrating such a use case, is shown below.

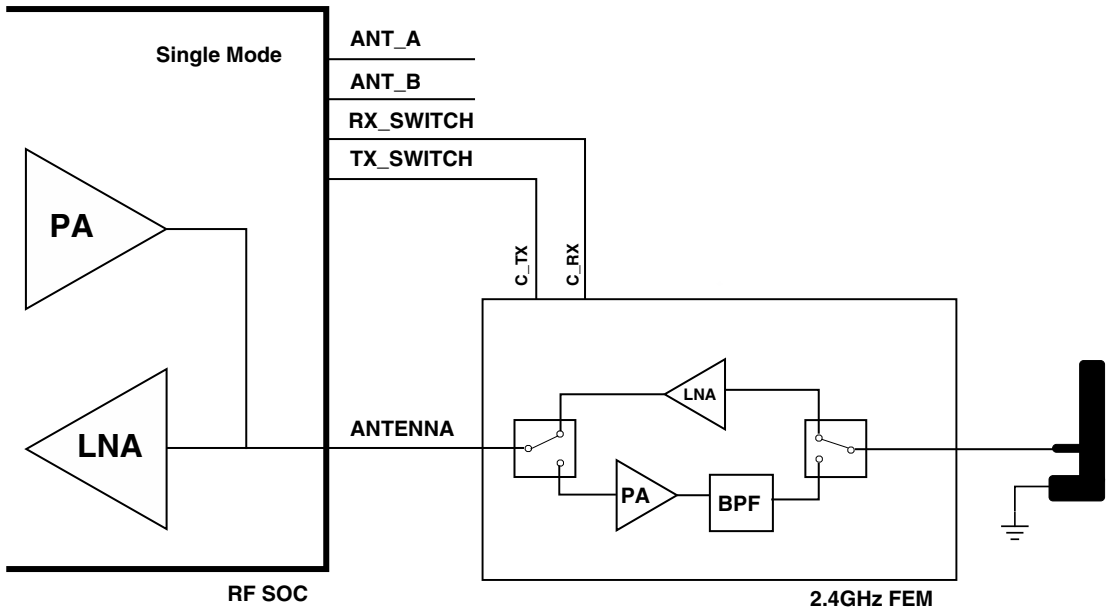


Figure 369. FEM Interface Single Antenna Configuration

55.4.7.10 LDO Calibration

The LDO Calibration module uses a binary search to find the best 6bit calibration code (ldo_cal_trim[5:0]) for the LDO calibration circuit in the analog, and then performs subsequent incremental trims for PLL, VCO and RTXHF LDOs. It is expected that the LDO calibration will be performed on each TX and RX warmup. The entire LDO calibration sequence with incremental calibration requires no more than 8us for all XO reference frequencies supported.

The calibration timing is illustrated in Figure 370 and Figure 371 below.

The calibration sequence is initiated by a TSM control signal (tsm_ldocal_en). The ldo_cal_trim[5:0] is initialized to 6'b10_0000 (6'h20) while the trim codes for the PLL, VCO, RXTXHF and RXTXLF LDOs are intialized to 6'b01_1111 (6'h1F). During each of the 6 steps of the calibration, each bit of the ldo_cal_trim[5:0] is individually set, and a comparator output from the alaog (ldo_trim_cmpout) is used to determine whether the bit should remain set of be cleared. By default, the comparator output signal is captured 2 clk cycles after the ldo_trim_sample signal is de-asserted, but this can also be programmed to 1clk or 3clks if desired. At the end of the 6 steps, the best ldo_cal_trim[5:0] value has been found. The PLL, VCO, RXTXHF and RXTXLF LDO's trim codes are then updated using ldo_cal_trim[5:0] and signed, 4-bit offsets which can be programmed differently for PLL, VCO, RXTXHF and RXTXLF as needed.

The total time needed to determine the ldo_cal_trim is as follows, where LDO_TRIM_SMPL_DLY is a programmable value which is 2 by default but which can also be programmed for values of 1 or 3

1clk + 6*(0.5us + 1clk + LDO_TRIM_SMPL_DLY clks)

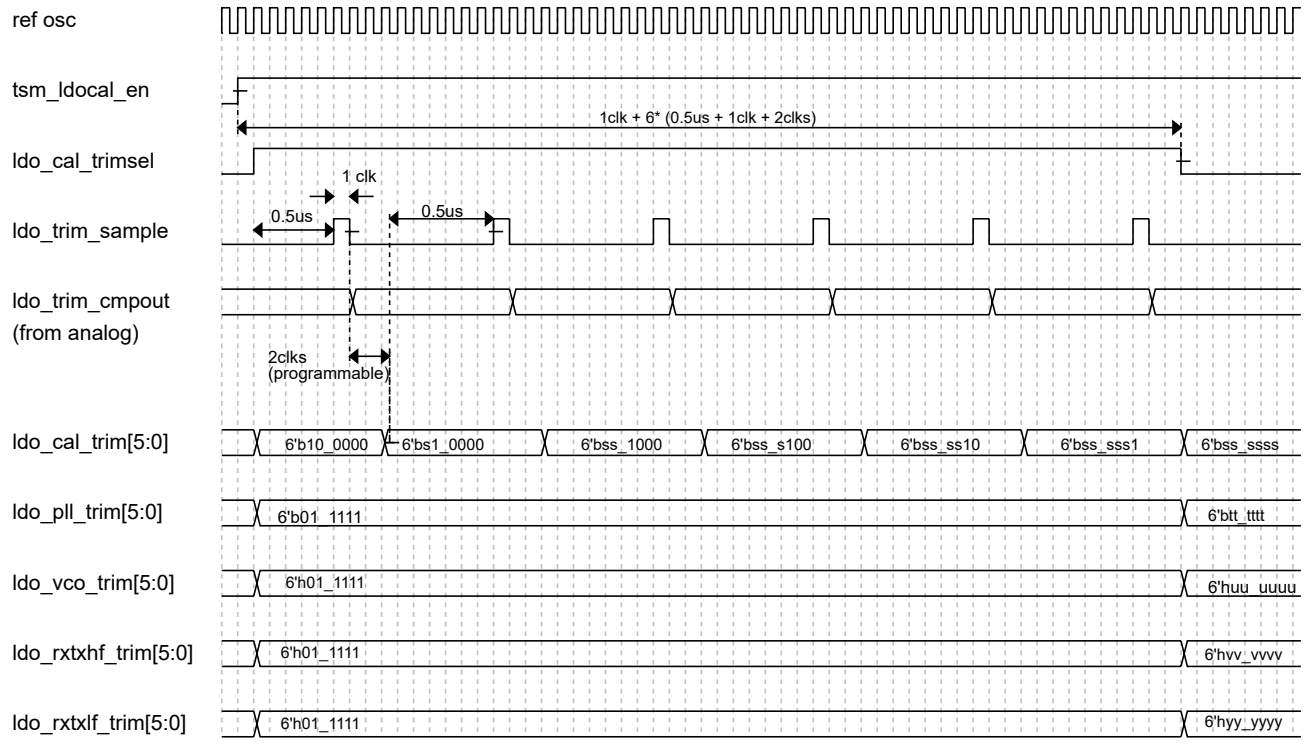


Figure 370. LDO trim part 1: ldo_cal_trim

Incremental calibration allows the PLL, VCO and RXTXHF LDO trim codes to be tested individually and adjusted by +1 trim code value. After the best ldo_trim_code[5:0] has been found from the main calibration, the ldocal module will wait at least 2us but no more than 3us. Following this, trim select signals for the PLL, VCO and RXTXHF LDOs will be sequentially asserted so that the trim code for each of these LDOs can be individually tested. In each case, based on the comparator value the LDO's trim code will either be left as is or incremented by +1 (there is no option to decrease the trim code by 1). Note that the RXTXLF LDO is not incrementally calibrated.

Excluding the +2us wait time before incremental trim, the total time for the incremental calibration is as follows:

3*(0.5us + 1clk + LDO_TRIM_SMPL_DLY clks + 1 clk) + 1 clk

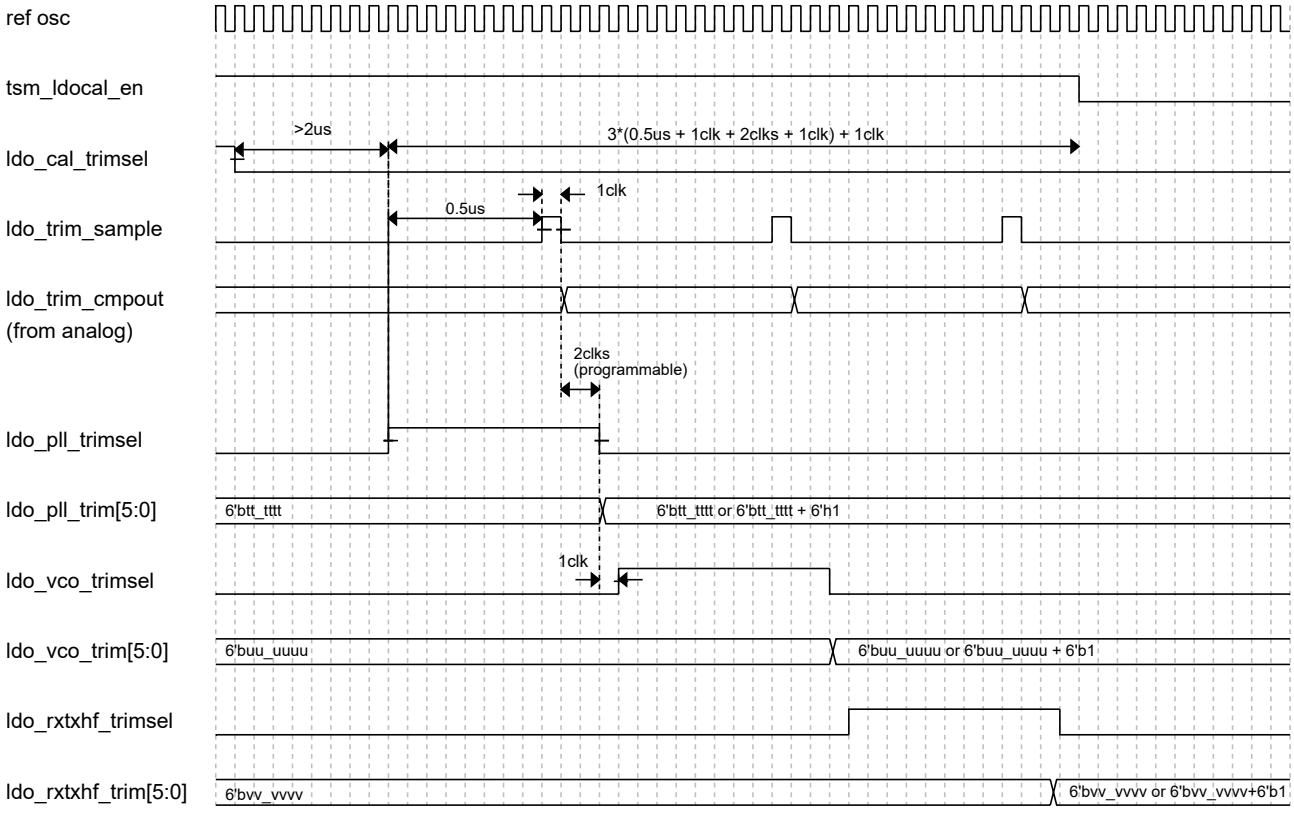


Figure 371. LDO trim part 2: incremental calibrations

The total time for the both parts of the calibrations is as follows:

$ceiling(1clk + 6*(0.5uw + 1clk + LDO_TRIM_SMPL_DLY clks)) + 2us + 3*(0.5us + 1clk + LDO_TRIM_SMPL_DLY clks + 1 clk) + 1 clk

For the default value of LDO_TRIM_SMPL_DLY=2, this is less than 8us for all supported XO frequencies.

Normally, the tsm_ldocal_en remains asserted after the ldo_cal_trim[5:0] has been found in order to perform the incremental calibrations. However, if incremental calibrations are not needed, the tsm_ldocal_en can be de-asserted after ldo_cal_trim[5:0] has been found. This scenario is shown in the figure below. In this case, for the default value of LDO_TRIM_SMPL_DLY=2, only 4us is required for the LDO calibration.

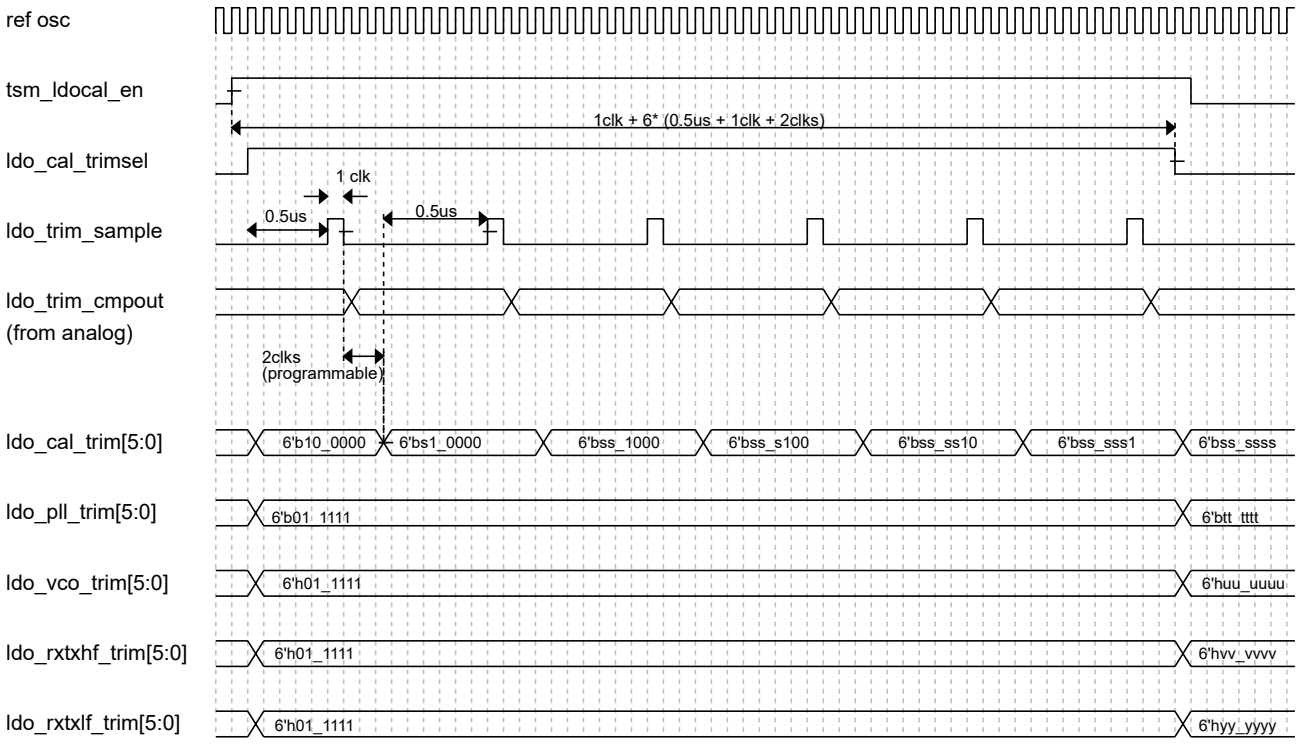


Figure 372. LDO trim without incremental calibrations

55.4.7.11 IPS Fast Overwrite Module

The IPS_FO module provides the capapility to pre-store several address/data pairs that are quickly applied when RX or TX operation begins. Each entry consists of 3 parts:

- Address and config: The ADDR field contains the address to be written, and the ENTRY_TX/RX bits denote if the entry is to be used in TX, RX, or both modes.
- DRS0_DATA: Contains the full 32-bit word that will be written to ADDR as RX/TX begins and the data rate select is 0.
- DRS1_DATA: Contains the full 32-bit word that will be written to ADDR as RX/TX begins and the data rate select is 1.

55.4.7.12 Transceiver Analog and Misc Register Definition

55.4.7.12.1 XCVR_ANALOG register descriptions

55.4.7.12.1.1 XCVR_ANALOG_ADDR memory map

XCVR_ANALOG base address: 48A0_7C00h

Offset	Register	Width (In bits)	Access	Reset value
0h	RF Analog Baseband LDO Control 1 (LDO_0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	RF Analog Baseband LDO Control 2 (LDO_1)	32	RW	0000_0080h
8h	RF Analog XO DIST Control (XO_DIST)	32	RW	0000_0000h
Ch	RF Analog PLL Control (PLL)	32	RW	0000_4040h
10h	RF Analog RX Control0 (RX_0)	32	RW	1000_0002h
14h	RF Analog RX Control1 (RX_1)	32	RW	0002_2008h
18h	RF Analog TX DAC PA Control (TX_DAC_PA)	32	RW	0100_0000h
1Ch	RF Analog DIAG Control 1 (DIAG)	32	RW	4000_0000h
20h	RF Analog SPARE Control (SPARE)	32	RW	0000_0000h

55.4.7.12.1.2 RF Analog Baseband LDO Control 1 (LDO_0)

Offset

Register	Offset
LDO_0	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		LDOTRIM_TRIM_VR...		LDO_CAL...	LDO_CAL_PTAT_BU...		LDO_CAL...	LDO_VCO...	LDO_VCO_PTAT_BU...		LDO_VCO...	LDO_PLL...	LDO_PLL_PTAT_BU...		LDO_PLL...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LDO_RXT...	LDO_RXTXLF_PTAT...		LDO_RXT...	LDO_RXT...	LDO_RXTXHF_PTAT...		LDO_RXT...	0	LDO_LV_...	LDO_LV_TRIM		BG_FORCE	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28	reg_ldotrim_trim_vref_dig[1:0]

Table continues on the next page...

Table continued from the previous page...

Field	Function
LDOTRIM_TRIM_VREF	Trim of vref in ldotrim block. 00b - 0.810 01b - 0.832 10b - 0.854 11b - 0.788
27 LDO_CAL_BYPASS	reg_ldo_cal_bypass_dig LDO bypass control. 0b - disable CAL bypass 1b - eable CAL bypass
26-25 LDO_CAL_PTAT_BUMP	reg_ldo_vco_ptat_bump_dig[1:0] LDO PTAT current boost 00b - nominal 01b - +30% 10b - nominal 11b - +30%
24 LDO_CAL_FORCE	reg_ldo_cal_force_dig LDO force control 0b - Force disable 1b - Force enable
23 LDO_VCO_BYPASS	reg_ldo_vco_bypass_dig LDO bypass control. 0b - disable VCO bypass 1b - eable VCO bypass
22-21 LDO_VCO_PTAT_BUMP	reg_ldo_vco_ptat_bump_dig[1:0] LDO PTAT current boost 00b - nominal 01b - +30% 10b - nominal 11b - +30%
20 LDO_VCO_FORCE	reg_ldo_vco_force_dig LDO force control

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Force disable 1b - Force enable
19 LDO_PLL_BYPASS	reg_ldo_pll_bypass_dig LDO bypass control. 0b - Bypass disabled. 1b - Bypass enabled
18-17 LDO_PLL_PTAT_BUMP	reg_ldo_pll_ptat_bump_dig[1:0] LDO PTAT current boost 00b - nominal 01b - +30% 10b - nominal 11b - +30%
16 LDO_PLL_FORCE	reg_ldo_pll_force_dig LDO force control 0b - force disable 1b - force enable
15 LDO_RXTXLF_BYPASS	reg_ldo_rxtxlf_bypass_dig LDO bypass control. 0b - Bypass disable 1b - Bypass enable
14-13 LDO_RXTXLF_PTAT_BUMP	reg_ldo_rxtxlf_ptat_bump_dig[1:0] LDO PTAT current boost 00b - nominal 01b - +30% 10b - nominal 11b - +30%
12 LDO_RXTXLF_FORCE	reg_ldo_rxtxlf_force_dig LDO force control 0b - disable force 1b - enable force
11	reg_ldo_rxtxihf_bypass_dig

Table continues on the next page...

Table continued from the previous page...

Field	Function
LDO_RXTXHF_BYPASS	LDO bypass control
10-9 LDO_RXTXHF_PTAT_BUMP	reg_ldo_rxtxhf_ptat_bump_dig LDO PTAT current boost 00b - nominal 01b - +30% 10b - nominal 11b - +30%
8 LDO_RXTXHF_FORCE	reg_ldo_rxtxhf_force_dig LDO force control. 0b - Force disabled. 1b - Force enabled
7 —	Reserved
6 LDO_LV_BYPASS	reg_ldo_lv_bypass_dig bypass control for LDO_LV. 0b - disable bypass for ldo_lv 1b - enable bypass for ldo_lv
5-4 LDO_LV_TRIM	reg_ldo_lv_trim_dig[1:0] Trim control for LDO_LV. 00b - 0.91V Default LDO output 01b - 0.86V 10b - 0.97V 11b - 1.3V
3 BG_FORCE	reg_bg_force_dig bandgap force startup 0b - force disable 1b - force enable
2-0 —	Reserved

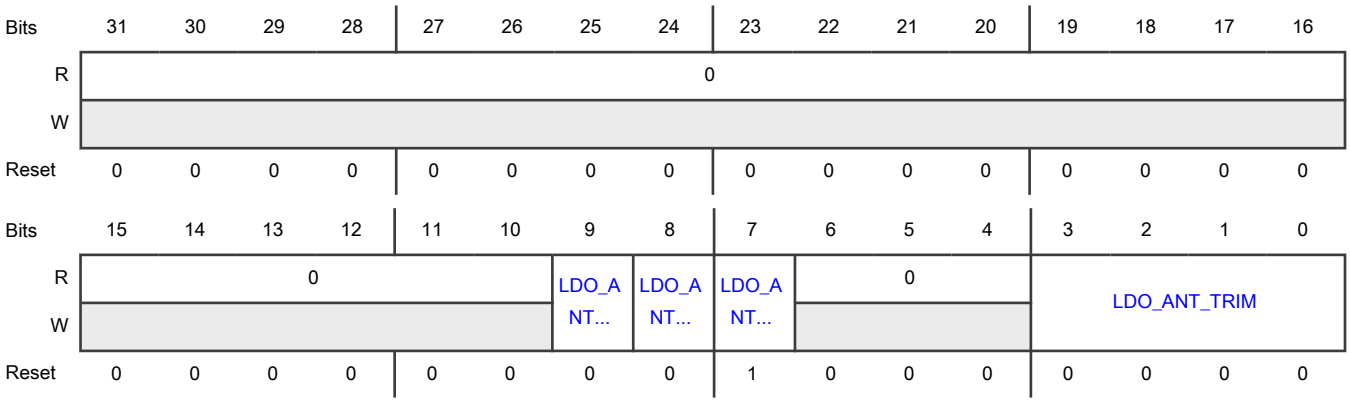
55.4.7.12.1.3 RF Analog Baseband LDO Control 2 (LDO_1)

Offset

Register	Offset
LDO_1	4h

Function

Diagram



Fields

Field	Function
31-10 —	Reserved
9 LDO_ANT_REF_SEL	reg_ldo_ant_ref_sel_dig Input reference voltage type select 0b - sel type disable (Default) 1b - sel type enable
8 LDO_ANT_BYPASS	reg_ldo_ant_bypass_dig LDO bypass control. 0b - ANT bypass disable 1b - ANT bypass enable
7 LDO_ANT_HIZ	reg_ldo_ant_hiz_dig LDO output high-impedance control 0b - high-impedance disabled. 1b - high-impedance enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-4 —	Reserved
3-0 LDO_ANT_TRIM	reg_ldo_ant_trim_dig[3:0] LDO output voltage trim control.SETTING IN RX MODE: 0,SETTING IN TX MODE: depends on desired output power 0000b - 0.91 V (Default) 0001b - 0.97 V 0010b - 1.04 V 0011b - 1.12 V 0100b - 1.21 V 0101b - 1.32 V 0110b - 1.45 V 0111b - 1.52 V 1000b - 1.61 V 1001b - 1.80 V 1010b - 2.06 V 1011b - 2.13 V 1100b - 2.21 V 1101b - 2.30 V 1110b - 2.39 V 1111b - 2.50 V

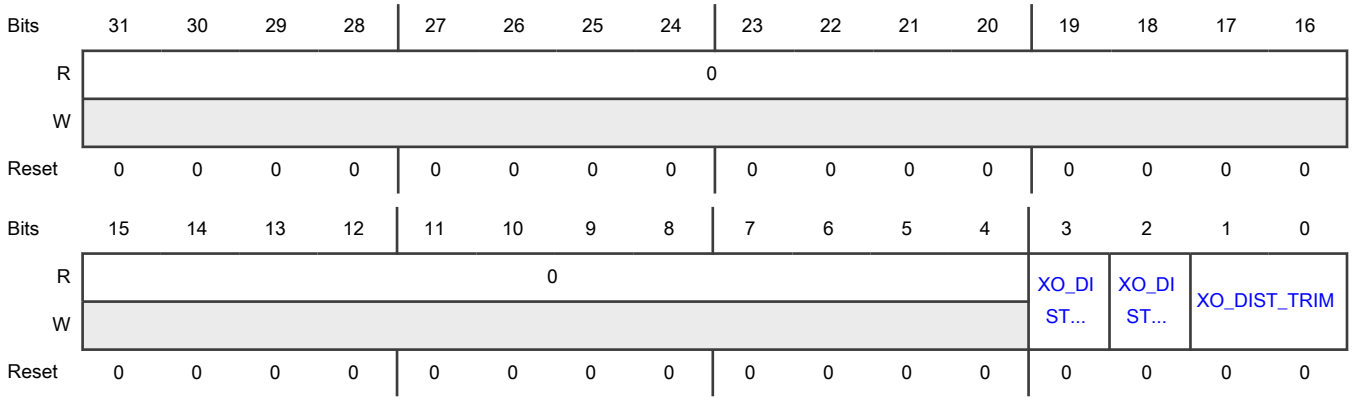
55.4.7.12.1.4 RF Analog XO DIST Control (XO_DIST)

Offset

Register	Offset
XO_DIST	8h

Function

Diagram



Fields

Field	Function
31-4 —	Reserved
3 XO_DIST_BYPASS	reg_xo_dist_bypass XO DIST bypass 0b - XO DIST not bypass 1b - XO DIST bypass
2 XO_DIST_FLIP	reg_xo_dist_flip_dig XO DIST flip output clock relative to input clock 0b - XO DIST doesn't flip the output clock relative to input clock 1b - XO DIST flip the output clock relative to input clock
1-0 XO_DIST_TRIM	reg_xo_dist_trim_dig[1:0] XO DIST LDO output voltage trim 00b - 0.9 V (Default) 01b - 0.86 V 10b - 0.95 V 11b - 1.21 V

55.4.7.12.1.5 RF Analog PLL Control (PLL)

Offset

Register	Offset
PLL	Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PLL_FCA...	0			PLL_PD_TRIM_FCA...		0		PLL_PD_...	PLL_PD_...	0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0			0			PLL_VCO...	0	PLL_VCO_TRIM_KVT			0	0		
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
31 PLL_FCAL_EN_STATIC_RES	reg_fcal_en_static_res_dig Resistor from LDO output to ground control. 0b - resistor is dynamically switched during FCAL operation 1b - resistor is always on during FCAL operation
30-28 —	Reserved
27-26 PLL_PD_TRIM_FCAL_BIAS	reg_pd_trim_fcal_bias_dig[1:0] Sets the ouput of PD positive precharge output 00b - 0.276V 01b - 0.164V 10b - 0.320V 11b - 0.391V
25-24 —	Reserved
23 PLL_PD_EN_VPD_PULLUP	reg_pd_en_vpd_pullup_dig If '1' pulls up the PD precharger output that drives the PD positive output to test the Full Scale range. Needs seq_pd_en_pd_drv_dig='0'. 0b - not pull up vpd output 1b - pull up vpd output
22 PLL_PD_EN_VPD_PULLDN	reg_pd_en_vpd_pulldn_dig If '1' pulls down the PD precharger output that drives the PD positive output to test the Full Scale range. Needs seq_pd_en_pd_drv_dig='0'.

Table continues on the next page...

Table continued from the previous page...

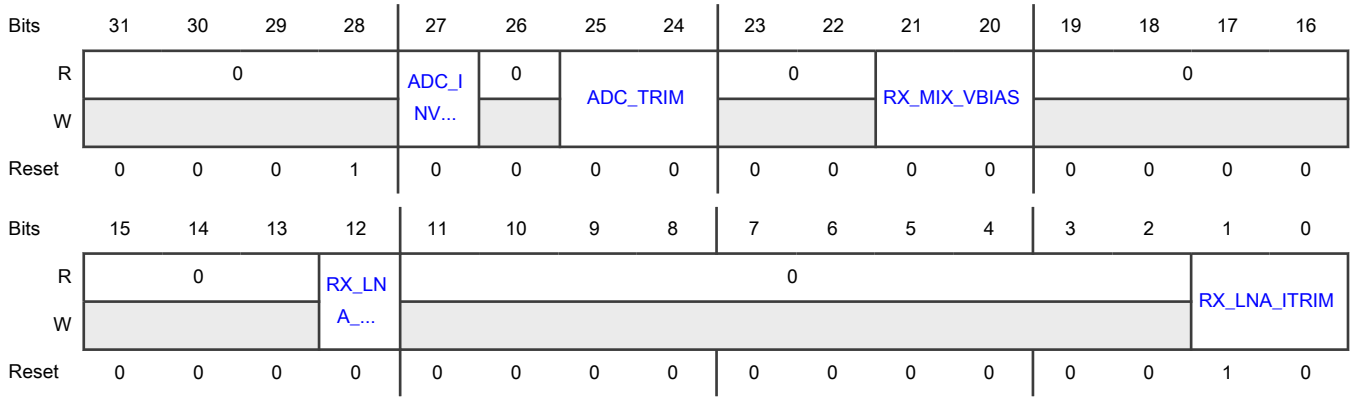
Field	Function
	0b - not pull down vpd output 1b - pull down vpd output
21-15 —	Reserved
14-12 —	Reserved
11-9 —	Reserved
8 PLL_VCO_EN_PKDET	reg_vco_en_pkdet_dig enable peak detector operation 0b - PKDET disable 1b - PKDET enable
7 —	Reserved
6-4 PLL_VCO_TRIM_KVT	reg_vco_trim_kvt_dig[2:0] kv tune control (for loop gain adjustment vs. fref frequency) 000b - 50MHz/V 100b - 60MHz/V for fref = 32M 110b - 70MHz/V 111b - 80MHz/V for fref = 26M
3 —	Reserved
2-0 —	Reserved

55.4.7.12.1.6 RF Analog RX Control0 (RX_0)

Offset

Register	Offset
RX_0	10h

Diagram



Fields

Field	Function
31-28 —	Reserved
27 ADC_INVERT_CLK	reg_adc_invert_clk_dig Invert the ADC clock 0b - not invert clk 1b - invert clk
26 —	Reserved
25-24 ADC_TRIM	reg_adc_trim_dig[1:0] trim of the adc reference current 00b - 0.965V 01b - 0.935V 10b - 0.905V 11b - 0.875V
23-22 —	Reserved
21-20 RX_MIX_VBIAS	reg_rx_mix_vbias_dig[1:0] Mixer bias voltage control 00b - 0.800V 01b - 0.742V 10b - 0.689V 11b - 0.857V

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-13 —	Reserved
12 RX_LNA_PTAT_FORCE_START	reg_rtfe_ptat_force_dig RTFE PTAT biasing force startup.
11-2 —	Reserved
1-0 RX_LNA_ITRIM	reg_rx_lna_itrim_dig[1:0] lna PTAT biasing current trim 00b - 3.7u -25% 01b - 4.4u -15% 10b - 5.1u Default 11b - 5.6u +10%

55.4.7.12.1.7 RF Analog RX Control1 (RX_1)

Offset

Register	Offset
RX_1	14h

Function

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														CBPF_TRIM_SHORT...	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		CBPF_VCM_TRIM		0		CBPF_TRIM_Q		0		CBPF_TRIM_I		CBPF_TY...		0	
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-18 —	Reserved
17-16 CBPF_TRIM_SHORT_DCBIAS	reg_cbpf_trim_short_dcbias_dig[1:0] Trim of dc voltage to which the inputs of the ADC will be shorted when reg_cbpf_short_dig=1 00b - 470mV 01b - 438mV 10b - 413mV 11b - 385mV
15-14 —	Reserved
13-12 CBPF_VCM_TRIM	reg_cbpf_vcm_trim_dig[1:0] vcm trim bits 00b - 480mV 01b - 453mV 10b - 426mV 11b - 401mV
11-10 —	Reserved
9-8 CBPF_TRIM_Q	reg_cbpf_trim_q_dig[1:0] trim of the amplifier reference current (q-path) 00b - 5u (Default) 01b - 6.25u 10b - 7.5u 11b - 8.75u
7-6 —	Reserved
5-4 CBPF_TRIM_I	reg_cbpf_trim_i_dig[1:0] trim of the amplifier reference current (i-path) 00b - 5u (Default) 01b - 6.25u 10b - 7.5u

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - 8.75u
3 CBPF_TYPE	reg_cbpf_type_dig Filter type 0b - Real 1b - Complex,
2-0 —	Reserved

55.4.7.12.1.8 RF Analog TX DAC PA Control (TX_DAC_PA)

Offset

Register	Offset
TX_DAC_PA	18h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				DAC_TRIM_CFBK_D...		DAC_TRIM_CFBK		0				TX_PA_VBIAS			
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				DAC_TRIM_IBIAS		DAC_TRIM_RLOAD		0				DAC_INV...	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-26 DAC_TRIM_CFBK_DRS	reg_dac_trim_cfbk_dig[1:0] trim of amplifier feedback capacitor when datarate is 2Mbps

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 675fF 01b - 1.35pF 10b - 1.35pF 11b - 2.025pF
25-24 DAC_TRIM_CFBK	reg_dac_trim_cfbk_dig[1:0] trim of amplifier feedback capacitor 00b - 675fF 01b - 1.35pF 10b - 1.35pF 11b - 2.025pF
23-18 —	Reserved
17-16 TX_PA_VBIAS	reg_tx_pa_vbias_dig[1:0] TX PA bias voltage control 00b - 0.460V 01b - 0.431V 10b - 0.403V 11b - 0.375V
15-12 —	Reserved
11-10 DAC_TRIM_IBIAS	reg_dac_trim_ibias_dig[1:0] trim of DAC bias current 00b - 3.0uA (I_lsb=250nA) 01b - 2.5uA 10b - 3.8uA 11b - 5.0uA
9-8 DAC_TRIM_RLOAD	reg_dac_trim_rload_dig[1:0] trim of DAC resistor load 00b - 3K 01b - 2.25K 10b - 3.75K

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - 4.5K
7-4 —	Reserved
3 DAC_INVERT_CLK	reg_dac_invert_clk_dig Invert clk coming to DAC. Changes the polarity in which the input data is captured
2-0 —	Reserved

55.4.7.12.1.9 RF Analog DIAG Control 1 (DIAG)

Offset

Register	Offset
DIAG	1Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ATX_ON_...	DIAG_DIS	DAC_AMP...	LDO_ANT...	XO_DIST...	0	DAC_DIA...	VCO_DIA...	PD_DIAG...	0	ADC_DIA...	CBPF_EN...	CBPF_Q_...	CBPF_Q_...	CBPF_I_...	CBPF_I_...
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RTFE_DI...	0	PROC_MO...	LDOTRIM...	BG_DIAG...	LDO_LV_...	LDO_RXT...	LDO_RXT...	0		LDO_PLL...	LDO_VCO...	LDO_CAL...	DIAG_CODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ATX_ON_2P4GHZ	reg_2p4ghz_atx_on_dig Enable the pathway to route diag into the SOC
30	reg_diag_dis_dig

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIAG_DIS	diag_sel control signal
29 DAC_AMP_DIAG_SEL	reg_dac_amp_diag_sel_dig diag_sel control signal for DAC AMP
28 LDO_ANT_DIAG_SEL	reg_ldo_ant_diag_sel_dig diag_sel control signal for LDO
27 XO_DIST_DIAG_SEL	reg_xo_dist_diag_sel_dig diag_sel control signal for XO DIST
26 —	Reserved
25 DAC_DIAG_SEL	reg_dac_diag_sel_dig diag_sel control signal for DAC
24 VCO_DIAG_SEL	reg_vco_diag_sel_dig diag_sel control signal for VCO
23 PD_DIAG_SEL	reg_pd_diag_sel_dig diag_sel control signal for PD
22 —	Reserved
21 ADC_DIAG_SEL	reg_adc_diag_sel_dig diag_sel control signal for ADC
20 CBPF_EN_DIAG_MEAS	reg_cbpf_en_diag_meas_dig Enable diag measurement through ADC. The ouput of CBPF is put in Hiz in diag mode and the diag voltages are routed to the ADC
19 CBPF_Q_DIAG_SEL_2	reg_cbpf_q_diag_sel_2_dig diag_sel control signal for CBPF Q
18	reg_cbpf_q_diag_sel_1_dig

Table continues on the next page...

Table continued from the previous page...

Field	Function
CBPF_Q_DIAG_SEL_1	diag_sel control signal for CBPF Q
17 CBPF_I_DIAG_SEL_2	reg_cbpf_i_diag_sel_2_dig diag_sel control signal for CBPF I
16 CBPF_I_DIAG_SEL_1	reg_cbpf_i_diag_sel_1_dig diag_sel control signal for CBPF I
15 RTFE_DIAG_SEL	reg_rtfe_diag_sel_dig diag_sel control signal
14 —	Reserved
13 PROC_MON_DIAG_SEL	reg_proc_mon_diag_sel_dig diag_sel control signal
12 LDOTRIM_DIAG_SEL	reg_ldotrim_diag_sel_dig diag_sel control signal
11 BG_DIAG_SEL	reg_bg_diag_sel_dig diag_sel control signal
10 LDO_LV_DIAG_SEL	reg_ldo_lv_diag_sel_dig diag_sel control signal
9 LDO_RXTXHF_DIAG_SEL	reg_ldo_rxtxhf_diag_sel_dig diag_sel control signal
8 LDO_RXTXLF_DIAG_SEL	reg_ldo_rxtxlf_diag_sel_dig diag_sel control signal
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 LDO_PLL_DIAG_SEL	reg_ldo_pll_diag_sel_dig diag_sel control signal
4 LDO_VCO_DIAG_SEL	reg_ldo_vco_diag_sel_dig diag_sel control signal
3 LDO_CAL_DIAG_SEL	reg_ldo_cal_diag_sel_dig diag_sel control signal
2-0 DIAG_CODE	reg_diag_code_dig[2:0] Diag code

55.4.7.12.1.10 RF Analog SPARE Control (SPARE)

Offset

Register	Offset
SPARE	20h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		SPARE_DIAG_SEL		0				SPARELV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 SPARE_DIAG_SEL	reg_spare_diag_sel_dig[1:0] diag_sel control signal for RX peak detector.
11-7 —	Reserved
6-0 SPARELV	reg_sparelv_dig[1:0] Spare register input signals

55.4.7.12.2 XCVR_MISC register descriptions

55.4.7.12.2.1 XCVR_MISC_ADDR memory map

XCVR_MISC base address: 48A0_7D00h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRANSCEIVER CONTROL (XCVR_CTRL)	32	RW	0000_0140h
4h	TRANSCEIVER STATUS (XCVR_STATUS)	32	RW	See section
8h	FAD CONTROL (FAD_CTRL)	32	RW	0000_F080h
Ch	TRANSCEIVER DMA CONTROL (DMA_CTRL)	32	RW	0000_0000h
10h	DBG Ram control register (DBG_RAM_CTRL)	32	RW	0000_0000h
14h	DBG RAM ADDRESS (DBG_RAM_ADDR)	32	RW	47FC_0000h
18h	DBG RAM STOP ADDRESS (DBG_RAM_STOP_ADDR)	32	R	0000_0000h
1Ch	LDO TRIM Configuration 0 (LDO_TRIM_0)	32	RW	0000_0000h
20h	LDO TRIM Configuration 1 (LDO_TRIM_1)	32	RW	0000_0000h
24h	RF Analog LDO Trim Res Control 0 (LDO_TRIM_RES_0)	32	R	0000_0000h
28h	RF Analog LDO Trim Res Control 1 (LDO_TRIM_RES_1)	32	R	0000_0000h
2Ch	LCL CTRL CFG 0 (LCL_CFG0)	32	RW	0000_0000h
30h	LCL CTRL CFG 1 (LCL_CFG1)	32	RW	0000_0000h
34h	LCL CTRL TX CONFIG0 (LCL_TX_CFG0)	32	RW	0000_0000h
38h	LCL CTRL TX CONFIG1 (LCL_TX_CFG1)	32	RW	0000_0000h
3Ch	LCL CTRL TX CONFIG2 (LCL_TX_CFG2)	32	RW	0000_0000h
40h	LCL CTRL RX CONFIG0 (LCL_RX_CFG0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

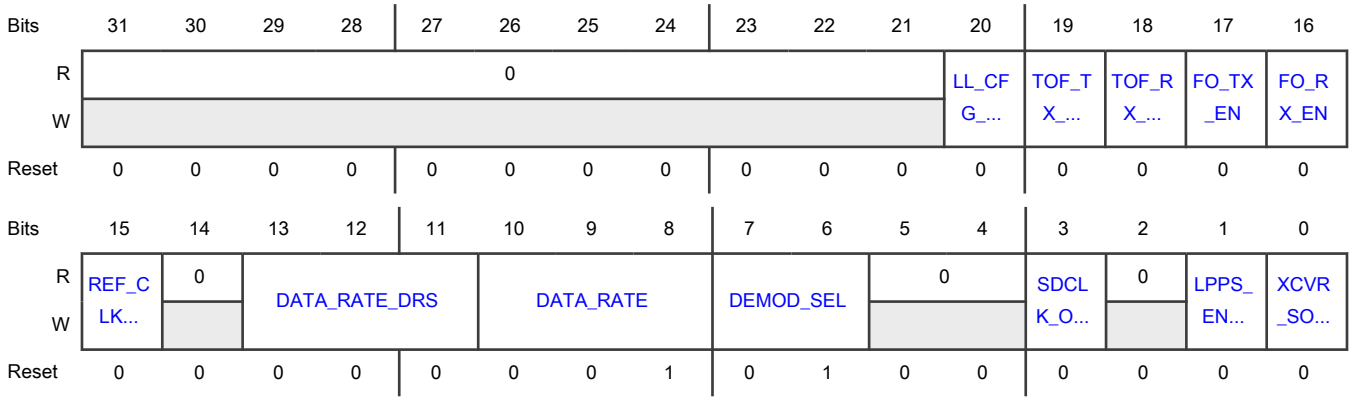
Offset	Register	Width (In bits)	Access	Reset value
44h	LCL CTRL RX CONFIG1 (LCL_RX_CFG1)	32	RW	0000_0000h
48h	LCL CTRL RX CONFIG2 (LCL_RX_CFG2)	32	RW	0000_0000h
4Ch	LCL CTRL PM MSB (LCL_PM_MSB)	32	RW	0000_0000h
50h	LCL CTRL PM LSB (LCL_PM_LSB)	32	RW	0000_0000h
54h	LCL GPIO CTRL 0 (LCL_GPIO_CTRL0)	32	RW	0000_0000h
58h	LCL GPIO CTRL 1 (LCL_GPIO_CTRL1)	32	RW	0000_0000h
5Ch	LCL GPIO CTRL 2 (LCL_GPIO_CTRL2)	32	RW	0000_0000h
60h	LCL GPIO CTRL 3 (LCL_GPIO_CTRL3)	32	RW	0000_0000h
64h	LCL GPIO CTRL 4 (LCL_GPIO_CTRL4)	32	RW	0000_0000h
68h	LCL_DMA_MASK_DELAY (LCL_DMA_MASK_DELAY)	32	RW	0000_0000h
6Ch	LCL_DMA_MASK_PERIOD (LCL_DMA_MASK_PERIOD)	32	RW	0000_0000h
70h	Ranging Sequence Manager Control and Status (RSM_CSR)	32	RW	0000_0000h
74h	Ranging Sequence Manager Control (RSM_CTRL0)	32	RW	0000_0000h
78h	Ranging Sequence Manager Control (RSM_CTRL1)	32	RW	0000_0000h
7Ch	Ranging Sequence Manager Control (RSM_CTRL2)	32	RW	0000_0000h
80h	Ranging Sequence Manager Control (RSM_CTRL3)	32	RW	0000_0000h
84h	Ranging Sequence Manager Control (RSM_CTRL4)	32	RW	0000_0000h
9Ch	RF DFT CTRL (RF_DFT_CTRL)	32	RW	0000_0000h
A0h - BCh	IPS FAST OVERWRITE ADDRESS (IPS_FO_ADDR0 - IPS_FO_ADDR7)	32	RW	0000_0000h
C0h - DCh	IPS FAST OVERWRITE DRS0 DATA (IPS_FO_DRS0_DATA0 - IPS_FO_DRS0_DATA7)	32	RW	0000_0000h
E0h - FCh	IPS FAST OVERWRITE DRS1 DATA (IPS_FO_DRS1_DATA0 - IPS_FO_DRS1_DATA7)	32	RW	0000_0000h

55.4.7.12.2.2 TRANSCEIVER CONTROL (XCVR_CTRL)

Offset

Register	Offset
XCVR_CTRL	0h

Diagram



Fields

Field	Function
31-21 —	Reserved
20 LL_CFG_CAPT_DIS	Link Layer Configuration Capture Disable A few of the link layer configuration signals used by the transceiver can be captured at the start of TX to ensure that they do not change value until after the TSM completes warmdown. This feature is normally enabled but it can be disabled by setting this bit. 0b - Enabled: Link Layer configuration inputs are captured. 1b - Disabled: Link Layer configurations are not captured.
19 TOF_TX_SEL	Time-of-Flight TX Select Selects the signal output to SOC timer for time-of-flight timestamping, when TSM is in tx_mode. 0b - TSM: tx_dig_en 1b - TXDIG: pa_wu_complete
18 TOF_RX_SEL	Time-of-Flight RX Select Selects the signal output to SOC timer for time-of-flight timestamping, when TSM is in rx_mode. 0b - PHY: aa_fnd_to_ll 1b - Localization Control: pattern_found
17 FO_TX_EN	Fast Overwrite TX Enable If set, Fast Overwrite is enabled for TX
16 FO_RX_EN	Fast Overwrite RX Enable If set, Fast Overwrite is enabled for RX
15	transciever ref clk freq control This register selects the Reference Clock Frequency for the Radio.

Table continues on the next page...

Table continued from the previous page...

Field	Function
REF_CLK_FREQ	0b - 32MHz 1b - 26MHz
14 —	Reserved
13-11 DATA_RATE_DRS	Radio data rate setting, Data Rate Switch This register selects the data rate for TXDIG when the alternate data rate is selected (Bluetooth LE sel_2mbps=1, or Generic LL datarate_config_sel=1). 000b - 2Mbps 001b - 1Mbps 010b - 500Kbps 011b - 250Kbps 1xxb - Reserved
10-8 DATA_RATE	Radio data rate setting This register selects the data rate for TXDIG when the primary data rate is selected (Bluetooth LE sel_2mbps=0, or Generic LL datarate_config_sel=0). 000b - 2Mbps 001b - 1Mbps 010b - 500Kbps 011b - 250Kbps 1xxb - Reserved
7-6 DEMOD_SEL	Demodulator Selector This bit selects the demodulator used during reception 00b - No demodulator selected 01b - Use NXP Multi-standard PHY demodulator 10b - Use Legacy 802.15.4 demodulator 11b - Reserved
5-4 —	Reserved
3 SDCLK_OUT_EN	sdclk out control 0b - no sdclk out 1b - enable sdclk out
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 LPPS_ENABLE	transciever lpps enable control 0b - no lpps feature 1b - enable lpps feature
0 XCVR_SOFT_RESET	transciever soft reset control 0b - no soft reset 1b - enable soft reset on transciever

55.4.7.12.2.3 TRANSCEIVER STATUS (XCVR_STATUS)

Offset

Register	Offset
XCVR_STATUS	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TX_MODE	RX_MODE	TSM_BUSY	0			TSM_IRQ1	TSM_IRQ0	TSM_COUNT							
W							W1C	W1C								
Reset	u	u	0	0	0	0	0	0	u	u	u	u	u	u	u	u

Fields

Field	Function
31-16 —	Reserved
15 TX_MODE	Transmit Mode Indicates an TX transceiver operation is in progress.
14	Receive Mode

Table continues on the next page...

Table continued from the previous page...

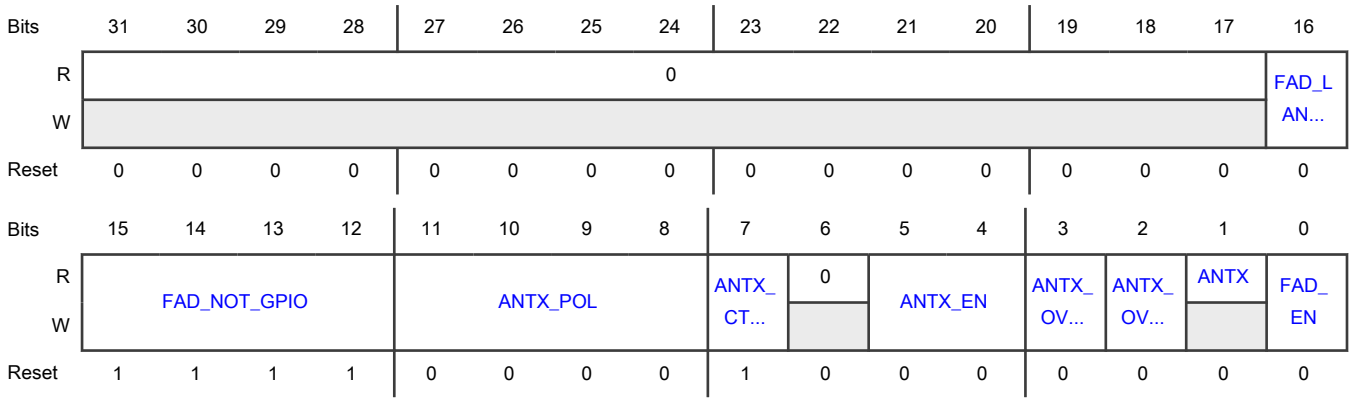
Field	Function
RX_MODE	Indicates an RX transceiver operation is in progress.
13 TSM_BUSY	tsm busy status Indicates TSM is busy or not, active high
12-10 —	Reserved
9 TSM_IRQ1	TSM Interrupt #1 0b - TSM Interrupt #1 is not asserted. 1b - TSM Interrupt #1 is asserted. Write '1' to this bit to clear it.
8 TSM_IRQ0	TSM Interrupt #0 0b - TSM Interrupt #0 is not asserted. 1b - TSM Interrupt #0 is asserted. Write '1' to this bit to clear it.
7-0 TSM_COUNT	TSM_COUNT Reflects the instantaneous value of the TSM counter.

55.4.7.12.2.4 FAD CONTROL (FAD_CTRL)

Offset

Register	Offset
FAD_CTRL	8h

Diagram



Fields

Field	Function
31-17 —	Reserved
16 FAD_LANT_SEL	FAD versus LANT_LUT_GPIO Selector Controls the mux which select whether the LANT_LUT_GPIO[3:0] or the FAD control signals are output to RF_GPO[11:8]. Refer also to the figure in the Coexistence/FEM/LANT connections section of the Radio Control chapter. 0b - LANT_LUT_GPIO[3:0] 1b - {ANT_B, ANT_A, RX_SWITCH, TX_SWITCH}
15-12 FAD_NOT_GPIO	FAD versus GPIO Mode Selector **xxx1** : The ANT_A pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **xxx0** : The ANT_A pad is controlled directly by the TSM output gpio0_trig_en **xx1x** : The ANT_B pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **xx0x** : The ANT_B pad is controlled directly by the TSM output gpio1_trig_en **x1xx** : The TX_SWITCH pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **x0xx** : The TX_SWITCH pad is controlled directly by the TSM output gpio2_trig_en **1xxx** : The RX_SWITCH pad is controlled by the antenna-selection algorithm, modified by the register settings above {ANTX_CTRLMODE, ANTX_EN, ANTX_POL} **0xxx** : The RX_SWITCH pad is controlled directly by the TSM output gpio3_trig_en
11-8 ANTX_POL	FAD Antenna Controls Polarity Control the polarity of the FAD pins: ANTX_POL<0> : This bit currently has no functionality ANTX_POL<1> : This bit currently has no functionality ANTX_POL<2>=1 : invert the TX_SWITCH output ANTX_POL<3>=1 : invert the RX_SWITCH output
7 ANTX_CTRLMODE	Antenna Diversity Control Mode When ANTX_CTRLMODE=1 (dual mode): ANT_A=NOT(ANTX) AND (GPIO3_TRIG_EN OR GPIO2_TRIG_EN) ANT_B=ANTX AND (GPIO3_TRIG_EN OR GPIO2_TRIG_EN) TX_SWITCH=GPIO2_TRIG_EN RX_SWITCH=GPIO3_TRIG_EN

Table continues on the next page...

Table continued from the previous page...

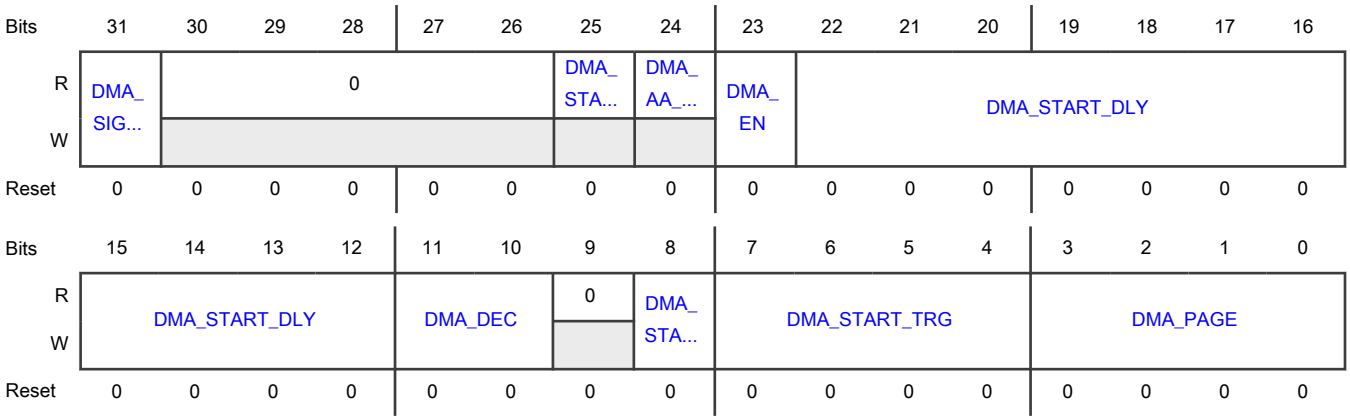
Field	Function
	When ANTX_CTRLMODE=0 (single mode): ANT_A=NOT(ANTX) AND (GPIO3_TRIG_EN OR GPIO2_TRIG_EN) ANT_B=ANTX AND (GPIO3_TRIG_EN OR GPIO2_TRIG_EN) TX_SWITCH=GPIO2_TRIG_EN RX_SWITCH=(GPIO3_TRIG_EN OR GPIO2_TRIG_EN) GPIO2_TRIG_EN and GPIO3_TRIG_EN are outputs of the Transceiver Sequence Manager (TSM). The TSM timing registers associated with GPIO2_TRIG_EN and GPIO3_TRIG_EN should be programmed with the desired TX_SWITCH and RX_SWITCH timing , before enabling Fast Antenna Diversity.
6 —	Reserved
5-4 ANTX_EN	FAD Antenna Controls Enable For any of the 4 FAD-related outputs {ANT_B, ANT_A, RX_SWITCH, TX_SWITCH}, when the associated register control bit (FAD_NOT_GPIO[x]=1) puts that output in FAD mode, ANTX_EN[1:0] determines which pairs of FAD related outputs are enabled: 00b - all disabled (held low) 01b - only RX/TX_SWITCH enabled 10b - only ANT_A/B enabled 11b - all enabled
3 ANTX_OVRD	Antenna State Override Value When ANTX_OVRD_EN=1, the FAD antenna state (ANTX) is overriden by the value of this register; otherwise, antenna state is determined by the 802.15.4 FAD state machine, if XCVR_CTRL[DEMOD_SEL]=0b10, or the FSK PHY FAD state machine if XCVR_CTRL[DEMOD_SEL]=0b01.
2 ANTX_OVRD_EN	Antenna State Override Enable When ANTX_OVRD_EN=1, the FAD antenna state (ANTX) is overriden by the value of the ANTX_OVRD register; otherwise, antenna state is determined by the 802.15.4 FAD state machine, if XCVR_CTRL[DEMOD_SEL]=0b10, or the FSK PHY FAD state machine if XCVR_CTRL[DEMOD_SEL]=0b01.
1 ANTX	Antenna Selection State The ANTX bit reflects the state of antx_out signal.
0 FAD_EN	Fast Antenna Diversity Enable 0b - Fast Antenna Diversity disabled 1b - Fast Antenna Diversity enabled

55.4.7.12.2.5 TRANSCEIVER DMA CONTROL (DMA_CTRL)

Offset

Register	Offset
DMA_CTRL	Ch

Diagram



Fields

Field	Function
31 DMA_SIGNAL_VALID_MASK_EN	DMA Signal Valid Mask Enable **NOTE** This bit should not be set if RXDIG's RX_IQ_PH_OUTPUT_COND bit is set. 0b - Disable use of dma_signal_valid_mask. 1b - Enable use of dma_signal_valid_mask.
30-26 —	Reserved
25 DMA_START_TRIGGERED	DMA Start Trigger Occurred This read-only status bit becomes set, when, during a DMA session (DMA_PAGE > 0), the trigger source selected by DMA_START_TRG[2:0] occurs, with the edge sensitivity selected by DMA_START_EDGE. This bit is cleared when the DMA is re-armed by programming DMA_PAGE to 0 and then to a non-zero value.
24 DMA_AA_TRIGGERED	DMA Access Address triggered This status bit becomes set when the PHY indicates that a Network Address match has occurred. In some cases it may be desirable to delay the start of DMA transfers until such a match occurs; in such cases, software can poll this bit, and once it becomes set, immediately select the desired DMA_PAGE to initiate DMA transfers. This bit is intended for use with DMA_PAGE=GEN4-PHY, but is available on all pages. Its usage is optional.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 DMA_EN	DMA Enable Setting DMA_EN=1 enables clock gating to the Transciever DMA engine. This bit should be set prior to the start of a DMA session, and remain set for the duration of the session.
22-12 DMA_START_DLY	DMA Start Trigger Delay When a start trigger is used (DMA_START_TRG not equal to 0), the DMA transfers will be delayed by DMA_START_DLY microseconds after the trigger occurs.
11-10 DMA_DEC	DMA Decimation Rate Controls whether every data is captured on every data valid strobe, or only every 2nd, 4th or 8th data valid strobe. 00b - Data is captured on every data valid 01b - Data is captured on every 2nd data valid 10b - Data is captured on every 4th data valid 11b - Data is captured on every 8th data valid
9 —	Reserved
8 DMA_START_EDGE	DMA Start Trigger Edge Selector DMA_START_EDGE selects the edge sensitivity (rising or falling) of the selected start-trigger. 0b - Trigger fires on a rising edge of the selected trigger source 1b - Trigger fires on a falling edge of the selected trigger source
7-4 DMA_START_TRG	DMA Start Trigger Selector The DMA Start Trigger, if desired, can be selected from the sources as shown below 0000b - no trigger 0001b - PHY: pd found 0010b - PHY: aa found 0011b - Zigbee_PHY: pd found 0100b - Zigbee_PHY: sfd detect 0101b - RXDIG: agc_gain_chg 0110b - TSM: rx_dig_en 0111b - TSM: tsm_irq0_start_trig 1000b - CRC pass 1001b - CRC done (Not used for 15.4LL) 1010b - Localization control: pattern match

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1011b - GenericLL: cte_present, Bluetooth LE: cte_en 1100b - Ranging sequence manager: dma_trigger
3-0 DMA_PAGE	Transceiver DMA Page Selector Selects the page of receiver data for storage to system memory whenusing Transceiver DMA Debug Mode. Setting this register to a non-zero value enablesthe DMA interface logic. Setting this register to zero disables the interface. 0000b - DMA idle 0001b - RXDIG-IQ: Select from (mixer, decimator, pd channel filter, src, cfo-mixer, demod chan filter) in RXDIG. 11bit signed data, MSB aligned 0010b - RXDIG-IQ-ALT: Same as above + signals on unused LSBs : {antenna switch, GenLL or 802.15.4LL CRC_FAIL} on "Q" LSBs, {preamble_found, aa_sfd_matched} on "I" LSBs. 0011b - ADC-IQ: 11bit samples are MSB aligned in each 16bit half-word 0100b - PHASE: Select from (sync phase, demod phase) within RXDIG. MSB aligned 0101b - RSSI-PHASE: select from 8bit Narrowband (rssi, rssi raw, lqi, snr, noise) and Wideband(rssi, rssi raw) + 8bit high-resolution PHASE 0110b - MAG-PHASE: RSSI magnitude + 8bit high-resolution PHASE 0111b - GEN4-PHY 1000b - DETERMINISTIC

55.4.7.12.2.6 DBG Ram control register (DBG_RAM_CTRL)

Offset

Register	Offset
DBG_RAM_CTRL	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DBG_STOP_TRG				DBG_RAM...	DBG_STO...	DBG_STA...	DBG_AA_...	DBG_EN	DBG_START_DLY						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DBG_START_DLY				DBG_DEC		DBG_STO...	DBG_STA...	DBG_START_TRG				DBG_SIG...	DBG_PAGE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 DBG_STOP_TRG	Packet RAM Debug Stop Trigger Selector The Packet RAM Debug Stop Trigger, if desired, can be selected from the sources as shown below 0000b - no trigger 0001b - PHY: pd found 0010b - PHY: aa found 0011b - Zigbee_PHY: pd found 0100b - Zigbee_PHY: sfd detect 0101b - RXDIG: agc_gain_chg 0110b - TSM: rx_dig_en 0111b - TSM: tsm_irq1_stop_trig 1000b - CRC fail 1001b - CRC done (Not used for 15.4LL) 1010b - RBME: error 1011b - GenericLL header fail 1100b - PLL unlock
27 DBG_RAM_FULL	DBG_RAM_FULL Status Bit indicating that logic has written to the DBG_RAM_LAST address when a Stop Trigger is not being use. In this case, any subsequent data will not alter the contents of the RAM. Set DBG_PAGE=0 to clear the DBG_RAM_FULL bit. 0b - Packet RAM is not full 1b - Packet RAM is full
26 DBG_STOP_TRIGGERED	DBG Stop Trigger Occurred This read-only status bit becomes set, when, during a DBG session (DBG_PAGE > 0), the trigger source selected by DBG_START_TRG[2:0] occurs, with the edge sensitivity selected by DBG_STOP_EDGE. This bit is cleared when the DBG is re-armed by programming DBG_PAGE to 0 and then to a non-zero value.
25 DBG_START_TRIGGERED	DBG Start Trigger Occurred This read-only status bit becomes set, when, during a DBG session (DBG_PAGE > 0), the trigger source selected by DBG_START_TRG[2:0] occurs, with the edge sensitivity selected by DBG_START_EDGE. This bit is cleared when the DBG is re-armed by programming DBG_PAGE to 0 and then to a non-zero value.
24 DBG_AA_TRIGGERED	DBG Access Address triggered This status bit becomes set when the PHY indicates that a Network Address match has occurred. In some cases it may be desirable to delay the start of DBG transfers until such a match occurs; in such cases, software can poll this bit, and once it becomes set, immediately select the desired DBG_PAGE to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	initiate DBG transfers. This bit is intended for use with DBG_PAGE=GEN4-PHY, but is available on all pages. Its usage is optional.
23 DBG_EN	DBG Enable Setting DBG_EN=1 enables clock gating to the Transciever DBG engine. This bit should be set prior to the start of a DBG session, and remain set for the duration of the session.
22-12 DBG_START_DLY	DBG Start Trigger Delay When a start trigger is used (DBG_START_TRG not equal to 0), the Packet RAM Debug transfers will be delayed by DBG_START_DLY microseconds after the trigger occurs.
11-10 DBG_DEC	DBG Decimation Rate Controls whether every data is captured on every data valid strobe, or only every 2nd, 4th or 8th data valid strobe. 00b - Data is captured on every data valid 01b - Data is captured on every 2nd data valid 10b - Data is captured on every 4th data valid 11b - Data is captured on every 8th data valid
9 DBG_STOP_EDGE	DBG Stop Trigger Edge Selector DBG_STOP_EDGE selects the edge sensitivity (rising or falling) of the selected start-trigger. 0b - Trigger stops on a rising edge of the selected trigger source 1b - Trigger stops on a falling edge of the selected trigger source
8 DBG_START_EDGE	DBG Start Trigger Edge Selector DBG_START_EDGE selects the edge sensitivity (rising or falling) of the selected start-trigger. 0b - Trigger fires on a rising edge of the selected trigger source 1b - Trigger fires on a falling edge of the selected trigger source
7-4 DBG_START_TRG	DMA Start Trigger Selector The DMA Start Trigger, if desired, can be selected from the sources as shown below 0000b - no trigger 0001b - PHY: pd found 0010b - PHY: aa found 0011b - Zigbee_PHY: pd found 0100b - Zigbee_PHY: sfd detect 0101b - RXDIG: agc_gain_chg 0110b - TSM: rx_dig_en 0111b - TSM: tsm_irq0_start_trig

Table continues on the next page...

Table continued from the previous page...

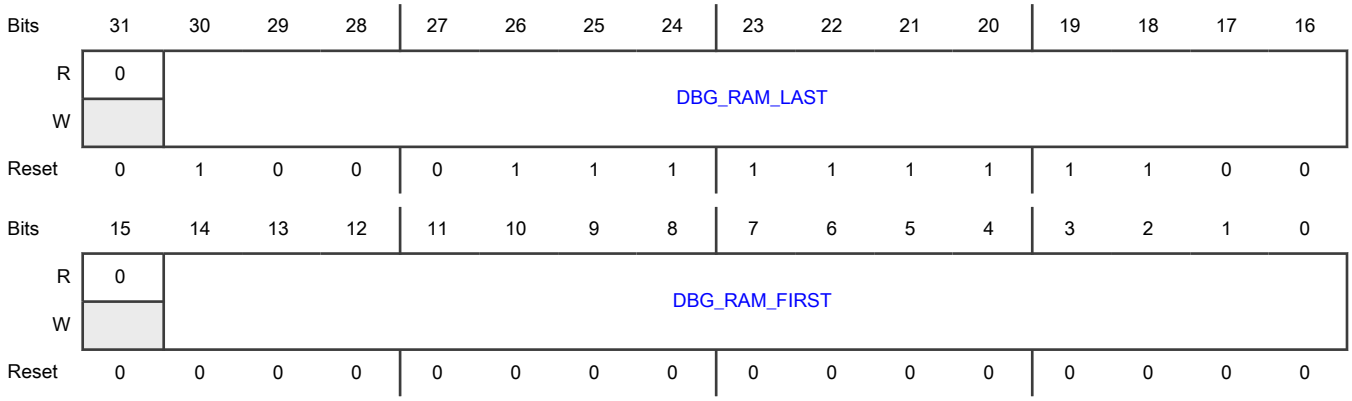
Field	Function
	1000b - CRC pass 1001b - CRC done (Not used for 15.4LL) 1010b - Localization control: pattern match 1011b - GenericLL: cte_present, Bluetooth LE: cte_en 1100b - Ranging sequence manager: dma_trigger
3 DBG_SIGNAL_VALID_MASK_EN	DBG Signal Valid Mask Enable **NOTE** This bit should not be set if RXDIG's RX_IQ_PH_OUTPUT_COND bit is set. 0b - Disable use of dbg_signal_valid_mask. 1b - Enable use of dbg_signal_valid_mask.
2-0 DBG_PAGE	Packet RAM Debug Page Selector Selects the page of receiver data for storage to Packet RAM using Transceiver Packet RAM Debug Mode. Setting this register to a non-zero value enablesthe Packet RAM Debug Mode interface logic. Setting this register to zero disables the interface. 000b - DMA idle 001b - RXDIG-IQ: Select from (mixer, decimator, pd channel filter, src, cfo-mixer, demod chan filter) in RXDIG. 11bit signed data, MSB aligned 010b - RXDIG-IQ-ALT: Same as above + signals on unused LSBs : {antenna switch, GenLL or 802.15.4LL CRC_FAIL} on "Q" LSBs, {preamble_found, aa_sfd_matched} on "I" LSBs. 011b - ADC-IQ: 11bit samples are MSB aligned in each 16bit half-word 100b - PHASE: Select from (sync phase, demod phase) within RXDIG. MSB aligned 101b - RSSI-PHASE: select from 8bit Narrowband (rssi, rssi raw, lqi, snr, noise) and Wideband(rssi, rssi raw) + 8bit high-resolution PHASE 110b - MAG-PHASE: RSSI magnitude + 8bit high-resolution PHASE 111b - GEN4-PHY

55.4.7.12.2.7 DBG RAM ADDRESS (DBG_RAM_ADDR)

Offset

Register	Offset
DBG_RAM_ADDR	14h

Diagram



Fields

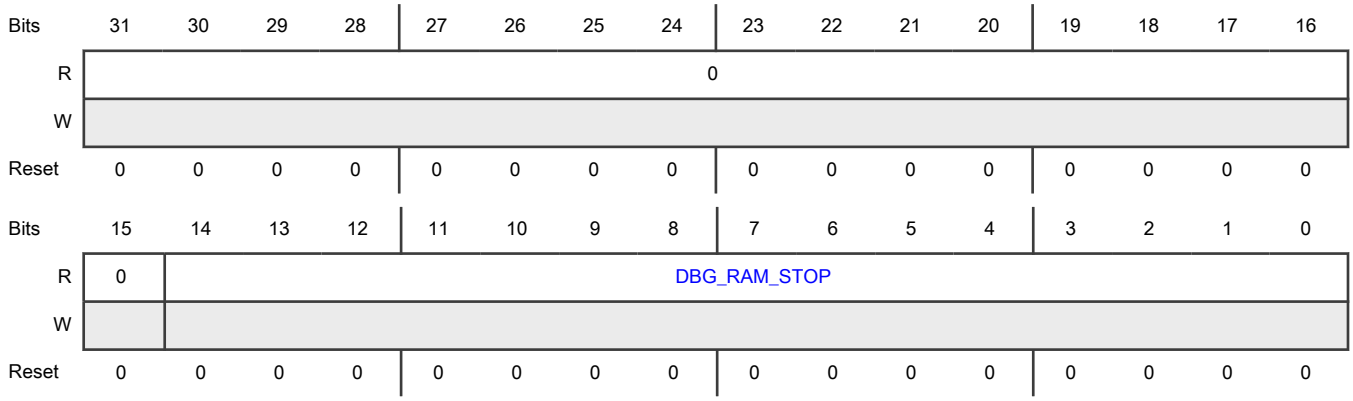
Field	Function
31 —	Reserved
30-16 DBG_RAM_LAST	DBG RAM Last Address Represents the last valid RAM address which can be written. The 2 LSBs should always be 0. The TX_PACKET_RAM is accessed from address 0x0000 to 0x0FFC, and the RX_PACKET_RAM from address 0x1000 to 0x17FC. Through programming of DBG_RAM_FIRST and DBG_RAM_LAST, data can be captured solely to the TX_PACKET_RAM, solely to the RX_PACKET_RAM, or it can span the TX_PACKET_RAM and RX_PACKET_RAM spaces. The only restrictions are that DBG_RAM_LAST should be greater than DBG_RAM_FIRST, and DBG_RAM_LAST should be no larger than the last address of the RX_PACKET_RAM (0x17FC).
15 —	Reserved
14-0 DBG_RAM_FIRST	DBG RAM First Address Represents the first RAM address which will be written. The 2 LSBs should always be 0. The TX_PACKET_RAM is accessed from address 0x0000 to 0x0FFC, and the RX_PACKET_RAM from address 0x1000 to 0x17FC. Through programming of DBG_RAM_FIRST and DBG_RAM_LAST, data can be captured solely to the TX_PACKET_RAM, solely to the RX_PACKET_RAM, or it can span the TX_PACKET_RAM and RX_PACKET_RAM spaces. The only restrictions are that DBG_RAM_LAST should be greater than DBG_RAM_FIRST, and DBG_RAM_LAST should be no larger than the last address of the RX_PACKET_RAM (0x17FC).

55.4.7.12.2.8 DBG RAM STOP ADDRESS (DBG_RAM_STOP_ADDR)

Offset

Register	Offset
DBG_RAM_STOP_ADDR	18h

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 DBG_RAM_STOP	DBG RAM Stop Address After a Packet RAM Debug Mode acquisition, RAM_STOP_ADDR represents the last (most recent) RAM location filled with sample data before the acquision terminated. The acquisition terminates when a stop-trigger occurs, or due to a manual termination by setting DBG_PAGE=0.

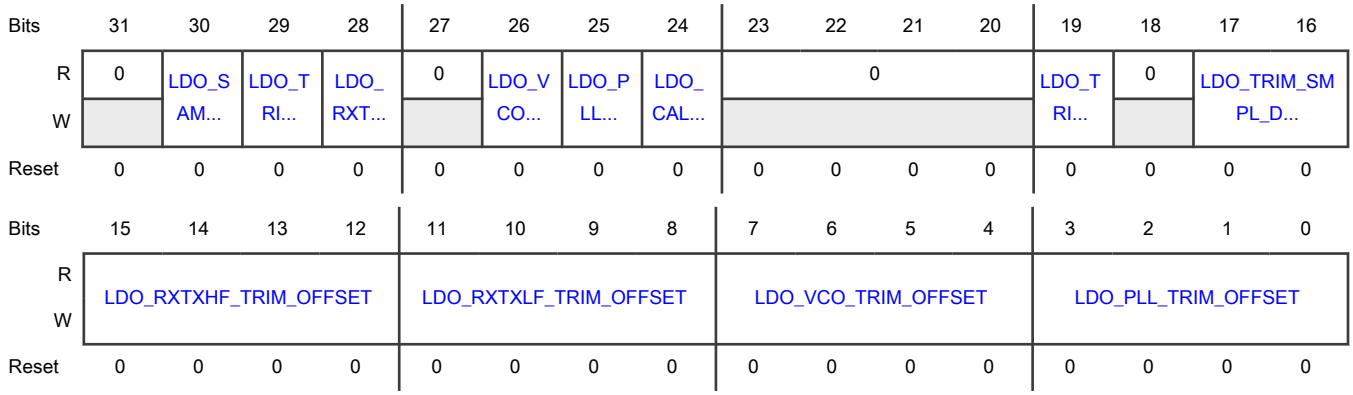
55.4.7.12.2.9 LDO TRIM Configuration 0 (LDO_TRIM_0)

Offset

Register	Offset
LDO_TRIM_0	1Ch

Function
LDO_TRIM_0

Diagram



Fields

Field	Function
31 —	Reserved
30 LDO_SAMPLE_TRIMSEL_OVRD_EN	LDO SAMPLE TRIMSEL Override Enable If set, the sample and trimsel signals are overridden with values in LDO_TRIM_SAMPLE_OVRD, LDO_RXTXHF_TRIMSEL_OVRD, LDO_PLL_TRIMSEL_OVRD, LDO_VCO_TRIMSEL_OVRD, and LDO_CAL_TRIMSEL_OVRD bits.
29 LDO_TRIM_SAMPLE_OVRD	LDO_TRIM_SAMPLE Override Value If LDO_SAMPLE_TRIMSEL_OVRD_EN is set, the LDO_TRIM_SAMPLE signal is overridden with this value.
28 LDO_RXTXHF_TRIMSEL_OVRD	LDO_RXTXHF_TRIMSEL Override Value If LDO_SAMPLE_TRIMSEL_OVRD_EN is set, the LDO_RXTXHF_TRIMSEL signal is overridden with this value.
27 —	Reserved
26 LDO_VCO_TRIMSEL_OVRD	LDO_VCO_TRIMSEL Override Value If LDO_SAMPLE_TRIMSEL_OVRD_EN is set, the LDO_VCO_TRIMSEL signal is overridden with this value.
25 LDO_PLL_TRIMSEL_OVRD	LDO_PLL_TRIMSEL Override Value If LDO_SAMPLE_TRIMSEL_OVRD_EN is set, the LDO_PLL_TRIMSEL signal is overridden with this value.
24 LDO_CAL_TRIMSEL_OVRD	LDO_CAL_TRIMSEL Override Value If LDO_SAMPLE_TRIMSEL_OVRD_EN is set, the LDO_CAL_TRIMSEL signal is overridden with this value.
23-20 —	Reserved
19 LDO_TRIM_CMPOUT_INV	LDO TRIM CMPOUT Invert If set, the LDO trim comparator output signal from the analog is inverted before it is used by the calibration logic.
18 —	Reserved
17-16	LDO TRIM Sample Delay Specifies when the LDO trim comparator output signal from the analog is captured.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LDO_TRIM_SMPL_DLY	
15-12 LDO_RXTXHF_TRIM_OFFSET	LDO RXTXHF TRIM Offset Signed offset applied to the RXTXHF trim calculated by the calibration logic
11-8 LDO_RXTXLF_TRIM_OFFSET	LDO RXTXLF TRIM Offset Signed offset applied to the RXTXLF trim calculated by the calibration logic.
7-4 LDO_VCO_TRIM_OFFSET	LDO VCO TRIM Offset Signed offset applied to the VCO trim calculated by the calibration logic. The resolution of this offset is 4 codes: this signed value is arithmetically left-shifted by 2bits before it is added to VCO trim calculated by the calibrated logic.
3-0 LDO_PLL_TRIM_OFFSET	LDO PLL TRIM Offset Signed offset applied to the PLL trim calculated by the calibration logic.

55.4.7.12.2.10 LDO TRIM Configuration 1 (LDO_TRIM_1)

Offset

Register	Offset
LDO_TRIM_1	20h

Function
LDO_TRIM_1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	LDO_RXT...	LDO_RXTXHF_TRIM_OVRD						0	LDO_RXT...	LDO_RXTXLF_TRIM_OVRD					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	LDO_VCO...	LDO_VCO_TRIM_OVRD						0	LDO_PLL...	LDO_PLL_TRIM_OVRD					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 LDO_RXTXHF_TRIM_OVRD_EN	LDO RXTXHF TRIM Override Enable If set, the LDO_RXTXHF_TRIM_OVRD is used for the RXTXHF TRIM instead of the value determined by the calibration logic
29-24 LDO_RXTXHF_TRIM_OVRD	LDO RXTXHF TRIM Override Value If LDO_RXTXHF_TRIM_OVRD_EN=1, this value is used for the LDO RXTXHF TRIM.
23 —	Reserved
22 LDO_RXTXLF_TRIM_OVRD_EN	LDO RXTXLF TRIM Override Enable If set, the LDO_RXTXLF_TRIM_OVRD is used for the RXTXLF TRIM instead of the value determined by the calibration logic
21-16 LDO_RXTXLF_TRIM_OVRD	LDO RXTXLF TRIM Override Value If LDO_RXTXLF_TRIM_OVRD_EN=1, this value is used for the LDO RXTXLF TRIM.
15 —	Reserved
14 LDO_VCO_TRIM_OVRD_EN	VCO TRIM Override Enable If set, the VCO_TRIM_OVRD is used for the VCO TRIM instead of the value determined by the calibration logic.
13-8 LDO_VCO_TRIM_OVRD	LDO VCO TRIM Override Value If LDO_VCO_TRIM_OVRD_EN=1, this value is used for the LDO VCO TRIM.
7 —	Reserved
6 LDO_PLL_TRIM_OVRD_EN	LDO PLL TRIM Override Enable If set, the LDO_PLL_TRIM_OVRD is used for the PLL TRIM instead of the value determined by the calibration logic.
5-0 LDO_PLL_TRIM_OVRD	LDO PLL TRIM Override Value If LDO_PLL_TRIM_OVRD_EN=1, this value is used for the LDO PLL TRIM.

55.4.7.12.2.11 RF Analog LDO Trim Res Control 0 (LDO_TRIM_RES_0)

Offset

Register	Offset
LDO_TRIM_RES_0	24h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		LDO_RXTXHF_TRIM						0		LDO_RXTXLF_TRIM					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		LDO_VCO_TRIM						0		LDO_PLL_TRIM					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-24 LDO_RXTXHF_TRIM	LDO_RXTXHF_TRIM Result LDO RXTXHF Trim value
23-22 —	Reserved
21-16 LDO_RXTXLF_TRIM	LDO_RXTXLF_TRIM Result LDO Rxtxlf Trim value
15-14 —	Reserved
13-8 LDO_VCO_TRIM	LDO_VCO_TRIM Result LDO Vco Trim value
7-6	Reserved

Table continues on the next page...

Table continued from the previous page...

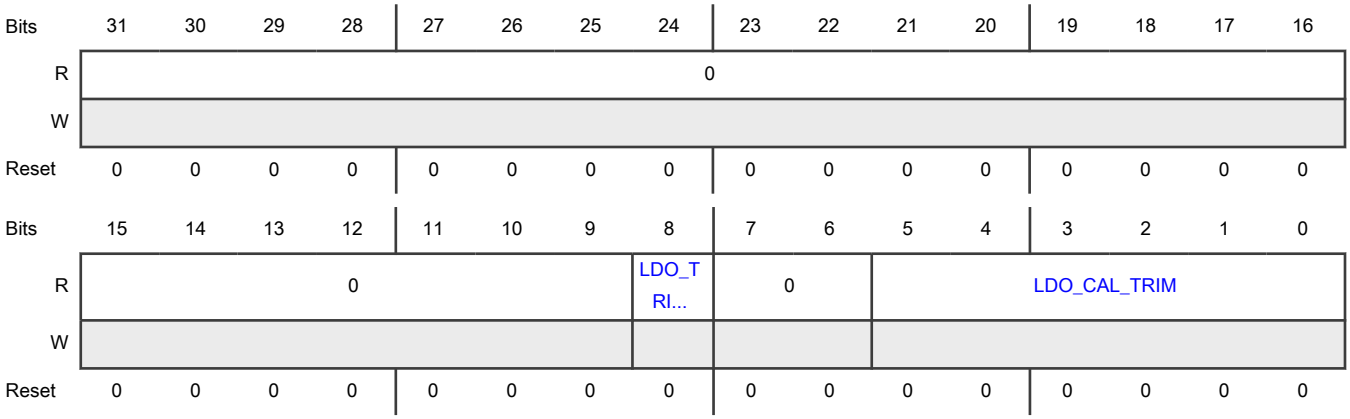
Field	Function
—	
5-0 LDO_PLL_TRIM	LDO_PLL_TRIM Result LDO Pll Trim value

55.4.7.12.2.12 RF Analog LDO Trim Res Control 1 (LDO_TRIM_RES_1)

Offset

Register	Offset
LDO_TRIM_RES_1	28h

Diagram



Fields

Field	Function
31-9 —	Reserved
8 LDO_TRIM_CMPOUT	LDO TRIM CMPOUT Instantaneous value of the LDO_TRIM_CMPOUT signal
7-6 —	Reserved
5-0	LDO_CAL_TRIM Result

Table continues on the next page...

Table continued from the previous page...

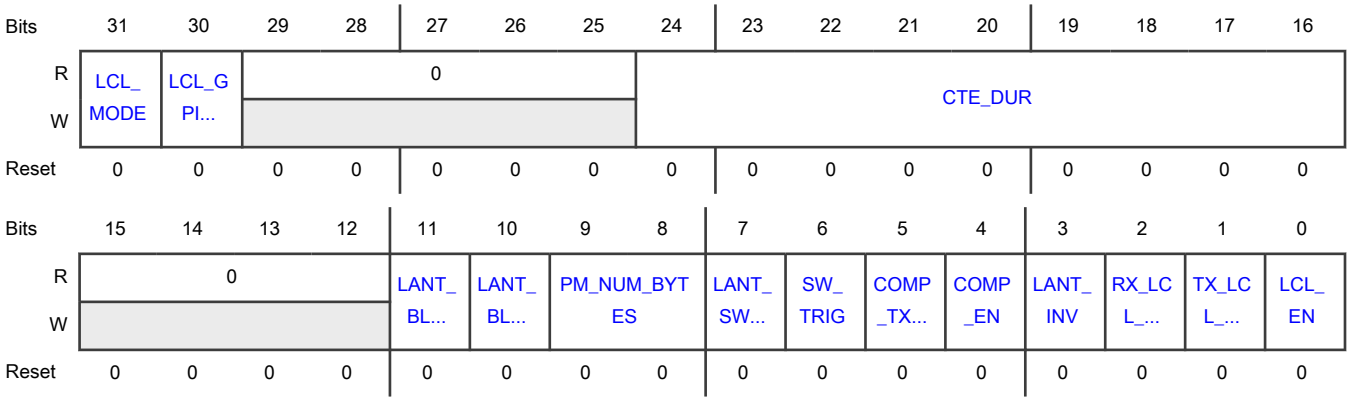
Field	Function
LDO_CAL_TRIM	LDO Cal Trim Value

55.4.7.12.2.13 LCL CTRL CFG 0 (LCL_CFG0)

Offset

Register	Offset
LCL_CFG0	2Ch

Diagram



Fields

Field	Function
31 LCL_MODE	Localization Mode Configure localization control module for GenLL or Bluetooth LE LL. 0b - GenLL configuration. 1b - Bluetooth LE LL configuration.
30 LCL_GPIO_SEL	Localization GPIO Select Reserved. This bit should be programmed to 0
29-25 —	Reserved
24-16 CTE_DUR	Total Switching Duration LCL antenna switching will continue for this many doublets, where a doublet is (RX/TX_HI_PER + RX/TX_LO_PER).

Table continues on the next page...

Table continued from the previous page...

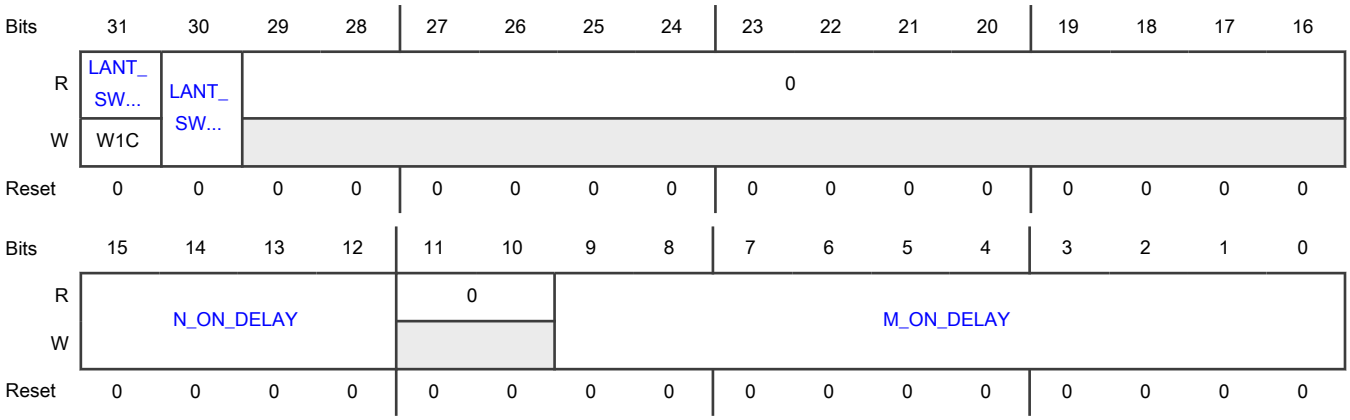
Field	Function
15-12 —	Reserved
11 LANT_BLOCK_RX	Block LANT_SW for RX Setting this bit will cause the Localization Control Module to block its LANT_SW output, for RX. NOTE: the PHY's ant_out will still pass through on the LANT_SW output.
10 LANT_BLOCK_TX	Block LANT_SW for TX Setting this bit will cause the Localization Control Module to block its LANT_SW output, for TX. NOTE: the PHY's ant_out will still pass through on the LANT_SW output.
9-8 PM_NUM_BYTES	Number of Bytes to Match This field indicates how many bytes are used to compare in the bit stream for a pattern match. 00b - 4 bytes 01b - 5 bytes 10b - 6 bytes 11b - 8 bytes
7 LANT_SW_WIGGLE	LANT_SW Wiggle If set, 3 clock pulses will be driven on LANT_SW after the localization control module is triggered, to initialize the SOC TPM.
6 SW_TRIG	Software Trigger. Can be used with either RX or TX Setting this bit will cause a trigger event in RX and TX, depending on the trigger select configuration.
5 COMP_TX_EN	Pattern Matching Enable in TX Normally, pattern matching is used for RX. If it is desired for TX, as well, this bit should be set.
4 COMP_EN	Pattern Matching Enable This enables the pattern matching feature.
3 LANT_INV	Invert Antenna Switch Output
2 RX_LCL_EN	Enable Switching in RX
1 TX_LCL_EN	Enable Switching in TX
0 LCL_EN	Localization Control Module Enable Note: This bit, when 0, will still allow the PHY FAD antenna switch signal to propagate.

55.4.7.12.2.14 LCL CTRL CFG 1 (LCL_CFG1)

Offset

Register	Offset
LCL_CFG1	30h

Diagram



Fields

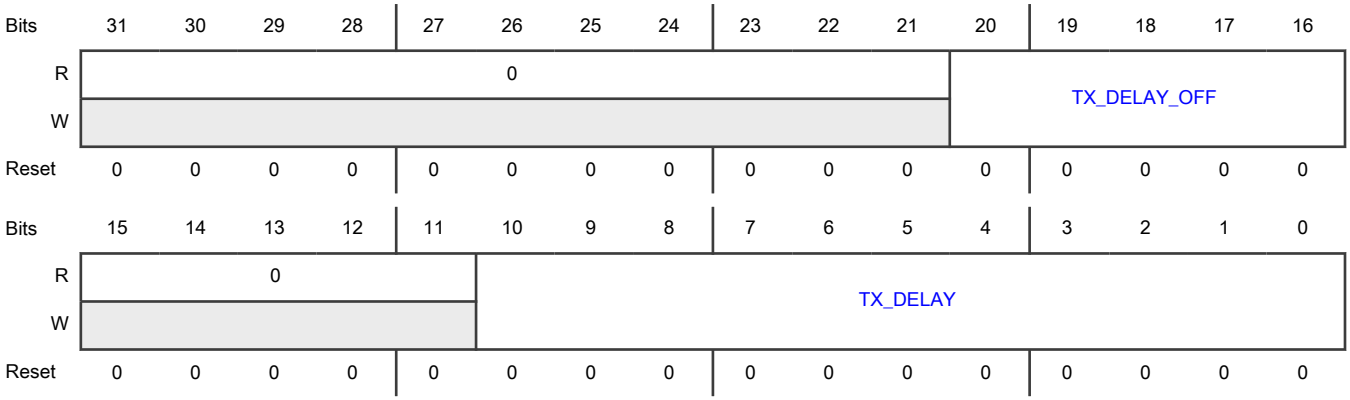
Field	Function
31 LANT_SW_FLAG	Localization Antenna Switch Flag The bit is set when a rising-edge on lant_sw occurs. Write 1 to clear the bit.
30 LANT_SW_IE	Localization Antenna Switch Interrupt Enable Configures the radio to generate an interrupt when the LANT_SW_FLAG is set. 0b - Localization Antenna Switch interrupt disabled 1b - Localization Antenna Switch interrupt enabled
29-16 —	Reserved
15-12 N_ON_DELAY	N on Delay rx/tx_lcl_on will de-assert M_ON_DELAY intervals + N_ON_DELAY doublets earlier than the antenna switching stops.
11-10 —	Reserved
9-0 M_ON_DELAY	M on Delay rx/tx_lcl_on will de-assert M_ON_DELAY intervals + N_ON_DELAY doublets earlier than the antenna switching stops.

55.4.7.12.2.15 LCL CTRL TX CONFIG0 (LCL_TX_CFG0)

Offset

Register	Offset
LCL_TX_CFG0	34h

Diagram



Fields

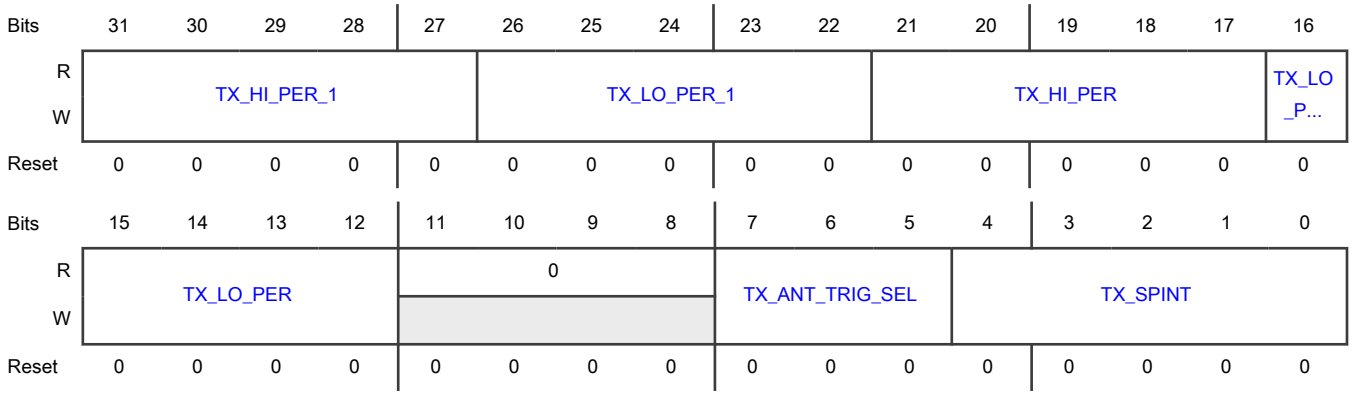
Field	Function
31-21 —	Reserved
20-16 TX_DELAY_OFF	Fine sample delay after TX_DELAY. This field alows for fine tuning the switching start point by several samples after TX_DELAY.
15-11 —	Reserved
10-0 TX_DELAY	Interval delay before TX switching begins. After a legitimate trigger event in TX, antenna switching will begin this many intervals later, where an interval is defined as TX_SPINT samples.

55.4.7.12.2.16 LCL CTRL TX CONFIG1 (LCL_TX_CFG1)

Offset

Register	Offset
LCL_TX_CFG1	38h

Diagram



Fields

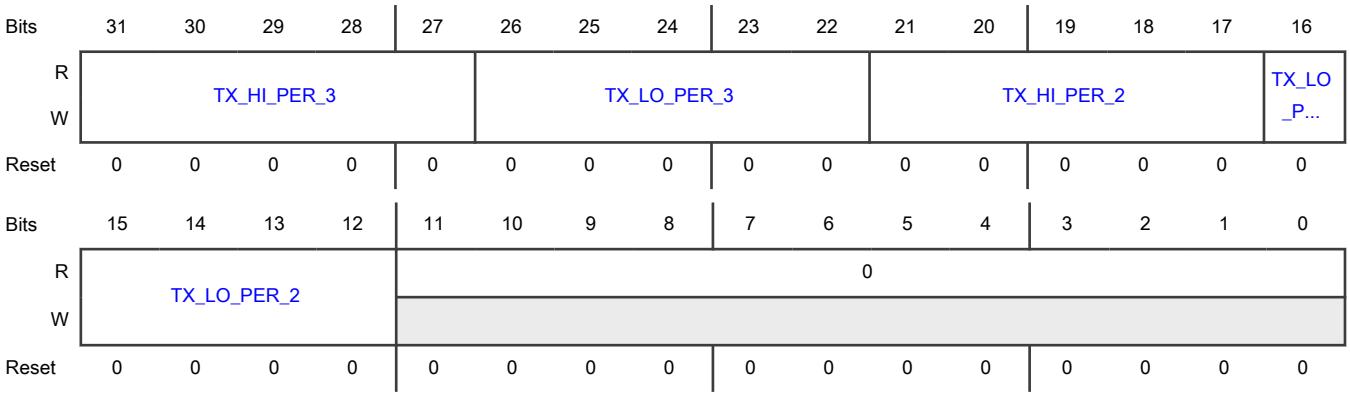
Field	Function
31-27 TX_HI_PER_1	Alternate Number of intervals for antenna HIGH. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 1, when switching is active. A value of 0 is not recommened when TX switching is enabled.
26-22 TX_LO_PER_1	Alternate Number of intervals for antenna LOW. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 0, when switching is active. A value of 0 is not recommened when TX switching is enabled.
21-17 TX_HI_PER	Primary Number of intervals for antenna HIGH Number of intervals for which lant_sw should be 1, when switching is active. A value of 0 is not recommended when TX switching is enabled.
16-12 TX_LO_PER	Primary Number of intervals for antenna LOW Number of intervals for which lant_sw should be 0, when switching is active. A value of 0 is not recommended when TX switching is enabled.
11-8 —	Reserved
7-5 TX_ANT_TRIG_SEL	Selects Trigger for TX 000b - Software Trigger 001b - LCL Pattern Found 010b - CRC Complete 011b - PA Warmup Complete 100b - RBME tx_done_pre 101b - Bluetooth LE cte_en 110b - Ranging sequence manager lcl_tx_trigger
4-0 TX_SPINT	Number of TX Samples that define the length of an Interval, where 0=1sample, 1=2sample, etc.

55.4.7.12.2.17 LCL CTRL TX CONFIG2 (LCL_TX_CFG2)

Offset

Register	Offset
LCL_TX_CFG2	3Ch

Diagram



Fields

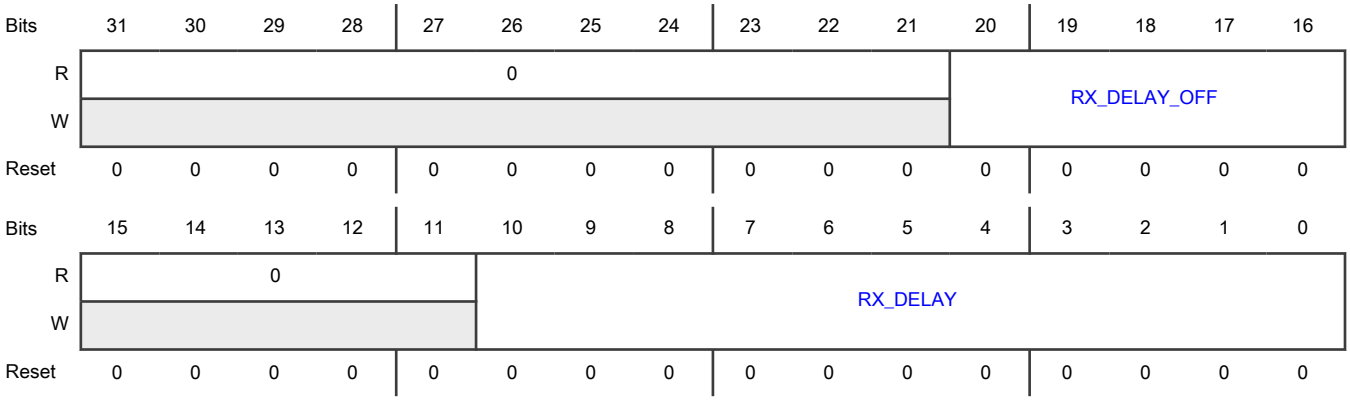
Field	Function
31-27 TX_HI_PER_3	Alternate Number of intervals for antenna HIGH. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 1, when switching is active. A value of 0 is not recommened when TX switching is enabled.
26-22 TX_LO_PER_3	Alternate Number of intervals for antenna LOW. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 0, when switching is active. A value of 0 is not recommened when TX switching is enabled.
21-17 TX_HI_PER_2	Alternate Number of intervals for antenna HIGH. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 1, when switching is active. A value of 0 is not recommened when TX switching is enabled.
16-12 TX_LO_PER_2	Alternate Number of intervals for antenna LOW. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 0, when switching is active. A value of 0 is not recommened when TX switching is enabled.
11-0 —	Reserved

55.4.7.12.2.18 LCL CTRL RX CONFIG0 (LCL_RX_CFG0)

Offset

Register	Offset
LCL_RX_CFG0	40h

Diagram



Fields

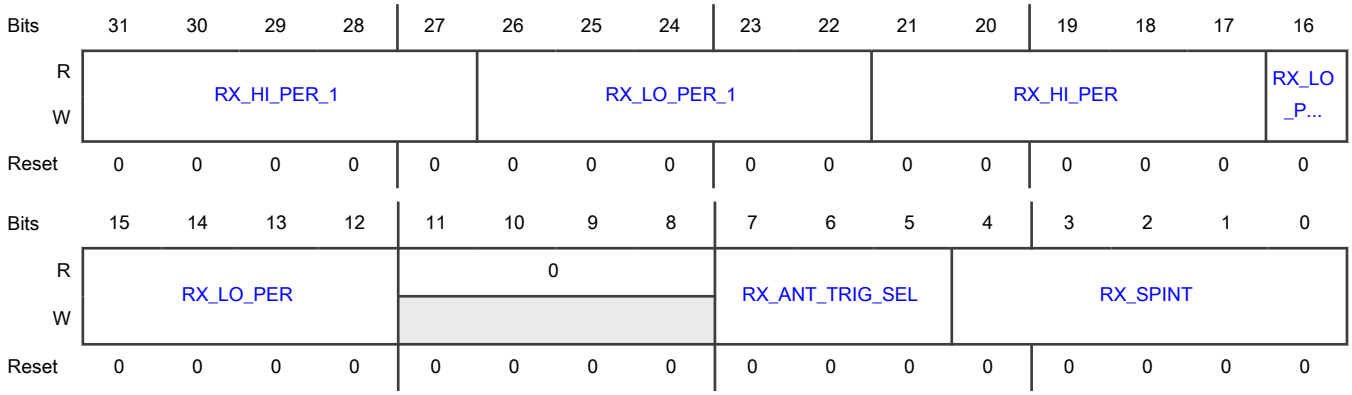
Field	Function
31-21 —	Reserved
20-16 RX_DELAY_OFF	Fine sample delay after RX_DELAY. This field alows for fine tuning the switching start point by several samples after RX_DELAY.
15-11 —	Reserved
10-0 RX_DELAY	Interval delay before RX switching begins. After a legitimate trigger event in RX, antenna switching will begin this many intervals later, where an interval is defined as RX_SPINT samples.

55.4.7.12.2.19 LCL CTRL RX CONFIG1 (LCL_RX_CFG1)

Offset

Register	Offset
LCL_RX_CFG1	44h

Diagram



Fields

Field	Function
31-27 RX_HI_PER_1	Alternate Number of intervals for antenna HIGH. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 1, when switching is active. It also indicates the number of intervals that the dma_mask should be 1. A value of 0 is not recommended when RX switching is enabled.
26-22 RX_LO_PER_1	Alternate Number of intervals for antenna LOW. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 0, when switching is active. It also indicates the number of intervals that the dma_mask should be 0. A value of 0 is not recommended when RX switching is enabled.
21-17 RX_HI_PER	Primary Number of intervals for antenna HIGH Number of intervals for which lant_sw should be 1, when switching is active. It also indicates the number of intervals that the dma_mask should be 1. A value of 0 is not recommended when RX switching is enabled.
16-12 RX_LO_PER	Primary Number of intervals for antenna LOW Number of intervals for which lant_sw should be 0, when switching is active. It also indicates the number of intervals that the dma_mask should be 0. A value of 0 is not recommened when RX switching is enabled.
11-8 —	Reserved
7-5 RX_ANT_TRIG_SEL	Selects Trigger for RX 000b - Software Trigger 001b - LCL Pattern Found 010b - CRC Complete 011b - CRC Pass 100b - GenericLL: cte_present, Bluetooth LE: cte_en

Table continues on the next page...

Table continued from the previous page...

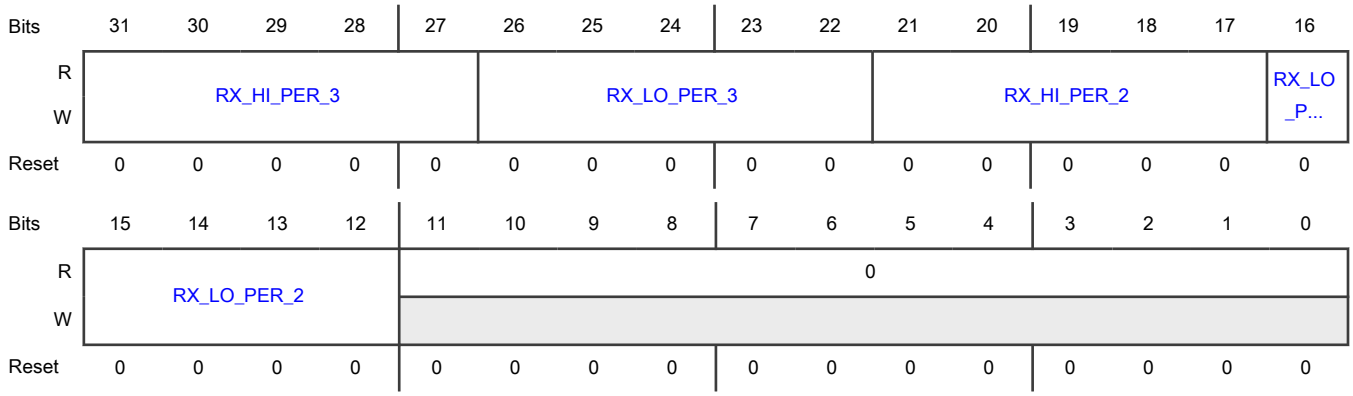
Field	Function
	101b - aa_fnd_to_ll 110b - Ranging sequence manager lcl_rx_trigger
4-0 RX_SPINT	Number of RX Samples that define the length of an Interval, where 0=1sample, 1=2sample, etc.

55.4.7.12.2.20 LCL CTRL RX CONFIG2 (LCL_RX_CFG2)

Offset

Register	Offset
LCL_RX_CFG2	48h

Diagram



Fields

Field	Function
31-27 RX_HI_PER_3	Alternate Number of intervals for antenna HIGH. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 1, when switching is active. It also indicates the number of intervals that the dma_mask should be 1. A value of 0 is not recommended when RX switching is enabled.
26-22 RX_LO_PER_3	Alternate Number of intervals for antenna LOW. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 0, when switching is active. It also indicates the number of intervals that the dma_mask should be 0. A value of 0 is not recommended when RX switching is enabled.
21-17 RX_HI_PER_2	Alternate Number of intervals for antenna HIGH. Used only in conjunction with RSM.

Table continues on the next page...

Table continued from the previous page...

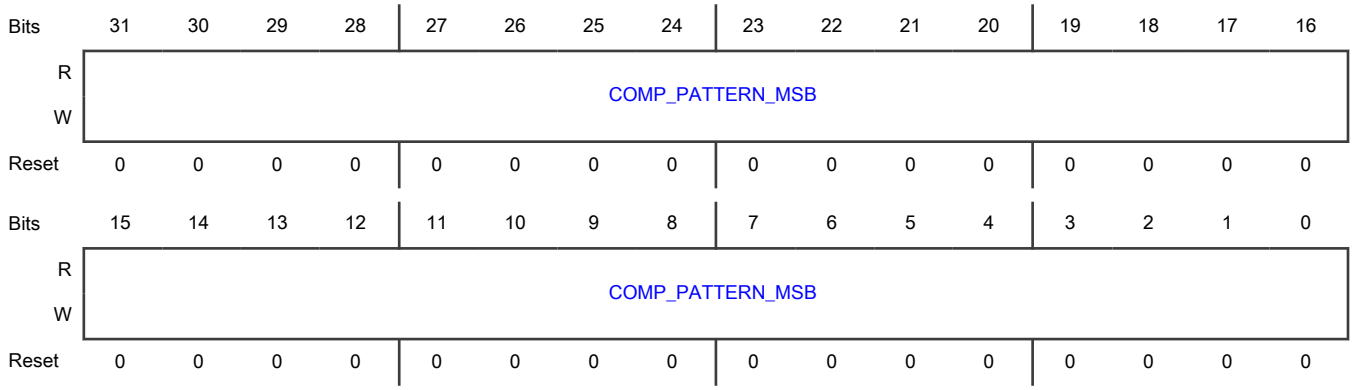
Field	Function
	Number of intervals for which lant_sw should be 1, when switching is active. It also indicates the number of intervals that the dma_mask should be 1. A value of 0 is not recommended when RX switching is enabled.
16-12 RX_LO_PER_2	Alternate Number of intervals for antenna LOW. Used only in conjunction with RSM. Number of intervals for which lant_sw should be 0, when switching is active. It also indicates the number of intervals that the dma_mask should be 0. A value of 0 is not recommended when RX switching is enabled.
11-0 —	Reserved

55.4.7.12.2.21 LCL CTRL PM MSB (LCL_PM_MSB)

Offset

Register	Offset
LCL_PM_MSB	4Ch

Diagram



Fields

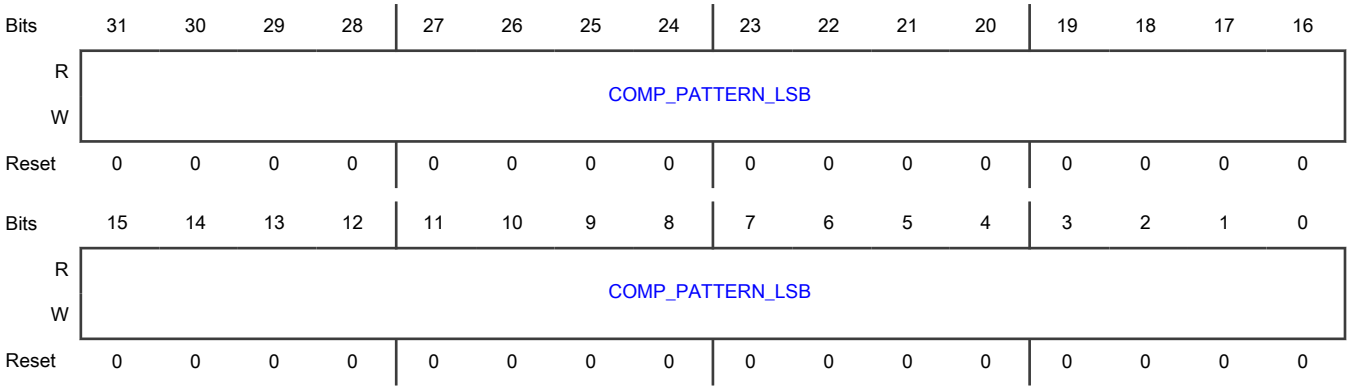
Field	Function
31-0 COMP_PATTERN_MSB	Upper bytes of pattern to be matched, bits 63:32

55.4.7.12.2.22 LCL CTRL PM LSB (LCL_PM_LSB)

Offset

Register	Offset
LCL_PM_LSB	50h

Diagram



Fields

Field	Function
31-0 COMP_PATTERN_LSB	Lower bytes of pattern to be matched, bits 31:0

55.4.7.12.2.23 LCL GPIO CTRL 0 (LCL_GPIO_CTRL0)

Offset

Register	Offset
LCL_GPIO_CTRL0	54h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LUT_7				LUT_6				LUT_5				LUT_4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LUT_3				LUT_2				LUT_1				LUT_0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 LUT_7	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
27-24 LUT_6	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
23-20 LUT_5	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
19-16 LUT_4	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
15-12 LUT_3	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
11-8 LUT_2	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
7-4 LUT_1	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
3-0 LUT_0	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).

55.4.7.12.2.24 LCL GPIO CTRL 1 (LCL_GPIO_CTRL1)

Offset

Register	Offset
LCL_GPIO_CTRL1	58h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LUT_15				LUT_14				LUT_13				LUT_12			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LUT_11				LUT_10				LUT_9				LUT_8			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 LUT_15	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
27-24 LUT_14	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
23-20 LUT_13	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
19-16 LUT_12	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
15-12 LUT_11	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
11-8 LUT_10	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 LUT_9	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
3-0 LUT_8	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).

55.4.7.12.2.25 LCL GPIO CTRL 2 (LCL_GPIO_CTRL2)

Offset

Register	Offset
LCL_GPIO_CTRL2	5Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LUT_23				LUT_22				LUT_21				LUT_20			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LUT_19				LUT_18				LUT_17				LUT_16			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 LUT_23	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
27-24 LUT_22	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
23-20 LUT_21	GPIO antenna state LUT entry

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
19-16 LUT_20	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
15-12 LUT_19	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
11-8 LUT_18	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
7-4 LUT_17	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
3-0 LUT_16	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).

55.4.7.12.2.26 LCL GPIO CTRL 3 (LCL_GPIO_CTRL3)

Offset

Register	Offset
LCL_GPIO_CTRL3	60h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LUT_31				LUT_30				LUT_29				LUT_28			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LUT_27				LUT_26				LUT_25				LUT_24			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

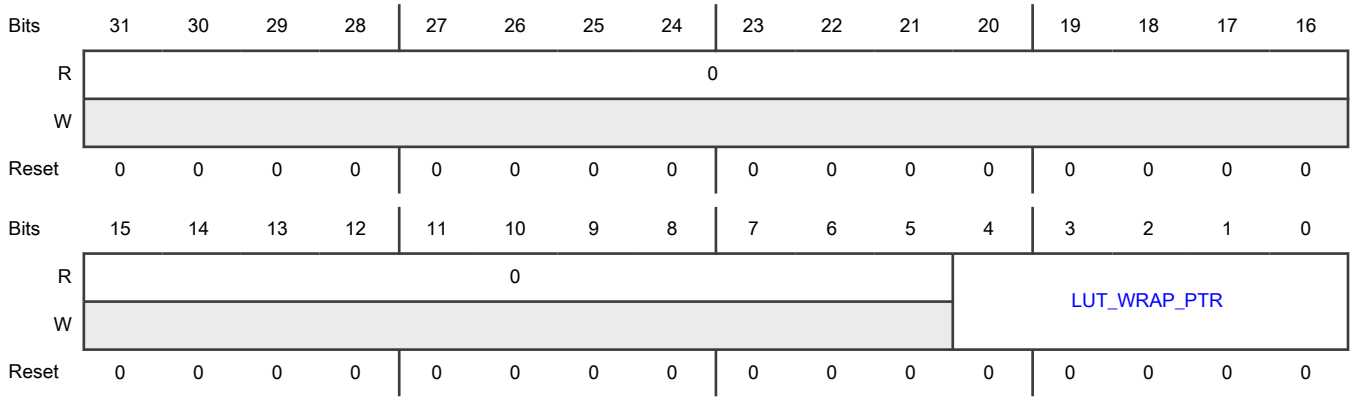
Field	Function
31-28 LUT_31	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
27-24 LUT_30	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
23-20 LUT_29	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
19-16 LUT_28	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
15-12 LUT_27	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
11-8 LUT_26	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
7-4 LUT_25	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).
3-0 LUT_24	GPIO antenna state LUT entry Each entry corresponds to a state of the 4 GPIOs, set at the beginning of each switch period (each doublet).

55.4.7.12.2.27 LCL GPIO CTRL 4 (LCL_GPIO_CTRL4)

Offset

Register	Offset
LCL_GPIO_CTRL4	64h

Diagram



Fields

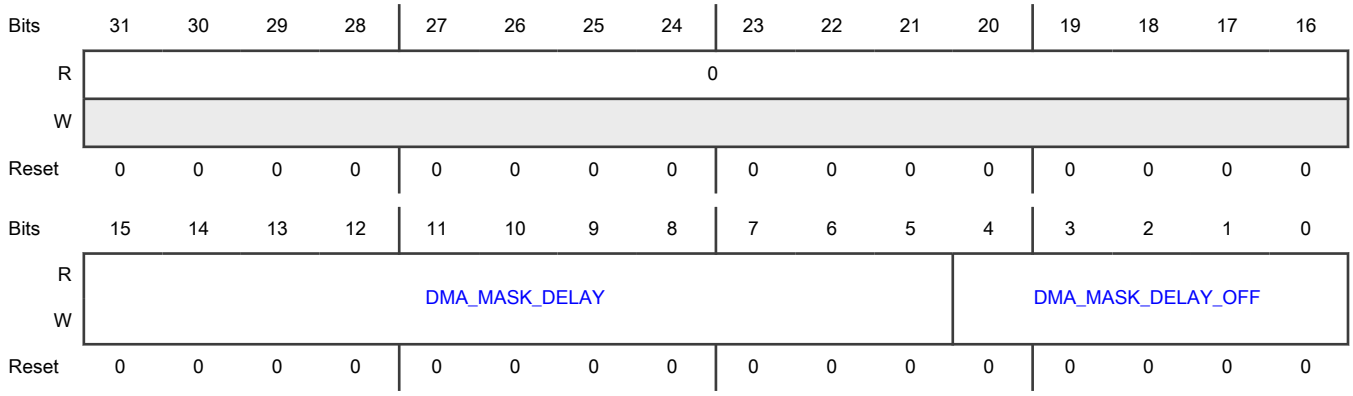
Field	Function
31-5 —	Reserved
4-0 LUT_WRAP_PTR	Wrap point for the LUT table in generating the 4 antenna GPIO wire states. The programmed value represents the last LUT entry that will be used before wrapping back to use LUT_0; if programmed to value "N", when the LCL_CTRL is triggered the output will be LUT_0, ..., LUT_N, LUT_0, ... etc. If programmed to "0", only LUT_0 will be used.

55.4.7.12.2.28 LCL_DMA_MASK_DELAY (LCL_DMA_MASK_DELAY)

Offset

Register	Offset
LCL_DMA_MASK_DELAY	68h

Diagram



Fields

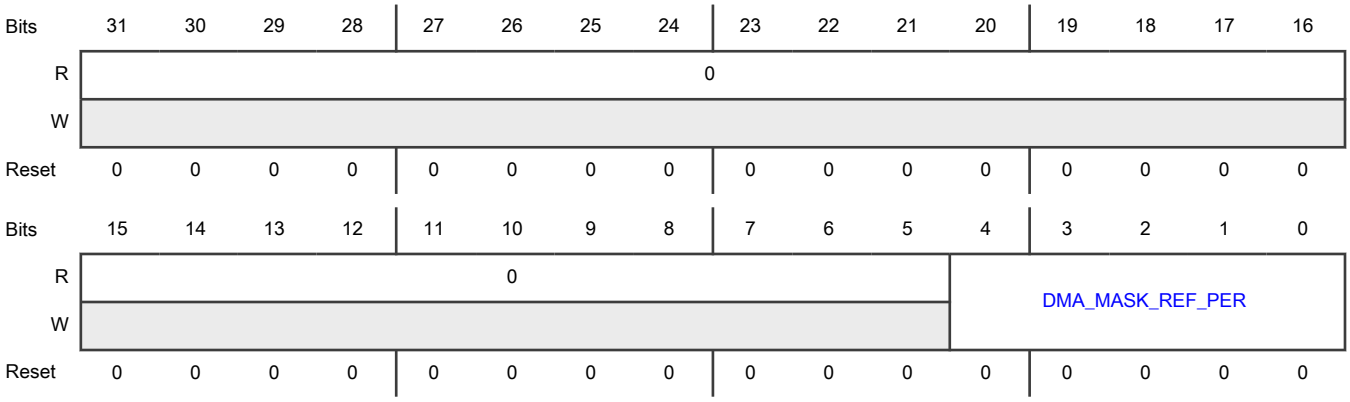
Field	Function
31-16 —	Reserved
15-5 DMA_MASK_DELAY	DMA_MASK_DELAY Coarse (intervals) offset delay from the trigger to the time at which DMA mask asserts.
4-0 DMA_MASK_DELAY_OFF	DMA_MASK_DELAY_OFF Fine (samples) offset delay from the trigger to the time at which DMA mask asserts.

55.4.7.12.2.29 LCL_DMA_MASK_PERIOD (LCL_DMA_MASK_PERIOD)

Offset

Register	Offset
LCL_DMA_MASK_PERIOD	6Ch

Diagram



Fields

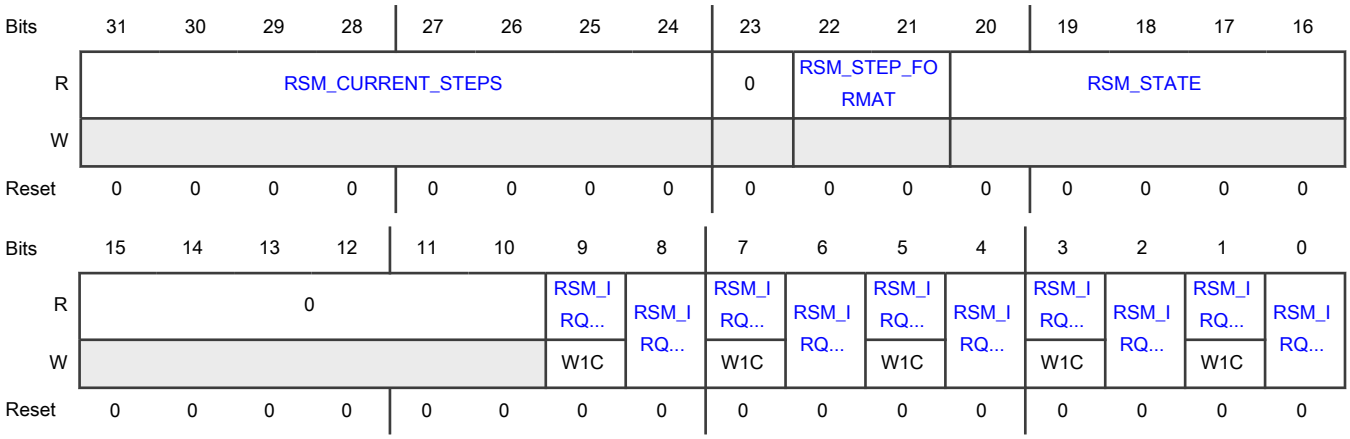
Field	Function
31-5 —	Reserved
4-0 DMA_MASK_REF_PER	DMA_MASK_REF_PER Duration (intervals) of the reference period, during which dma_mask remains asserted.

55.4.7.12.2.30 Ranging Sequence Manager Control and Status (RSM_CSR)

Offset

Register	Offset
RSM_CSR	70h

Diagram



Fields

Field	Function
31-24 RSM_CURRENT_STEPS	RSM_CURRENT_STEPS The current frequency step.
23 —	Reserved
22-21 RSM_STEP_FORMAT	RSM_STEP_FORMAT The current step format.
20-16 RSM_STATE	RSM_STATE The current state of the module. 0_0000b - IDLE 0_0001b - DELAY. Used only for the trigger delay in SQTE 0_0010b - EXT_TX (Extend TX). Used only for PDE 0_0011b - EXT_RX (Extend RX). Used only for PDE 0_0100b - WU (Warmup). Used only for SQTE

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0_0101b - DT_TX (Packet TX). Used only for SQTE 0_0110b - DT_RX (Packet RX). Used only for SQTE 0_0111b - DT_RX_SYNC (Packet RX Sync). Used only for SQTE 0_1000b - FM_TX (Frequency Measurement TX). Used only for SQTE 0_1001b - FM_RX (Frequency Measurement RX). Used only for SQTE 0_1010b - PM_TX (Phase Measurement TX). 0_1011b - PM_RX (Phase Measurement RX). 0_1100b - IP1_RX2TX (Interlude Period 1 RX2TX). Used only for SQTE 0_1101b - IP1_TX2RX (Interlude Period 1 TX2RX). Used only for SQTE 0_1110b - S_RX2RX (Short Period RX2RX). Used only for SQTE 0_1111b - S_TX2TX (Short Period TX2TX). Used only for SQTE 1_0000b - IP2_RX2TX (Interlude Period 2 RX2TX). 1_0001b - IP2_TX2RX (Interlude Period 2 TX2RX). 1_0010b - FC_RX2TX (Frequency Change RX2TX). 1_0011b - FC_TX2RX (Frequency Change TX2RX). 1_0100b - WD (Warmdown)
15-10 —	Reserved
9 RSM_IRQ_ABORT	RSM_IRQ_ABORT Flag The bit is set when the sequence is aborted due to pll_abort. This condition can occur during any RSM state except for the IDLE or WD states. Write 1 to clear the bit. Note that RX_EN or TX_EN must be set in order to clear this bit, so it is recommended that software ensures this W1C bit is cleared at the end of each RSM sequence before clearing RX_EN or TX_EN.
8 RSM_IRQ_ABORT_EN	RSM_IRQ_ABORT_EN Enable an interrupt when the sequence is aborted due to pll_abort.
7 RSM_IRQ_EOS	RSM_IRQ_EOS Flag The bit is set to indicate that the sequence is complete. It sets when the RSM transitions from the WD to the IDLE state. Write 1 to clear the bit. Note that RSM_RX_EN or RSM_TX_EN must be set in order to clear this bit, so it is recommended that software ensures this W1C bit is cleared at the end of each RSM sequence before clearing RSM_RX_EN or RSM_TX_EN to disable the module.
6 RSM_IRQ_EOS_EN	RSM_IRQ_EOS_EN Enable an interrupt when the sequence is complete.

Table continues on the next page...

Table continued from the previous page...

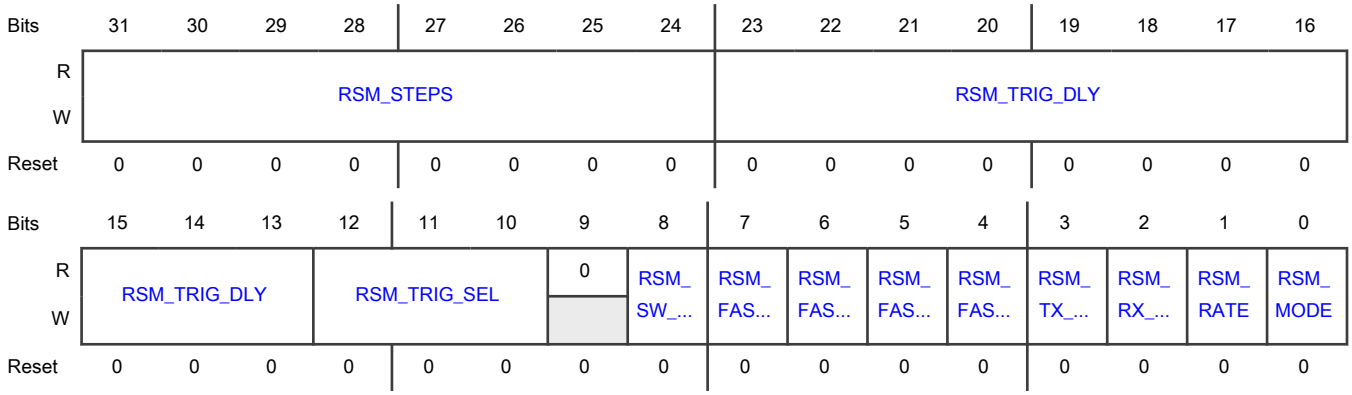
Field	Function
5 RSM_IRQ_FC	RSM_IRQ_FC Flag The bit is set on entry to FC_RX2TX or FC_TX2RX states. Write 1 to clear the bit. Note that RSM_RX_EN or RSM_TX_EN must be set in order to clear this bit, so it is recommended that software ensures this W1C bit is cleared at the end of each RSM sequence before clearing RSM_RX_EN or RSM_TX_EN to disable the module.
4 RSM_IRQ_FC_EN	RSM_IRQ_FC_EN Enable an interrupt on entry to FC_RX2TX or FC_TX2RX states.
3 RSM_IRQ_IP2	RSM_IRQ_IP2 Flag The bit is set on entry to IP2_TX2RX or IP2_RX2TX states. Write 1 to clear the bit. Note that RSM_RX_EN or RSM_TX_EN must be set in order to clear this bit, so it is recommended that software ensures this W1C bit is cleared at the end of each RSM sequence before clearing RSM_RX_EN or RSM_TX_EN to disable the module.
2 RSM_IRQ_IP2_EN	RSM_IRQ_IP2_EN Enable an interrupt asserted on entry to IP2_TX2RX or IP2_RX2TX states.
1 RSM_IRQ_IP1	RSM_IRQ_IP1 Flag The bit is set on entry to IP1_RX2TX or IP1_TX2RX states. Write 1 to clear the bit. Note that RSM_RX_EN or RSM_TX_EN must be set in order to clear this bit, so it is recommended that software ensures this W1C bit is cleared at the end of each RSM sequence before clearing RSM_RX_EN or RSM_TX_EN to disable the module.
0 RSM_IRQ_IP1_EN	RSM_IRQ_IP1_EN Enable an interrupt asserted on entry to IP1_RX2TX or IP1_TX2RX states.

55.4.7.12.2.31 Ranging Sequence Manager Control (RSM_CTRL0)

Offset

Register	Offset
RSM_CTRL0	74h

Diagram



Fields

Field	Function
31-24 RSM_STEPS	RSM_FREQUENCY_STEP Number of frequency steps. (For SQTE this also includes the FCS.) The maximum number supported is 128, limited by RX Packet RAM used by RSM. A value of 0 should not be used
23-13 RSM_TRIG_DLY	RSM_TRIG_DLY • SQTE mode: delay from trigger to when rx_en or tx_en is asserted. • PDE mode: delay from trigger to when the first rx2tx or tx2rx occurs. For PDE mode, this should be programmed to a value greater than 0.
12-10 RSM_TRIG_SEL	RSM_TRIG_SEL 000b - software trigger 001b - crc_vld 010b - aa_fnd_to_ll 011b - tx_dig_en 100b - seq_spare3 101b - lcl pattern_match 110b-111b - Reserved
9 —	Reserved
8 RSM_SW_ABORT	RSM_SW_ABORT Used to gracefully shut down the module prior to normal end of sequence. Once set, this bit should remain set until the RSM has returned to the IDLE state at the end of the sequence.
7 RSM_FAST_FC_TX_WU	RSM_FAST_FC_TX_WU Forces TSM to fast TX wu during FC_RX2TX and WU state (ipr_tx_en=1).

Table continues on the next page...

Table continued from the previous page...

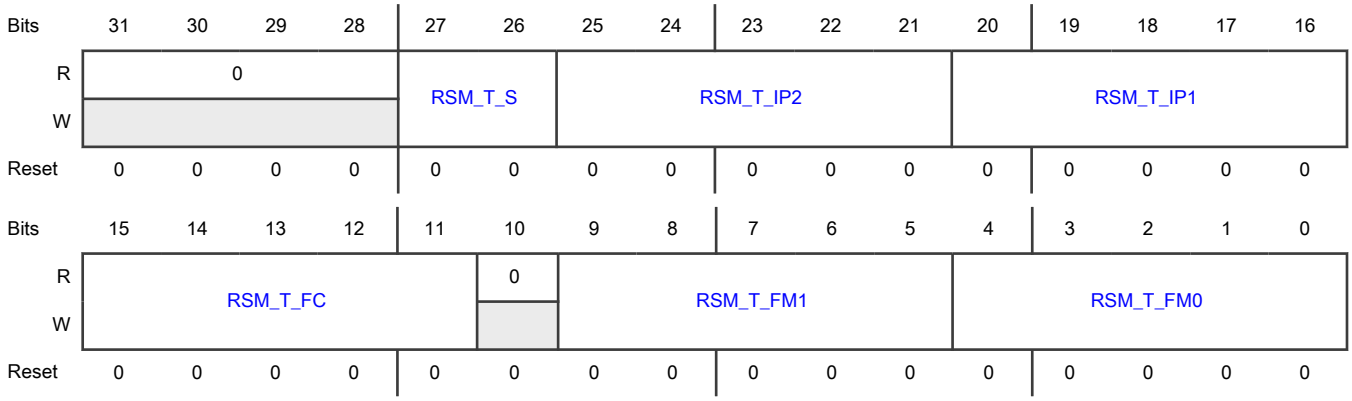
Field	Function
6 RSM_FAST_FC_RX_WU	RSM_FAST_FC_RX_WU Forces TSM to fast RX wu during FC_TX2RX and WU state (ipr_rx_en=1).
5 RSM_FAST_IP_TX_WU	RSM_FAST_IP_TX_WU Forces TSM to fast TX wu during IP1_RX2TX and IP2_RX2TX.
4 RSM_FAST_IP_RX_WU	RSM_FAST_IP_RX_WU Forces TSM to fast RX wu during IP1_TX2RX and IP2_TX2RX.
3 RSM_TX_EN	RSM_TX_EN Arms the module for operation starting with TX. Set this bit for SQTE Initiator/PDE MD. Once set, this bit should remain set until the RSM has returned to the IDLE state at the end of the sequence.
2 RSM_RX_EN	RSM_RX_EN Arms the module for operation starting with RX. Set this bit for SQTE Reflector/PDE RD. Once set, this bit should remain set until the RSM has returned to the IDLE state at the end of the sequence.
1 RSM_RATE	RSM_RATE Only used for SQTE. Indicates how the transceiver is configured, and the transfer rate for packets. 0b - 1Mbps 1b - 2Mbps
0 RSM_MODE	RSM_MODE 0b - SQTE 1b - PDE

55.4.7.12.2.32 Ranging Sequence Manager Control (RSM_CTRL1)

Offset

Register	Offset
RSM_CTRL1	78h

Diagram



Fields

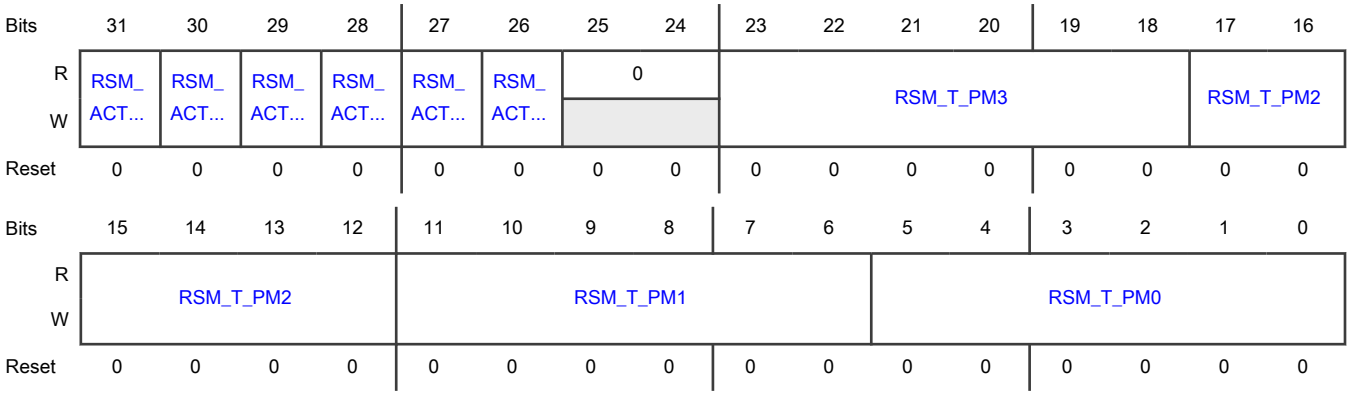
Field	Function
31-28 —	Reserved
27-26 RSM_T_S	RSM_T_S Only applies to SQTE. S (short interlude period) duration, units of 5us. 1=5us, 2=10us, 3=15us. A value of 0 should not be used.
25-21 RSM_T_IP2	RSM_T_IP2 IP2 (interlude period 2) duration, units of 5us. 1=5us, ..., 0x1F=155us. A value of 0 should not be used.
20-16 RSM_T_IP1	RSM_T_IP1 Only applies to SQTE. IP1 (interlude period 1) duration, units of 5us. 1=5us, ..., 0x1F=155us. A value of 0 should not be used
15-11 RSM_T_FC	RSM_T_FC FC (frequency change) duration, units of 5us. 1=5us, ... 0x1F=155us. A value of 0 should not be used.
10 —	Reserved
9-5 RSM_T_FM1	RSM_T_FM1 Only applies to SQTE. FM (frequency measurement) tone duration, units of 10us. 0=10us, ... 0x1F=320us. The T_FM used (T_FM0 or T_FM1) depends on t_pm_sel[0] field.
4-0 RSM_T_FM0	RSM_T_FM0 Only applies to SQTE. FM (frequency measurement) tone duration, units of 10us. 0=10us, ... 0x1F=320us. The T_FM used (T_FM0 or T_FM1) depends on t_pm_sel[0] field.

55.4.7.12.2.33 Ranging Sequence Manager Control (RSM_CTRL2)

Offset

Register	Offset
RSM_CTRL2	7Ch

Diagram



Fields

Field	Function
31 RSM_ACTIVE_OVRD_EN_RXDIG	RSM_ACTIVE_OVRD_EN_RXDIG 0b - Disable override rsm_active of RXDIG module. 1b - Enable override rsm_active of RXDIG module.
30 RSM_ACTIVE_OVRD_RXDIG	RSM_ACTIVE_OVRD_RXDIG When RSM_ACTIVE_OVRD_EN_RXDIG is set to 1, override rsm_active to RXDIG module with RSM_ACTIVE_OVRD_RXDIG value.
29 RSM_ACTIVE_OVRD_EN_TXDIG	RSM_ACTIVE_OVRD_EN_TXDIG 0b - Disable override rsm_active of TXDIG module. 1b - Enable override rsm_active of TXDIG module.
28 RSM_ACTIVE_OVRD_TXDIG	RSM_ACTIVE_OVRD_TXDIG When RSM_ACTIVE_OVRD_EN_TXDIG is set to 1, override rsm_active to TXDIG module with RSM_ACTIVE_OVRD_TXDIG value.
27 RSM_ACTIVE_OVRD_EN_LCL	RSM_ACTIVE_OVRD_EN_LCL 0b - Disable override rsm_active of LCL_CTRL module. 1b - Enable override rsm_active of LCL_CTRL module.

Table continues on the next page...

Table continued from the previous page...

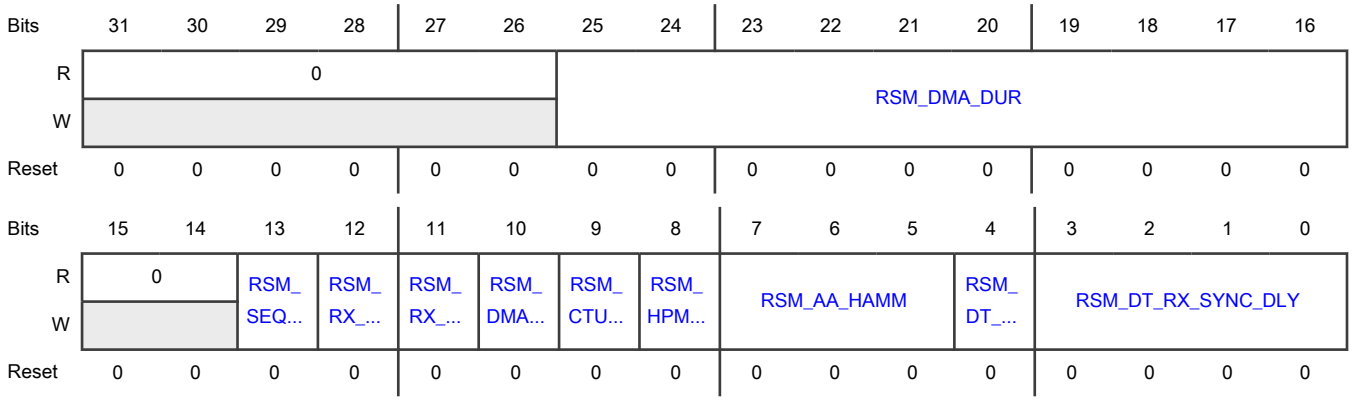
Field	Function
26 RSM_ACTIVE_OVRD_LCL	RSM_ACTIVE_OVRD_LCL When RSM_ACTIVE_OVRD_EN_LCL is set to 1, override rsm_active to LCL_CTRL module with RSM_ACTIVE_OVRD_LCL value.
25-24 —	Reserved
23-18 RSM_T_PM3	RSM_T_PM3 PM (phase measurement) tone duration, units of 10us. 0=10us, ..., 0x3F=640us. For SQTE, this is the T_PM*N_AP duration. The T_PM used (T_PM0, T_PM1, T_PM2 or T_PM3) depends on t_pm_sel[1:0] field.
17-12 RSM_T_PM2	RSM_T_PM2 PM (phase measurement) tone duration, units of 10us. 0=10us, ..., 0x3F=640us. For SQTE, this is the T_PM*N_AP duration. The T_PM used (T_PM0, T_PM1, T_PM2 or T_PM3) depends on t_pm_sel[1:0] field.
11-6 RSM_T_PM1	RSM_T_PM1 PM (phase measurement) tone duration, units of 10us. 0=10us, ..., 0x3F=640us. For SQTE, this is the T_PM*N_AP duration. The T_PM used (T_PM0, T_PM1, T_PM2 or T_PM3) depends on t_pm_sel[1:0] field.
5-0 RSM_T_PM0	RSM_T_PM0 PM (phase measurement) tone duration, units of 10us. 0=10us, ..., 0x3F=640us. For SQTE, this is the T_PM*N_AP duration. The T_PM used (T_PM0, T_PM1, T_PM2 or T_PM3) depends on t_pm_sel[1:0] field.

55.4.7.12.2.34 Ranging Sequence Manager Control (RSM_CTRL3)

Offset

Register	Offset
RSM_CTRL3	80h

Diagram



Fields

Field	Function
31-26 —	Reserved
25-16 RSM_DMA_DUR	DMA Duration Duration of the dma_mask signal, in microseconds. Used in PM_RX state. NOTE: If used in conjunction with the RXDIG's window averager, to ensure that the same number of samples are captured during each time the dma_mask asserts, the duration should be a multiple of the window averager's duration.
15-14 —	Reserved
13 RSM_SEQ_RCCAL_PUP_MASK_DIS	RSM_SEQ_RCCAL_PUP_MASK_DIS If set, disables the seq_rccal_pup_mask output (seq_rccal_pup_mask output will always be 1).
12 RSM_RX_SIGNALS_MASK_DIS	RSM_RX_SIGNALS_MASK_DIS If set, disables the rx_signals_mask output (rx_signals_mask output will always be 1).
11 RSM_RX_PHY_EN_MASK_DIS	RSM_RX_PHY_EN_MASK_DIS Only applies to SQTE. If set, the rx_phy_en_mask output is always asserted. If clear, when the RSM is active the rx_phy_en_mask output is only asserted for packet reception.
10 RSM_DMA_RX_EN	RSM_DMA_RX_EN Enables dma_mask to assert in PM_RX state. (this bit should be cleared if antenna switching / LCL is being used). dma_mask always asserts in SQTE FCS FM state.

Table continues on the next page...

Table continued from the previous page...

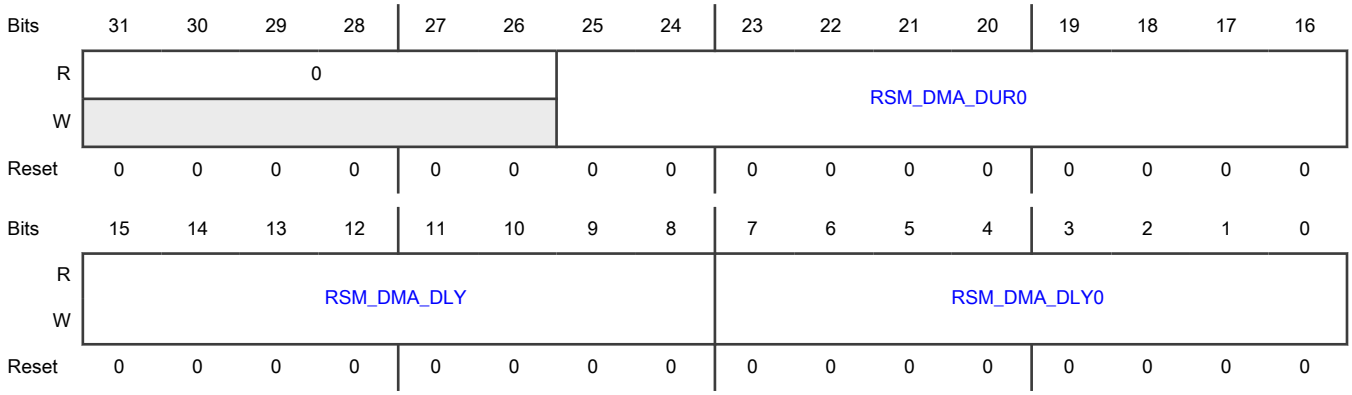
Field	Function
9 RSM_CTUNE	RSM_CTUNE If set, RSM will send coarse tune values read from the Packet RAM to the PLL. If clear, the PLL will perform coarse tune calibration
8 RSM_HPM_CAL	RSM_HPM_CAL If set, RSM will send HPM cal values read from the Packet RAM to the PLL. If clear, the PLL will perform HPM calibration
7-5 RSM_AA_HAMM	RSM_AA_HAMM Only applies to SQTE. AA hamming distance to PHY.
4 RSM_DT_RX_SYNC_DIS	RSM_DT_RX_SYNC_DIS Only applies to SQTE. Disables the auto-sync feature for Reflector in FCS RX packet.
3-0 RSM_DT_RX_SYNC_DLY	RSM_DT_RX_SYNC_DLY Only applies to SQTE, and only used by the Reflector device (RX_EN=1) when DT_RX_SYNC_DIS=0. In this configuration the RSM uses the PHY's AA_MATCHED output to sync the Reflector device to the Initiator device's timing. When RTT_SEQ_LEN=0, this corresponds to the delay in microseconds from the PHY's AA_MATCHED to the end of DT_RX duration (the packet duration including preamble and trailer, plus RX_SETTLING_DELAY). The value used for RTT_SEQ_LEN=0 can also be used when RTT_SEQ_LEN=1, though in the latter configuration the RSM internally offsets this value to compensate for the longer packets (since AA_MATCHED uses only the first 32bits of the PN sequence regardless of the PN sequence length).

55.4.7.12.2.35 Ranging Sequence Manager Control (RSM_CTRL4)

Offset

Register	Offset
RSM_CTRL4	84h

Diagram



Fields

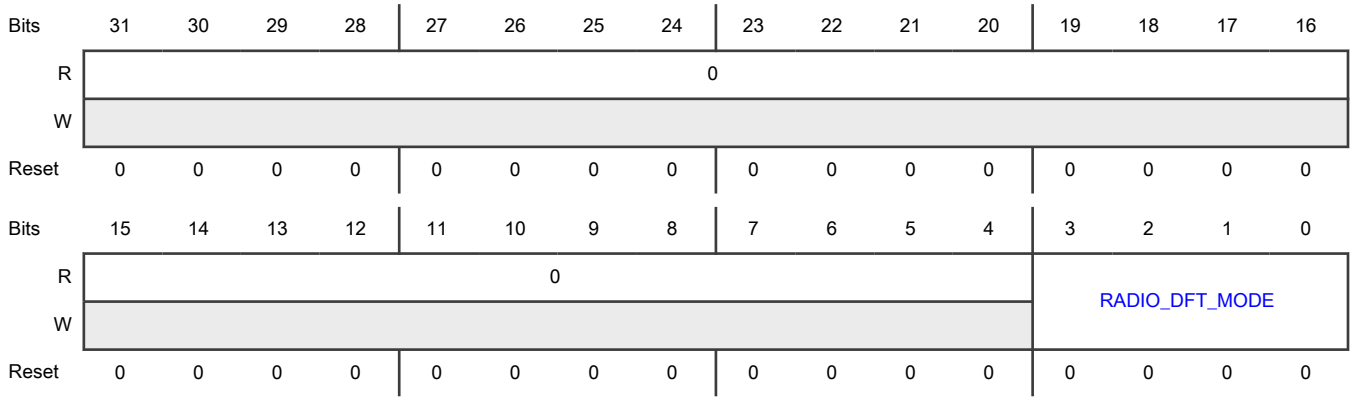
Field	Function
31-26 —	Reserved
25-16 RSM_DMA_DUR0	DMA Duration 0 Duration of the dma_mask signal, in microseconds. Used in FM_RX state (SQTE) and EXT_RX state (PDE). NOTE: If used in conjunction with the RXDIG's window averager, to ensure that the same number of samples are captured during each time the dma_mask asserts, the duration should be a multiple of the window averager's duration.
15-8 RSM_DMA_DLY	DMA Delay Delay from end of RX warmup to assertion of dma_mask signal, in microseconds. Used in PM_RX state.
7-0 RSM_DMA_DLY0	DMA Delay 0 Delay from end of RX warmup to assertion of dma_mask signal, in microseconds. Used in FM_RX state (SQTE) and EXT_RX state (PDE).

55.4.7.12.2.36 RF DFT CTRL (RF_DFT_CTRL)

Offset

Register	Offset
RF_DFT_CTRL	9Ch

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RADIO_DFT_MODE	Radio DFT mode control These bits control Radio DFT mode used by TXDIG and PLL. Values other than those shown below are Reserved. 0000b - Normal Mode 0001b - Carrier Only 0010b - Pattern Register 0011b - LFSR 0100b - RAM Modulation 1010b - Coarse Tune BIST, no modulation 1011b - PLL Locking BIST, no modulation 1100b - HPM DAC Cal BIST, no modulation

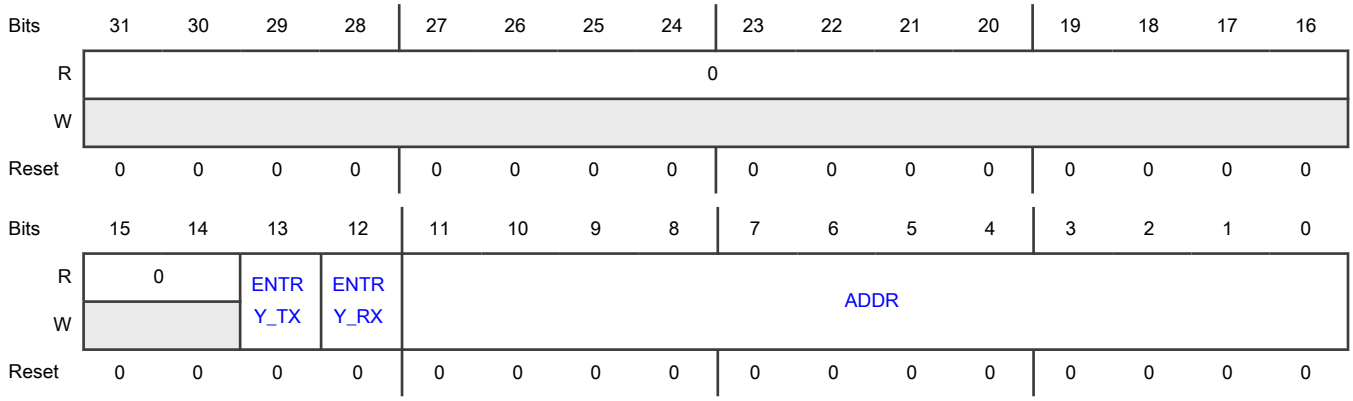
55.4.7.12.2.37 IPS FAST OVERWRITE ADDRESS (IPS_FO_ADDR0 - IPS_FO_ADDR7)

Offset

For a = 0 to 7:

Register	Offset
IPS_FO_ADDRa	A0h + (a × 4h)

Diagram



Fields

Field	Function
31-14 —	Reserved
13 ENTRY_TX	Enable Entry for TX If set, the Nth fast overwrite entry is enabled for TX
12 ENTRY_RX	Enable Entry for RX If set, the Nth fast overwrite entry is enabled for RX
11-0 ADDR	IPS Address This specifies the address associated with the Nth fast overwrite entry

55.4.7.12.2.38 IPS FAST OVERWRITE DRS0 DATA (IPS_FO_DRS0_DATA0 - IPS_FO_DRS0_DATA7)

Offset

For a = 0 to 7:

Register	Offset
IPS_FO_DRS0_DATAa	C0h + (a × 4h)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DRS0_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DRS0_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DRS0_DATA	Fast Overwrite DRS0 data This specifies the 32bit data associated with the Nth fast overwrite entry, for DRS0.

55.4.7.12.2.39 IPS FAST OVERWRITE DRS1 DATA (IPS_FO_DRS1_DATA0 - IPS_FO_DRS1_DATA7)

Offset

For a = 0 to 7:

Register	Offset
IPS_FO_DRS1_DATAa	E0h + (a × 4h)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DRS1_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DRS1_DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DRS1_DATA	Fast Overwrite DRS1 data This specifies the 32bit data associated with the Nth fast overwrite entry, for DRS1.

55.4.8 Radio RAM

55.4.8.1 TX Packet Ram

55.4.8.1.1 RADIO_PACKET_RAM register descriptions

The Radio Packet RAM is directly accessed by various logic blocks in the radio. If software performs an access to Packet RAM concurrently with an access by the radio logic, software reads will be wait-stated but software writes will be ignored. Therefore it the Packet RAM is written by software at a time when the radio logic might be accessing the Packet RAM, it is advisable that software verify the writes were completed successfully by reading the Packet RAM values at the addresses written.

55.4.8.1.1.1 RADIO_PACKET_RAM_ADDR memory map

TX_PACKET_RAM base address: 48A0_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h - FFCh	Shared Packet RAM for multiple Link Layer usage. (PACKET_RAM_0 - PACKET_RAM_1023)	32	RW	See section

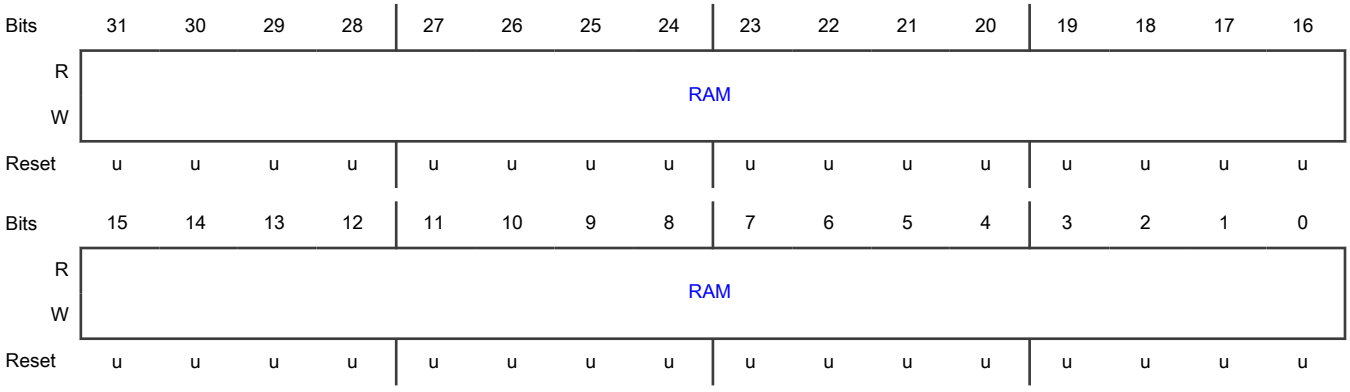
55.4.8.1.1.2 Shared Packet RAM for multiple Link Layer usage. (PACKET_RAM_0 - PACKET_RAM_1023)

Offset

For a = 0 to 1023:

Register	Offset
PACKET_RAM_a	0h + (a × 4h)

Diagram



Fields

Field	Function
31-0 RAM	One entry in the packet RAM One word of the packet RAM, whatever the defined width is.

55.4.8.2 RX Packet Ram

55.4.8.2.1 RADIO_PACKET_RAM register descriptions

The Radio Packet RAM is directly accessed by various logic blocks in the radio. If software performs an access to Packet RAM concurrently with an access by the radio logic, software reads will be wait-stated but software writes will be ignored. Therefore it the Packet RAM is written by software at a time when the radio logic might be accessing the Packet RAM, it is advisable that software verify the writes were completed successfully by reading the Packet RAM values at the addresses written.

55.4.8.2.1.1 RADIO_PACKET_RAM_ADDR memory map

RX_PACKET_RAM base address: 48A0_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7FCh	Shared Packet RAM for multiple Link Layer usage. (PACKET_RAM_0 - PACKET_RAM_511)	32	RW	See section

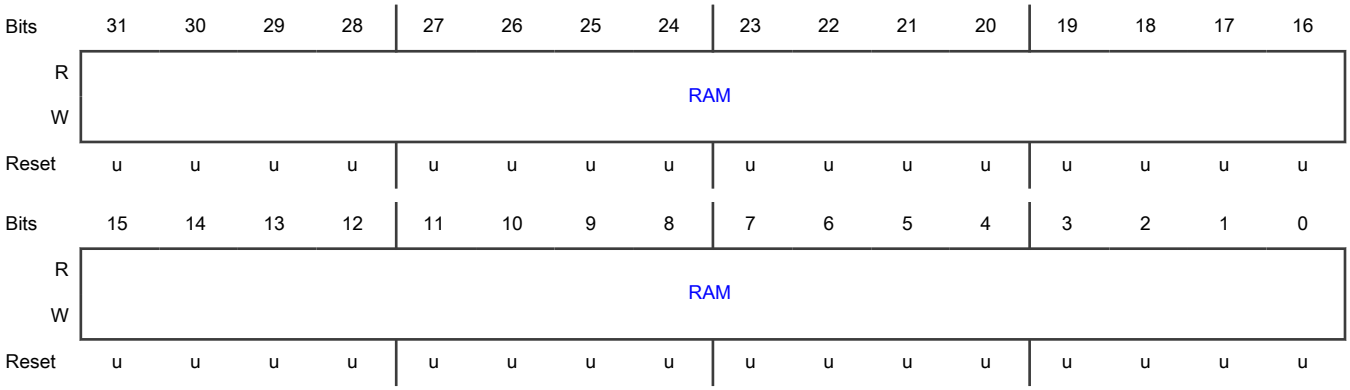
55.4.8.2.1.2 Shared Packet RAM for multiple Link Layer usage. (PACKET_RAM_0 - PACKET_RAM_511)

Offset

For a = 0 to 511:

Register	Offset
PACKET_RAM_a	0h + (a × 4h)

Diagram



Fields

Field	Function
31-0 RAM	One entry in the packet RAM One word of the packet RAM, whatever the defined width is.

55.4.8.3 Radio Packet RAM memory map

NOTE

The RAM entries shown for Bluetooth LE LL are not actually part of the "Packet RAM" but are included in the table for reference.

RAM	Start Address	Size(Byte)
Bluetooth LE LL		
Bluetooth LE LL IMEM	0xA0100000(CPU2)/0x48900000(SoC)	2K
Bluetooth LE LL DMEM	0xA0140000(CPU2)	48K
Bluetooth LE LL EBRAM	0xA9024000(CPU2)	8928
Bluetooth LE LL SMU2	0xB0000000(CPU2)/489C0000(SoC)	40K
Generic LL		
Generic LL TX_RAM Buffer	TX_RAM_BASE_ADDR	2048+256
Generic LL IE	TX_RAM_BASE_ADDR+0xAC0	64
Generic LL SAM	TX_RAM_BASE_ADDR+0xB00	256
Generic LL PIRK	TX_RAM_BASE_ADDR+0xB00	128
Generic LL LIRK	TX_RAM_BASE_ADDR+0xB80	128
Generic LL RPA-ID	TX_RAM_BASE_ADDR+0xC00	64
Generic LL PRPA	TX_RAM_BASE_ADDR+0xC40	64
Generic LL LRPA	TX_RAM_BASE_ADDR+0xC80	64
Generic LL WL0	TX_RAM_BASE_ADDR+0xD00	256
Generic LL WL1	TX_RAM_BASE_ADDR+0xE00	256
Generic LL RX_RAM Buffer	RX_RAM_BASE_ADDR	2048
Zigbee LL		
Zigbee LL TX_RAM Buffer	TX_RAM_BASE_ADDR	128
Zigbee LL RX_RAM Buffer	TX_RAM_BASE_ADDR+0x80	128
Zigbee LL SAM	TX_RAM_BASE_ADDR+0x100	256
Wake on Radio		
WOR_HOP	TX_RAM_BASE_ADDR+0x900	256

Table continues on the next page...

Table continued from the previous page...

WOR_TIME	TX_RAM_BASE_ADDR+0xA00	160
Ranging sequence manager		
RSM_PCBD_RAM	RX_RAM_BASE_ADDR	128
RSM_RTT_RAM	RX_RAM_BASE_ADDR+0x80	424
RSM_PN_RAM	RX_RAM_BASE_ADDR+0x230	848
RSM_FSTEP_RAM	RX_RAM_BASE_ADDR+0x580	640

55.4.9 Link Layer Blocks

55.4.9.1 802.15.4 Link Layer

55.4.9.1.1 802.15.4 Link Layer Controller

55.4.9.1.1.1 Introduction

The 802.15.4 Link Layer Controller implements all hardware functionality of the 802.15.4 Link Layer.

55.4.9.1.1.2 Overview

The 802.15.4 Link Layer implementation is divided into software and hardware components. The 802.15.4 Link Layer Controller represents the hardware component of the link layer. Link Layer hardware functionality is further sub-divided into functional blocks. These blocks include:

- Packet Processor block.
- Sequence Manager block.
- Packet Storage block.
- Event Timer block.
- Interrupts block.
- Clear Channel Assessment/Energy Detect/Link Quality Indication block.
- Bit Streaming Mode block.

Together, these blocks implement and accelerate critical functions of the 802.15.4 Link Layer. The non-timing critical functions are implemented in software on a MCU.

55.4.9.1.1.3 Block Diagram

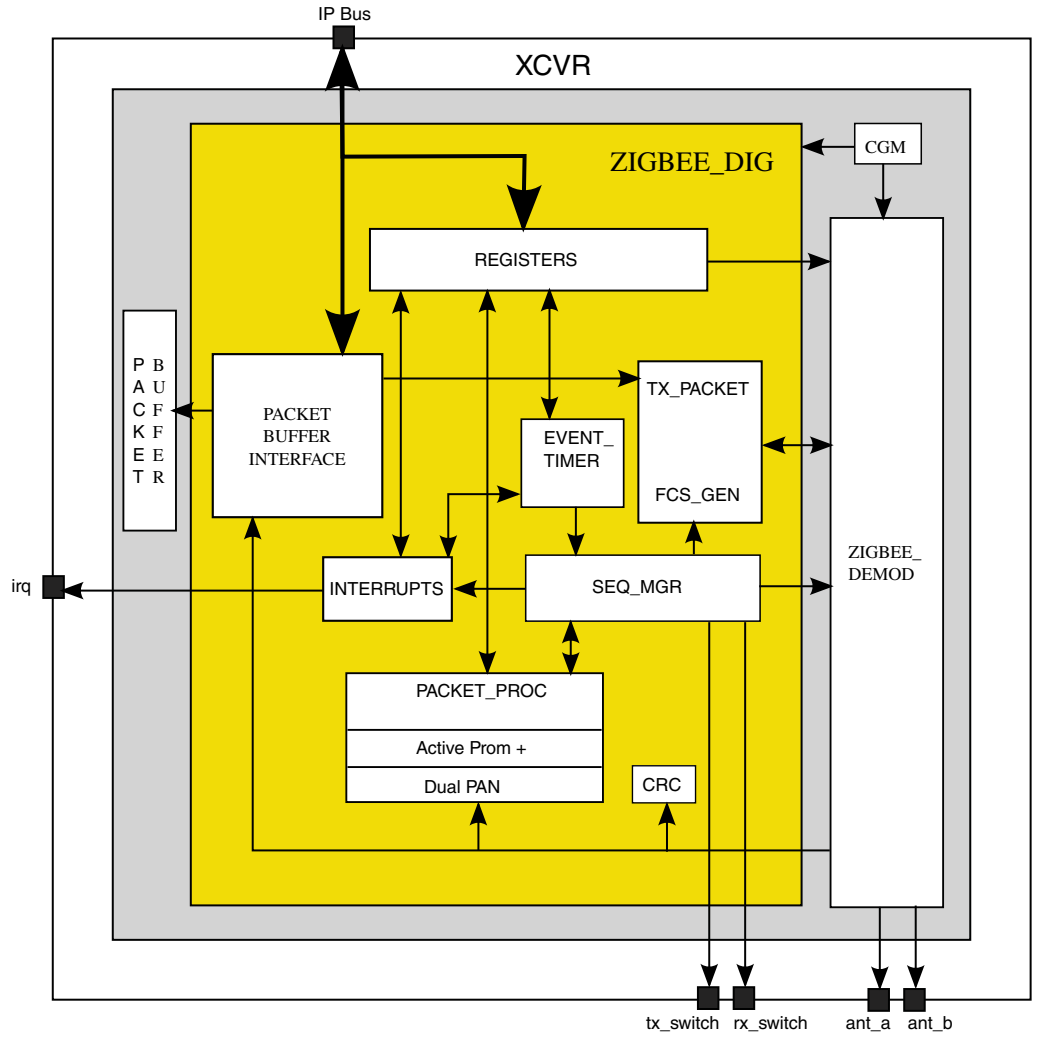


Figure 373. 802.15.4 Link Layer Controller Block Diagram

55.4.9.1.2 Functional Description

55.4.9.1.2.1 Packet Processor

55.4.9.1.2.1.1 Introduction

The 802.15.4 Packet Processor performs sophisticated hardware filtering of the incoming received packet, to determine whether the packet is both PHY- and MAC-compliant, whether the packet is addressed to this device, and if the device is a PAN Coordinator, whether a message is pending for the sending device. The packet processor greatly reduces the packet filtering burden on software, allowing software to tend to higher-layer tasks with a lower latency and smaller software footprint.

Table 472. References

Revision	Title
IEEE Std 802.15.4 -2003	Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)
IEEE Std 802.15.4 -2006	802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)
IEEE Std 802.15.4 -2011	Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)
IEEE Std 802.15.4e™-2012	Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) (Amendment to IEEE Std 802.15.4™-2011)
IEEE Std 802.15.4 -2015	Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) (Amendment to IEEE Std 802.15.4-2011)

55.4.9.1.2.1.1.1 Overview

The 802.15.4 Packet Processor performs aggressive hardware filtering of the incoming packet as it is being received, to increase battery life by not awakening the MCU to process packets which are non-compliant or not addressed to the device, or that violate the 802.15.4 MAC Frame structure. The advanced frame parsing enables the sequence manager to handle complex sequences, by, for example, determining whether a receive frame requires an acknowledge, and generating the Sequence Number for that acknowledge, so that the transmit/receive combination sequence can be executed without any MCU intervention. Finally, the packet processor includes a hardware accelerator that allows the device to handle a critical timing requirement that had been problematic for 8-bit MCU's in the past: the quick-turnaround requirement for an indirect queue-lookup to determine the correct response to an end device sending a data polling request. The packet processor contains a register-based table of end devices which have messages pending, allowing a single-cycle hardware table lookup to be performed, and a minimum-turnaround-time acknowledge packet to be sent to the end device without MCU intervention. The packet processor also includes support for Dual PAN mode, which allows the device to simultaneously reside on 2 networks, with 2 separate IEEE addresses. And the packet processor also supports Active Promiscuous mode, which allows address filtering rules to be bypassed, while still enabling the device to automatically acknowledge packets which are addressed to the device.

55.4.9.1.2.1.1.2 Features

- Aggressive packet filtering to enable long, uninterrupted MCU sleep periods
- Fully compliant with both 2003 and 2006 versions of the 802.15.4 wireless standard
- Supports all Frame Types, including reserved types
- Supports all valid 802.15.4 Frame Lengths
- Enables auto-Tx Acknowledge frames (no MCU intervention) by parsing of Frame Control Field and Sequence Number
- Supports all Source and Destination Address modes, and also PAN ID Compression
- Supports Broadcast address for PAN ID and short address mode
- Supports "Promiscuous" mode, to receive all packets regardless of address- and rules-checking
- Allows Frame Type-specific filtering (e.g., reject all but Beacon frames)
- Supports SLOTTED and non-SLOTTED modes
- Includes special filtering rules for PAN Coordinator devices
- Enables minimum-turnaround TX-Acknowledge frames for data-polling requests by automatically determining message-pending status
- Assists MCU in locating pending messages in its indirect queue for data-polling end devices
- Assists MCU in determining if a poll request originated from a valid child end device
- Makes available to MCU detailed status of frames which fail address- or rules-checking

- Supports Dual PAN mode, allowing the device to exist on 2 PAN's simultaneously
- Supports 2 IEEE addresses for the device
- Supports Active Promiscuous Mode
- Includes provisions to enable software support of 802.15.4e and 802.15.4-2015 frame types and versions

55.4.9.1.2.1.1.3 Block diagram

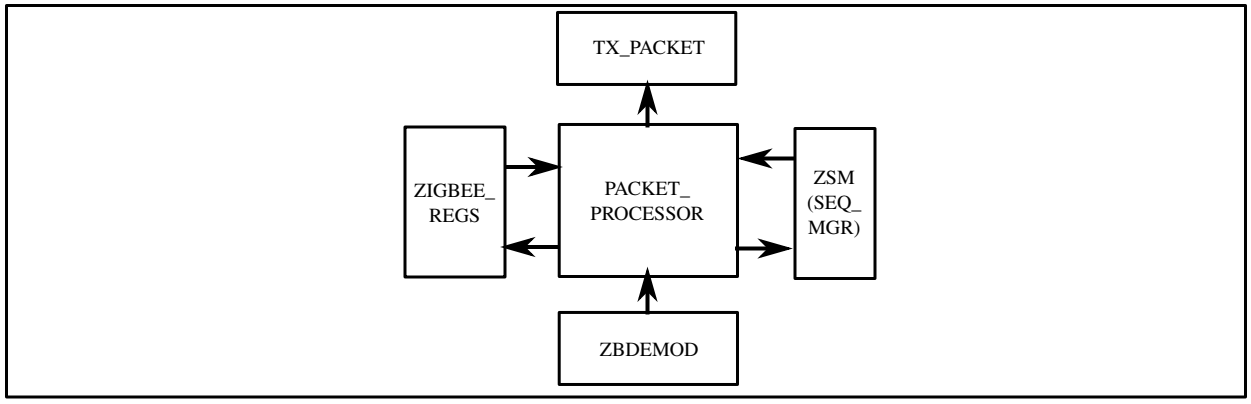


Figure 374. Block diagram

55.4.9.1.2.1.2 Memory Map and register definition

Numerous registers exist to provide configuration and status for the 802.15.4 Packet Processor.

The memory-mapped registers which affect the Packet Processor are described in the table below.

Field	R/W	Description
PI	r	Poll Indication: 1: the received packet was a data request, regardless of whether a source-address table-match occurred, and regardless of whether source-address matching is enabled (see SRCADDR_EN bit); 0: the received packet was not a data request PI is a read-only bit.
SRCADDR	r	If Source Address Management is engaged, meaning at least one of the following bits is set: • SAP0_EN • SAA0_EN • SAP1_EN • SAA1_EN then SRCADDR will be set to 1 if the packet just received is a poll request (PI=1), and at least one of the following conditions is met: • SAP0_EN && SAP0_ADDR_PRESENT

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		• SAA0_EN && SAA0_ADDR_ABSENT • SAP1_EN && SAP1_ADDR_PRESENT • SAA1_EN && SAA1_ADDR_ABSENT If SRCADDR=1, this indicates to SW that the Packet Processor has determined that an auto-TxACK frame must be transmitted with the FramePending subfield of the FrameControlField set to 1. HW will assemble and transmit this Ack packet. If the above conditions are not met, SRCADDR will be cleared to 0.
SAP0_EN	rw	1: Enables SAP0 Partition 0: Disables SAP0 Partition
SAA0_EN	rw	1: Enables SAA0 Partition 0: Disables SAA0 Partition
SAP1_EN	rw	1: Enables SAP1 Partition 0: Disables SAP1 Partition
SAA1_EN	rw	1: Enables SAA1 Partition 0: Disables SAA1 Partition
SAA0_START[6:0]	rw	First Index of SAA0 partition.
SAP1_START[6:0]	rw	First Index of SAP1 partition.
SAA1_START[6:0]	rw	First Index of SAA1 partition.
SAM_CHECKSUM[15:0]		Software-computed source address checksum, to be installed into a table index
SAM_INDEX[6:0]		Contains the table index to be enabled or invalidated. Software must ensure that the index is within the range of the desired partition.
SAM_INDEX_WR	w	For 32-bit writes, SAM_INDEX_WR must be set to indicate that the table entry specified by SAM_INDEX[6:0] is to be written; if SAM_INDEX_WR=0, the table entry is not written, but the SAM_INDEX[6:0] register is updated. For 8-bit writes, this bit is ignored.
SAM_INDEX_EN	w	Enable the index selected by SAM_INDEX[6:0] for searching.
SAM_INDEX_INV	w	Invalidate the index selected by SAM_INDEX[6:0]
FIND_FREE_IDX	w	After modifying Valid bits (enabling or invalidating), write this bit to 1 to force HW to update the “First Free Index” registers to account for the changed Valid bits. This HW update process takes 4us. SW can poll SAM_BUSY to determine when the table update is complete. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
SAM_BUSY	r	HW is in the process of updating the Source Address table, either in response to a poll indication from the packet processor, or due to

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		SW setting FIND_FREE_IDX=1. In the latter case, SW should poll SAM_BUSY until low before accessing the “First Free Index” registers. Read-only bit.
INVALIDATE_ALL	w	Writing a 1 to this bit clears all 128 Valid bits. Invalidates the entire table. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
SAP0_1ST_FREE_IDX[6:0]	r	First non-enabled (invalid) index in the SAP0 partition. Read-only.
SAA0_1ST_FREE_IDX[6:0]	r	First non-enabled (invalid) index in the SAA0 partition. Read-only.
SAP1_1ST_FREE_IDX[6:0]	r	First non-enabled (invalid) index in the SAP1 partition. Read-only.
SAA1_1ST_FREE_IDX[6:0]	r	First non-enabled (invalid) index in the SAA1 partition. Read-only
PROMISCUOUS	rw	Bypasses most packet filtering. 1: all packet filtering except frame length checking (FrameLength>=5 and FrameLength<=127) is bypassed. 0: normal mode **Note:** Promiscuous Mode is not intended for Sequence TR with RXACKRQD=1 (auto-RXACK mode), because the contents of auto-RXACK packets are not stored in the Packet Buffer. If auto-RXACK packets are to be stored in the PB, use ACTIVE_PROMISCUOUS instead.
PANCORDNTR0	rw	Device is a PAN Coordinator on PAN0. Allows device to receive packets with no destination address, if Source PAN ID matches.
MACPANID0[15:0]	rw	MAC PAN ID for PAN0. The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet Source PAN ID against this register. Also, if PANCORDNTR0=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet Source PAN ID against this register.
MACSHORTADDRS0[15:0]	rw	MAC Short Address for PAN0, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
MACLONGADDRS0[63:0]	rw	MAC Long Address for PAN0, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
BEACON_FT	rw	1: Beacon frame type enabled.

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		0: reject all Beacon frames
ACK_FT	rw	1: Acknowledge frame type enabled. 0: reject all Acknowledge frames
DATA_FT	rw	1: Data frame type enabled. 0: reject all Data frames
CMD_FT	rw	1: MAC Command frame type enabled. 0: reject all MAC Command frames
LLDN_FT	rw	1: LLDN frame type enabled (Frame Type 4). 0: reject all MAC Command frames
MULTIPURPOSE_FT	rw	1: MULTIPURPOSE frame type enabled (Frame Type 5). 0: reject all MAC Command frames
EXTENDED_FT	rw	1: EXTENDED frame type enabled (Frame Type 7). 0: reject all MAC Command frames
NS_FT	rw	1: Not-specified (reserved) frame type enabled. No packet filtering is performed, except for frame length checking (FrameLength>=5 and FrameLength<=127). 0: reject all reserved frame types
EXTENDED_FCS_CHK	rw	Verify FCS on Frame Type Extended 1: Packet Processor will check FCS at end-of-packet based on packet length derived from PHR, for Frame Type EXTENDED 0: Packet Processor will not check FCS for Frame Type EXTENDED (default)
ACTIVE_PROMISCUOUS	rw	1: Provide Data Indication on all received packets under the same rules which apply in PROMISCUOUS mode, however acknowledge those packets under rules which apply in non-PROMISCUOUS mode 0: normal operation (Default)
FRM_VER_FILTER[3:0]	rw	The incoming packet's Frame Control Field is parsed to obtain the FrameVersion subfield, and that value is compared against this register, in accordance with the following: xxx1: Accept received packets with FrameVersion=00 xx1x: Accept received packets with FrameVersion=01 x1xx: Accept received packets with FrameVersion=10

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		1xxx: Accept received packets with FrameVersion=11 These filtering rules apply to Beacon, Acknowledge, Data, and MAC Command Frame Types, since these frame types require a 2-octet Frame Control Field which embeds a 2-bit FrameVersion subfield. Later frame types introduced in the 802.15.4e addendum (LLDN, Multipurpose) don't guarantee a FrameVersion subfield with the original meaning, so these filtering rules do not apply to these frame types. See registers LLDN_FT and MULTIPURPOSE_FT. For Acknowledge frames, FrameVersion bits are ignored by the Packet Processor, irrespective of FRM_VER_FILTER.
MACPANID1[15:0]	rw	MAC PAN ID for PAN1. The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet Source PAN ID against this register. Also, if PANCORDNTR1=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet Source PAN ID against this register.
MACSHORTADDRS1[15:0]	rw	MAC Short Address for PAN1, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
MACLONGADDRS1[63:0]	rw	MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
ACTIVE_NETWORK	rw	Selects the PAN on which to transceive, by activating a PAN parameter set (PAN0 or PAN1). In Manual Dual PAN mode (or Single PAN mode), this bit selects the active PAN parameter set (channel and addressing parameters) which governs all autosequences. In Auto Dual PAN mode, this bit selects the PAN on which to begin transceiving, latched at the point at which DUAL_PAN_DWELL register is written. 1: Select PAN1 0: Select PAN0 (Default)
DUAL_PAN_AUTO	rw	Activates automatic Dual PAN operating mode. In this mode, PAN-switching is controlled by hardware at a pre-programmed rate, determined by DUAL_PAN_DWELL. 1: Auto Dual PAN Mode 0: Manual Dual PAN mode (or Single PAN mode). Default.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>R/W</th><th>Description</th></tr>
<tr><td></td><td></td><td>Whenever DUAL_PAN_AUTO=0, CURRENT_NETWORK=ACTIVE_NETWORK at all times. In other words, software directly controls which PAN is selected.

Whenever DUAL_PAN_AUTO=1, CURRENT_NETWORK is controlled by hardware</td></tr>
<tr><td>PANCORDNTR1</td><td>rw</td><td>Device is a PAN Coordinator on PAN1. Allows device to receive packets with no destination address, if Source PAN ID matches.</td></tr>
<tr><td>CURRENT_NETWORK</td><td>r</td><td>This read-only bit indicates which PAN is currently selected by hardware in automatic Dual PAN mode

1: PAN1 is selected

0: PAN0 is selected</td></tr>
<tr><td>DUAL_PAN_DWELL[7:0]</td><td>rw</td><td>Channel Frequency Dwell Time. In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to DUAL_PAN_DWELL, always re-initializes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the sequence warmup.
<table>
<tr><th>PRESCALER (bits [1:0])</th><th>TIMEBASE (bits [7:2])</th><th>RANGE (min) - (max)</th></tr>
<tr><td>00</td><td>0.5ms</td><td>0.5 - 32ms</td></tr>
<tr><td>01</td><td>2.5ms</td><td>2.5 - 160ms</td></tr>
<tr><td>10</td><td>10ms</td><td>10 - 640ms</td></tr>
<tr><td>11</td><td>50ms</td><td>50ms - 3.2seconds</td></tr>
</table>
A write to DUAL_PAN_DWELL also causes the value of ACTIVE_NETWORK to get latched into the hardware. This latched value will be the starting point for the automatic dual-pan mode (i.e., start on PAN0 or on PAN1). The starting value takes effect immediately (if sequence is underway and DUAL_PAN_AUTO=1), or is otherwise delayed until sequence starts and DUAL_PAN_AUTO=1.</td></tr>
<tr><td>DUAL_PAN_REMAIN[5:0]</td><td></td><td>This read-only register indicates time remaining before next PAN switch in auto Dual PAN mode. The units for this register, depend on the PRESCALER setting (bits [1:0]) in the DUAL_PAN_DWELL register, according to the following table:</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>R/W</th><th>Description</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>DUAL_PAN_DWELL PRESCALER</th><th>DUAL_PAN_REMAIN UNITS</th></tr>
<tr><td>00</td><td>0.5ms</td></tr>
<tr><td>01</td><td>2.5ms</td></tr>
<tr><td>10</td><td>10ms</td></tr>
<tr><td>11</td><td>50ms</td></tr>
</table>

The readback value indicates that between N-1 and N timebase units remain until the next PAN switch. For example, a DUAL_PAN_REMAIN readback value of 3, with a DUAL_PAN_DWELL PRESCALER setting of 2 (10ms), indicates that between 20ms (2*10ms) and 30ms (3*10ms), remain until the next automatic PAN switch.</td></tr>
<tr><td>RECD_ON_PAN0</td><td>r</td><td>Indicates the packet which was just received, was received on PAN0.

In Dual PAN mode operating on 2 different channels, RECD_ON_PAN0 will be set if CURRENT_NETWORK=0 when the packet was received, regardless of FILTERFAIL status.

In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN0 will be set only if a valid packet was received on PAN0 (PAN0’s FILTERFAIL_FLAG is deasserted).</td></tr>
<tr><td>RECD_ON_PAN1</td><td>r</td><td>Indicates the packet which was just received, was received on PAN1.

In Dual PAN mode operating on 2 different channels, RECD_ON_PAN1 will be set if CURRENT_NETWORK=1 when the packet was received, regardless of FILTERFAIL status.

In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN1 will be set only if a valid packet was received on PAN1 (PAN1’s FILTERFAIL_FLAG is deasserted).</td></tr>
<tr><td>FILTERFAIL_CODE[9:0]</td><td>r</td><td>Status of packet processor filtering. For incoming packets which have been rejected by the packet processor, FILTERFAIL_CODE will indicate which violated packet-filtering rule(s) led to the rejection. This is intended for debug only. The contents of this register are valid at the FILTERFAIL_IRQ interrupt. The packet processor maintains 2 independent copies of FILTERFAIL_CODE[9:0], one for each PAN. When reading this register, software can select for which PAN the FILTERFAIL_CODE applies, using the FILTERFAIL_PAN_SEL bit. See Appendix A.</td></tr>
<tr><td>FILTERFAIL_PAN_SEL</td><td>rw</td><td>1: FILTERFAIL_CODE[9:0] will report the FILTERFAIL status of PAN1

0: FILTERFAIL_CODE[9:0] will report the FILTERFAIL status of PAN0</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
LENIENCY[42:0]	rw	The LENIENCY register allows individual packet filtering rules to be overridden, on a rule-by-rule basis. In other words, a specific packet filtering rule, such as an address-mode check, which would result in an incoming packet being rejected, can be overridden by setting the corresponding bit in the LENIENCY register, which would then result in the packet being accepted. This is similar to PROMISCUOUS mode, except that it allows checks to be overridden individually. This will facilitate debug, and potentially allow the device to communicate with devices which may not be 100% MAC-compliant. The mapping of LENIENCY register bits to rules is shown in Appendix B of this document.

55.4.9.1.2.1.3 Functional Description

The packet processor is responsible for receiving symbols from the demodulator, combining them into octets, assembling the octets into packets, filtering the packets in compliance with the 802.15.4 wireless MAC standard, and storing the packets into the Packet Buffer (Packet RAM).

The packet processor parses the incoming packets on-the-fly to determine:

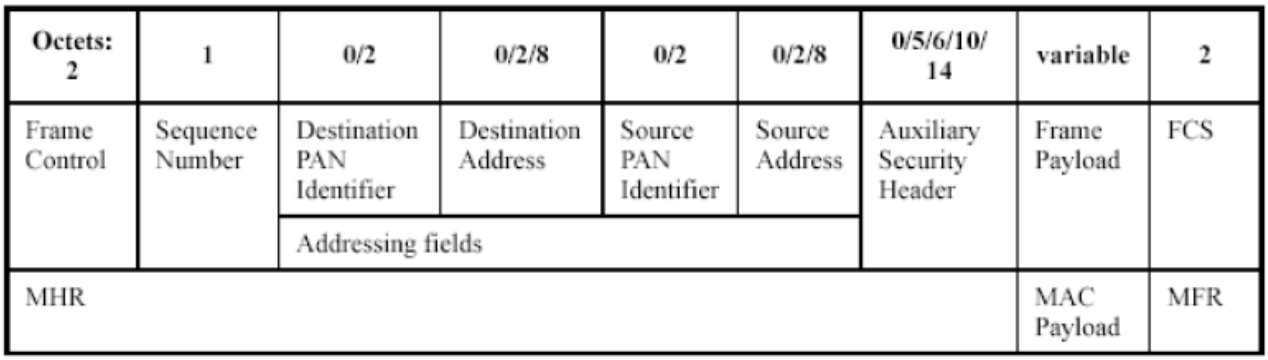
1. whether the packet is MAC-compliant, and if it is,
2. whether the packet is addressed to the end device.

The packet processor performs aggressive filtering on incoming packets, in order to reduce unnecessary MCU wake-ups, to allow longer MCU sleep durations, and to provide sufficient hardware automation so as to enable complex sequences to be executed by the sequence manager. The packet processor also includes hardware support for Dual PAN operation (allowing the device to reside on two networks simultaneously), and for Active Promiscuous mode (allowing the device to receive all packets, but acknowledging only those addressed to the device).

The 802.15.4 packet processor also includes functionality to allow a critical-timing requirement to be met, which in the past has been problematic for low-MIPS 802.15.4 receiving devices: the ability of a coordinator device to respond to a MAC Command data request from an end device, with an Acknowledge frame containing a FramePending bit accurately reflecting the presence of a message for the end device in the coordinator's indirect queue, or the absence of the end device in the coordinator's neighbor table, within the prescribed 192us window. This functionality is described in more detail in the Source Address Matching section.

55.4.9.1.2.1.3.1 Packet Filtering Background

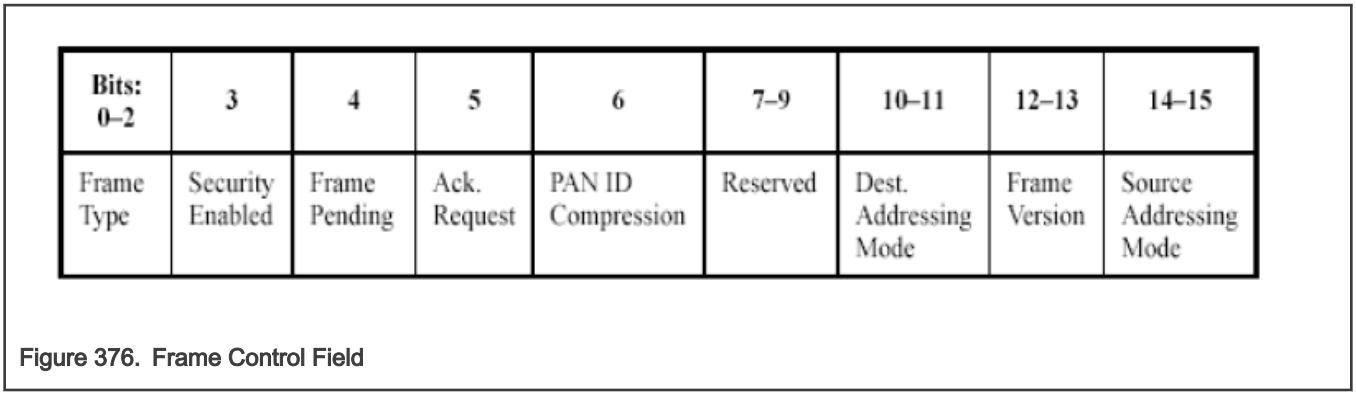
The packet processor parses packets to verify compliance with the 802.15.4 MAC frame format. The basic MAC frame format is shown in the figure below.



Octets: 2	1	0/2	0/2/8	0/2	0/2/8	0/5/6/10/14	variable	2
Frame Control	Sequence Number	Destination PAN Identifier	Destination Address	Source PAN Identifier	Source Address	Auxiliary Security Header	Frame Payload	FCS
		Addressing fields						
MHR							MAC Payload	MFR

Figure 375. MAC Frame Format

The packet processor parses the incoming octets on the fly, as they are received. The Frame Control Field, two octets in length, contains subfields which encode “instructions” on how to parse the remainder of the MHR (MAC Header). The structure of the Frame Control Field is shown in the figure below.



Bits: 0–2	3	4	5	6	7–9	10–11	12–13	14–15
Frame Type	Security Enabled	Frame Pending	Ack. Request	PAN ID Compression	Reserved	Dest. Addressing Mode	Frame Version	Source Addressing Mode

Figure 376. Frame Control Field

The packet processor utilizes the Frame Control Fields subfields in the following manner:

FCF Subfield	Utilization by Packet Processor
FrameType	Interprets the remaining MHR as specific to Beacon, Ack, Data, Command, or Reserved frame types. Each frame type has a unique MHR structure, and so different parsing rules apply.
SecurityEnabled	If SecurityEnabled=1 and FrameVersion>0 (frame versions 2006 and later), an Auxiliary Security Header field will be present in the MHR and will need to be further parsed by the packet processor if this is a MAC Command frame (see Section Source Address Management below). There is no other use of SecurityEnabled by the packet processor.
FramePending	ignored by packet processor
Ack. Request	will be copied into the **pp_ack_request** signal to the sequence manager. An auto-TxAck frame will follow the incoming receive frame if necessary conditions are met
PAN ID Compression	used for addressing mode rules-checking and addressing field parsing.

Table continues on the next page...

Table continued from the previous page...

FCF Subfield	Utilization by Packet Processor
Reserved	ignored by packet processor
Destination Addressing Mode	used for addressing mode rules-checking and addressing field parsing
FrameVersion	For Beacon, Data, and MAC Command frames, FrameVersion is checked against the allowed frame versions according to FRM_VER_FILTER For Acknowledge frames, FrameVersion is ignored. If SecurityEnabled=1 and FrameVersion>0 (frame versions 2006 and later), an Auxiliary Security Header field will be present in the MHR and the MHR will need to be further parsed by the packet processor if this is a MAC Command frame (see Section Source Address Management).
Source Addressing Mode	used for addressing mode rules-checking and addressing field parsing

The Packet Processor parses the FrameType field of the Frame Control Field per the following table:

Frame type value b_2 b_1 b_0	Description
000	Beacon
001	Data
010	Acknowledgment
011	MAC command
100–111	Reserved

Figure 377. Frame Type Field

Directly following the Frame Control Field is the Sequence Number field. The Sequence Number field of the MHR is captured by the packet processor. If an auto-TxAck frame follows the incoming receive frame, the captured Sequence Number will be copied to the transmitted Acknowledge packet, and the Packet Processor will insert FrameVersion=00 into the Frame Control Field of the transmitted frame, regardless of the received FrameVersion subfield of the original frame.

The Addressing Fields follow the Sequence Number. The format of the Addressing Fields depends upon the Source and Destination Addressing Mode subfields of the Frame Control Field. The Addressing Modes are defined in the table below:

Addressing mode value b_1 b_0	Description
00	PAN identifier and address fields are not present.
01	Reserved.
10	Address field contains a 16-bit short address.
11	Address field contains a 64-bit extended address.

Figure 378. Addressing Mode Fields

The packet processor uses these Source and Destination Address Modes shown above, to extract the Source PAN ID and Address (if present), and the Destination PAN ID and Address (if present), from the MHR, according to the table below:

FrameType	Destination Addressing Mode	Source Addressing Mode	PanID-Compression	Addressing Fields
Acknowledge only (reject all other frame types)	0	0	0	None
Data or Command (reject Ack and Beacon frames)	2 or 3	0	0	DstPanID + DstAddr
Beacon (all devices), or Data or Command (PAN Coord Only) (reject Ack frames)	0	2 or 3	0	SrcPanID + SrcAddr
Data or Command (reject Ack and Beacon frames)	2 or 3	2 or 3	0	DstPanID + DstAddr + SrcPanId + SrcAddr
Reject all frames (Illegal as per Section 7.2.2.1.5)	0	0	1	-
Reject all frames (Illegal as per Section 7.2.2.1.5)	2 or 3	0	1	-
Reject all frames	0	2 or 3	1	-

Table continues on the next page...

Table continued from the previous page...

FrameType	Destination Addressing Mode	Source Addressing Mode	PanID-Compression	Addressing Fields
(Illegal as per Section 7.2.2.1.5)				
Data or Command (reject Ack or Beacon frames)	2 or 3	2 or 3	1	DstPanID + DstAddr + SrcAddr
Reject all frames (Illegal as per Section 7.2.2.1.6)	1	-	-	-
Reject all frames (Illegal as per Section 7.2.2.1.8)	-	1	-	-

The Source PAN ID and Address, and Destination PAN ID and Address, extracted by the packet processor, are then checked against the MACPANID, MACSHORTADDRS, and MACLONGADDRS register settings, depending on the FrameType of the incoming packet, to determine:

1. Is the addressing mode combination valid for this frame type?
2. Do the address fields indicate that the packet is indeed addressed to the end device?

The details on how the packet processor answers those 2 questions, are described in the following sections.

55.4.9.1.2.1.3.2 Packet Filtering Detail

The figure below is a high-level representation of the packet filtering performed by the 802.15.4 packet processor. The sections which follow provide more detail on the packet filter implementation.

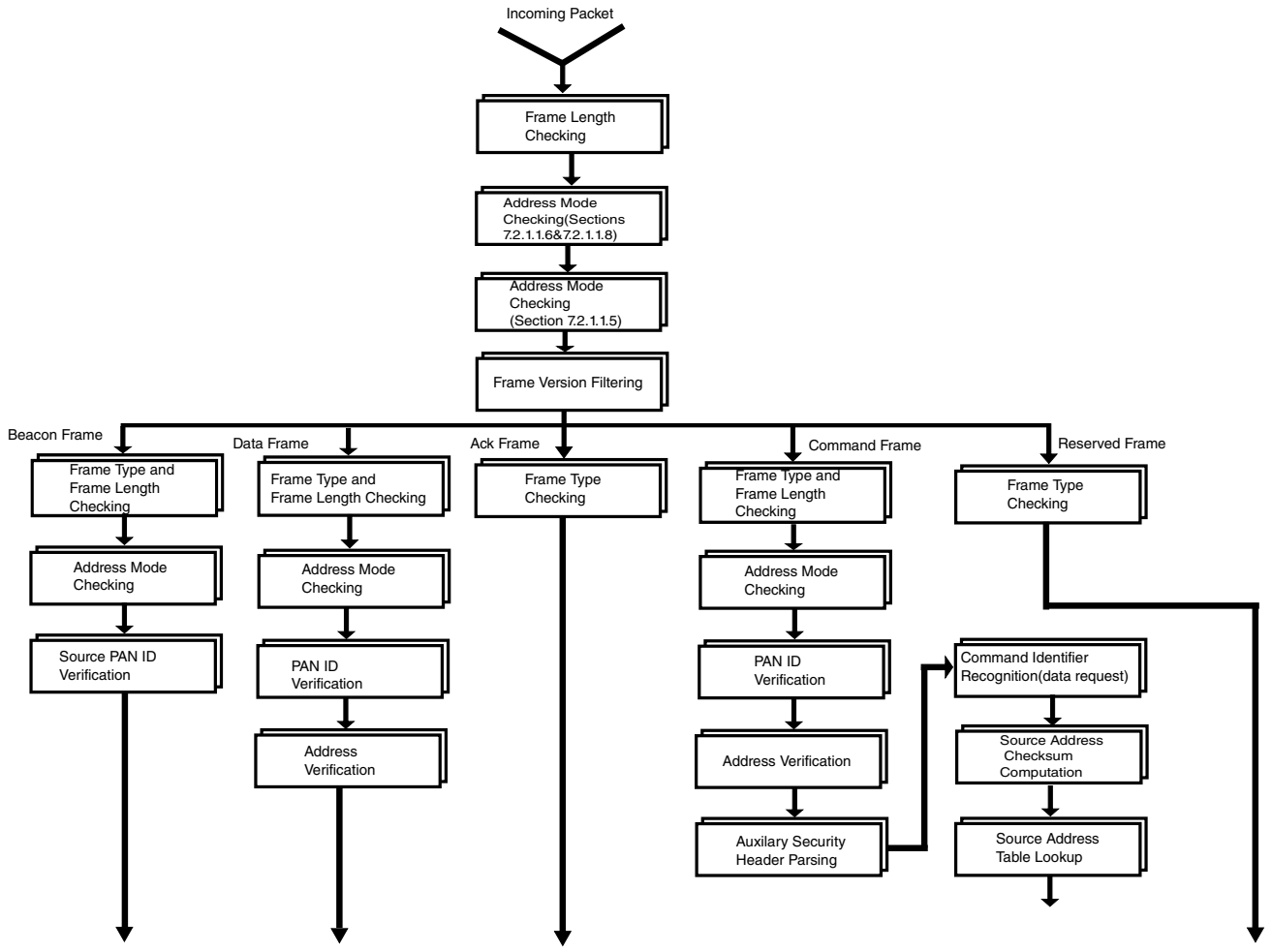


Figure 379. Packet Filtering Flow Diagram

First-Stage Packet Filtering

The first stage of packet filtering examines the incoming packet, and implements basic rule-checking, based on the 2006 version of the 802.15.4 standard, and is applied to *all* frame types. First stage rule-checking is described in the table below:

	Filtering Step	Description
1	Frame Length Checking	Check that all frames are at least 5 octets long (minimum frame length required by the standard), and no longer than 127 octets long (maximum frame length allowed by the standard).
2	Address Mode Checking (Sections 7.2.1.1.6 and 7.2.1.1.8)	Check that neither Source nor Destination Address Mode set to 1 (illegal addressing mode)
3	Address Mode Checking (Section 7.2.1.1.5)	Check for PAN ID Compression enabled, with either Source or Destination Address Mode set to 0 (illegal addressing mode)

Table continues on the next page...

Table continued from the previous page...

	Filtering Step	Description
4	FrameVersion Checking	Check that incoming packet's Frame Version is allowed based on FRM_VER_FILTER setting. (FRM_VER_FILTER is a programmable register, see Register Descriptions Section).

Packets which pass first stage filtering, proceed to the second stage of packet filtering, which is *frame type-specific* filtering. Packets which fail initial rule checking are marked as rejected.

Second-Stage Packet Filtering

The second stage of packet filtering is *frame type-specific*. Different variations of the MAC Header (MHR) apply to different frame types, so different parsing mechanisms apply to each.

Note: In the following section, the following references are to register bits; see Register Description Section for register descriptions: BEACON_FT, ACK_FT, DATA_FT, CMD_FT, NS_FT, PANCORDNTR, MACPANID, MACSHORTADDRS, and MACLONGADDRS.

1) For Beacon frames, these are the basic filtering steps:

- Check that Beacon frames are enabled by software (BEACON_FT)
- Check that FrameLength >= 9
- Check for illegal combinations of PAN ID Compression and Source/Destination Address Modes
- Verify the Source PAN ID matches MACPANID, or that MACPANID = 0xffff.

2) For Acknowledge frames, these are the basic filtering steps:

- Check that Ack frames are enabled by software (ACK_FT)
- If Sequence TR is active and RXACKRQD = 1, check the received Sequence Number matches the transmitted Sequence Number; FrameVersion bits are ignored, as per section 7.2.2.3.1 of the 802.15.4 standard.

Note: It is recommended to use Sequence TR to receive Acknowledge frames, not Sequence R. See Chapter ZSM Sequence Manager, Section Sequence TR (Transmit/Receive).

3) For Data frames, these are the basic filtering steps:

- Check that Data frames are enabled by software (DATA_FT)
- Check that FrameLength >= 9
- Check for illegal combinations of Source and Destination Address Modes
- If PANCORDNTR=0, check for missing Destination Address field or present Source Address field
- If no Destination Address field (PANCORDNTR=1 only), check Source PAN ID matches MACPANID
- If Destination Address present, check Destination PAN ID matches MACPANID or broadcast (0xffff)
- If Short Destination Address field, check address matches MACSHORTADDRS or broadcast (0xffff)
- If Long Destination Address field, check address matches MACLONGADDRS.

4) For MAC Command frames, these are the basic filtering steps:

- Check that Command frames are enabled by software (CMD_FT)
- Check that FrameLength >= 9

- Check for illegal combinations of Source and Destination Address Modes
- If PANCORDNTR=0, check for missing Destination Address field or present Source Address field
- If no Destination Address field (PANCORDNTR=1 only), check Source PAN ID matches MACPANID
- If no Destination Address field (PANCORDNTR=1 only), capture Source PAN ID and Source Address for later use
- If Destination Address present, check Destination PAN ID matches MACPANID or broadcast (0xffff)
- If Short Destination Address field, check address matches MACSHORTADDRS or broadcast (0xffff)
- If Long Destination Address field, check address matches MACLONGADDRS.
- At end of the addressing field, if incoming packet's FrameVersion=0 or SecurityEnabled=0, capture the next octet (Command Identifier)
- At end of the addressing field, if incoming packet's FrameVersion!=0 and SecurityEnabled=1, parse the Auxiliary Security Header, then capture the next octet (Command Identifier)
- If Command Identifier indicates the MAC Command is a data request, compute a "Source Address Checksum" to uniquely identify the sending device.

55.4.9.1.2.1.3.3 Source Address Management

The 802.15.4 wireless MAC standard envisions a scenario whereby an end device may interrogate a coordinator as to whether the coordinator is storing data (i.e., a pending message) for the end device. The situation arises in a beacon-enabled network, when a coordinator includes in its transmitted beacon frame the MAC address of the end device in its "Pending Address Fields". The "Pending Address Fields" are part of the required MAC payload of the beacon frame. The "Pending Address Fields" contain a list of end device addresses for which messages are pending. In this scenario, an end device which finds its address included in the "Pending Address Fields" of the received beacon frame, must respond to the beacon (coordinator) with a MAC Command of type "data request". Alternatively, in non-beacon-enabled networks, an end device may periodically wake up and "poll" a coordinator, to determine if a message is pending for the end device. In either case, the coordinator stores messages for its end devices in its "indirect queue". The coordinator must respond to an incoming MAC Command data request (which *must* have AckRequest=1 in its Frame Control Field), with an Acknowledge frame containing a FramePending subfield indicating the presence (or absence) of a message for the requesting end device, in the coordinator's indirect queue. For a coordinator built around a low-MIPS MCU, a significant amount of MCU time is typically required, simply to ascertain whether a given end device has a message pending or not; the indirect queue can be quite large, and for 8-bit MCU's, the queue must be searched byte-by-byte. This makes it difficult, or impossible, for an underpowered MCU to establish frame-pending status within the 192us RX-to-TX turnaround time required by the standard.

Also, coordinators may include, in software, a "Neighbor Table", the contents of which consist of all the end devices with which the coordinator is in communication. The neighbor table requires periodic maintenance by coordinator (parent) software, as end devices (children) come and go within the network, join different parents, or simply never return. Child tables have limited capacity and parents need to create space for new devices that may show up. A timeout mechanism is the only way that can guarantee that child tables will eventually be cleaned up. Coordinator software must "age out" end devices with which there has not been recent communication. In order for an end device to refresh its status within its parent's Neighbor Table, the end device may periodically send a "keepalive" message in the form of a MAC Data Poll Request. Upon receipt of such a poll request, the parent must search its neighbor table to determine if the source address of the end device is a valid table entry. If the end device exists in the table, an Acknowledgment packet must be transmitted to the end device, with the FramePending subfield set to 0, and coordinator software resets that end device's timeout counter; if the child is not present in the table, an Acknowledgment packet must be transmitted to the end device, with the FramePending subfield set to 1, and the parent device will follow up by transmitting a "Leave Packet" to the end device, instructing the end device to sever its child status with the parent. The end device may opt to rejoin at a later time. Due to the size of the parent's Neighbor table, the search a child's source address can be time consuming, and may not always be possible within the 192us RX-to-TX turnaround time required by the standard.

To alleviate both of these problems, the 802.15.4 packet processor includes hardware acceleration to expedite the source address table search, greatly relieving the software bottleneck during the critical 192us turnaround time. The table search acceleration consists of 2 components: 1) a search for the *presence* of a specific end device's source address, which expedites the process of determining whether a message is pending for the device; and 2) a search for the *absence* of a specific end device's source address, which expedites the process of determining whether the end device is a valid child of the parent.

The packet processor maintains a register table (128 deep, 16-bit wide). The table is divided into multiple partitions, one to perform the “Source Address Present” (SAP) search, and another to perform the “Source Address Absent” (SAA) search. In the SAP partition of the table, coordinator software will store the address of end devices for which it has a message pending; in the SAA partition of the table, coordinator software will store the address of end devices for which it is the parent. To reduce area, the address information is stored in a “compressed” format. Software compresses the address information, and stores it into the table partitions, during a non-critical time. Then, when the coordinator’s packet processor receives a MAC Command data request, it extracts the source address information from the incoming packet, and compresses it using the *same* compression algorithm used by software to populate the source address table. The packet processor performs 2 steps simultaneously:

1. it compares the just-received, compressed source address information, or “checksum”, against all of the enabled entries in the SAP partition of the table. If a match is found in the SAP partition, the packet processor *asserts* the **rx_frame_pending** signal to the transmit block (TX_PACKET), so that an auto-TxAck packet can be sent to the requesting end device, with the FramePending subfield set to 1, indicating that the end device has a message pending in the coordinator’s indirect queue. A message packet from the coordinator to the end device will follow.
2. it compares the just-received, compressed source address information, or “checksum”, against all of the enabled entries in the SAA partition of the table. If *no* match is found in the SAA partition, the packet processor *asserts* the **rx_frame_pending** signal to the transmit block (TX_PACKET), so that an auto-TxAck packet can be sent to the requesting end device, with the FramePending subfield set to 1, indicating that the end device is not a valid child of this parent, and a “Leave Packet” from the coordinator to the end device, will follow.

If neither condition is met, i.e, there is no address match in the SAP partition, and a positive match in the SAA partition, the packet processor *deasserts* the **rx_frame_pending** signal, so that an auto-TxAck packet can be sent indicating to the end device that it has no data pending, and that it remains a valid child of this parent; the end device may then deactivate its receiver immediately. For the packet processor, the entire table lookup occurs in 128 clock cycles, once the last source address octet has been received. The SAP and SAA lookups are performed simultaneously. The checksum is also computed on-the-fly by hardware as the source address octets are received, taking no more than 1 clock cycle per octet. Finally, table-search status is reported to the MCU:

1. after the packet processor completes its SAP table search, the status bit SAP0_ADDR_PRESENT will be set to 1 if a match is found in the SAP partition, and the table index which matched will be indicated in the SAP0_MATCH[6:0] register, to further assist software in locating the relevant message within its indirect queue; if no match is found in the SAP partition, the status bit SAP0_ADDR_PRESENT will be set to 0.
2. after the packet processor completes its SAA table search, the status bit SAA0_ADDR_ABSENT will be set to 1 if *no* match is found in the SAA partition; if a positive match is found in the SAA partition, the status bit SAA0_ADDR_ABSENT will be set to 0, and the table index which matched will be indicated in the SAA0_MATCH[6:0] register, to further assist software in locating the child device within its neighbor table.

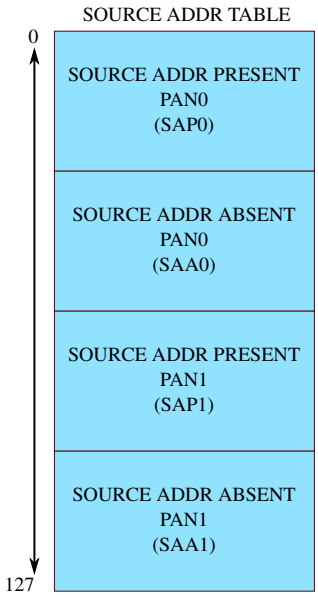
Both features, SAP and SAA acceleration, are optional, and functionally independent. Coordinator software can elect to enable both, either, or neither feature. The control bit SAP0_EN enables SAP acceleration. The control bit SAA0_EN enables SAA acceleration. If either feature is disabled, that table partition (SAP or SAA) is not searched, and so is not a factor in determining FramePending status for the transmitted Acknowledge frame. If *both* features are disabled, FramePending status defaults to software control, and merely tracks the state of the ACK_FRM_PND register bit; in this case software must manage FramePending status for all auto-TxAck frames.

Software can also take direct control of FramePending status for the next auto-TxAck frame at any time (regardless of whether SAP or SAA acceleration is enabled), by setting the ACK_FRM_PND_CTRL bit; in this case, hardware FramePending status will track the state of ACK_FRM_PND.

In order to support Dual PAN mode, a device could theoretically be a PAN Coordinator on both networks. Thus, a coordinator device may need to maintain distinct message pending lists and/or neighbor tables for the 2 PANs. Accordingly, the packet processor further subdivides the Source Address Management table into 4, instead of 2, partitions:

1. SAP0: Source Address Present Partition for PAN0
2. SAA0: Source Address Absent Partition for PAN0
3. SAP1: Source Address Present Partition for PAN1
4. SAA1: Source Address Absent Partition for PAN1

SOURCE ADDRESS TABLE PARTITIONS



Four register control bits enable each partition individually. If a partition is enabled, and a MAC poll request is received on the corresponding PAN (e.g. SAP0 or SAA0 partitions on PAN0), then the partition is searched by the packet processor, looking for a source address match (or absence), and the result of the search affects the FramePending status; if a partition is disabled, the partition is not searched, and will not influence FramePending status. The register bits are described in the table below.

REGISTER NAME	DESCRIPTION
SAP0_EN	Enables SAP0 Partition
SAA0_EN	Enables SAA0 Partition
SAP1_EN	Enables SAP1 Partition
SAA1_EN	Enables SAA1 Partition

For maximum flexibility in supporting all combinations of message-pending status, neighbor-table status, and dual PAN mode, the dividing-line between all the partitions is fully programmable. The SAP0 partition always starts at table index 0. Three registers control the starting point for the other 3 partitions. The starting point is defined the first table index (0-127) of that partition. The registers are described below, along with some software restrictions to ensure table consistency.

REGISTER NAME	DESCRIPTION	RESTRICTIONS
SAA0_START[6:0]	First Index of SAA0 partition.	If SAA0_START=0, then there is no SAP0 partition (SAP0_EN should be set to 0)
SAP1_START[6:0]	First Index of SAP1 partition.	Software must ensure that: **SAP1_START ≥ SAA0_START** If SAP1_START = SAA0_START, then there is no SAA0 partition (SAA0_EN should be set to 0)
SAA1_START[6:0]	First Index of SAA1 partition.	Software must ensure that:

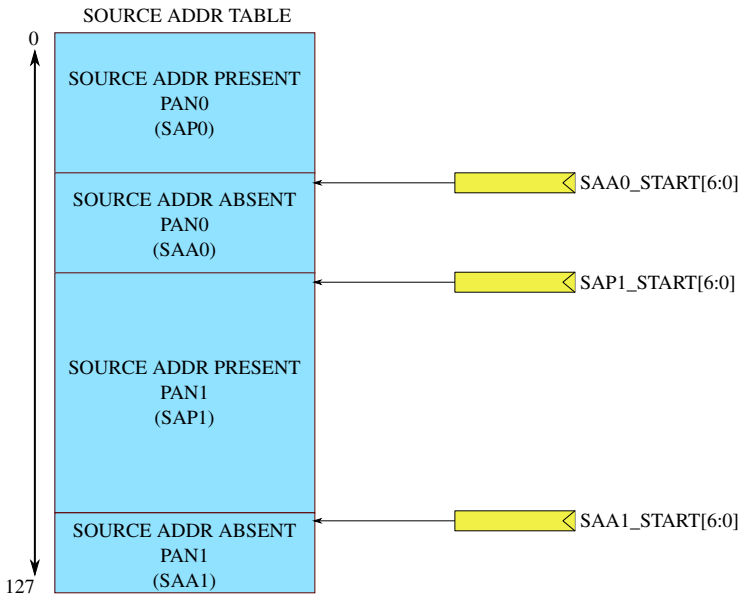
Table continues on the next page...

Table continued from the previous page...

REGISTER NAME	DESCRIPTION	RESTRICTIONS
		SAA1_START ≥ SAP1_START If SAA1_START = SAP1_START, then there is no SAP1 partition (SAP1_EN should be set to 0)

A diagram which depicts one example of how the dividing-line registers affect partitioning of the table, is shown below.

SOURCE ADDRESS TABLE PARTITIONS -- NON-EQUAL SIZES (EXAMPLE)



Each table partition consists of one or more indices, which can be used to store source address checksums. The first index for any partition, is defined by the aforementioned *PARTITION*_START[6:0] register, where *PARTITION*_ is one of {SAP0, SAP1, SAA0, SAA1}. The last index for any partition, is defined by the next higher partition's *PARTITION*_START[6:0] register, minus 1. When populating the table with source address information, software must be sure to select an index that is within the defined range for the partition. For example, when installing a checksum in the SAP1 partition:

SAP1_START ≤ Selected_Index ≤ (SAA1_START – 1)

To populate the table partitions, a 7-bit register SAM_INDEX[6:0] is provided, as well as a "index enable bit", and a "index invalidate" bit. A 16-bit SAM_CHECKSUM[15:0] is provided, into which software will write the computed checksum. Installing a checksum into any partition is referred to as "Enabling an Index", and removing a checksum from any partition is referred to as "Invalidating an Index". A description of the registers is provided in the following table:

REGISTER NAME	DESCRIPTION
SAM_CHECKSUM[15:0]	Software-computed source address checksum, to be installed into a table index
SAM_INDEX[6:0]	Contains the table index to be enabled or invalidated. Software must ensure that the index is within the range of the desired partition.

Table continues on the next page...

Table continued from the previous page...

REGISTER NAME	DESCRIPTION
SAM_INDEX_WR	For 32-bit writes, this must be set to indicate that the table entry specified by SAM_INDEX[6:0] is to be written; if SAM_INDEX_WR=0, the table entry is not written, but the SAM_INDEX[6:0] register is updated. For 8-bit writes, this bit is ignored.
SAM_INDEX_EN	Enable the index selected by SAM_INDEX[6:0] for searching.
SAM_INDEX_INV	Invalidate the index selected by SAM_INDEX[6:0]
FIND_FREE_IDX	After modifying Valid bits (enabling or invalidating), write this bit to 1 to force HW to update the “First Free Index” registers to account for the changed Valid bits. This HW update process takes 4us. SW can poll SAM_BUSY to determine when the table update is complete. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
SAM_BUSY	HW is in the process of updating the Source Address table, either in response to a poll indication from the packet processor, or due to SW setting FIND_FREE_IDX=1. In the latter case, SW should poll SAM_BUSY until low before accessing the “First Free Index” registers. Read-only bit.
INVALIDATE_ALL	Writing a 1 to this bit clears all 128 Valid bits. Invalidates the entire table. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.

All the index-control registers reside within a single 32-bit word, so that a single 32-bit write can store a new checksum in the table, and simultaneously enable, or invalidate, the index. For 32-bit writes, SAM_INDEX_WR must be set to 1 to cause the SAM_CHECKSUM[15:0] to be loaded into the table at the address specified by SAM_INDEX[6:0]. For 32-bit reads, it is necessary first to update SAM_INDEX[6:0] to the desired table index, if it is not already set to the desired index; this requires a write to SAM_INDEX[6:0] with SAM_INDEX_WR=0, so as to not update the table contents. Then, a subsequent 32-bit read will return the table contents at the desired index.

For systems that don’t support 32-bit writes, the register fields should be written in the following order:

1. SAM_INDEX[6:0]
2. SAM_CHECKSUM (the 2 bytes can be written in either order)
3. SAM_INDEX_EN (to enable), or SAM_INDEX_INV (to invalidate)

During operation, when a packet is received and the packet processor searches the table for a source address match, only enabled indices within each partition are searched. Invalidated indices are ignored.

To assist software in finding an available index to store the next source address checksum, the packet processor makes available the following 4 read-only registers:

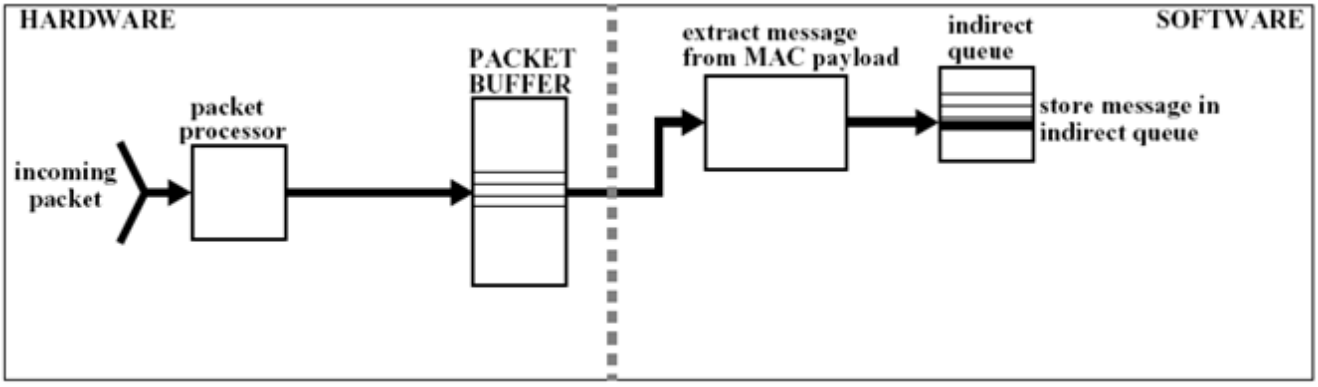
REGISTER NAME	DESCRIPTION
SAP0_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAP0 partition
SAA0_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAA0 partition
SAP1_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAP1 partition
SAA1_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAA1 partition

Operational Use

The following flow description illustrates how hardware and software interact to utilize the Source Addressing Management feature to facilitate pending-message handling and neighbor table management. This flow assumes both SAP and SAA acceleration are enabled, and Dual PAN mode is in effect. There are 11 sequential steps.

Step 1: Message Packet Reception, Extracting the Message, Indirect Queue Management

A message (data frame) is received by a coordinator. The message is received from one end device, and is intended for another end device. The coordinator's packet processor receives the packet, and stores it in the packet buffer. Coordinator software downloads the packet data from the packet buffer, extracts the message from the MAC payload, and stores the message in its indirect queue.



Step 2: Software computes address checksum

Coordinator software also extracts the destination information (Destination PANID and Destination Address), from the data packet stored in the packet buffer. Software uses this information to compute a 16-bit checksum. This checksum uniquely identifies the sending device, with a high degree of reliability. The checksum is computed according to the following rules:

Destination Addressing Mode	Rule
Mode 2 (short address)	Checksum = (Destination PAN ID + DstAddr[15:0]) % 65536
Mode 3 (long address)	Checksum = (Destination PAN ID + DstAddr[15:0]) % 65536 Checksum = (Checksum + DstAddr[31:16]) % 65536 Checksum = (Checksum + DstAddr[47:32]) % 65536 Checksum = (Checksum + DstAddr[63:48]) % 65536

Step 3: Software populates the address table

Coordinator software chooses an index in the Source Address Table in which it will store the just-computed checksum. If the data packet was received on PAN0, an index in the SAP0 partition should be chosen. Software may consult the register SAP0_1ST_FREE_IDX[6:0] to find the first available index in this partition. If the data packet was received on PAN1, an index in the SAP1 partition should be chosen. Software may consult the register SAP1_1ST_FREE_IDX[6:0] to find the first available index in this partition.

If an index is chosen which is already valid, software should first invalidate the index with the following procedure:

1. Set SAM_INDEX[6:0] to the selected index
2. Set SAM_INDEX_EN=0 and SAM_INDEX_INV=1

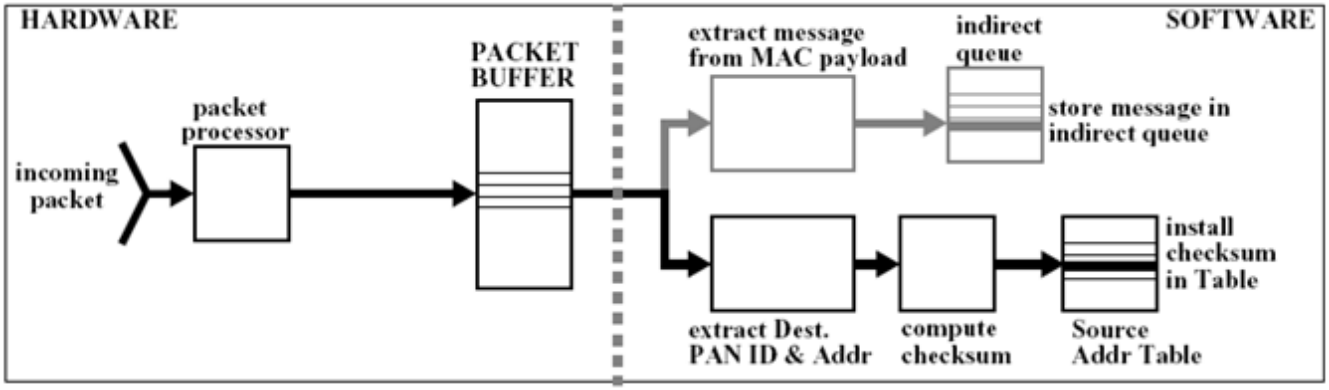
These steps can be performed in a single, 32-bit register write.

To install the source address checksum into the index and enable the index, the following procedure should be used:

1. Program SAM_CHECKSUM[15:0] to the computed checksum
2. Set SAM_INDEX[6:0] to the selected index
3. Set SAM_INDEX_EN=1 and SAM_INDEX_INV=0

These steps can be performed in a single, 32-bit register write.

The index is now enabled, and the packet processor will include it in its search when the next MAC Poll Request is received.



Step 4: Incoming Poll request, Packet Processor parses the Auxiliary Security Header

An incoming packet is received. The packet processor performs routine packet filtering. After determining that the packet is type 'MAC Command', the packet processor must perform additional packet parsing in order to locate the packet's Command Frame Identifier. This is the octet that distinguishes this command as being of type "data request", from other MAC Command types. For the packet processor, this extra parsing is more complex, because the Command Frame Identifier is part of the MAC payload; this is the only scenario where the packet processor is required to parse the payload. This is further complicated by the fact that the revision 2006 (and subsequent) of the 802.15.4 wireless MAC standard, includes a variable-length "Auxiliary Security Header", prior to the MAC payload. This header is only present for FrameVersion > 0, and SecurityEnabled = 1. The packet processor must conditionally parse this header in order to locate the start of the MAC payload, where the Command Frame Identifier is located. The packet processor uses 2 subfields of the Frame Control Field to parse the Auxiliary Security Header (A.S.H.): FrameVersion, and SecurityEnabled. The packet processor parses the Auxiliary Security Header (A.S.H.) according to the following table:

Length of Auxiliary Security Header (Octets)	SecurityEnabled (Frame Control Field)	FrameVersion (Frame Control Field)	KeyIdentifierMode (Security Control Field)	Remarks
0	0	00 or 01	-	If !SecurityEnabled, no ASH regardless of FrameVersion
0	1	00	0	If packet is 2003-compliant, no ASH
5 Security Control (1 octet) + Frame Counter (4 octet)	1	01	0x00	ASH present. Length of ASH computed by HW.
6 Security Control (1 octet) +	1	01	0x01	ASH present. Length of ASH computed by HW.

Table continues on the next page...

Table continued from the previous page...

Length of Auxiliary Security Header (Octets)	SecurityEnabled (Frame Control Field)	FrameVersion (Frame Control Field)	KeyIdentifierMode (Security Control Field)	Remarks
Frame Counter (4 octet) + Key Identifier (1 octet)				
10 Security Control (1 octet) + Frame Counter (4 octet) + Key Identifier (5 octet)	1	01	0x2	ASH present. Length of ASH computed by HW.
14 Security Control (1 octet) + Frame Counter (4 octet) + Key Identifier (9 octet)	1	01	0x3	ASH present. Length of ASH computed by HW.
-	-	10 or 11	-	HW will handle these FrameVersion's indentically to FrameVersion 01

The contents of the A.S.H. are irrelevant to the packet processor; only its length is important. This is because the octet which immediately follows the A.S.H. is the Command Frame Identifier. This is all the packet processor needs to know.

Step 5: Hardware extracts the Command Frame Identifier

After parsing the A.S.H. (if present), the next octet in the MAC payload, in accordance with the 802.15.4 standard, is the Command Frame Identifier.

Command frame identifier	Command name	RFD		Subclause
		Tx	Rx	
0x01	Association request	X		5.3.1
0x02	Association response		X	5.3.2
0x03	Disassociation notification	X	X	5.3.3
0x04	Data request	X		5.3.4
0x05	PAN ID conflict notification	X		5.3.5
0x06	Orphan notification	X		5.3.6
0x07	Beacon request			5.3.7
0x08	Coordinator realignment		X	5.3.8
0x09	GTS request			5.3.9
0x0a–0xff	Reserved			—

The packet processor checks to see if the Command Frame Identifier = 0x4. If it is not, packet processing for this frame is complete, the remainder of the packet is received, without inspection by the packet processor, and is stored the packet buffer. If the Command Frame Identifier = 0x4, then this is a data request, and assuming the prior rules-checking and address-checking on this packet have already passed, then the Source Address Matching function will be applied to this packet, continuing in STEP 6.

Step 6: Hardware Computes the Source Address Checksum

The packet processor uses the received Source PAN ID and Source Address for this packet to compute a 16-bit Source Address checksum. This checksum uses precisely the same algorithm that software uses to compute its Address checksum (see STEP 2), except that the *Source* PAN ID and address are used, rather than the Destination PAN ID and address. This checksum uniquely identifies the sending device, with a high degree of reliability. Note if the incoming packet's Frame Control Field has the PANIDCompression subfield asserted, the packet processor will substitute "Destination PAN ID" for "Source PAN ID" in the computation, since Source PAN ID will not be present under these circumstances.

Step 7: Hardware performs "Source Address Present" Table Search

If the MAC poll request was received on PAN0, and SAP0_EN=1 to enable searches of the PAN0 "Source Address Present" partition, the packet processor uses its Source Address checksum to look for a match in the SAP0 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If a single table match is found (i.e., 1 software-computed checksum in SAP0 partition matches the hardware-computed checksum for the incoming packet), this means that there is a message for this requestor in the coordinator's message queue. For a SAP0 match, the packet processor takes these 2 steps:

1. Set SAP0_ADDR_PRESENT=1
2. Set SAP0_MATCH[6:0] to the index which matched

If multiple table matches are found (i.e., more than 1 software-computed checksum in the SAP0 partition matches the hardware-computed checksum for the incoming packet), this means that there are multiple messages for this requestor in the coordinator's message queue. In this case, the packet processor takes the same steps as for a single match, with the SAP0_MATCH[6:0] register being set to the lowest-value matching index (i.e., the smallest number) within the SAP0 partition.

If no source address match is found in the SAP0 table search, the packet processor will set SAP0_ADDR_PRESENT=0.

If, on the other hand, the MAC poll request was received on PAN1, and SAP1_EN=1 to enable searches of the PAN1 “Source Address Present” partition, the packet processor uses its Source Address checksum to look for a match in the SAP1 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If a single table match is found (i.e., 1 software-computed checksum in SAP1 partition matches the hardware-computed checksum for the incoming packet), this means that there is a message for this requestor in the coordinator’s message queue. For a SAP1 match, the packet processor takes these 2 steps:

1. Set SAP1_ADDR_PRESENT=1
2. Set SAP1_MATCH [6:0] to the index which matched

If multiple table matches are found (i.e., more than 1 software-computed checksum in the SAP1 partition matches the hardware-computed checksum for the incoming packet), this means that there are multiple messages for this requestor in the coordinator’s message queue. In this case, the packet processor takes the same steps as for a single match, with the SAP1_MATCH[6:0] register being set to the lowest-value matching index (i.e., the smallest number) within the SAP1 partition.

If no source address match is found in the SAP1 table search, the packet processor will set SAP1_ADDR_PRESENT=0.

Step 8: Hardware performs “Source Address Absent” Table Search

If the MAC poll request was received on PAN0, and SAA0_EN=1 to enable searches of the PAN0 “Source Address Absent” partition, the packet processor uses its Source Address checksum to look for a match in the SAA0 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If one (or more) table match is found (i.e., a software-computed checksum in SAA0 partition matches the hardware-computed checksum for the incoming packet), this means that the end device which sent the poll request is a valid child of this parent. For a SAA0 match, the packet processor takes these 2 steps:

1. Set SAA0_ADDR_ABSENT=0
2. Set SAA0_MATCH[6:0] to the first index which matched

If *no* source address match is found in the SAA0 table search, the packet processor will set SAA0_ADDR_ABSENT=1.

If, on the other hand, the MAC poll request was received on PAN1, and SAA1_EN=1 to enable searches of the PAN1 “Source Address Absent” partition, the packet processor uses its Source Address checksum to look for a match in the SAA1 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If one (or more) table match is found (i.e., a software-computed checksum in SAA1 partition matches the hardware-computed checksum for the incoming packet), this means that the end device which sent the poll request is a valid child of this parent. For a SAA1 match, the packet processor takes these 2 steps:

1. Set SAA1_ADDR_ABSENT=0
2. Set SAA1_MATCH[6:0] to the first index which matched

If *no* source address match is found in the SAA1 table search, the packet processor will set SAA0_ADDR_ABSENT=1.

Step 9: Hardware determines FramePending status

The packet processor maintains an internal flag called **rx_frame_pending**, which is sent to the TX_PACKET block after every received packet, to directly control the state of the FramePending subfield of the Frame Control Field in the upcoming auto-TxAck frame (if enabled with AUTOACK=1).

If the Source Address Management feature is completely disabled, i.e.,

SAP0_EN = SAA0_EN = SAP1_EN = SAA1_EN = 0

then the packet processor will set the **rx_frame_pending flag** to the value of the ACK_FRM_PND register, putting the state of FramePending under direct software control.

Otherwise, if *any* of the following conditions are met, the packet processor will set its **rx_frame_pending** flag to 1:

1. The MAC Poll Request was received on PAN0, SAP0_EN=1, and SAP0_ADDR_PRESENT=1
2. The MAC Poll Request was received on PAN1, SAP1_EN=1, and SAP1_ADDR_PRESENT=1
3. The MAC Poll Request was received on PAN0, SAA0_EN=1, and SAA0_ADDR_ABSENT=1
4. The MAC Poll Request was received on PAN1, SAA1_EN=1, and SAA1_ADDR_ABSENT=1

If none of the conditions are met, **rx_frame_pending** flag will be set to 0.

Step 10: Receive interrupt RXIRQ

The packet processor receives the remainder of the packet, and verifies the FCS. Assuming the CRC check passes, the RXIRQ interrupt is issued. In the coordinator software's RXIRQ Interrupt Service Routine, software can determine if the received packet was a MAC Poll Request by reading the PI bit of the IRQSTS register. If PI=0, the received packet was not a poll request, and no action relating to Source Address Matching need be taken.

Otherwise, if *either* SAP0_ADDR_PRESENT=1 or SAP1_ADDR_PRESENT=1, an auto-TxAck packet is being sent with the FramePending subfield set to 1, so software should follow up by constructing a data packet (or packets) for the requesting end device, with a payload consisting of the message for this device which had been stored in the coordinator's indirect queue.

Also, if *either* SAA0_ADDR_ABSENT=1 or SAA1_ADDR_ABSENT=1, an auto-TxAck packet is being sent with the FramePending subfield set to 1, so software should follow up by constructing a "Leave Message" for the requesting end device.

Step 11: Software disables the spent index

In the final step, coordinator software must disable (invalidate) the index which was just used to locate and send the stored message to the requesting end device. To disable the index:

1. Set SAM_INDEX[6:0] to the selected index
2. Set SAM_INDEX_EN=0 and SAM_INDEX_INV=1

At this point, the disabled index is now an available table entry, which can be used to install a new Address checksum for another message in the indirect queue.

55.4.9.1.2.1.3.4 Dual PAN Mode

In the past, radio transceivers designed for 802.15.4 applications allow a device to associate to one and only one PAN (Personal Area Network), at any given time. The Packet Processor includes hardware support for a device to reside on two networks simultaneously. In optional Dual PAN mode, the device will alternate between the 2 PAN's, under hardware or software control. Hardware support for Dual PAN operation consists of 2 sets of PAN and IEEE addresses for the device, 2 different channels (one for each PAN), a programmable timer to automatically switch PAN's (including on-the-fly channel-changing) without software intervention, control bits to configure and enable Dual PAN mode, and read-only bits to monitor status in Dual PAN mode. A device can be configured to be a PAN Coordinator on either network, both networks, or neither.

For the purpose of defining a "PAN" in the context of Dual PAN mode, two sets of network parameters are maintained. In this document, "PAN0" and "PAN1" will be used to refer to the 2 PAN's. Each parameter set uniquely identifies a PAN for Dual PAN mode. These parameters are:

PAN0	PAN1
CHANNEL_NUM0	CHANNEL_NUM1
MACPANID0 (16-bit register)	MACPANID1 (16-bit register)

Table continues on the next page...

Table continued from the previous page...

PAN0	PAN1
MACSHORTADDRS0 (16-bit register)	MACSHORTADDRS1 (16-bit register)
MACLONGADDRS0 (64-bit registers)	MACLONGADDRS1 (64-bit registers)
PANCORDNTR0 (1-bit register)	PANCORDNTR1 (1-bit register)

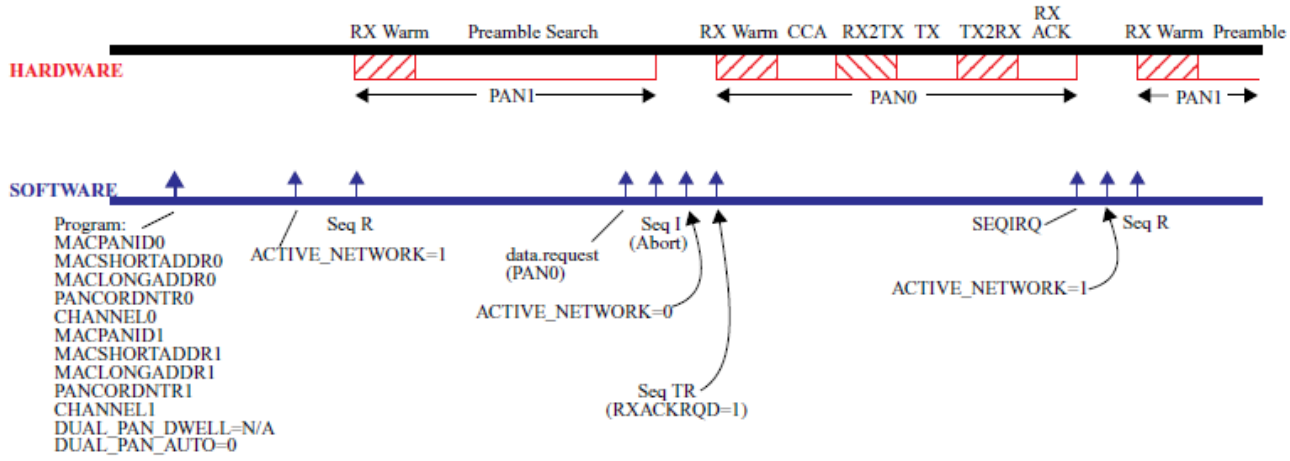
During device initialization, if Dual PAN mode is to be utilized, software will program both parameters sets, to configure the hardware for operation on two networks.

Two modes are available for Dual PAN operation: manual and automatic.

Manual Dual PAN

In manual Dual PAN mode, software controls which PAN is selected for transmission and reception, at all times. Software populates both PAN network parameter-set registers listed above. Software then selects a PAN to begin transmission or reception. A single control bit **ACTIVE_NETWORK**, selects the PAN. To select PAN0, **ACTIVE_NETWORK** should be set to 0; to select PAN1, **ACTIVE_NETWORK** should be set to 1. Software then initiates an autosequence by writing the appropriate value to XCVSEQ (PHY_CTRL1 Register). At any point, software can elect to switch to the alternate PAN. To do so, software aborts the sequence (if one is active) by writing XCVSEQ=IDLE. Software then toggles **ACTIVE_NETWORK**, and restarts a new sequence with XCVSEQ. Software is responsible for aborting and initiating sequences. Hardware is responsible for warming up the transceiver on the selected channel (CHANNEL_NUM0 or CHANNEL_NUM1), and performing address-filtering on the selected network's address parameters (PANCORDNTR, MacPanID, Short Address, and/or IEEE address). Software overhead is significantly reduced in this mode, since there is no need to re-program all of the the PAN-select registers every time the PAN is toggled in Dual PAN mode; only a single bit (**ACTIVE_NETWORK**) need be written. For manual Dual PAN operation, **DUAL_PAN_AUTO** should be set to 0.

The following diagram depicts Dual PAN operation in manual mode. In this example, software populates both PAN parameters sets during initialization. In this example, software elects to initiate packet reception on PAN1, so **ACTIVE_NETWORK** is set to 1. Software initiates the Sequence R by writing to XCVSEQ. Hardware warms up the receiver on CHANNEL_NUM1 and enters preamble-search. In this example, no packet is received, but a data request from higher-level software requires a switch to PAN0. Software responds by aborting the reception by writing Sequence I to XCVSEQ. Software then sets **ACTIVE_NETWORK** to 0 to select PAN0. In the example, software initiates a Sequence TR. Hardware warms up the receiver for CCA on CHANNEL_NUM0, performs CCA (channel is idle), warms down the receiver, and warms up the transmitter on CHANNEL_NUM0, transmits the requested packet, warms down the transmitter and warms up the receiver again (as per the autosequence) on CHANNEL_NUM0, to receive the acknowledge packet. The acknowledge packet is received and the autosequence completes successfully with RXIRQ and SEQIRQ. Software elects to resume packet reception on PAN1 (which was in progress before the data request). Software toggles **ACTIVE_NETWORK** back to 1, and re-initiates Sequence R.



Auto Dual PAN

In automatic Dual PAN mode, hardware is responsible for alternating the selected PAN, at a pre-programmed interval. Software programs a "PAN dwell time", which determines the amount of time during which the hardware will receive on a given PAN. This is done by programming the **DUAL_PAN_DWELL** register. Software sets the **DUAL_PAN_AUTO** bit to 1 for automatic Dual PAN operation, and sets **ACTIVE_NETWORK** to correspond to the PAN on which it desires packet reception to begin. Software then initiates a Sequence R. Hardware warms up the receiver on the selected PAN, and begins preamble search. When the dwell time expires, assuming preamble and SFD have not been detected, hardware will switch to the alternate PAN. This includes an on-the-fly channel change. Once switched to the alternate PAN, hardware will use the address filtering parameters (PANCORDNTR, MacPanID, Short Address, and/or IEEE address) programmed for the alternate PAN. Once the dwell time expires on the alternate PAN, assuming preamble and SFD have not been detected, hardware will switch back to the original PAN. This process will repeat indefinitely, until one of the following occurs:

1. A packet is successfully received (address match and CRC valid).
2. Software aborts the Sequence R
3. A TMR3 RX timeout, which automatically aborts the Sequence R
4. A PLL unlock automatically aborts the Sequence R (if unlock aborting is enabled)

If the hardware PAN dwell timer expires *during* a packet reception (i.e., preamble and SFD have been detected during a Sequence R), the PAN-switch will be delayed until the packet is completely received. If the packet is received successfully (address match and CRC valid), the receive sequence will terminate with RXIRQ and SEQIRQ (if an auto-Acknowledge was requested and enabled, the sequence will complete after the Acknowledge packet was transmitted, and TXIRQ will also be set, in addition to RXIRQ and SEQIRQ); the PAN switch will occur at this point, after the Sequence has returned to IDLE.

If the received packet fails FCS (CRC check), or, if the packet was not intended for the device (address mismatch), the hardware will toggle the PAN, and return to preamble-search on the new PAN.

If the hardware PAN dwell timer expires during a state *other* than RX preamble search during Sequence R (i.e., if expiration occurs during TX, CCA, ED, Sequence TR, or any warmup or transitional state), PAN-switch will be delayed until either:

1. Preamble Search state is reached during a Sequence R (not Sequence TR)
2. Sequence returns to IDLE.

At any point, software can elect to interrupt the automatic Dual PAN operation, by aborting the sequence with a XCVSEQ=IDLE, and the setting the **DUAL_PAN_AUTO** bit to 0. In response, hardware will freeze the current state of the dwell timer, and capture the currently-selected PAN, so that hardware-controlled PAN alternation can resume where it left off, later. Software can then tend to that tasks that caused the Dual PAN interruption. When software wants to resume Dual PAN operation, it then sets **DUAL_PAN_AUTO**=1, and re-initiates Sequence R. Hardware will resume the dwell timer from the point it was

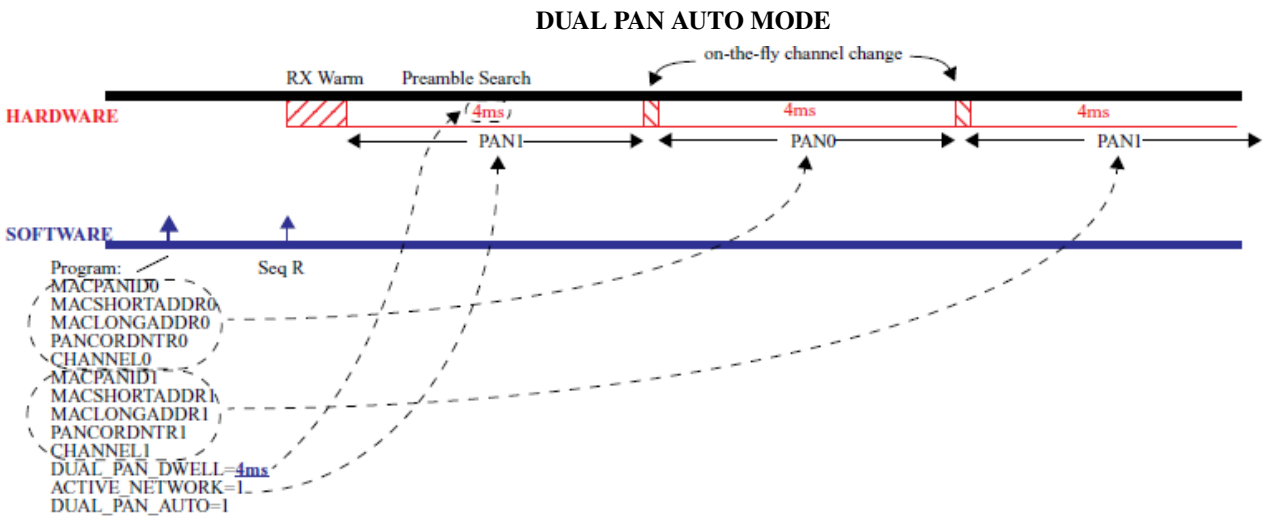
frozen earlier, on the same PAN, and will warmup the receiver to receive on the PAN which was being monitored before the interruption. Hardware will dwell on that PAN until the timer expires, and then switch PAN's (assuming no preamble and SFD detected, subject to the conditions listed above). Software can always determine the selected PAN at any time, in any mode, by reading **CURRENT_NETWORK** bit. If **DUAL_PAN_AUTO**=0, then **CURRENT_NETWORK** = **ACTIVE_NETWORK**. If **DUAL_PAN_AUTO**=1, then **CURRENT_NETWORK** is controlled by hardware.

To initiate auto Dual PAN operation for the first time, or to reset the dwell timer to its programmed value and allow software to select the initial PAN for auto Dual PAN operation, software should program the desired initial PAN to **ACTIVE_NETWORK** (0 for PAN0, 1 for PAN1), and then program the desired dwell time into the **DUAL_PAN_DWELL** register. Hardware always responds to a write to **DUAL_PAN_DWELL**, by resetting the dwell timer to its programmed value, and selecting the initial PAN for auto Dual PAN operation, based on the state of **ACTIVE_NETWORK**. A write to **DUAL_PAN_DWELL**, always re-initializes the DWELL TIMER to the programmed value. If a write to **DUAL_PAN_DWELL** occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to **DUAL_PAN_DWELL** occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the autosequence warmup. Under all circumstances, **DUAL_PAN_AUTO** must be asserted to allow the DWELL TIMER to count. If **DUAL_PAN_AUTO**=0, the DWELL TIMER will freeze and hold its state, until **DUAL_PAN_AUTO**=1. However, a write to **DUAL_PAN_DWELL** will still initialize the DWELL TIMER to its programmed value, regardless of the state of **DUAL_PAN_AUTO**.

At any point during auto Dual PAN mode, software can ascertain the PAN currently being serviced by hardware, by reading the **CURRENT_NETWORK** bit. Software can also determine the time-remaining in the current dwell, by reading the **DUAL_PAN_REMAIN** Register.

In auto Dual PAN mode, software retains responsibility for initiating and aborting sequences. There is no provision for hardware-initiated sequences. TMR2 may be used to initiate an autosequence at a future time. TMR3 may be used as an RX hardware-abort mechanism.

The following diagram depicts Dual PAN operation in automatic mode. In this example, software populates both PAN parameters sets during initialization. In this example, software wants to initiate packet reception in auto Dual PAN mode, on PAN1, so **ACTIVE_NETWORK** is set to 1. For auto Dual PAN operation, software programs the desired PAN-switching interval into **DUAL_PAN_DWELL** register, in this case, 4ms. Software initiates the Sequence R by writing to XCVSEQ. The dwell timer begins counting as soon as the receiver warmup begins. Hardware warms up the receiver on CHANNEL_NUM1 (since PAN1 was selected as the initial PAN in this example), and begins preamble search. If a packet is received during this interval, hardware applies PAN1 address filtering. In this example, no preamble is detected at the point at which the dwell timer expires. So, at this point, hardware executes a PAN-switch to PAN0. which includes an on-the-fly channel change. The on-the-fly channel change requires approximately 93us and no preamble detection is possible during this "blind spot". (For more details on how the hardware executes the Dual PAN on-the-fly channel change, see Chapter ZSM Sequence Manager, Section Dual PAN Mode.) After the PAN switch, hardware resumes preamble search on CHANNEL_NUM0, and applies PAN0 address filtering if a packet is received. In this example, no preamble is detected at the point at which the dwell timer expires. So, at this point, hardware performs a PAN-switch back to PAN1. And so on and so forth.



Two-PAN-one-channel

A special case occurs if a device is operating in Dual PAN mode (either Auto or Manual mode), and both PAN's occupy the same channel; in other words, CHANNEL_NUM0=CHANNEL_NUM1. In this case, hardware will detect this condition automatically, and apply both sets of address-filtering parameters simultaneously. Hardware will provide Data Indication if the received packet matches either the PAN0 address parameter set {**MACPANID0, MACSHORTADDRS0, MACLONGADDRS0, PANCORDNTR0**}, or the PAN1 address parameter set {**MACPANID1, MACSHORTADDRS1, MACLONGADDRS1, PANCORDNTR1**}. At Data Indication, two status bits can be queried by software to quickly determine on which PAN the packet was received, **RECD_ON_PAN0** or **RECD_ON_PAN1**. In the "Two-PAN-one-channel" scenario, it is possible for a packet to satisfy both sets of addressing parameters (PAN0 and PAN1). When this happens, both **RECD_ON_PAN0** and **RECD_ON_PAN1** will be set. In the "Two-PAN-one-channel" scenario, the **CURRENT_NETWORK** bit should be ignored, since both networks are simultaneously active.

If Dual PAN mode is not to be utilized, **ACTIVE_NETWORK** should be maintained at 0 (default) to select the PAN0 parameters for transceiving. In single PAN operation, the PAN1 parameter registers need not be programmed, and can be used as scratchpad. CHANNEL_NUM1 should not be programmed, and should instead be left in its default state, if Dual PAN mode is not in use.

Dual PAN Channel Override

In Dual PAN mode, in case there is a need to generate a frequency which may be offset from the 16 prescribed 5MHz-spaced channels, to, for example, avoid interference on one of the Dual PAN channels, a method has been provided to do that, by designating one of the two PAN channels to use the transceiver's set of direct frequency-programming registers, instead of CHANNEL_NUMx. Programming the direct frequency-programming registers – integer, numerator, and denominator, allows an RF frequency to be selected with much more precision than the 5MHz granularity of the 802.15.4 mapped-channel registers, CHANNEL_NUM0 and CHANNEL_NUM1. This is referred to as "Dual PAN Channel Override". For more details on this feature, see Chapter ZSM Sequence Manager, Section Dual PAN Channel Override.

55.4.9.1.2.1.3.5 Active Promiscuous Mode

Functional description

The 802.15.4 Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs) defines the following behavior for Promiscuous Mode operation:

> "In promiscuous mode, the MAC sublayer shall pass all frames received after the first filter directly to the upper layers without applying any more filtering or processing."

Where first level filtering is described as:

> "For the first level of filtering, the MAC sublayer shall discard all received frames that do not contain a correct value in their FCS field in the MFR (see 7.2.1.9)"

Promiscuous mode allows a device to receive and process all 802.15.4 frames which pass FCS check (CRC), regardless of whether or not they are addressed to the device. This allows a device to receive and process all valid traffic on a network. In Promiscuous mode, the Packet Processor will provide a Data Indication on all such packets. This capability allows a simple network sniffer to be constructed, which can display comprehensive packet data for all packets transmitted on a network within receiving range of the device.

Since a device operating in promiscuous mode is designed to do so discreetly, automatic acknowledgement of all received packets is inhibited, regardless of the state of the Ack Request field in the Frame Control Field of the received packet; otherwise, the sniffing device would be acknowledging received frames for which it was not the addressee.

The Packet Processor features an "Active Promiscuous Mode". In this mode, the Packet Processor will process packets according to the same rules that it applies to Promiscuous mode, except that it will automatically transmit an acknowledgement packet *for frames that are addressed to the device*. In other words, in Active Promiscuous mode, if the received frame meets all the address and packet filtering requirements that would have applied in non-promiscuous mode, an acknowledgement packet will be transmitted; a Data Indication will still be provided for all valid packets (those which pass FCS check). Note that all existing packet processor rules governing transmission of acknowledge packets, are still in effect; i.e., AUTOACK bit of the PHY_CTRL1 Register must be set to 1, and the Ack Request field in the Frame Control Field of the received packet must be set to 1.

The Packet Processor can be also programmed to provide a Data Indication for packets which fail FCS check, by setting CRC_MSK=0. Received packets which fail FCS check are never acknowledged, under any circumstances.

Special Handling for Broadcast Packets

Special rules apply to received packets which contain a broadcast PAN Identifier (0xffff), and/or a broadcast short address (0xffff). According to the 802.15.4 standard,

> “For valid frames that are not broadcast, if the Frame Type subfield indicates a data or MAC command frame and the Acknowledgment Request subfield of the Frame Control field is set to one, the MAC sublayer shall send an acknowledgment frame."

An exception is made for received packets which contain a broadcast PAN Identifier, and a long address which matches the device’s IEEE address. Such packets are acknowledged, because the IEEE address uniquely identifies the device, regardless of PAN. This special case will apply also in Active Promiscuous mode; such packets will be acknowledged. The following diagram describes the packet processor’s "acknowledgement decision" for all permutations of broadcast indicators in the received packet’s addressing fields.

ACKNOWLEDGE-ON-BROADCAST (ALL PERMUTATIONS)

DESTINATION ADDRESS MODE	PAN ID = !BROADCAST DSTADDR = !BROADCAST	PAN ID = !BROADCAST DSTADDR = BROADCAST	PAN ID = BROADCAST DSTADDR = BROADCAST	PAN ID = BROADCAST DSTADDR = !BROADCAST
SHORT	ACK	NO ACK	NO ACK	NO ACK
LONG	ACK			ACK

Special Handling for Broadcast Packets

The Packet Processor features an autosequence to automatically receive, and verify, Acknowledge Frames after a Transmit frame has been sent by the device. This is accomplished by programming a Sequence TR with RXACKRQD=1. During normal operation, these receive Acknowledge Frames are not stored in the Packet Buffer. They are merely checked for matching Sequence Number and Frame Version, with a Data Indication issued for a valid match on both. In Active Promiscuous Mode, receive Acknowledge frames *will* be stored into the Packet Buffer, just like any other received frame. This saves software from having to re-construct the contents of the Acknowledge packet, from the contents of the packet which was originally transmitted. Ordinary Promiscuous Mode should therefore not be used with Sequence TR with RXACKRQD=1. Use Active Promiscuous instead.

55.4.9.1.2.1.3.6 Provisions to Enable Software Support for 802.15.4e and 802.15.4-2015

Addendum 802.15.4e to the 802.15.4-2011 standard introduces 3 new variations on the 802.15.4 MAC Frame Structure:

1. FrameVersion=2 (Frame Types: Beacon, Acknowledge, Data, and MAC Command)
2. LLDN Frame Type
3. Multipurpose Frame Type

In addition, the 802.15.4-2015 revision to the standard introduces a new Frame Type: Extended (Frame Type = 0b111).

All three of the “4e-compliant packets”, as well as the Extended Frame Type, modify the format and definition of the Frame Control Field in such a way as to render parsing of the MHR by the packet processor impossible, without major upgrades to the hardware. For example, the packet processor hardware does not support 8-bit (simple) addressing, nor Sequence Number Suppression.

As in interim solution, provisions have been added to the packet processor to allow these new packet variations to be recognized and conditionally admitted (Data Indication to software). Three new Frame Type control bits, and a modification of the Frame Version Filtering bits, are included in the Packet Processor, and affect hardware packet parsing as indicated in the following table:

REGISTER	DEFINITION
FRM_VER_FILTER[3:0]	Frame Version selector. The incoming packet's Frame Control Field is parsed to obtain the FrameVersion subfield, and that value is compared against this register, in accordance with the following: xxx1: Accept received packets with FrameVersion=00 xx1x: Accept received packets with FrameVersion=01 x1xx: Accept received packets with FrameVersion=10 1xxx: Accept received packets with FrameVersion=11 These filtering rules apply to Beacon, Acknowledge, Data, and MAC Command Frame Types, since these frame types require a 2-octet Frame Control Field which embeds a 2-bit FrameVersion subfield. Later frame types introduced in the 802.15.4e addendum (LLDN, Multipurpose) don't guarantee a FrameVersion subfield with the original meaning, so these filtering rules do not apply to those frame types. See registers LLDN_FT and MULTIPURPOSE_FT.
LLDN_FT	1: LLDN frame type enabled (Frame Type 4) 0: reject all LLDN frames
MULTIPURPOSE_FT	1: Multipurpose frame type enabled (Frame Type 5) 0: reject all Multipurpose frames
EXTENDED_FT	1: Extended frame type enabled (Frame Type 7) 0: reject all Extended frames

The Extended Frame Type does not comply with the General MAC Frame Format, which is shown in the following diagram:

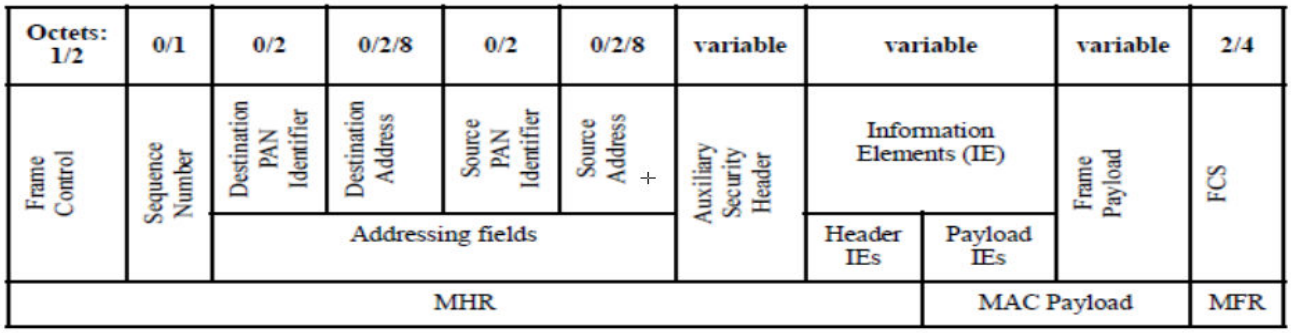


Figure 380. General MAC Frame Format

Instead, the Extended Frame has its own unique format, which is shown in the following diagram:

Bits: 0-2	3-5	variable
Frame Type	Extended Frame Type	Extended Frame Payload

Figure 381. Extended Frame Format

Because the Extended Frame Payload, which is not defined in the 802.15.4-2015 revision to the standard, would be intended to contain and fully specify the FCS, the Packet Processor will not be able to verify FCS on received packets with Frame Type Extended. FCS checking will need to be implemented in software.

Note: If the EXTENDED_FT bit is set to enable Data Indication on packets of Frame Type Extended, software should expect a substantial increase in the frequency of notifications, as a consequence of the lack of hardware verification of FCS, which will have implications for software workload as well as power consumption in the SoC.

When the packet processor receives any of the 4e-compliant frames, if the corresponding control bit is set, the hardware will skip Stage 2 filtering, and merely verify FCS. If FCS passes, the packet processor will notify software of the packet, without inspection of the addressing fields or payload. For such packets, software will be responsible for parsing the MHR and payload sections of the frame. If the corresponding control bit is not set, the packet processor will reject the received packet and there will be no Data Indication.

When the packet processor receives a packet of Frame Type Extended, if the EXTENDED_FT bit is set, the hardware will skip Stage 2 filtering, and after the entire packet is received, the packet processor will merely notify software of the packet, without inspection of the Extended Frame Type bits or the Extended Frame Payload. For such packets, software will be responsible for parsing the Extended Frame Type and the Extended Frame Payload sections of the frame. If the corresponding control bit is not set, the packet processor will reject the received packet and there will be no Data Indication.

Like all existing frames prior to 802.15.4e and 802.15.4-2015, a received frame may require an acknowledgement. In the case of the new 802.15.4e and 802.15.4-2015 frames, the required timing for the acknowledge frame transmission, has been modified. The acknowledgement no longer immediately follows the packet reception. For this reason, upon reception of any of these 4e-compliant 2015-compliant packets, the packet processor will cancel any auto-ACK transmission, regardless of the state of the PHY_CTRL[AUTOACK] bit and AR field of the received packet's Frame Control Field. Software will be responsible for constructing the acknowledgment packet and transmitting it at the appropriate time using a Sequence T.

To help expedite software processing of the 4e- and 2015-compliant Frame Types and Versions, additional status bits have been added to the 802.15.4 register set, which are described in the following table:

STATUS BIT	RESIDES IN REGISTER	DEFINITION
ENH_PKT_STATUS	IRQSTS	1: The last packet received was 4e- or 2015-compliant (SW should query the RX_FRAME_FILTER register for additional status bits) 0: The last packet received was neither 4e- nor 2015-compliant
BEACON_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Beacon with FrameVersion 2 0: The last packet received was not a FrameType Beacon with FrameVersion 2
DATA_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Data with FrameVersion 2

Table continues on the next page...

Table continued from the previous page...

		0: The last packet received was not a FrameType Data with FrameVersion 2
ACK_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Acknowledge with FrameVersion 2 0: The last packet received was not a FrameType Acknowledge with FrameVersion 2
CMD_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType MAC Command with FrameVersion 2 0: The last packet received was not a FrameType MAC Command with FrameVersion 2
LLDN_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType LLDN 0: The last packet received was not a FrameType LLDN
MULTIPURPOSE_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Multipurpose with FrameVersion 2 0: The last packet received was not a FrameType Multipurpose
EXTENDED_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Extended 0: The last packet received was not a FrameType Extended

The new status bits are mapped into the 802.15.4 Address space as shown in the diagram below:

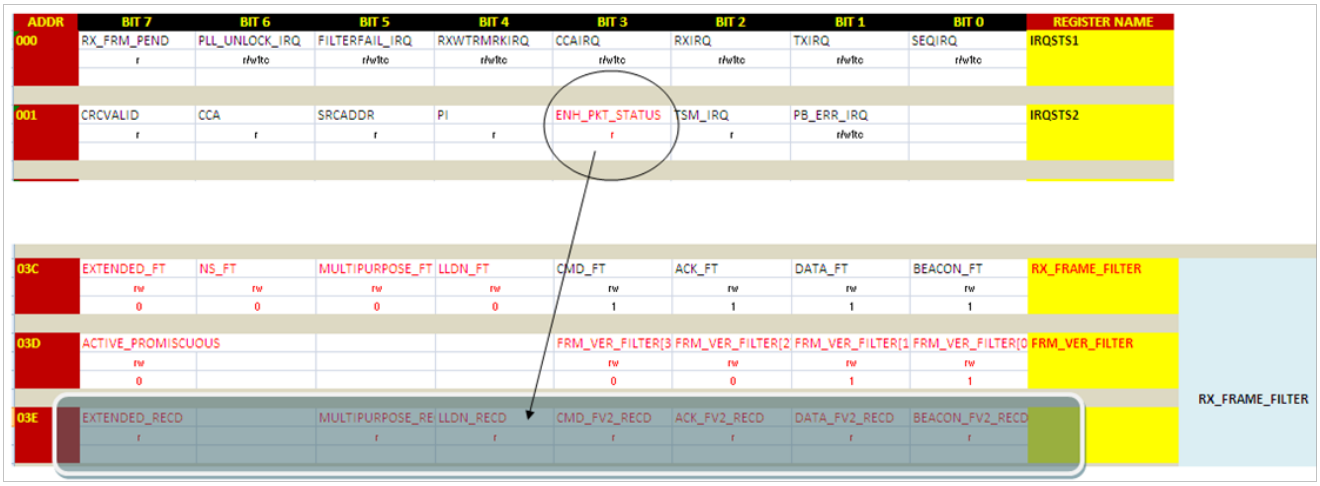


Figure 382. Enhanced Packet Status Bits

55.4.9.1.2.1.3.7 Clocks

Packet Processor is a fully synchronous module. It has a single clock input **clk_250khz**. The clock frequency, 250KHz, is the 802.15.4 bit rate. This clock input is driven by a divide-down of the RF Oscillator, generated in the CRM (Clocks and Resets module). The Packet Processor has no asynchronous interfaces.

55.4.9.1.2.1.3.8 Reset

The Packet Processor has a single, active-low, asynchronous reset input: **rst_b** . At integration, this reset should be tied to the master reset for the entire transceiver. There are no special reset requirements.

55.4.9.1.2.1.3.9 Interrupts

The Packet Processor does not generate any interrupts.

55.4.9.1.2.1.3.10 Appendix A

FILTERFAIL CODES

A description of the FILTERFAIL codes can be found in the following table.

FILTERFAIL CODE BIT	REASON FOR FILTERFAIL
[0]	Fails Stage 1 Frame Length Checking (FL < 5 or FL > MAXFRAMELENGTH) **Note:** FL < 3 will not generate an SFD, so this bit will not be set
[1]	Fails Stage 1 Section 7.2.1.1.6 or Section 7.2.1.1.8 Checking (DST_ADDR_MODE or SRC_ADDR_MODE = 1)
[2]	For frame version 0 or 1, fails Stage 1 Section 7.2.1.1.5 Checking (Illegal PAN_ID_COMPRESSION Usage); For frame version 2, fails Stage 1 checking not supported cases in 802.15.4-2015 Table 7-2, see leniency[40] for the not supported cases.
[3]	Fails Stage 1 Frame Version Checking
[4]	Fails Stage 2 Auto-RxAck Checking (Illegal Ack Frame Format in Sequence TR)
[5]	Fails Stage 2 Frame Type Checking (Incorrect Frame Filter Bit setting)
[6]	Fails Stage 2 Frame Length Checking (Illegal Beacon, Data, or Cmd FL)
[7]	Fails Stage 2 Addressing Mode Checking (Illegal Addressing Mode for Beacon, Data, OR Cmd)
[8]	Fails Stage 2 Sequence Number Matching (Sequence TR Only)
[9]	Fails Stage 2 PAN ID or Address Checking (Beacon, Data, or Cmd)

55.4.9.1.2.1.3.11 Appendix B

LENIENCY BITS

The Packet Processor performs filtering on all received packets, in order to determine whether the packet is intended for the device. The packet filtering is based on rules. In case any of the packet filtering rules need to be overridden, a 43-bit "leniency register" has been provided. When the leniency register is programmed to its default value (0), all hardware packet filtering rules are in effect, and if an incoming packet violates any rule, a "Filter Fail" will occur (packet will be rejected). When a given leniency register bit is asserted, the packet filtering rule assigned to that bit will not be in effect, and if any incoming packet violates that rule (but no other rules), then a "Filter Fail" will not occur, the packet will not be rejected, the packet will be treated as "intended for the device", and software will be notified of the incoming packet. The table below shows the assignment of leniency bits to packet filtering rules.

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[0]	Override Stage 1 Frame Length Checking (FL < 5 or FL > MAXFRAMELENGTH)

Table continues on the next page...

Table continued from the previous page...

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[1]	Override Stage 1 Section 7.2.1.1.6 Checking (DST_ADDR_MODE = 1)
leniency[2]	Override Stage 1 Section 7.2.1.1.8 Checking (SRC_ADDR_MODE = 1)
leniency[3]	Override Stage 1 Section 7.2.1.1.5 Checking (Illegal PAN_ID_COMPRESSION Usage)
leniency[4]	Override Stage 2 Auto-RxAck Checking (Illegal Ack Frame Format in Sequence TR)
leniency[5]	Override Stage 2 Frame Length Checking (Illegal Beacon, Data, or Cmd FL)
leniency[6]	Override Stage 2 Beacon Frame Address Mode Violations
leniency[7]	Override Stage 2 Data Frame Address Mode Violations
leniency[8]	Override Stage 2 MAC Command Frame Address Mode Violations
leniency[9]	Override Stage 2 Sequence Number Matching
leniency[10]	Override Stage 2 Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_SHORT (Beacon Only)
leniency[11]	Override Stage 2 Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_SHORT (Data and MAC Command Only)
leniency[12]	Override Stage 2 Dst PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_NONE (Data and MAC Command Only)
leniency[13]	Override Stage 2 Dst Short Addr Filter for DST_ADDR_MODE_SHORT/ SRC_ADDR_MODE_NONE (Data and MAC Command Only)
leniency[14]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Beacon Only)
leniency[15]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[16]	Override Stage 2 Dst Short Addr Filter for DST_ADDR_MODE_SHORT/ SRC_ADDR_MODE_SHORT/PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[17]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[18]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESS ION (Data and MAC Command Only)
leniency[19]	Override Stage 2 Dst Addr Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[20]	Override Stage 2 Src PAN ID Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[21]	Override Stage 2 Src PAN ID Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[22]	Override Stage 2 Dst PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_NONE (Data and MAC Command Only)
leniency[23]	Override Stage 2 Dst Long Addr Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_NONE (Data and MAC Command Only)

Table continues on the next page...

Table continued from the previous page...

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[24]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ PAN_ID_COMPRESSION (Beacon Only)
leniency[25]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[26]	Override Stage 2 Dst Long Addr Filter for DST_ADDR_MODE_SHORT/ SRC_ADDR_MODE_LONG/PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[27]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Beacon Only)
leniency[28]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Data and Mac Command Only)
leniency[29]	Override Stage 2 Dst Long Addr Filter for DST_ADDR_MODE_LONG/ SRC_ADDR_MODE_SHORT/PAN_ID_COMPRESSION (Data and Mac Command Only)
leniency[30]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[31]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[32]	Override Stage 2 Short Addr Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[33]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[34]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSI ON (Data and MAC Command Only)
leniency[35]	Override Stage 2 Long Addr Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[36]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_LONG (Beacon Only)
leniency[37]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_LONG (Data and MAC Command Only)
leniency[38]	Override Stage 2 Long Addr Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_LONG (Data and MAC Command Only)
leniency[39]	Allow an auto-TxAck frame to be sent, after a receive frame which has all of the following parameters: 1. Destination PAN ID = Broadcast (0xFFFF) 2. Destination Address = !Broadcast (not 0xFFFF) 3. Destination Address Mode = Short Nominally, the SEQ_MGR inhibits auto-TxAck on such frames.
leniency[40]	Override Stage 1 for frame version 2 not supported cases in 802.15.4-2015 Table 7-2. See detail in Chapter "Enhanced Acknowledgment (Enh-Ack) Support".

Table continues on the next page...

Table continued from the previous page...

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[41]	Override Stage 2 Auto-RxAck Checking (Illegal EnhAck Frame Format in Sequence TR). 1. The received PAN ID Compression matches the transmitted PAN ID Compression 2. The received Sequence Number Suppression matches the transmitted Sequence Number Suppression 3. The received Destination Address Mode matches the transmitted Source Address Mode
leniency[42]	Override Stage 2 Auto-RxAck Checking (Illegal EnhAck Frame Format in Sequence TR). 1. The Destination Address field, when present, shall contain the value of the Source Address field from the frame that is being acknowledged. 2. The Destination PAN ID, when present, contains the value from the Source PAN ID field from the frame that is being acknowledged.

55.4.9.1.2.2 Sequence Manager

55.4.9.1.2.2.1 Introduction

This Section describes the 802.15.4 Sequence Manager.

55.4.9.1.2.2.1.1 Overview

The 802.15.4 Sequence Manager (ZSM) is a hardware state machine which controls the timing of all transmit, receive, and CCA operations for the 802.15.4 Link Layer. The ZSM manages high-level 802.15.4 required timing (autosequences), and interfaces directly with a lower-level, protocol-neutral sequence manager called the Transceiver Sequence Manager (TSM), which manages the direct control of the digital, analog, and RF components of the 2.4GHz transceiver. See the TSM Chapter for more details.

For each digital and analog element in the 2.4GHz transceiver, the ZSM and TSM operate seamlessly and collaboratively, to perform some or all of the following functions:

- activation
- deactivation
- initialization
- mode transitions
- initiating calibration routines
- clock enabling and disabling
- interrupt triggering and status generation

Designed into the ZSM are a variety of complex sequences, called autosequences, which are sequential concatenations of simpler, building-block sequences. The hardware autosequences provide the following advantages:

- smaller SW memory footprint: HW automates operations previously done by SW
- increases MCU sleep times, reducing current consumption
- handles timing-critical sequences (rx-to-tx), e.g., data-polling, not possible in SW

Sequences can be initiated by software directly, or alternatively, can be initiated automatically at the expiration of a hardware timer. The general parameters of the sequences can be programmed by software prior to initiating the sequences. Such parameters allow software to select, for example:

- whether CCA is required ahead of a transmit sequence
- whether an automatic Ack frame should be transmitted after a receive sequence

- slotted vs. unslotted mode
- type of CCA (e.g., Energy Detect, CCA Mode 2)
- whether the device is a PAN Coordinator or not
- whether the device resides on 2 networks simultaneously (Dual PAN mode)

The ZSM also includes support for Dual PAN Mode, and Active Promiscuous Mode.

55.4.9.1.2.2.1.2 Features

- Supports complex sequences
- Unburdens software by automating many transceiver operations
- Data-polling accelerator hardware enables critical-timing sequences
- Includes Continuous CCA for low-latency-transmit applications
- Includes capability to conditionally perform CCA prior to every transmit operation
- Includes capability to conditionally transmit an Ack frame after every receive operation
- Supports software- or timer-initiated sequences
- Fully supports 2003 and 2006 versions of the 802.15.4 standard
- Supports slotted and unslotted modes
- Supports beacon-enabled and non-beacon-enabled networks
- Supports orderly transceiver shutdown on PLL unlock event
- Increases software visibility by making SEQ_STATE state readable
- Supports Dual PAN Mode, including on-the-fly channel changing
- Supports Active Promiscuous Mode

55.4.9.1.2.2.1.3 Modes and operations

The ZSM controls, coordinates, and automates all 802.15.4 transceiver operations within the 2.4GHz radio. Software selects the desired sequence, sets the general parameters, and (optionally) configures a hardware timer to trigger the sequence at a precise time in the future. Subsequently, the MCU can enter a low-power state, or tend to other activities, while the sequence manager state machine conducts the programmed transceiver operations. When the sequence completes, an interrupt alerts the MCU. At this point the MCU can check the completion status of the sequence, download received data from the packet buffer, and then prepare the next sequence. Sequences can be simple transceiver operations. For example, transmit one frame, or, perform a CCA on a channel. Alternatively, the sequence manager supports complex sequences, which use simple operations as building blocks, strung together in a back-to-back fashion, with precise timing intervals between operations. For example, a “TR” sequence calls for a transmit frame followed by a receive frame. For complex sequences, interrupts can optionally be generated at the completion of each stage of the sequence, providing greater visibility for the MCU into the timeline of the sequence. Software can determine the precise state of the sequence manager state machine at any time. Software can opt to abort a sequence at any time; when this happens, the sequence manager will return to its IDLE state in an orderly fashion.

55.4.9.1.2.2.1.4 Block diagram

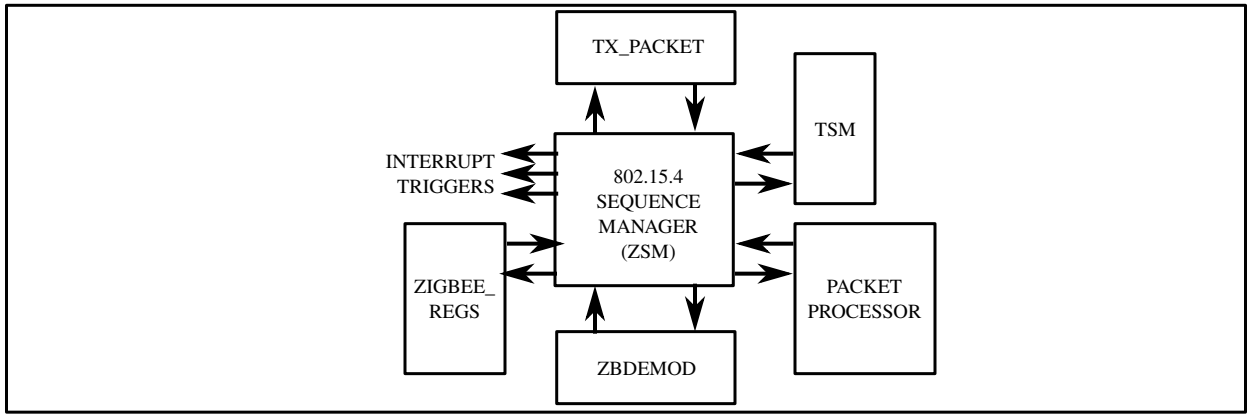


Figure 383. Block diagram

55.4.9.1.2.2.2 Memory Map and register definition

Numerous registers exist to provide configuration and status for the 802.15.4 Sequence Manager. See table below.

<table>
<tr><th>Field</th><th>R/W</th><th>Description</th></tr>
<tr><td>XCVSEQ</td><td>rw</td><td>The Transceiver Sequence Selector register selects a sequence for the sequence manager to execute. Sequence initiation can be immediate, or scheduled (see TMRTRIGEN). A write of XCVSEQ=IDLE will abort any ongoing sequence. A write of XCVSEQ=IDLE must always be performed after a sequence is complete, and before a new sequence is programmed. Any write to XCVSEQ other than XCVSEQ=IDLE during an ongoing sequence, shall be ignored. The mapping of XCVSEQ to sequence types is as follows:
<table>
<tr><th>XCVSEQ</th><th>SEQUENCE</th><th>DESCRIPTION</th></tr>
<tr><td>0</td><td>I</td><td>IDLE</td></tr>
<tr><td>1</td><td>R</td><td>RECEIVE</td></tr>
<tr><td>2</td><td>T</td><td>TRANSMIT</td></tr>
<tr><td>3</td><td>C</td><td>CCA</td></tr>
<tr><td>4</td><td>TR</td><td>TRANSMIT/RECEIVE</td></tr>
<tr><td>5</td><td>CCCA</td><td>CONTINUOUS CCA</td></tr>
</table></td></tr>
<tr><td>TMRTRIGEN</td><td>rw</td><td>Timer2 Trigger Enable.
1: allow timer TC2 (or TC2') to initiate a preprogrammed sequence (see XCVSEQ register).
0: programmed sequence initiates immediately upon write to XCVSEQ.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
CCABFRTX	rw	CCA Before TX. Applies only to Sequences T and TR, ignored during all other sequences. 1: at least one CCA measurement is required prior to the transmit operation (see also SLOTTED). 0: no CCA required, transmit operation begins immediately.
SLOTTED	rw	Slotted Mode, for beacon-enabled networks. Applies only to Sequences T, TR, and R, ignored during all other sequences. Used, in concert with CCABFRTX, to determine how many CCA measurements are required prior to a transmit operation. Also used during R sequence to determine whether the ensuing transmit acknowledge frame (if any) needs to be synchronized to a backoff slot boundary.
RXACKRQD	rw	Receive Acknowledge Frame required. Applies only to Sequence TR, ignored during all other sequences. 1: A receive Ack frame follows the transmit frame. 0: An ordinary receive frame (not an Ack frame) follows the transmit frame.
AUTOACK	rw	Auto Acknowledge Enable. Applies only to Sequence R and Sequence TR, ignored during other sequences. 1: sequence manager will follow a receive frame with an automatic hardware-generated Tx Ack frame, assuming other necessary conditions are met. 0: sequence manager will not follow a receive frame with a Tx Ack frame, under any conditions; the sequence will terminate after the receive frame.
CRC_MSK	rw	CRC Mask. 1: sequence manager requires CRCVALID=1 at the end of the received frame in order for the receive operation to complete successfully; if CRCVALID=0, sequence manager will return to preamble-detect mode after the last octet of the frame has been received. 0: sequence manager ignores CRCVALID and considers the receive operation complete after the last octet of the frame has been received.
PROMISCUOUS	rw	Prevents auto-TxAck on received packets 1: promiscuous mode, no auto-TxAck 0: auto-TxAck allowed, depending on other conditions.
CCATYPE[1:0]	rw	Clear Channel Assessment Type. Selects one of four possible functions for CCA or ED, per below. (See Chapter CCA/ED/LQI for more details). 00: ENERGY DETECT 01: CCA MODE 1

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		10: CCA MODE 2 11: CCA MODE 3
TC3TMOUT	rw	1: Allow a TMR3 timeout to abort any RX operation 0: Don't allow a TMR3 timeout to abort any RX operation See Appendix A for more details on how autosequences are affected by a TMR3 timeout
SEQ_STATE[4:0]	r	This read-only register reflects realtime 802.15.4 Sequence Manager (ZSM) state
ACTIVE_PROMISCUOUS	rw	1: Provide Data Indication on all received packets under the same rules which apply in PROMISCUOUS mode, however acknowledge those packets under rules which apply in non-PROMISCUOUS mode 0: normal operation (Default)
ACTIVE_NETWORK	rw	Selects the PAN on which to transceive, by activating a PAN parameter set (PAN0 or PAN1). In Manual Dual PAN mode (or Single PAN mode), this bit selects the active PAN parameter set (channel and addressing parameters) which governs all autosequences. In Auto Dual PAN mode, this bit selects the PAN on which to begin transceiving, latched at the point at which DUAL_PAN_DWELL register is written. 1: Select PAN1 0: Select PAN0 (Default)
DUAL_PAN_AUTO	rw	Activates automatic Dual PAN operating mode. In this mode, PAN-switching is controlled by hardware at a pre-programmed rate, determined by DUAL_PAN_DWELL. 1: Auto Dual PAN Mode 0: Manual Dual PAN mode (or Single PAN mode). Default. Whenever DUAL_PAN_AUTO=0, CURRENT_NETWORK=ACTIVE_NETWORK at all times. In other words, software directly controls which PAN is selected. Whenever DUAL_PAN_AUTO=1, CURRENT_NETWORK is controlled by hardware
CURRENT_NETWORK	r	This read-only bit indicates which PAN is currently selected by hardware in automatic Dual PAN mode 1: PAN1 is selected 0: PAN0 is selected
DUAL_PAN_DWELL[7:0]	rw	Channel Frequency Dwell Time. In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>R/W</th><th>Description</th></tr>
<tr><td></td><td></td><td>DUAL_PAN_DWELL, always re-initializes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the sequence warmup.

<table>
<tr><th>PRESCALER (bits [1:0])</th><th>TIMEBASE (bits [7:2])</th><th>RANGE (min) - (max)</th></tr>
<tr><td>00</td><td>0.5ms</td><td>0.5 - 32ms</td></tr>
<tr><td>01</td><td>2.5ms</td><td>2.5 - 160ms</td></tr>
<tr><td>10</td><td>10ms</td><td>10 - 640ms</td></tr>
<tr><td>11</td><td>50ms</td><td>50ms - 3.2seconds</td></tr>
</table>

A write to DUAL_PAN_DWELL also causes the value of ACTIVE_NETWORK to get latched into the hardware. This latched value will be the starting point for the automatic dual-pan mode (i.e., start on PAN0 or on PAN1). The starting value takes effect immediately (if sequence is underway and DUAL_PAN_AUTO=1), or is otherwise delayed until sequence starts and DUAL_PAN_AUTO=1.</td></tr>
<tr><td>DUAL_PAN_REMAIN[5:0]</td><td></td><td>This read-only register indicates time remaining before next PAN switch in auto Dual PAN mode. The units for this register, depend on the PRESCALER setting (bits [1:0]) in the DUAL_PAN_DWELL register, according to the following table:

<table>
<tr><th>DUAL_PAN_DWELL PRESCALER</th><th>DUAL_PAN_REMAIN UNITS</th></tr>
<tr><td>00</td><td>0.5ms</td></tr>
<tr><td>01</td><td>2.5ms</td></tr>
<tr><td>10</td><td>10ms</td></tr>
<tr><td>11</td><td>50ms</td></tr>
</table>

The readback value indicates that between N-1 and N timebase units remain until the next PAN switch. For example, a DUAL_PAN_REMAIN readback value of 3, with a DUAL_PAN_DWELL PRESCALER setting of 2 (10ms), indicates that between 20ms (2*10ms) and 30ms (3*10ms), remain until the next automatic PAN switch.</td></tr>
<tr><td>RECD_ON_PAN0</td><td>r</td><td>Indicates the packet which was just received, was received on PAN0.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		In Dual PAN mode operating on 2 different channels, RECD_ON_PAN0 will be set if CURRENT_NETWORK=0 when the packet was received, regardless of FILTERFAIL status. In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN0 will be set only if a valid packet was received on PAN0 (PAN0's FILTERFAIL_FLAG is deasserted).
RECD_ON_PAN1	r	Indicates the packet which was just received, was received on PAN1. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN1 will be set if CURRENT_NETWORK=1 when the packet was received, regardless of FILTERFAIL status. In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN1 will be set only if a valid packet was received on PAN1 (PAN1's FILTERFAIL_FLAG is deasserted).
ACKDELAY[5:0]	rw	Provides a fine-tune adjustment of the time delay between Rx warmdown and the beginning of Tx warmup for an Tx Acknowledge packet. ACKDELAY register will apply to both SLOTTED and UNSLOTTED TxAck, but only to TxAck (not T sequences). This is a two's complement value. The minimum permissible value is -19 (0x2D). Values less than -19 will lead to unexpected results. Resolution = 2us. Range = +/- 62us. Default = 0x0. Max ACKDELAY = 0x1F. Min ACKDELAY = 0x2d.
TXDELAY[5:0]	rw	Provides a fine-tune adjustment of the time delay between post-CCA Rx warm-down and the beginning of Tx warm-up for an Tx (non-Ack) packet. TXDELAY register will apply in both SLOTTED and UNSLOTTED modes, but only to T sequences (e.g., T, TR, and T(R)), not TxAck operations. This is a two's complement value. The minimum permissible value is -19 (0x2D). Values less than -19 will lead to unexpected results. Resolution = 2us. Range = +/- 62us. Default = 0x0. Max TXDELAY = 0x1F. Min TXDELAY = 0x2d

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
CLR_NEW_SEQ_INHIBIT	rw	when asserted, overrides the automatic hardware locking of the programmed XCVSEQ while an autosequence is underway. Asserting this feature will allow software to change the programmed autosequence “on-the-fly”, without aborting and returning to idle between sequences. Overriding the hardware lockout of XCVSEQ should be used with caution, since the Sequence Manager is not designed (or verified) for manual state transitions between one type of autosequence and other (i.e., Sequence T -> Sequence R).
NO_RX_RECYCLE	rw	when asserted, prevents the Sequence Manager from automatically re-starting the receiver when a packet is received which results in a FilterFail or CRC failure. Normally, the Sequence Manager jumps to the RX_CYC state, and then resumes from there with a new Rx warmup, in search of a new packet. When this bit is set, the Sequence Manager will instead return to idle state, and issue a SEQIRQ, after a FilterFail or CRC failure.
LATCH_PREAMBLE	rw	1: Make PREAMBLE_DET and SFD_DET bits of PHY_STS Register “sticky”, i.e., occurrences of preamble and SFD detection are latched and held until the start of the next autosequence 0: Don’t make PREAMBLE_DET and SFD_DET bits of PHY_STS Register “sticky”, i.e, these status bits reflect the realtime, dynamic state of preamble_detect and sfd_detect. (default)
XCVSEQ_ACTUAL[2:0]	r	indicates the programmed sequence that has been recognized by the Sequence Manager. Takes into account the fact that sequence-change commands from software are ignored while a sequence is underway (see new_seq_inhibit bit). Read-only bits.
SEQ_IDLE	r	when asserted, indicates that the Sequence Manager is in its idle state. Read-only bit.
NEW_SEQ_INHIBIT	r	when asserted, indicates that a new programmed autosequence has commenced (TMR2 match has occurred if TMRTRIGEN=1). Once this bit is asserted, software is blocked from commanding any new autosequences (other than Sequence I to abort the current sequence), until the current sequence completes. Hardware will ignore a sequence-change command from software while this bit is asserted. Hardware will automatically deassert this bit once the sequence completes. Read-only bit.
RX_TIMEOUT_PENDING	r	when asserted, indicates that a TMR3 timeout (RX timeout) flag has been set by Hardware, but the Sequence Manager has not yet aborted because an RX operation is not currently underway. This would be the case, for example, during a Sequence TR, if a TMR3 timeout were to occur during the transmit operation of this sequence; the sequence would not be aborted by Hardware until the receive operation begins. This bit will always be 0 if TC3TMOUT=0. Read-only bit.

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
RX_MODE	r	indicates that an RX operation is in progress. CCA and ED operations are considered RX operations. Read-only bit.
TMR2_SEQ_TRIG_ARMED	r	when asserted, indicates that TMR2 has been programmed and is armed to trigger a new autosequence, when Sequence Manager timer-triggering mode is selected (i.e., TMRTRIGEN=1). When timer-triggering mode is selected, TMR2 must be re-programmed (using either T2CMP or T2), in advance of each new sequence. Once TMR2 is programmed, this bit will be asserted, and will remain asserted until the new sequence commences (at TMR2 match). Hardware will deassert this bit when the new sequence starts. When TMRTRIGEN=0, this bit should be ignored. Read-only bit.
SW_ABORTED	r	when asserted, indicates that the autosequence has terminated due to an Software abort. Software can abort any programmed autosequence by writing Sequence I to XCVSEQ. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
TC3_ABORTED	r	when asserted, indicates that the autosequence has terminated due to an TC3 (TMR3) timeout during a receive operation. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
PLL_ABORTED	r	when asserted, indicates that the autosequence has terminated due to an PLL unlock event. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
CCCA_BUSY_CNT[5:0]	r	For Sequence CCCA mode only, this register indicates the number of “busy” CCA attempts which occurred during the autosequence, before the channel was detected to be idle. This register can also be read in real-time (during the autosequence) to determine how many busy CCA attempts have occurred to that point. The register saturates at 63 (i.e, if there are more than 63 busy attempts, the register will continue to read 63). This register is automatically cleared to zero by hardware when the next autosequence commences. Read-only register.
SEQ_T_STATUS[5:0]	r	Status of the just-completed (or ongoing) Sequence T or Sequence TR autosequence. This register is valid at all times during, and after, the Sequence T or Sequence TR. Not valid for other types of autosequences. This is a read-only register. The bits of this register map to status, according to the following table: [0] 1st CCA complete (CCABFRTX=1) [1] 2nd CCA complete (SLOTTED=1) [2] Tx operation complete [3] Rx Recycle occurred (Sequence TR only)

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		[4] Rx operation complete (Sequence TR only) [5] TxAck operation complete(Sequence TR only)
CONTINUOUS_EN	rw	Enable Continuous TX or RX mode (testmode). See Appendix B.
SLOT_PRELOAD[7:0]	rw	This register represents the number that gets loaded into the slot_timer at SFD detect, which ultimately determines when the next slot boundary will occur. Due to processing delays within the analog front-end and digital modem, the point at which SFD is detected by the modem, is delayed relative to over-the-air timing. Since this timing may not be known prior to until actual silicon, and since this is such a critical timing parameter for slotted operations, it has been made programmable. The default value is 0x74. This timing parameter is critical for the Sequence R autosequence in slotted mode, when an automatic TxAck is required.

55.4.9.1.2.2.3 Functional description

This section describes the supported sequences in more detail. All sequences can be initiated by software directly (instantaneously) by writing the desired sequence to the XCVSEQ register, with the TMRTRIGEN bit deasserted. Alternatively, software can schedule the initiation of a sequence at a precise time in the future, by programming the desired start time into the T2CMP (or T2PRIMECMP) timer compare register, and setting the TMRTRIGEN bit. The XCVSEQ register field is writable and readable, so software can read this register to determine which sequence it had programmed earlier. After the sequence completes (SEQIRQ interrupt), software should set the XCVSEQ to IDLE (0x0), before initiating a new sequence. If the sequence-complete interrupt (SEQIRQ) is masked, software can determine when the sequence is complete by polling the SEQ_STATE register, until it returns to SEQ_IDLE (0x0). While a sequence is ongoing, software can abort the sequence at any time by writing IDLE (0x0) to XCVSEQ; the sequence manager state machine will respond with an orderly return to the SEQ_IDLE state, and a SEQIRQ will be generated. While a sequence is underway, software should not attempt to change XCVSEQ (other than aborting the sequence as previously described); the sequence manager will ignore all mid-sequence attempts to change XCVSEQ. As mentioned previously, software should write the IDLE value to XCVSEQ before changing sequences. If TMR2 triggering is enabled (TMRTRIGEN=1), after the scheduled sequence is automatically launched at TMR2 match, and then completes, software must re-arm TMR2 triggering by writing to either T2CMP, or T2PRIMECMP. At least the LS byte of one of these registers must be written to re-arm. This hardware interlock is designed to prevent the inadvertent re-triggering of sequences, which would otherwise occur after the 24-bit Event Timer rolls over.

The supported sequences are shown in the following table.

Table 473. Supported Sequences

XCVSEQ	Sequence	Description
0	I	Idle
1	R	Receive Sequence - conditionally followed by a TxAck
2	T	Transmit Sequence
3	C	Standalone CCA
4	TR	Transmit /Receive Sequence - transmit unconditionally followed by either an R or RxAck

Table continues on the next page...

Table 473. Supported Sequences (continued)

XCVSEQ	Sequence	Description
5	CCCA	Continuous CCA - sequence completes when channel is idle
6	Reserved	
7	Reserved	

All sequences can be aborted by a PLL unlock detection. The unlock detection is qualified by the sequence manager; in other words, an unlock detection during the early stages of the PLL warmup is expected, and is not allowed to abort the sequence at that point. The primary rationale for the PLL unlock abort, is to prevent transmission on an incorrect, or unstable, frequency during a transmit operation, or to increase battery life by not prolonging a receive or CCA operation that is bound to fail. Software can disable the PLL unlock auto-abort. See Sequence Aborting.

55.4.9.1.2.2.3.1 Supported Sequences

55.4.9.1.2.2.3.1.1 Sequence I (Idle)

When the IDLE value is written to XCVSEQ, the sequence manager goes to its SEQ_IDLE state. If not already in SEQ_IDLE, the sequence manager executes an orderly warmdown to return to SEQ_IDLE. A SEQIRQ interrupt is then issued. Writing IDLE value to XCVSEQ is the proper way to abort a sequence.

When a sequence is aborted in this way (called a software abort), the SW_ABORTED bit of the ABORT_STS register in 802.15.4 space, will become set. This bit will be self-cleared at the start of the next autosequence.

55.4.9.1.2.2.3.1.2 Sequence R (Receive)

Sequence R is the basic receive sequence. Sequence R can be used to receive all types of 802.15.4 PHY- and MAC-compliant frames, including reserved frame types. However, reception of Acknowledge frames is not recommended using Sequence R. This is because reception of an Acknowledge frame, usually follows a transmitted frame, with a designated Sequence Number, so that the received Acknowledge frame can be verified for the matching Sequence Number. In a standalone Sequence R, there is no Sequence Number to verify against, so any Acknowledge frame is merely transferred to the Packet Buffer. (Note: the appropriate way to receive Acknowledge frames is with Sequence TR, with the RXACKRQD bit asserted). Using Sequence R, *all* frames which pass frame-filtering rules and CRC check, are transferred to the Packet Buffer.

The basic execution of a successful Sequence R is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU writes XCVSEQ=R
- Wait for TC2 match (if TMRTRIGEN=1)
- SEQ_MGR executes RxWarmup
- Preamble Detected
- SFD Detected
- FrameLength octet received, transferred to Packet Buffer
- *N* additional octets received (*N* = FrameLength), transferred to Packet Buffer. *Note if (FrameLength > 127), then N = 127.*
- If frame-filtering rules and CRC check pass:
- ---> RXIRQ interrupt issued
- ---> SEQ_MGR executes RxWarmdown
- ---> If received Frame Control Field indicates AckRequest=1
- ---- ---> SEQ_MGR executes TxWarmup

- ---- ---> an Ack frame is transmitted using the received Sequence Number
- ---- ---> SEQ_MGR executes TxWarmdown
- SEQIRQ interrupt issued

When Sequence R is initiated, the ZSM sequence manager warms up the receiver (analog and digital elements), via the TSM (Transceiver Sequence Manager). After warmup is complete, timer TMR3 can be enabled as a “bracketing” timer (software option). If a TMR3 timer match occurs, and software has asserted the TC3TMOUT bit, the sequence manager will warm down the receiver and return to SEQ_IDLE state. Assuming no TMR3 timeout, reception proceeds in preamble-search mode. Once a preamble is detected, reception continues in SFD-search mode. At this point, the sequence manager’s slot timer is activated. This is because a slotted “transmit acknowledge” frame (TxAck) may be required later in this sequence. The slot timer is loaded with the following value (in us):

Slot Timer Preload = 160 + TXWARMUPTIME + (2*ACKDELAY).

The 160 term arises because, at the point of SFD detection, we are exactly 10 symbols into the backoff slot, and the symbol period is 16us. The slot timer will count up and eventually rollover at 319, to 0. (There are 320us in a backoff slot). The receiver receives the next octet (FrameLength). Starting with this octet, and continuing through the remainder of the frame, each octet that is received is transferred to Packet Buffer. After each octet is transferred to Packet Buffer, the Packet Buffer address is incremented by 1. The next 2 octets (Frame Control Field) are then received, which are parsed to obtain the AckRequest field, and the FrameVersion field. The next octet is then received (SequenceNumber), and stored. The remainder of the frame, determined by FrameLength, is then received. As the octets are received, the packet data is parsed and subjected to packet-filtering rules (see Chapter Packet Processor). If any of the packet-filtering rules fail, or if the CRC check on the FCS field fails, the sequence manager will return the receiver to preamble-search mode, after the last octet has been received and transferred to Packet Buffer. Assuming the packet-filtering rules and CRC pass, an interrupt (RXIRQ) is issued to the MCU. At this point, the sequence manager must determine if a TxAck is required. A TxAck is required if the following 3 conditions are all met:

- AUTOACK=1 (register bit)
- PROMISCUOUS=0 (register bit)
- The received FrameControlField has AckRequest=1

If these conditions are met, and SLOTTED mode is not in effect, the sequence manager loads its TxAck timer with the following value (in us):

TxAck Timer Preload = 192 - TXWARMUPTIME - 2*ACKDELAY

The 192us is the non-slotted RX-to-TX turnaround requirement in the 802.15.4 standard. The TXWARMUPTIME takes into account the fact that the TxAck timer must expire early for the Tx warmup to be complete at exactly the 192us point. The ACKDELAY is a signed, fine-tune adjustment to the TxAck transmission time (+/- 62us).

If SLOTTED mode is in effect, the slot timer was previously loaded (at SFD detect), but the sequence manager needs to determine how much time is left in the current backoff slot. If less than 192us remain, the sequence manager must wait an *additional* backoff slot before initiating the Ack transmission. So it sets a flag indicating it must allow the slot counter to rollover an additional iteration.

Finally, at TxAck timer expiration (in non-SLOTTED mode), or at the qualified rollover of the slot timer (in SLOTTED mode), a Tx warmup is initiated. The conclusion of the Tx warmup will coincide precisely with a backoff slot boundary if SLOTTED mode is in effect. The sequence manager will then initiate the transmission of the Ack frame. The entire Ack frame will be built up by hardware; the Packet Buffer is not used for the auto-TxAck feature. The hardware will insert the following parameters into the Ack frame’s Frame Control Field:

Table 474. Frame Control Field for Hardware-generated Auto-TxAck Frame

FCF Field	Value
FrameType	Acknowledge
SecurityEnabled	0
FramePending	Determined by the Packet Processor.

Table continues on the next page...

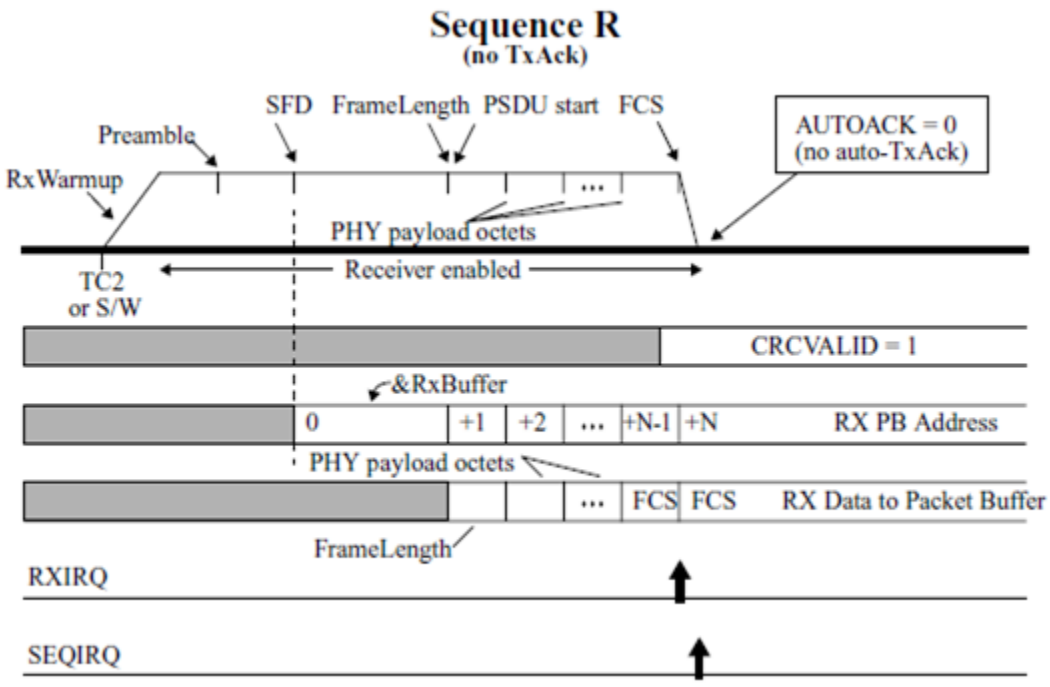
Table 474. Frame Control Field for Hardware-generated Auto-TxAck Frame (continued)

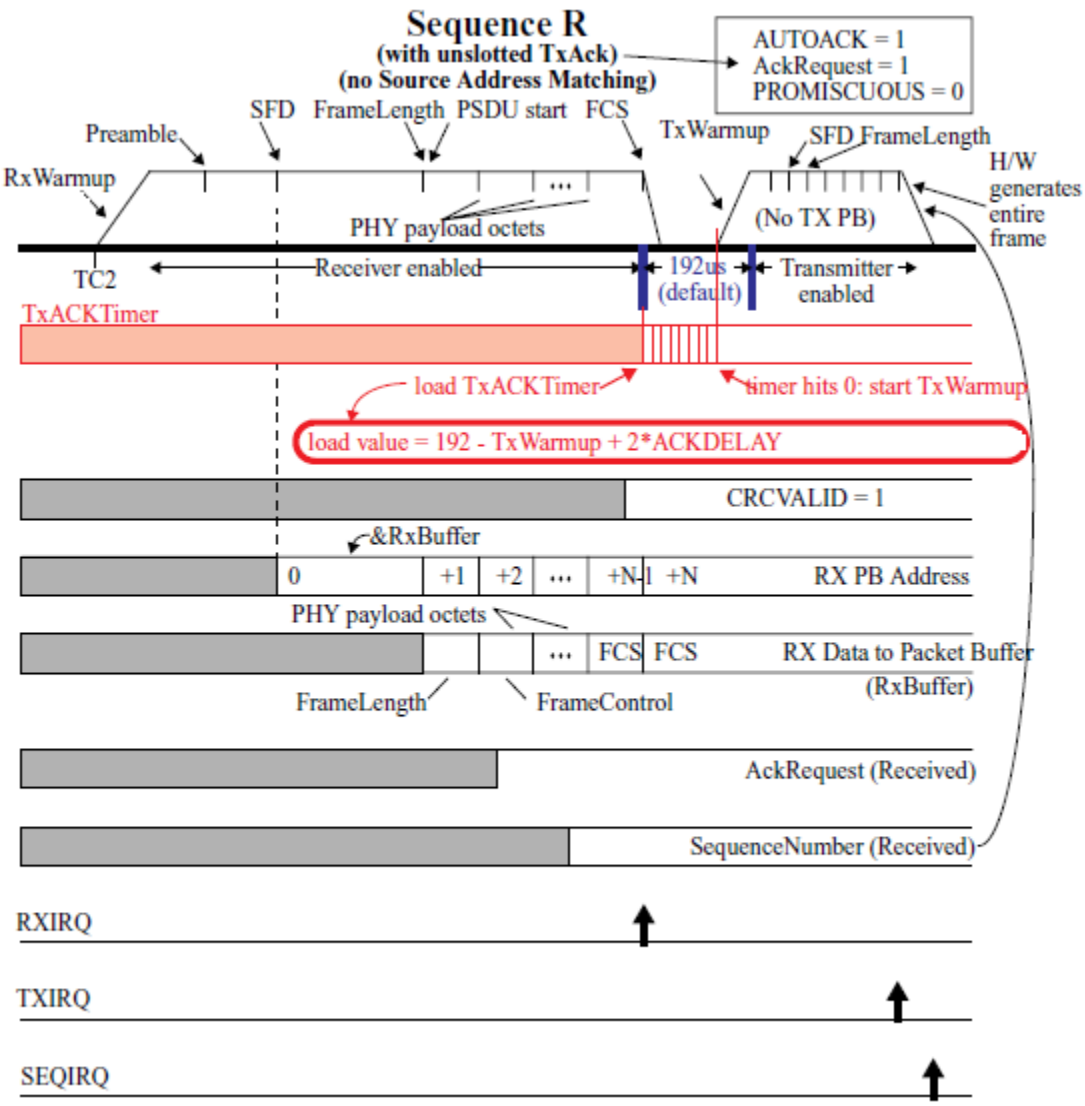
FCF Field	Value
	(see Packet Processor Chapter, Source Address Management.)
AckRequest	0
PanIDCompression	0
Reserved (bits 7-9)	000
DstAddrMode	0
FrameVersion	0
SrcAddrMode	0

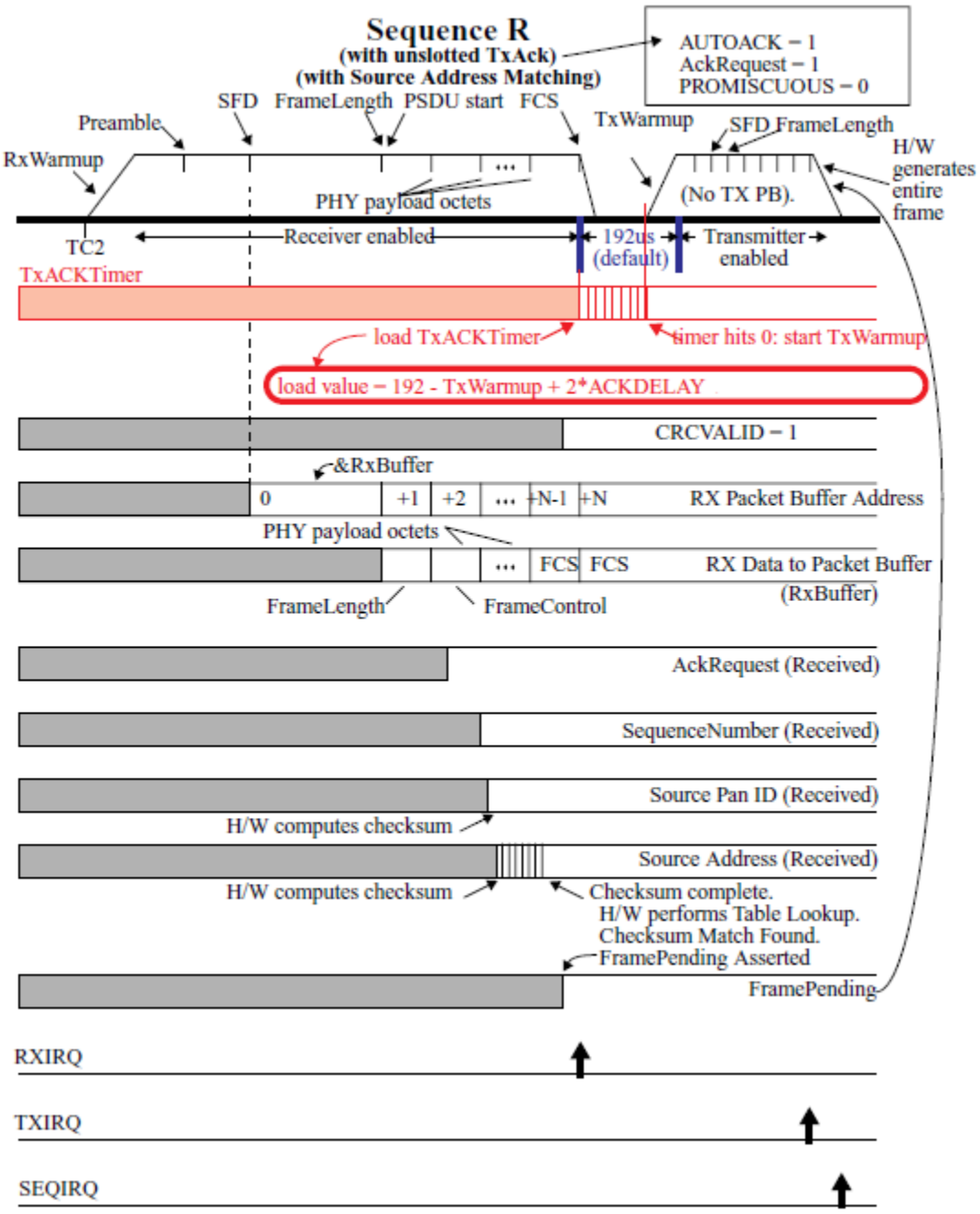
The Sequence Number for the Ack packet, will be the Sequence Number obtained from the previously received frame.

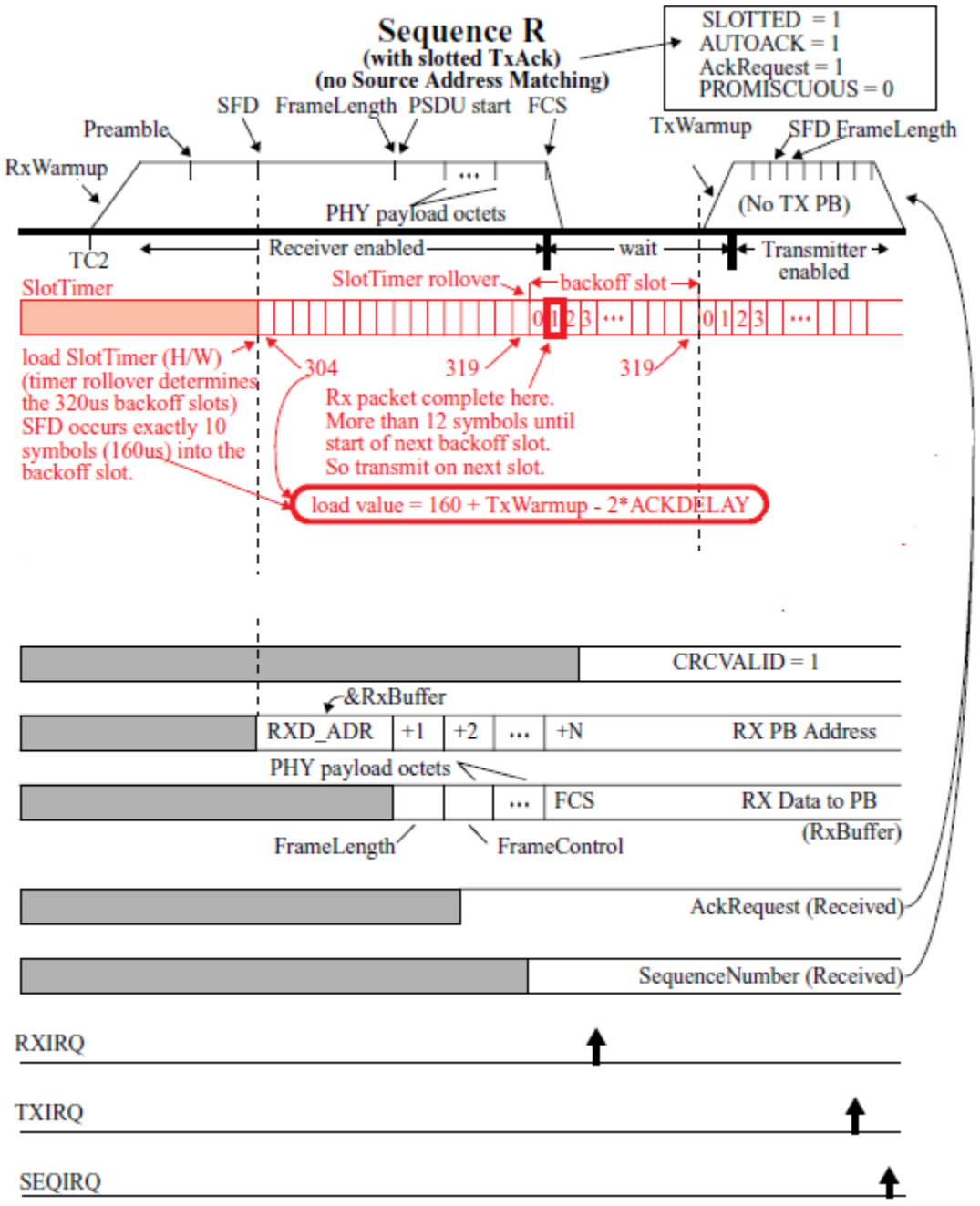
After the Ack frame has been transmitted, the sequence manager will issue a TXIRQ interrupt, and then perform a Tx warmdown. Finally, a SEQIRQ interrupt will be issued, and the sequence manager returns to SEQ_IDLE state.

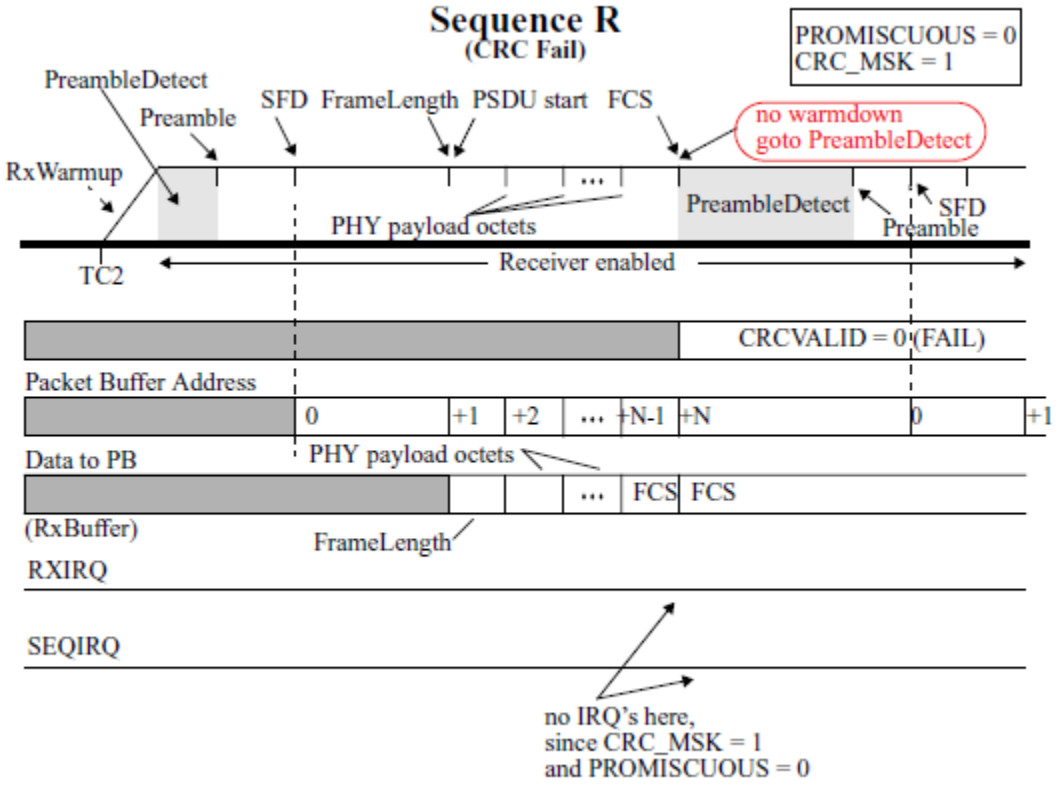
Sequence R Timing Diagrams

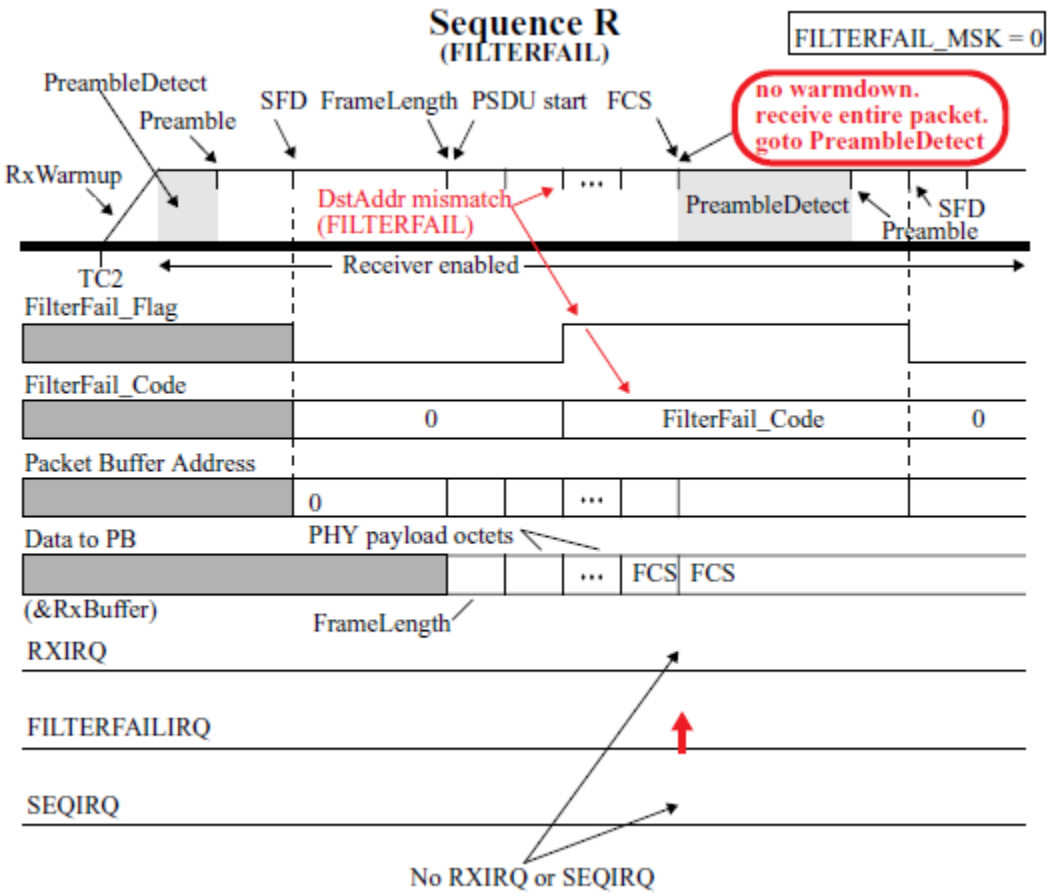



Sequence R
(with unslotted TxAck)
(no Source Address Matching)
AUTOACK = 1
AckRequest = 1
PROMISCUOUS = 0
Preamble
SFD
FrameLength
PSDU start
FCS
TxWarmup
SFD
FrameLength
H/W generates entire frame
RxWarmup
PHY payload octets
(No TX PB)
TC2
Receiver enabled
192us (default)
Transmitter enabled
TxACKTimer
load TxACKTimer
timer hits 0: start TxWarmup
load value = 192 - TxWarmup + 2*ACKDELAY
CRCVALID = 1
&RxBuffer
0
+1
+2
...
+N-1
+N
RX PB Address
PHY payload octets
FCS
FCS
RX Data to Packet Buffer (RxBuffer)
FrameLength
FrameControl
AckRequest (Received)
SequenceNumber (Received)
RXIRQ
TXIRQ
SEQIRQ


Sequence R
(with unslotted TxAck)
(with Source Address Matching)
AUTOACK = 1
AckRequest = 1
PROMISCUOUS = 0
SFD
FrameLength
PSDU start
FCS
TxWarmup
SFD FrameLength
Preamble
RxWarmup
H/W generates entire frame
PHY payload octets
(No TX PB).
TC2
Receiver enabled
192us (default)
Transmitter enabled
TxACKTimer
load TxACKTimer
timer hits 0: start TxWarmup
load value = 192 - TxWarmup + 2*ACKDELAY
CRCVALID = 1
&RxBuffer
0
+1
+2
+N-1
+N
RX Packet Buffer Address
PHY payload octets
FCS
FCS
RX Data to Packet Buffer (RxBuffer)
FrameLength
FrameControl
AckRequest (Received)
SequenceNumber (Received)
Source Pan ID (Received)
H/W computes checksum
Source Address (Received)
H/W computes checksum
Checksum complete.
H/W performs Table Lookup.
Checksum Match Found.
FramePending Asserted
FramePending
RXIRQ
TXIRQ
SEQIRQ


Sequence R
(with slotted TxAck)
(no Source Address Matching)
SLOTTED = 1
AUTOACK = 1
AckRequest = 1
PROMISCUOUS = 0
Preamble
SFD
FrameLength
PSDU start
FCS
TxWarmup
SFD
FrameLength
RxWarmup
PHY payload octets
(No TX PB)
TC2
Receiver enabled
wait
Transmitter enabled
SlotTimer
SlotTimer rollover
backoff slot
load SlotTimer (H/W) (timer rollover determines the 320us backoff slots) SFD occurs exactly 10 symbols (160us) into the backoff slot.
304
319
319
Rx packet complete here. More than 12 symbols until start of next backoff slot. So transmit on next slot.
load value = 160 + TxWarmup - 2*ACKDELAY
CRCVALID = 1
&RxBuffer
RXD_ADR
+1
+2
+N
RX PB Address
PHY payload octets
FCS
RX Data to PB (RxBuffer)
FrameLength
FrameControl
AckRequest (Received)
SequenceNumber (Received)
RXIRQ
TXIRQ
SEQIRQ


Sequence R
(CRC Fail)
PROMISCUOUS = 0
CRC_MSK = 1
PreambleDetect
Preamble
SFD
FrameLength
PSDU start
FCS
no warmdown
goto PreambleDetect
RxWarmup
PHY payload octets
PreambleDetect
SFD
Preamble
TC2
Receiver enabled
CRCVALID = 0 (FAIL)
Packet Buffer Address
0
+1
+2
+N-1
+N
0
+1
Data to PB
PHY payload octets
FCS
FCS
(RxBuffer)
FrameLength
RXIRQ
SEQIRQ
no IRQ's here,
since CRC_MSK = 1
and PROMISCUOUS = 0



55.4.9.1.2.2.3.1.3 Sequence T (Transmit)

Sequence T is the basic, standalone transmit sequence. Sequence T can be used to transmit all types of 802.15.4 PHY- and MAC-compliant frames. However, transmission of Acknowledge frames is not recommended using Sequence T. This is because transmission of an Acknowledge frame, usually follows a received frame, with a designated Sequence Number. The ZSM sequence manager automates the transmission of an Ack frame which follows a received frame, using Sequence R with the AUTOACK bit asserted. This is the recommended method for transmitting an Ack frame. This is especially beneficial, because the transmitted Ack frame octets are generated by hardware, and so no setup of a transmit packet in Packet Buffer, is required. However, sequence manager hardware does not prohibit Ack frames using Sequence T.

Sequence T allows for the insertion of 1 or 2 CCA (clear channel assessment) measurements prior to transmission, to ensure that the selected channel is idle. The CCA-before-TX feature is governed by two register bits, CCABFRTX and SLOTTED. All CCA measurements must indicate channel-idle, in order for the sequence manager to proceed to transmission; if the channel is determined by CCA to be busy, the sequence manager will terminate the sequence without a transmission. The number of CCA measurements attempted by the sequence manager, and the timing of the measurements, is shown in the following table.

Table 475. Timing of CCA Operations for Sequence T

CCABFRTX	SLOTTED	Number of CCA measurements	Timing of Initiation of CCA measurement #1	Timing of Initiation of CCA measurement #2	Timing of First Bit of Transmitted Frame (assumes channel idle)
0	X	0	-	-	Immediately follows TxWarmup
1	0	1	Immediately follows RxWarmup	-	Immediately follows RxWarmdown and TxWarmup
1	1	2	Immediately follows RxWarmup	320us after Initiation of CCA measurement #1	320us after Initiation of CCA measurement #2

Any CCA mode can be used for Sequence T and Sequence TR (CCA Modes 1, 2, and 3 are available. See Chapter CCA/ED/LQI). Software must configure the CCATYPE bits prior to initiating the autosequence, to select the CCA Mode.

The basic execution of a successful Sequence T is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU sets CCABFRTX if one or more CCA's are required prior to transmit
- MCU sets SLOTTED if in slotted mode (2 CCA's will be attempted)
- MCU writes XCVSEQ=T
- Wait for TC2 match (if TMRTRIGEN=1)
- If CCABFRTX=1 :
- ---> SEQ_MGR executes RxWarmup
- ---> SEQ_MGR initiates CCA measurement (takes 128us)
- ---> if CCA indicates channel busy, sequence terminates, CCAIRQ & SEQIRQ issued
- ---> otherwise, if CCA indicates channel idle:
- ---- ---> if SLOTTED=0, SEQ_MGR executes a RxWarmdown, followed by a TxWarmup
- ---- ---> if SLOTTED=1, SEQ_MGR initiates a 2nd CCA, 320us after initiating 1st CCA.
- ---- ---> if SLOTTED=1, and channel busy, sequence terminates, CCAIRQ & SEQIRQ issued
- ---- ---> if SLOTTED=1, and channel idle, SEQ_MGR executes a RxWarmdown, followed by a TxWarmup
- if SLOTTED=1, SEQ_MGR waits until 320us elapse after 2nd CCA initiation
- SEQ_MGR executes TxWarmup
- Preamble transmitted
- SFD transmitted
- FrameLength octet is obtained from Packet Buffer Address 0, and transmitted
- CRC engine is reset
- *N* additional octets are obtained from Packet Buffer and transmitted (*N* = FrameLength - 2)
- Each octet that is transmitted, is shifted through the CRC engine

- After Mth octet transmitted, FCS is available from CRC engine.
- FCS lower octet transmitted
- FCS upper octet transmitted
- SEQ_MGR issues TXIRQ
- SEQ_MGR executes TxWarmdown
- SEQ_MGR issues SEQIRQ

When Sequence T is initiated, the sequence manager first must determine if one or more CCA measurements are required. If CCABFRTX bit is asserted, the sequence manager warms up the receiver (analog and digital elements), via the TSM (Transceiver Sequence Manager). Immediately after the conclusion of the warmup, a CCA measurement is initiated. The CCA measurement takes 128us. If CCA indicates the channel is busy, the sequence manager warms down the receiver, issues CCAIRQ and SEQIRQ interrupts, and terminates the sequence. If, however, CCA indicates the channel is idle, the sequence manager inspects the SLOTTED bit to determine what to do next. If the SLOTTED bit is clear, the sequence manager issues a CCAIRQ interrupt, and then executes a warmdown of the receiver. If, however, the SLOTTED bit is set, the sequence manager does not issue a CCAIRQ interrupt; instead, the sequence manager initiates a second CCA measurement, precisely 320us after the initiation of the first CCA. This 320us interval is timed by the slot timer, which had been preloaded with 320 (microseconds) at the instant that Rx warmup was complete. At the conclusion of the second CCA, the sequence manager issues a CCAIRQ. If the second CCA indicates the channel is busy, the sequence manager warms down the receiver, issues a SEQIRQ interrupt, and terminates the sequence. If, however, CCA indicates the channel is idle, the sequence manager executes a warmdown of the receiver. Prior to executing the warmdown, the sequence manager reloads the slot timer, in order to precisely schedule the ensuing transmission. This time, the slot timer is loaded with a smaller value, in order to account for the fact that the timer must expire early in order to trigger the Tx warmup, which must complete at precisely the next backoff slot boundary. Thus, this time the slot timer is preloaded with:

Slot Timer Preload = 320 - TXWARMUPTIME + (2*TXDELAY).

The 320us is the duration of the backoff slot. The TXWARMUPTIME takes into account the fact that the TxAck timer must expire early for the Tx warmup to be complete at exactly the 320us point. The TXDELAY is a signed, fine-tune adjustment to the TxAck transmission time (+/- 62us).

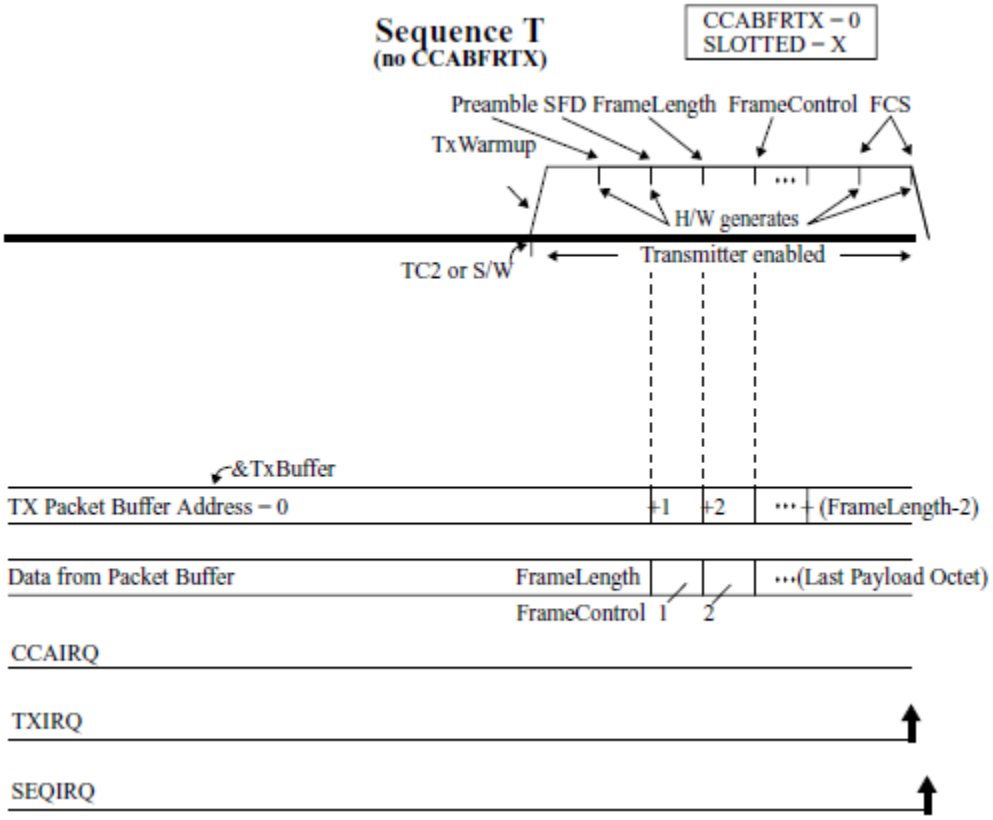
If the SLOTTED bit is asserted, once the slot timer expires, the sequence manager executes a Tx warmup. The conclusion of this warmup coincides with the backoff slot boundary. If the SLOTTED bit is deasserted, the sequence manager does not wait for the slot timer, and instead executes a Tx warmup immediately after the Rx warmdown.

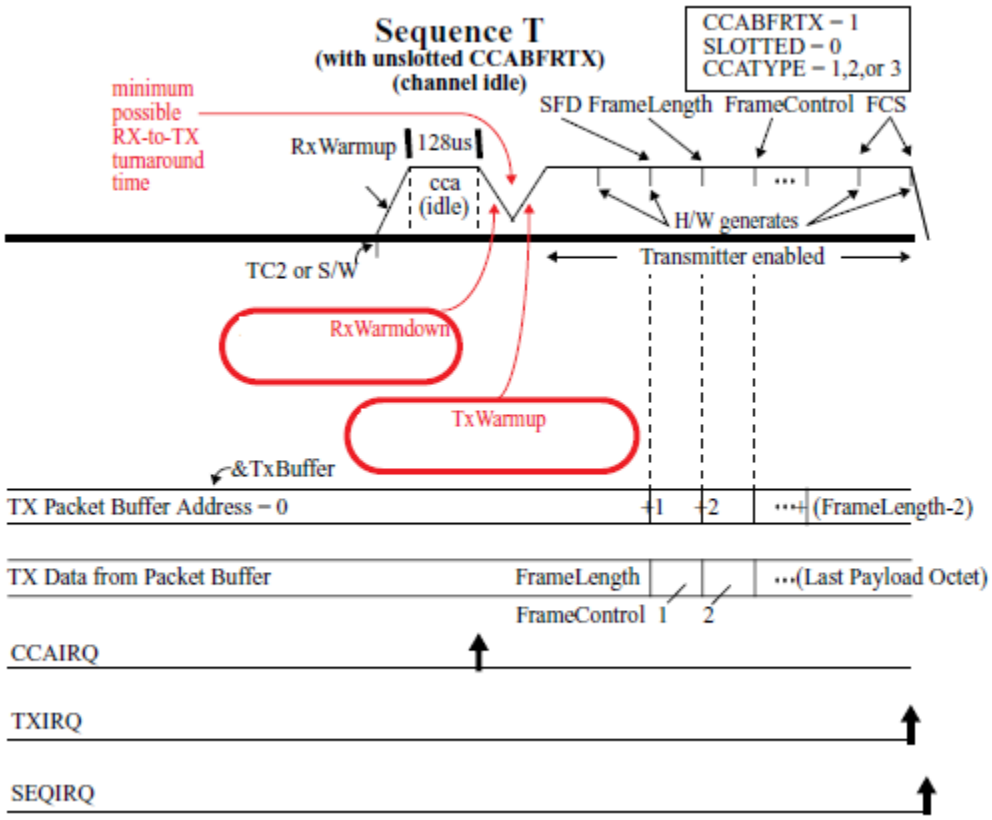
At this point, the sequence manager begins the transmit operation. The preamble and SFD are transmitted first. The preamble and SFD octets are generated by hardware, and are not included in the Packet Buffer. Then, the FrameLength octet is obtained from the Packet Buffer (address 0), and the buffer address pointer is incremented by one. The FrameLength octet is transmitted. The CRC engine is reset at this point. The FrameLength octet is used to determine how many more octets remain in the Packet Buffer. The number of remaining octets is:

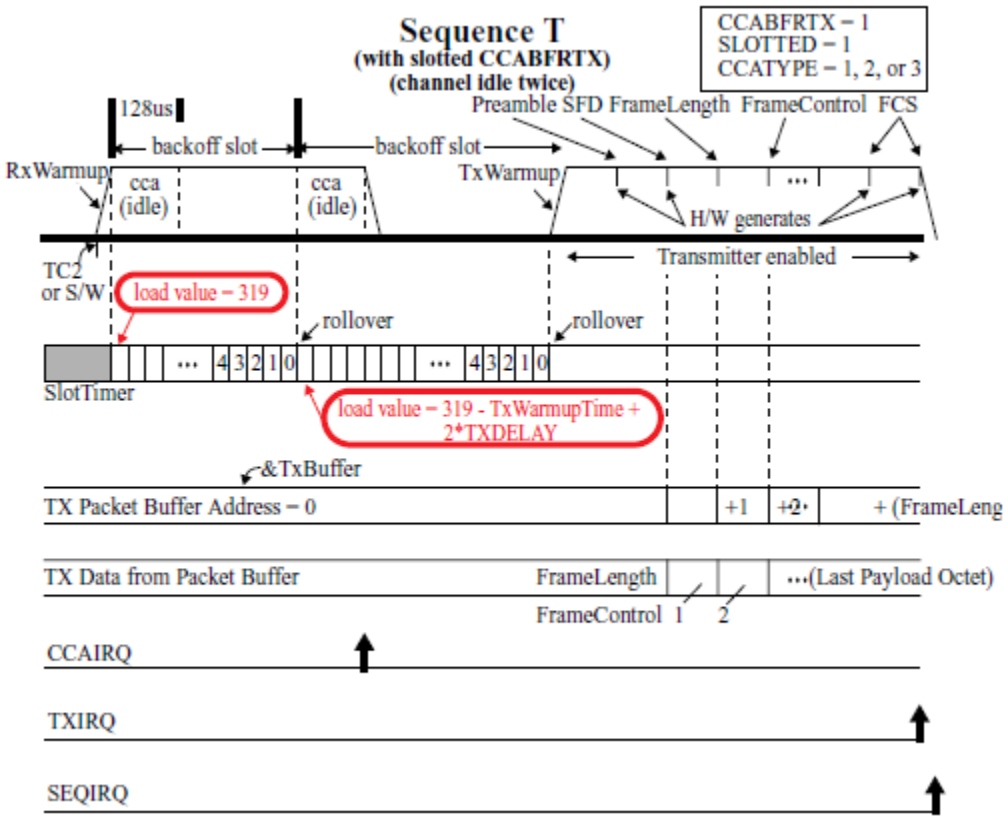
Number of octets remaining in Tx buffer = FrameLength - 2

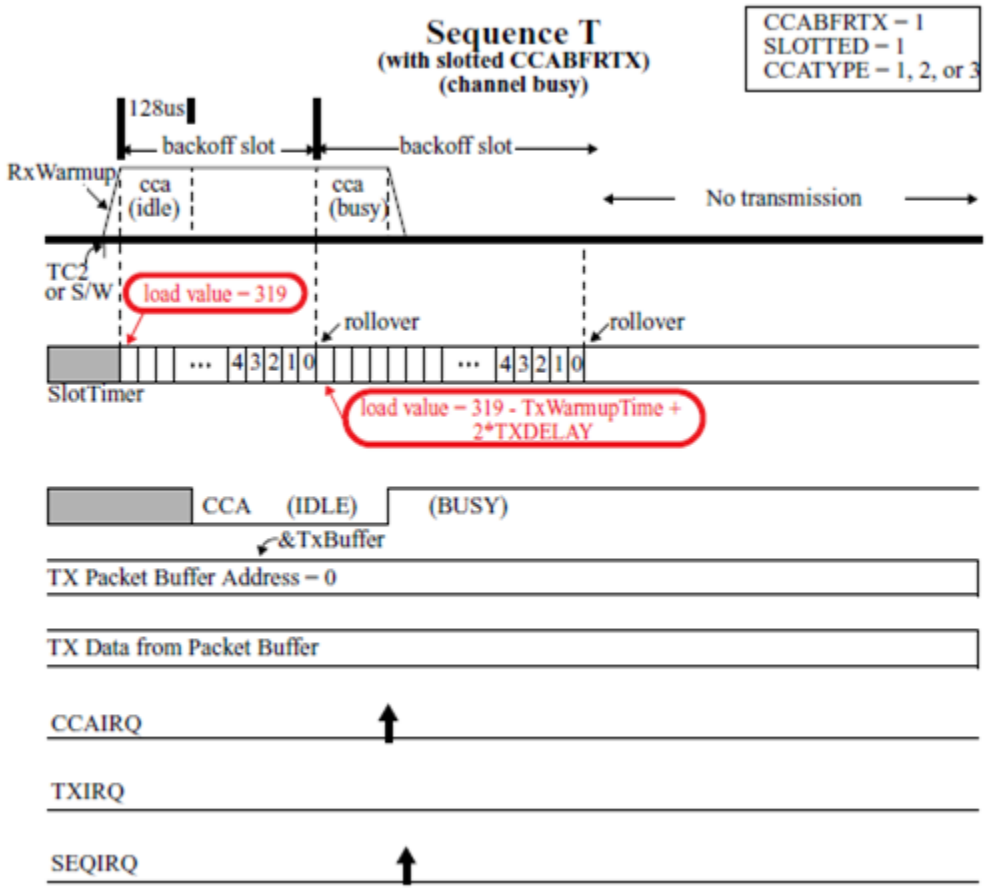
This is because the final 2 octets of the transmit frame constitute the FCS (Frame Check Sequence), and this value is hardware-computed. Thus, FCS is not part of the Tx buffer. Each of the octets in the Tx buffer is transmitted, and simultaneously shifted through CRC, to generate the FCS field. Once the Tx buffer is drained and all of its octets transmitted, the 2 FCS octets are transmitted, starting with the least significant octet. At this point, the transmit operation is complete. The sequence manager issues a TXIRQ interrupt. The sequence manager then executes a complete Tx warmdown, and follows up with a SEQIRQ interrupt. The sequence manager returns to SEQ_IDLE state.

Sequence T Timing Diagrams


Sequence T
(no CCABFRTX)
CCABFRTX = 0
SLOTTED = X
Preamble SFD FrameLength FrameControl FCS
TxWarmup
H/W generates
TC2 or S/W
Transmitter enabled
&TxBuffer
TX Packet Buffer Address = 0
+1
+2
(FrameLength-2)
Data from Packet Buffer
FrameLength
(Last Payload Octet)
FrameControl 1
2
CCAIRQ
TXIRQ
SEQIRQ


Sequence T
(with unslotted CCABFRTX)
(channel idle)
CCABFRTX = 1
SLOTTED = 0
CCATYPE = 1,2,or 3
minimum possible RX-to-TX turnaround time
RxWarmup
128us
cca (idle)
SFD
FrameLength
FrameControl
FCS
H/W generates
Transmitter enabled
TC2 or S/W
RxWarmdown
TxWarmup
&TxBuffer
TX Packet Buffer Address = 0
+1
+2
(FrameLength-2)
TX Data from Packet Buffer
FrameLength
(Last Payload Octet)
FrameControl 1
2
CCAIRQ
TXIRQ
SEQIRQ


Sequence T
(with slotted CCABFRTX)
(channel idle twice)
CCABFRTX = 1
SLOTTED = 1
CCATYPE = 1, 2, or 3
128us
backoff slot
backoff slot
Preamble SFD FrameLength FrameControl FCS
RxWarmup
cca
(idle)
cca
(idle)
TxWarmup
H/W generates
Transmitter enabled
TC2
or S/W
load value = 319
rollover
rollover
SlotTimer
load value = 319 - TxWarmupTime +
2*TXDELAY
&TxBuffer
TX Packet Buffer Address = 0
+1
+2
+ (FrameLeng
TX Data from Packet Buffer
FrameLength
(Last Payload Octet)
FrameControl 1
2
CCAIRQ
TXIRQ
SEQIRQ



55.4.9.1.2.2.3.1.4 Sequence C (CCA)

Sequence C is the standalone CCA sequence. During Sequence C, the sequence manager executes a Clear Channel Assessment (CCA). The result of the CCA measurement is reported back to software in the CCA bit of the IRQSTS2 Register. The CCA bit indicates either a busy channel (1), or an idle channel (0).

The basic execution of a Sequence C is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU writes XCVSEQ=C
- Wait for TC2 match (if TMRTRIGEN=1)
- SEQ_MGR asserts one of {**cca1_en, cca2_en, cca3_en, ed_en** } to CCA_DIG
- SEQ_MGR executes RxWarmup, via Transceiver Sequence Manager
- SEQ_MGR initiates CCA measurement by asserting **rx_cca_en** to CCA_DIG
- SEQ_MGR waits for CCA to complete
- CCA bit is set to the CCA status (1=busy 0=idle)
- CCAIRQ interrupt issued
- SEQ_MGR executes RxWarmdown
- SEQIRQ interrupt issued

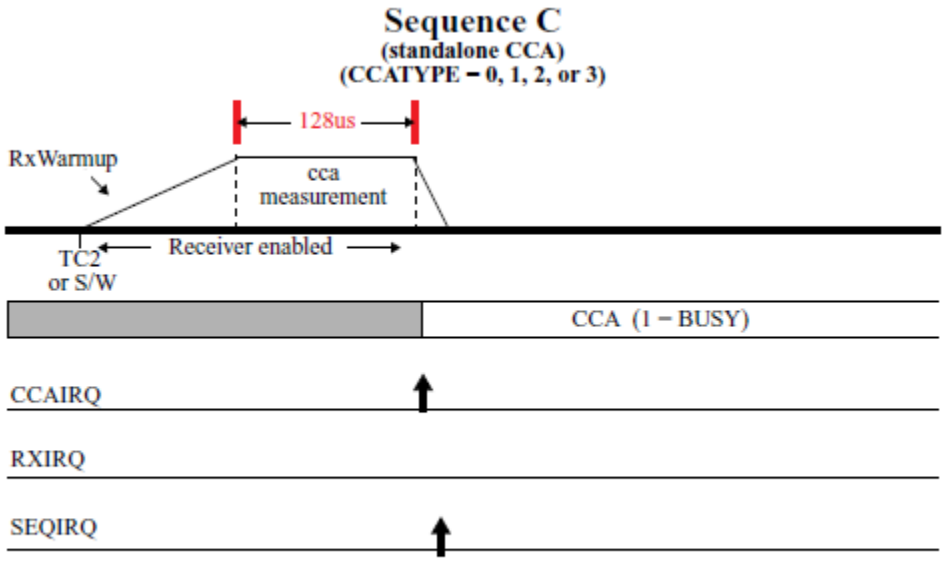
When Sequence C is initiated, the sequence manager asserts one of 4 control signals to CCA_DIG, based on which CCA or ED mode is programmed into CCATYPE[1:0]. The mapping of CCATYPE[1:0] to control signal is shown in the following table:

CCATYPE[1:0]	CCA_DIG CONTROL SIGNAL
00	ed_en
01	cca1_en
10	cca2_en
11	cca3_en

The asserted control signal will remain asserted for the duration of the Sequence C. The sequence manager warms up the receiver (analog and digital elements), via the Transceiver Sequence Manager (TSM). The TC3 “bracketing” timer has no effect on the sequence manager during Sequence C, so it cannot abort the sequence. At the completion of the warmup, the sequence manager initiates the CCA by asserting the **rx_cca_en** signal to CCA_DIG. While CCA is enabled, the CCA_DIG measures and averages the energy or signal on the channel. At the completion of the CCA process, the CCA bit is set to the status of the channel (idle or busy). The CCAIRQ is issued, the receiver is warmed down, the SEQIRQ is issued, and the sequence manager returns to SEQ_IDLE state.

Any CCATYPE setting may be used for Sequence C.

Sequence C Timing Diagrams



55.4.9.1.2.2.3.1.5 Sequence TR (Transmit/Receive)

Sequence TR is a combination Transmit/Receive sequence. The sequence is executed as a concatenation of 1 transmit operation followed by 1 receive operation, with a minimum TX-to-RX turnaround time in between. There are 2 permutations of Sequence TR, depending on the RXACKRQD bit. For both permutations, the Sequence T which constitutes the first half of a Sequence TR, is identical to the standalone Sequence T (XCVSEQ=2). This means that the transmit operation can be slotted or unslotted, and can be preceded by 1 or 2 CCA measurements, depending on the state of the CCABFRTX and SLOTTED bits.

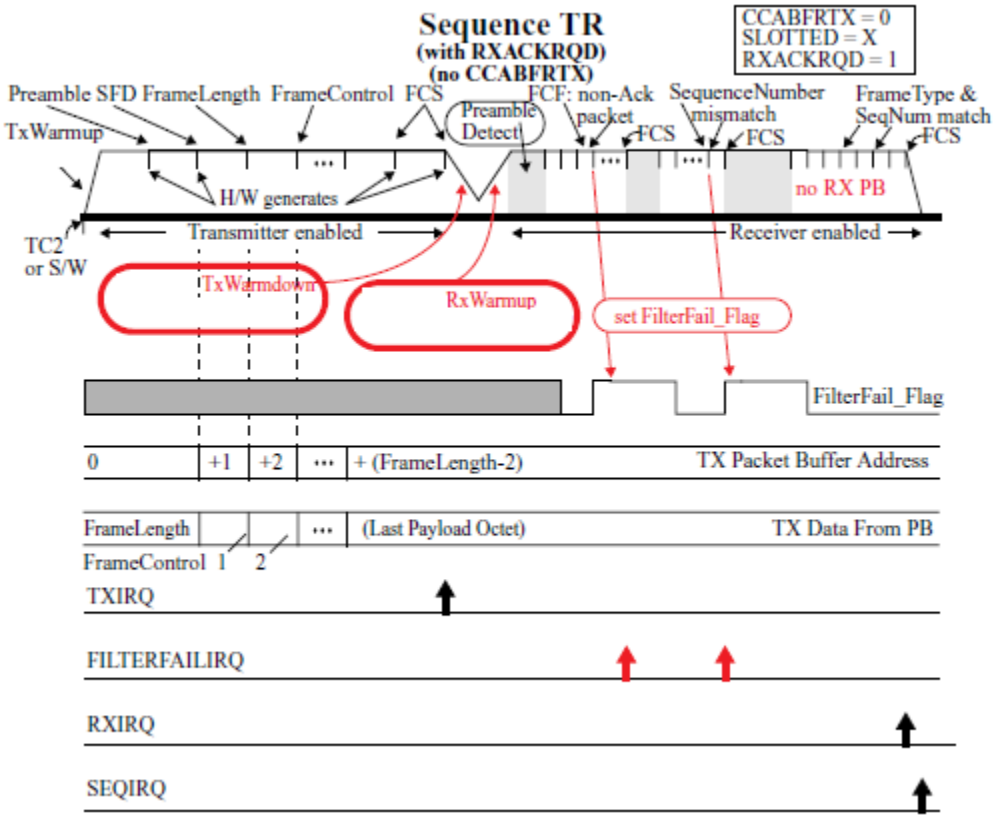
If RXACKRQD=0, then Sequence TR is executed as a Sequence T followed by a Sequence R, with a minimum TX-to-RX turnaround time in between. The Sequence R, which constitutes the second half of the Sequence TR, is identical to the standalone Sequence R (XCVSEQ=1). This means that the receive operation can be followed by an automatic, hardware-generated transmit Acknowledge frame, if all the necessary conditions are met. As with basic Sequence R, data for a successful receive operation is always transferred to Packet Buffer.

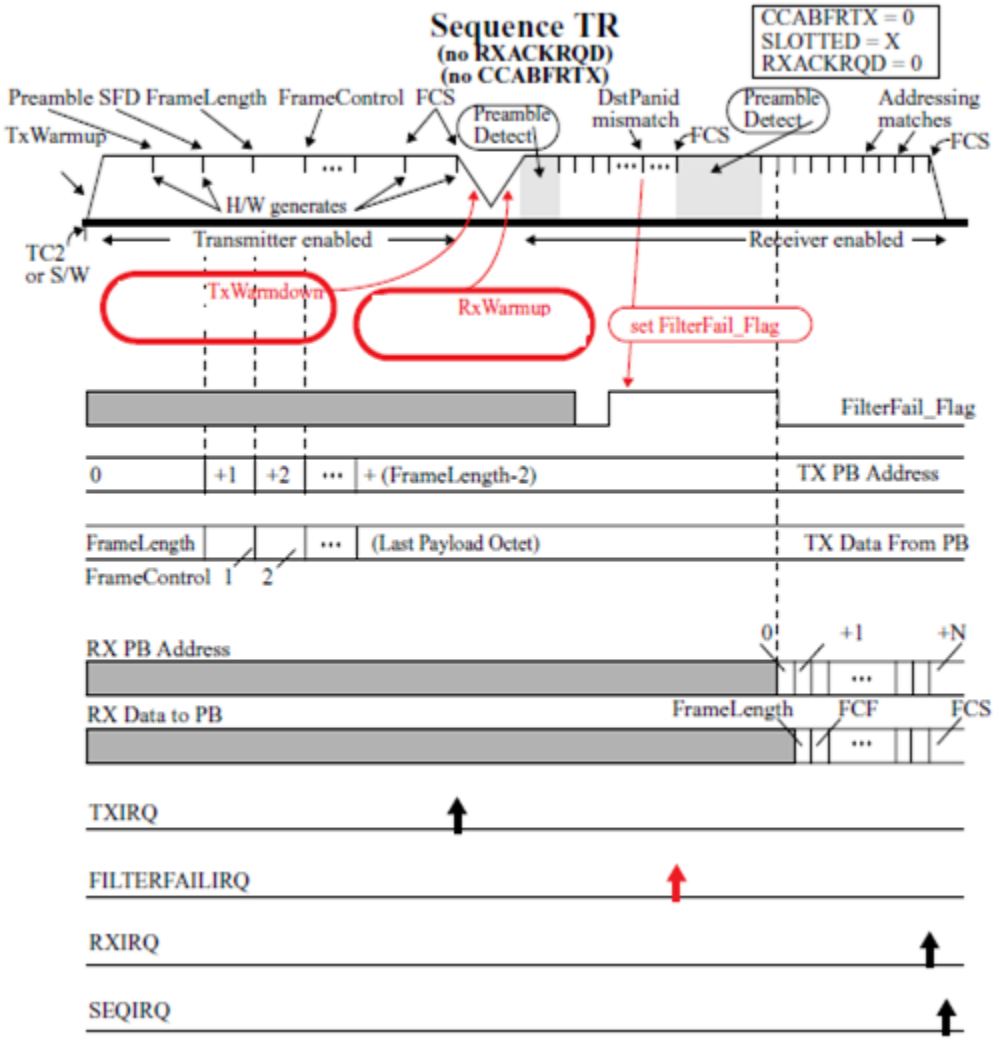
If RXACKRQD=1, then Sequence TR is executed as a Sequence T followed by a Receive-Acknowledge-Only frame. This type of receive operation is special, and not the same as a standalone Sequence R. The Ack-only receive operation filters all incoming frames, looking only for an Acknowledge frame whose Sequence Number matches the Sequence Number which was transmitted in the Sequence T portion of the sequence. All non-matching frames are discarded, and after each non-matching frame, the sequence manager will return the receiver to preamble-detect mode. This receive operation will continue until the matching Ack frame is received. Since this is a Receive-Acknowledge-Only operation, and not a Sequence R, there will be no receive octets transferred to Packet Buffer (an exception is made if ACTIVE_PROMISCUOUS=1; if so, all receive octets are transferred to the Packet Buffer, even for the Receive-Acknowledge-Only frame). The sequence will end with a RXIRQ interrupt, which indicates that the matching Ack frame was successfully received.

Needless to say, if during the transmit operation of a Sequence TR, an Acknowledge frame is being requested of the receiving end device (Frame Control Field: AckRequest=1), then Sequence TR with RXACKRQD=1 is the appropriate procedure. (Using Sequence TR with *RXACKREQ=0* is *not* recommended for this scenario, since there is no Sequence Number matching.)

Conversely, if during the transmit operation of a Sequence TR, an Acknowledge frame is *not* being requested of the receiving end device (Frame Control Field: AckRequest=0), then Sequence TR with RXACKRQD=1 should *never* be used, since a receive acknowledge frame will not be forthcoming. Instead, use Sequence TR with RXACKRQD=0, or simply Sequence T, if no followup receive frame is expected.

Sequence TR Timing Diagrams





55.4.9.1.2.2.3.1.6 Sequence CCCA (Continuous CCA)

Sequence CCCA is the Continuous CCA sequence. This sequence is designed to accommodate situations where channel availability may be infrequent, or low-duty-cycle, and a device needs to be able to transmit at the earliest available opportunity. During Sequence CCCA, the sequence manager repeats CCA measurements continuously until a channel-idle condition is found. The interval between the end of one CCA measurement, and the start of the next, is 63us. The actual interval between consecutive CCA measurements (i.e., from measurement-start to measurement-start), in the CCCA sequence, is 193us. After each CCA iteration, the CCA bit (IRQSTS2 Register) is set to the result of the measurement. As long as the CCA bit is high (busy) at the end of the iteration, the sequence manager will initiate a new measurement. Unlike Sequence C (standalone CCA), there is no CCAIRQ interrupt generated at the end of each CCA measurement, until the channel-idle condition is detected, at which point CCAIRQ is issued. The SLOTTED bit has no effect on this sequence. A Sequence CCCA will be aborted by the ZSM Sequence Manager if a TMR3 match occurs, and TC3TMOUT=1. CCA Modes 1, 2, or 3, can be used for Continuous CCA.

The basic execution of a Sequence CCCA is as follows:

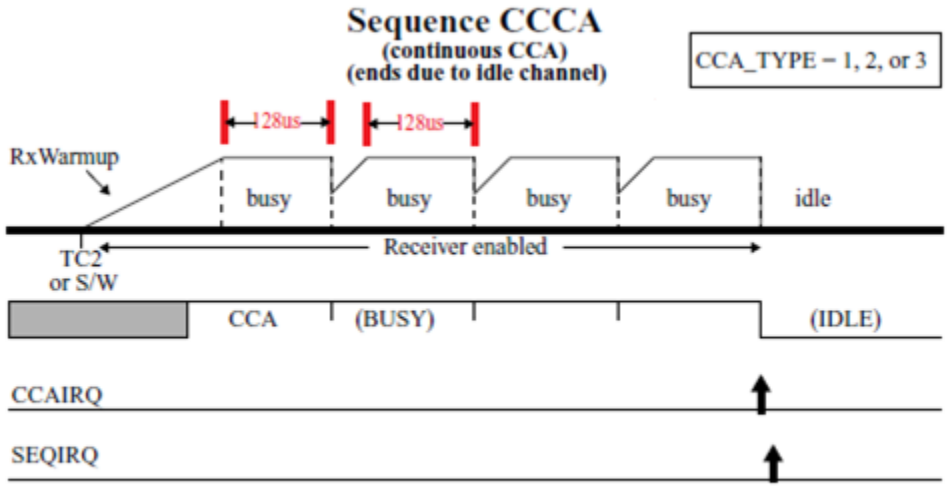
- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU writes XCVSEQ=CCCA
- Wait for TC2 match (if TMRTRIGEN=1)
- SEQ_MGR asserts one of {**cca1_en, cca2_en, cca3_en** } to CCA_DIG

- SEQ_MGR executes RxWarmup
- SEQ_MGR initiates CCA measurement by asserting **rx_cca_en** to CCA_DIG
- SEQ_MGR waits for CCA to complete
- CCA bit is set to the CCA status (1=busy 0=idle)
- If CCA=1, initiate a new CCA measurement immediately and repeat the previous 4 steps
- If CCA=0, CCAIRQ is issued
- SEQ_MGR executes RxWarmdown
- SEQIRQ interrupt issued

Note: The CCA bit (IRQSTS2 register) may change dynamically during the CCA measurement, and are only considered valid (and sampled by the Sequence Manager), at the end of the CCA measurement; therefore the user should not continuously poll CCA during CCCA and be expecting continuously-valid results

Note: The user can continuously monitor progress of the Sequence CCCA by reading the CCCA_BUSY_CNT register.

Sequence CCCA Timing Diagrams

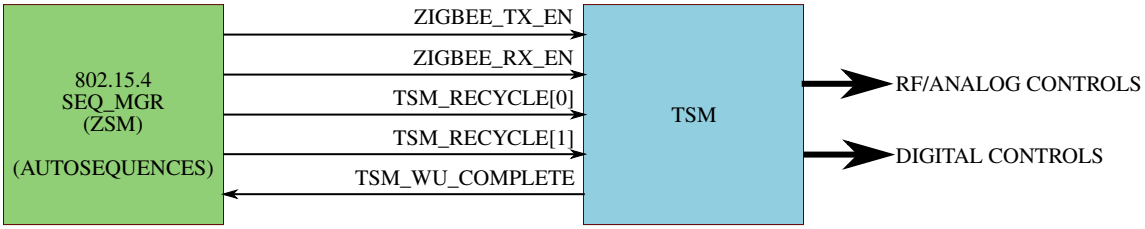


55.4.9.1.2.2.3.2 ZSM/TSM Interaction

The 802.15.4 Sequence Manager (ZSM) controls high-level autosequence timing, but offloads low level warmup and warmdown tasks to a dedicated, protocol-neutral Transceiver Sequence Manager, or TSM (see TSM Chapter). The ZSM initiates high-level functions, such as a CCA measurement operation or a transmit acknowledge operation, by commanding the TSM to execute an RX warmup, a TX warmup, a RX warmdown, a TX warmdown, or an RX recycle. Based on these commands, TSM generates the precise timing to control all of the digital, analog, and RF components of the TX and RX chains. The TSM is fully programmable, allowing all of the enabling and control signals to the transceiver components to be independently scheduled with 1us precision. TSM also allows all enable and control signals to be optionally overridden at any time, giving software complete control of each transceiver block, and ultimately the entire warmup/warmdown process if desired.

The ZSM and TSM sequence managers operate on the same clock domain, and both have 1us resolution. The TSM is fully programmable, while the ZSM is largely hardwired, with a few key exceptions. In the ZSM/TSM relationship, ZSM is the master and TSM is the slave. The key ZSM/TSM interface signaling is shown below:

ZSM/TSM INTERFACE SIGNALS



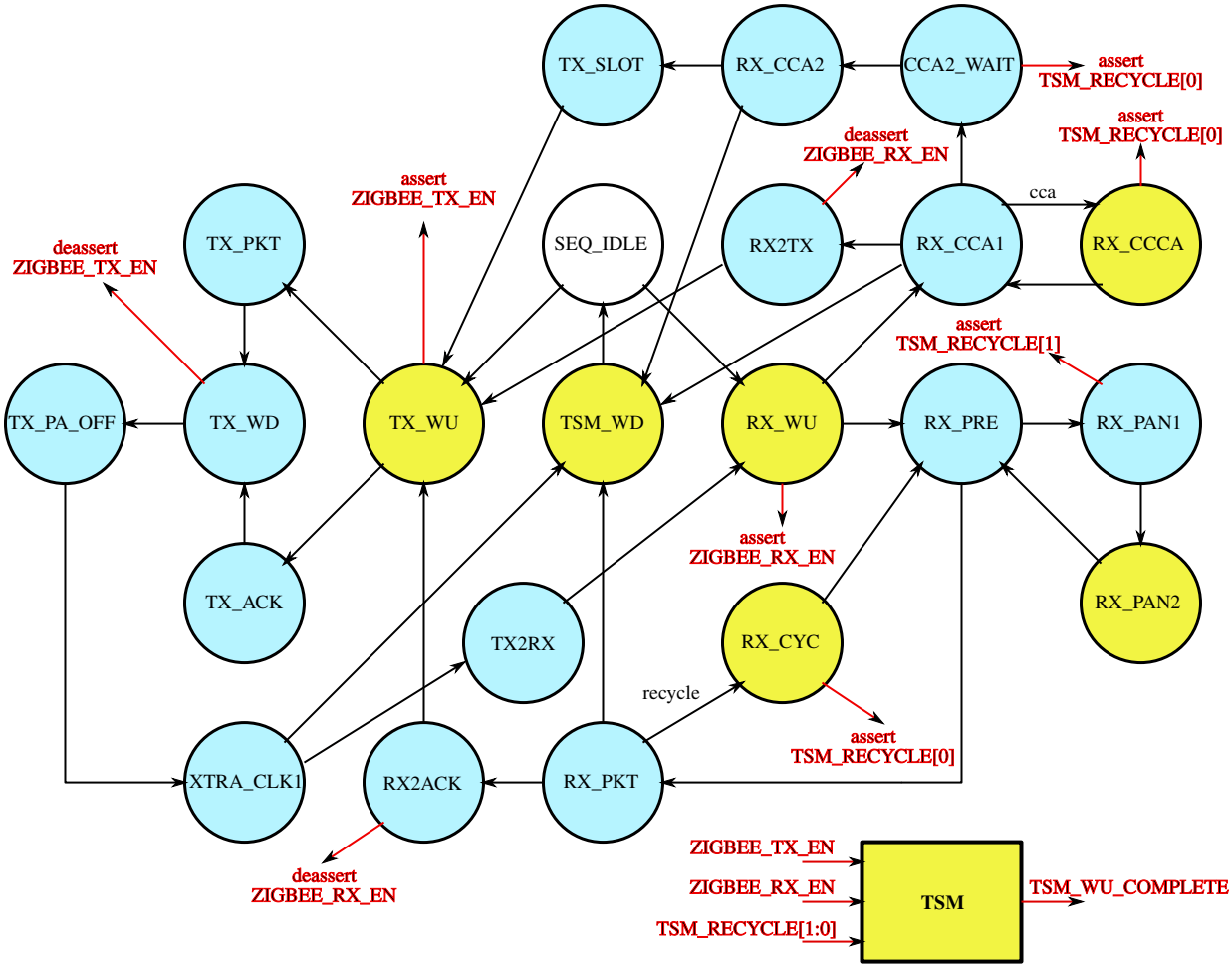
A description of the interface signals is provided in the table below.

ZSM/TSM INTERFACE SIGNAL	DESCRIPTION
ZIGBEE_TX_EN	A low-to-high transition initiates a TSM TX warmup, starting 1us after the transition. A high-to-low transition initiates a TSM TX warmdown, starting 1us after the transition.
ZIGBEE_RX_EN	A low-to-high transition initiates a TSM RX warmup, starting 1us after the transition. A high-to-low transition initiates a TSM RX warmdown, starting 1us after the transition.
TSM_RECYCLE[0]	This signal asserts in the ZSM RX_CYC state, and causes the TSM to jump from its “ON” phase, back to a point in the TSM RX warmup determined by the TSM’s RECYCLE_COUNT0[7:0] register in XCVR space. This process is used to execute an RX recycle. This signal is also asserted in the ZSM RX_CCCA state, with the same response by the TSM. This process is used to re-initialize the CCA function during Continuous CCA (CCCA) operations.
TSM_RECYCLE[1]	This signal asserts in the ZSM RX_PAN1 state, and causes the TSM to jump from its “ON” phase, back to a point in the TSM RX warmup determined by the TSM’s RECYCLE_COUNT1[7:0] register in XCVR space. This process is used to re-initialize the PLL digital on a new RF channel, during a Dual PAN mode on-the-fly channel change.
TSM_WU_COMPLETE	For the ZSM states which assert ZIGBEE_TX_EN or ZIGBEE_RX_EN, this 1us-wide signal from the TSM indicates that the warmup is complete, and the ZSM is now free to advance to its next state. The TSM_WU_COMPLETE is asserted 1us before the actual end of sequence, which is determined by the END_OF_TX_WU[7:0] register for TX warmup, and the END_OF_RX_WARMUP[7:0] register for RX warmup.

55.4.9.1.2.2.3.3 ZSM State Diagram

A ZSM state diagram is shown below. The states in yellow, are ZSM states which require interaction with the TSM. For example, a TSM control signal is asserted or deasserted, and/or a wait for warmup-complete from TSM is required; the assertions of the TSM control signals are shown in red. The states in blue have no interaction with TSM, so the timing of these states, and the transitions between such states, are governed by the ZSM itself.

802.15.4 SEQUENCE MANAGER AND TSM INTERACTION



Each ZSM state is represented in the hardware by a 5-bit state vector. The ZSM state can be monitored at any time by way of the read-only SEQ_STATE[4:0] register in 802.15.4 space. The mapping of ZSM state names to state vectors is shown in the table below:

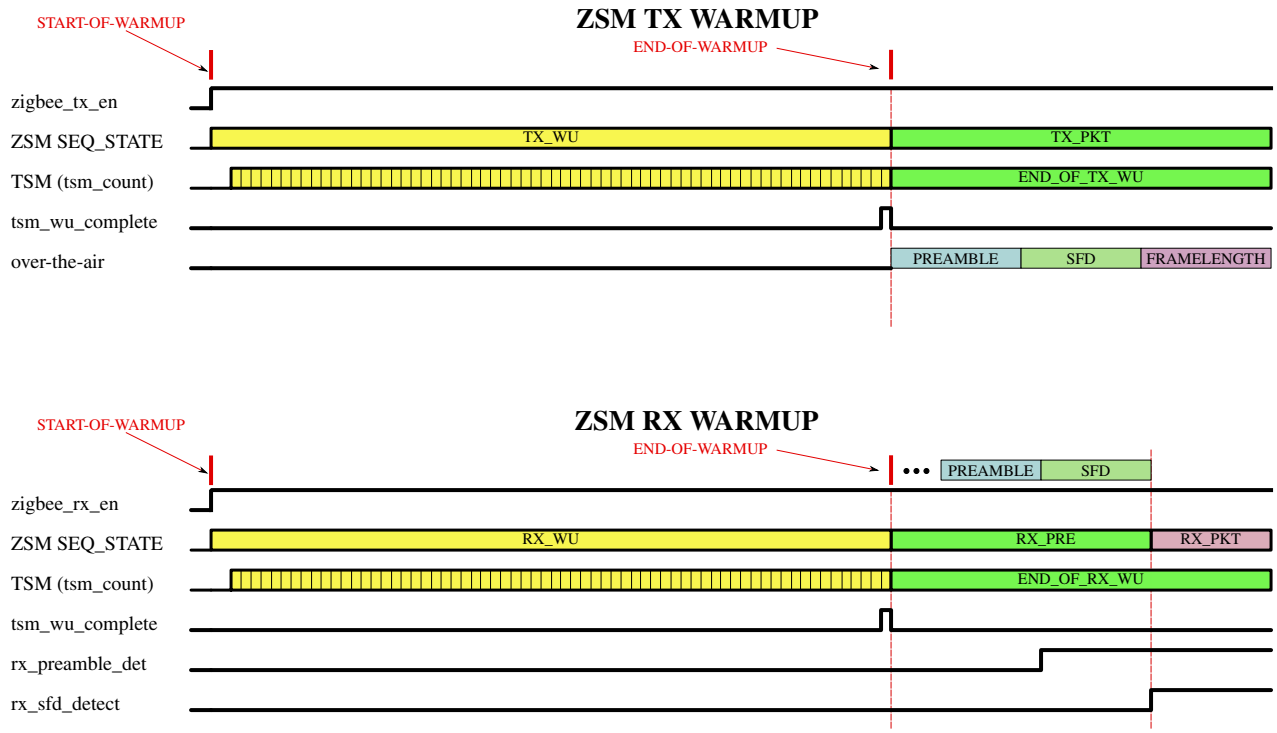
ZSM STATE	VECTOR
SEQ_IDLE	0x00
RX_WU	0x10
RX_PRE	0x14
RX_PKT	0x15
RX2ACK	0x16
RX_CYC	0x17
RX_PAN1	0x1D
RX_PAN2	0x1E
RX_CCA1	0x18

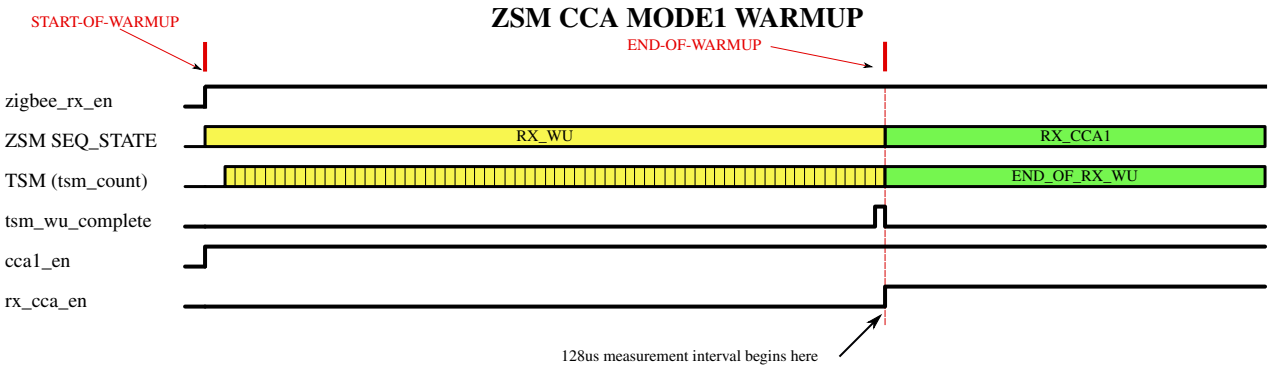
Table continues on the next page...

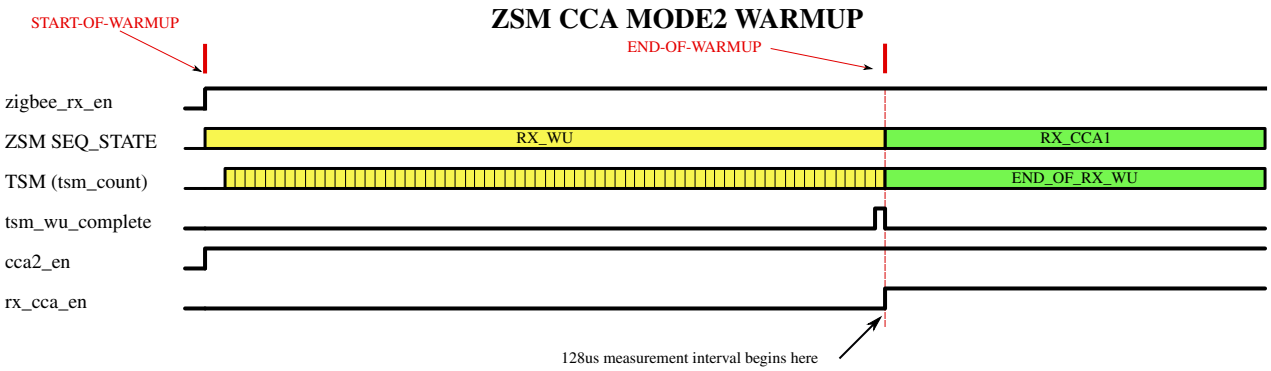
Table continued from the previous page...

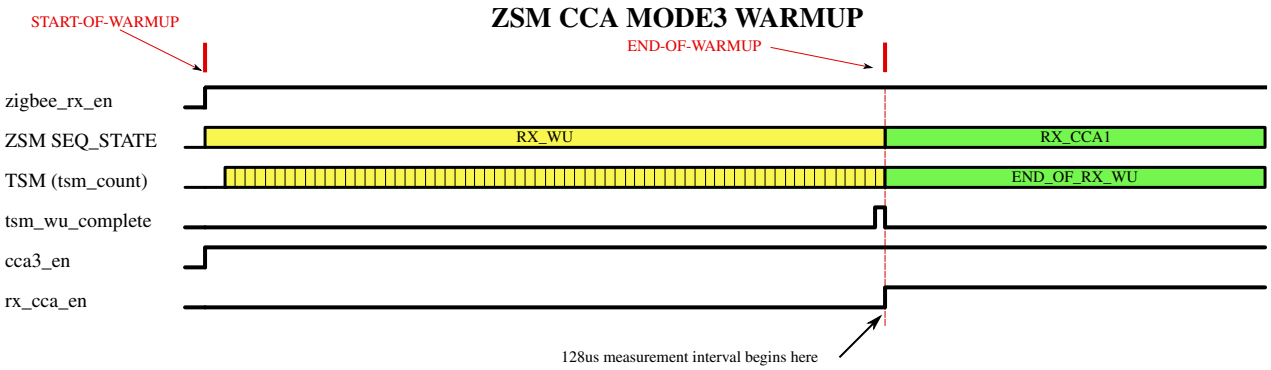
ZSM STATE	VECTOR
CCA2_WAIT	0x19
RX_CCA2	0x1A
RX_CCCA	0x1C
RX2TX	0x1B
TX_WU	0x01
TX_PKT	0x05
TX_ACK	0x06
TX_SLOT	0x07
TX_WD	0x08
TX_PA_OFF	0x09
XTRA_CLK1	0x0A
TX2RX	0x0B
TSM_WD	0x1F

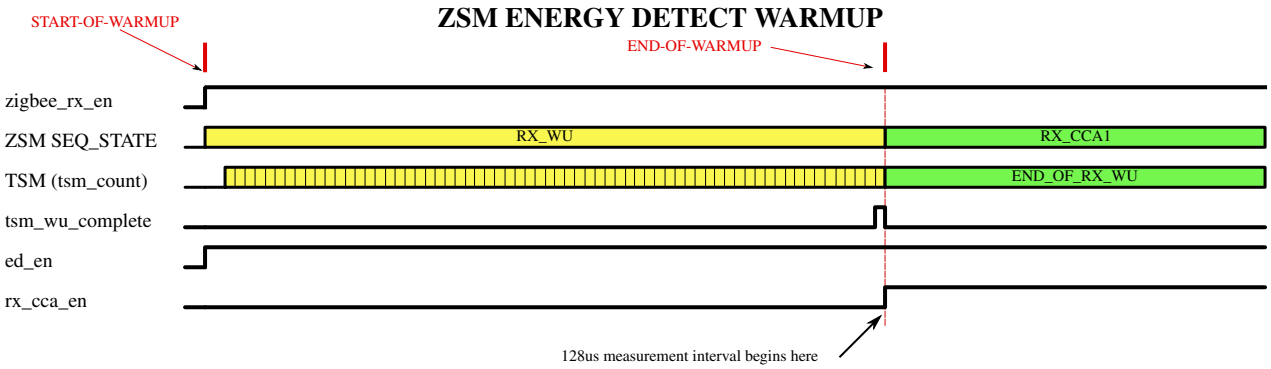
55.4.9.1.2.2.3.4 ZSM Warmup Timing




ZSM CCA MODE1 WARMUP
START-OF-WARMUP
END-OF-WARMUP
zigbee_rx_en
ZSM SEQ_STATE
RX_WU
RX_CCA1
TSM (tsm_count)
END_OF_RX_WU
tsm_wu_complete
cca1_en
rx_cca_en
128us measurement interval begins here


ZSM CCA MODE2 WARMUP
START-OF-WARMUP
END-OF-WARMUP
zigbee_rx_en
ZSM SEQ_STATE
RX_WU
RX_CCA1
TSM (tsm_count)
END_OF_RX_WU
tsm_wu_complete
cca2_en
rx_cca_en
128us measurement interval begins here


ZSM CCA MODE3 WARMUP
START-OF-WARMUP
END-OF-WARMUP
zigbee_rx_en
ZSM SEQ_STATE
RX_WU
RX_CCA1
TSM (tsm_count)
END_OF_RX_WU
tsm_wu_complete
cca3_en
rx_cca_en
128us measurement interval begins here


ZSM ENERGY DETECT WARMUP
START-OF-WARMUP
END-OF-WARMUP
zigbee_rx_en
ZSM SEQ_STATE
RX_WU
RX_CCA1
TSM (tsm_count)
END_OF_RX_WU
tsm_wu_complete
ed_en
rx_cca_en
128us measurement interval begins here

55.4.9.1.2.2.3.5 Dual PAN Mode

Dual PAN On-the-fly Channel Change

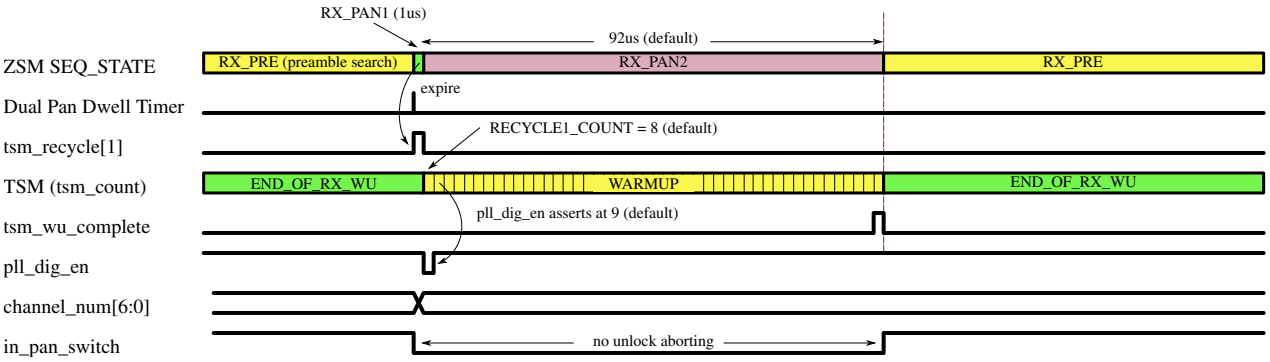
The 802.15.4 Sequence Manager (ZSM) supports Dual PAN mode, which allows the device to reside on 2 networks simultaneously, switching back and forth between the 2 networks under software, or hardware control. (See Packet Processor Chapter, Section Dual PAN Mode for more details on Dual PAN mode).

In Automatic Dual PAN Mode, the device must switch from one network to another under hardware control, at a software-programmed rate. Since the 2 networks often occupy different channels (frequencies), the sequence manager supports an "on-the-fly channel change", whereby, during a Sequence R, in preamble-search state (RX_PRE), the sequence manager will request the TSM to toggle **pll_dig_en** and switch to a new RF channel, wait out a PLL settling time, and then resume preamble search on the new channel. The entire on-the-fly channel change consumes 93us (based on default TSM programming), and consumes 2 ZSM states. The first state (RX_PAN1) issues **tsm_recycle[1]** to the TSM, which causes TSM to jump from its "ON" phase, back to the tsm_count value which is programmed into the RECYCLE_COUNT1[7:0] register in XCVR space. Based on the default programming for RECYCLE1_COUNT, along with the default programming for the TSM timing registers, this TSM recycle will cause the TSM to jump back to the point 1us before **pll_dig_en** is asserted, resulting in a 1us-long deassertion of **pll_dig_en**. The TSM will resume counting from that point, which will re-assert **pll_dig_en**, and the ZSM will simultaneously select the alternate Dual Pan channel. Meanwhile, the ZSM will advance to RX_PAN2 state and wait there pending **tsm_wu_complete** from the TSM. The TSM will resume counting until it once again until it reaches END_OF_RX_WU[7:0], and then will return to its ON phase. At the end of the TSM RX warmup, TSM will issue **tsm_wu_complete** to ZSM, which will return ZSM to RX_PRE (preamble search) state.

Normally, during RX sequences, the PLL is monitored for unlock conditions, and if such an unlock were to occur, the ZSM would abort the RX sequence by deasserting ZIGBEE_RX_EN to TSM. During RX_PAN1 and RX_PAN2 states, ZSM asserts **in_pan_switch** to the transceiver, which indicates that PLL unlock detection should be temporarily suspended while the channel change and subsequent PLL re-lock takes place. Once the ZSM transitions from RX_PAN2 state back to RX_PRE, ZSM will deassert **in_pan_switch** to resume PLL unlock monitoring.

As mentioned, the entire on-the-fly channel change consumes 93us, based on default TSM programming. This 93us represents a short "blind spot" during which preamble detection can't occur, while the device switches from one RF frequency to another. See diagram below.

ZSM DUAL PAN ON-THE-FLY CHANNEL CHANGE



Dual PAN Channel Override

When using Dual PAN mode, the channels are assigned to the 2 PAN's by programming the CHANNEL_NUM0[6:0] and CHANNEL_NUM1[6:0] registers in 802.15.4 space. This method allows a channel number, from 11 to 26, to be programmed into each register, and the channel number is processed downstream into the necessary Integer, Numerator, and Denominator components, required to generate the correct RF frequencies (see PLL Digital Chapter).

In Dual PAN mode, in case there is a need to generate a frequency which may be offset from the 16 prescribed 5MHz-spaced channels, to, for example, avoid interference on one of the Dual PAN channels, a method has been provided to do that, by designating one of the two PAN channels to use the transceiver's set of direct frequency-programming registers, instead of CHANNEL_NUMx. Programming the direct frequency-programming registers – integer, numerator, and denominator, allows an

RF frequency to be selected with much more precision than the 5MHz granularity of the 802.15.4 mapped-channel registers, CHANNEL_NUM0 and CHANNEL_NUM1.

Two bits have been provided in 802.15.4 space to realize this feature: ZB_DP_CHAN_OVRD_SEL and ZB_DP_CHAN_OVRD_EN. When ZB_DP_CHAN_OVRD_EN=1, this enables one of the Dual PAN channels to use the direct frequency programming. The ZB_DP_CHAN_OVRD_SEL bit determines *which* channel uses the direct programming, according to the following table:

ZB_DP_CHAN_OVRD_EN	ZB_DP_CHAN_OVRD_SEL	PAN0 Frequency Determined by …	PAN1 Frequency Determined by …
0	X	CHANNEL_NUM0[6:0]	CHANNEL_NUM1[6:0]
1	0	DIRECT FREQUENCY PROGRAMMING	CHANNEL_NUM1[6:0]
1	1	CHANNEL_NUM0[6:0]	DIRECT FREQUENCY PROGRAMMING

Direct Frequency Programming is accomplished by setting the PLL's Integer, Numerator, and Denominator registers to the appropriate values for the desired RF frequency. These registers are in XCVR address space, and are shown in the following table:

DIRECT FREQUENCY PROGRAMMING PARAMETER	REGISTER NAME (XCVR SPACE)	REGISTER MNEMONIC
Integer	LPM_INTG[6:0]	PLL_LP_SDM_CTRL1
Numerator	LPM_NUM[27:0]	PLL_LP_SDM_CTRL2
Denominator	LPM_DENOM[27:0]	PLL_LP_SDM_CTRL3

See the PLL Digital Chapter for RF frequency programming details.

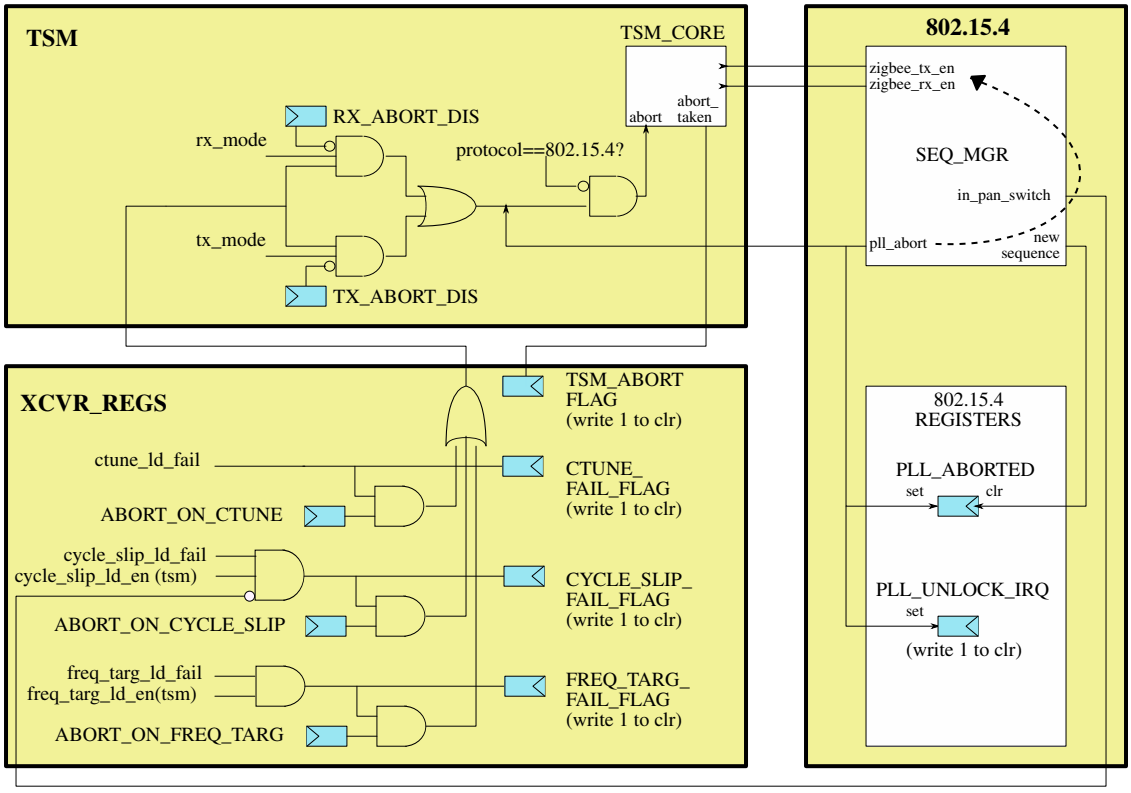
In Dual PAN automatic mode, if ZB_DP_CHAN_OVRD_EN=1, when the hardware switches to the PAN selected by ZB_DP_CHAN_OVRD_SEL as a direct-programming channel, the CHANNEL_NUMx[6:0] register is ignored and the directly programmed frequency is used instead; when the hardware switches back to the other channel, the CHANNEL_NUMx[6:0] register is used once again.

55.4.9.1.2.2.3.6 Sequence Aborting

All 802.15.4 sequences can be aborted by a PLL unlock detection. The unlock detection is qualified by the ZSM sequence manager; in other words, an unlock condition during the early stages of the PLL warmup is expected, and is not allowed to abort the sequence at that point. The primary rationale for the PLL unlock abort is to prevent transmission on an incorrect, or unstable, frequency during a transmit operation, or to increase battery life by not prolonging a receive or CCA operation that is bound to fail. Software can disable the PLL unlock auto-abort.

When a PLL unlock event occurs during a 802.15.4 autosequence, the TSM's abort facilities are not used, to allow the ZSM to handle the unlock condition directly, as shown in the following diagram:

802.15.4 PLL UNLOCK HANDLING



The combined-abort signal is blocked from reaching the TSM, and instead is routed to the 802.15.4 Sequence Manager (ZSM). This is due to the fact that the ZSM already includes abort-handling logic, which is being re-used from previous Freescale/NXP 802.15.4 products. Keeping the unlock handling intact, facilitates the porting of software from one NXP 802.15.4 product to the next.

For 802.15.4, the logic which determines which ZSM states allow aborting, is designed into the ZSM state machine. Not all states require an abort-on-unlock. When an unlock event occurs during a ZSM state that requires an abort, the ZSM will deassert its initiating signal to TSM (zigbee_tx_en or zigbee_rx_en), which results in a de facto abort, similar to a TSM-generated abort. For a ZSM abort, the PLL_UNLOCK_IRQ bit will become set, as will the PLL_ABORTED bit. The PLL_UNLOCK_IRQ bit is of type write-1-to-clear, and resides in the IRQSTS1 register of 802.15.4 space. The PLL_ABORTED bit is a read-only bit, and resides in the ABORT_STS register of 802.15.4 space; it will self-clear at the start of the next autosequence.

55.4.9.1.2.2.3.7 Clocks

The ZSM sequence manager use 2 clocks.

The main clock is **seq_mgr_clk**. This clock establishes the sequence manager timebase (1us), and runs the sequence manager state machine and auxiliary timers.

A secondary clock is “**ck”,** which is a gated 32mhz clock from the reference oscillator. This clock is used only for low-latency processing of a few incoming signals, such as the issuing of XCVSEQ sequence commands through the IPS bus interface, which can occur at a rate faster than 1us.

The **seq_mgr_clk** and **ck** derive from the same 32MHz source. Thus, these 2 clocks are in the same clock domain, and will be balanced (skew-controlled) during clock-tree synthesis. The **seq_mgr_clk** will be generated in the CRM. All inputs to the sequence manager are also in this same clock domain, including all the TSM interface signals, and the IPS bus-based control registers; there are no clock-domain-crossings or asynchronous interfaces in the ZSM sequence manager.

55.4.9.1.2.2.3.8 Reset

The 802.15.4 Sequence Manager has a single, active-low, asynchronous reset input: **rst_b** . At integration, this reset should be tied to the master reset for the entire transceiver. There are no special reset requirements.

55.4.9.1.2.2.3.9 Interrupts

The 802.15.4 Sequence Manager generates 4 interrupt "triggers":

1. rxirq_trig
2. txirq_trig
3. ccairq_trig
4. filterfail_irq_trig

These triggers are not a direct interrupts to the MCU, but are used to set individual 802.15.4 Interrupt Status bits to become 4 of the 12 802.15.4 Link Layer interrupt sources which, if enabled, will result in the 802.15.4 interrupt **ipi_int_zigbee** being asserted to the MCU. These 4 interrupt triggers will assert the 802.15.4 Interrupt Status Bits RXIRQ, TXIRQ, CCAIRQ, and FILTERFAIL_IRQ, respectively. All of these Interrupt Status Bits reside in the IRQSTS1 register of 802.15.4 Address Space. See Section Interrupts for more details on how the 802.15.4 Link Layer interrupt sources are consolidated into a single MCU interrupt line.

55.4.9.1.2.2.3.10 Appendix A

Effect of Timer TMR3 on Sequence Manager

Timer TMR3 can be used as a “bracketing” timer, to abort certain sequences after a software-determined timeout period. This timeout mechanism is enabled by setting the TC3TMOUT bit to 1. If TC3TMOUT=0, then the abort mechanism is disabled, and timer TMR3 functions merely as a software timer.

Timer TMR3’s hardware-abort functions are summarized in the following table.

Table 476. TMR3 Hardware Abort Functions

SEQUENCE	TC3TMOUT=0	TC3TMOUT=1
I	S/W timer	S/W timer
R	S/W timer	Abort sequence
T	S/W timer	S/W timer
C	S/W timer	S/W timer
TR	S/W timer	Abort sequence, (during receive operation only)
CCCA	S/W timer	Abort sequence

55.4.9.1.2.2.3.11 Appendix B

Continuous TX and RX Modes

Setting the bit CONTINUOUS_EN enables continuous transmission or reception, determined by which type of sequence is subsequently activated using the XCVSEQ register (i.e., Sequence T or Sequence R). Continuous TX and RX modes are designed to be used in tandem; i.e., two communicating devices, one serving as a continuous transmitter, and the other serving as a continuous receiver. The CONTINUOUS_EN resides in the SEQ_MGR_CTRL register of 802.15.4 address space. A description of the continuous TX and RX modes follows:

Continuous TX Mode

Continuous TX mode instructs the Sequence Manager to stay in a transmitting state forever (TX_PKT state), rather than shutting down the sequence after the 802.15.4 packet has been transmitted. In Continuous TX mode, data to be transmitted must be pre-established in the Packet Buffer prior to initiating a Sequence T. In Continuous TX mode, the first byte of the Packet Buffer

determines the length of the buffer, in bytes. In this mode, when the end of the buffer is reached, instead of transmitting FCS (CRC bytes), the TX modem circles back to the 2nd byte of the buffer (the byte after the length byte), and re-transmits the remainder of the buffer again, and this repeats continuously, until one of the following events occurs:

1) CONTINUOUS_EN is deasserted by SW

2) SW aborts the Sequence T by writing Sequence I to XCVSEQ

3) An PLL Unlock occurs, aborting the Sequence

To prepare for Continuous TX mode, the following steps should be taken:

1) Write the “length” byte to the first byte of the Packet Buffer

2) Write the remaining Packet Buffer contents (a total of “length” bytes)

3) Assert the CONTINUOUS_EN bit

4) Start the Transmit Sequence by writing Sequence T to XCVSEQ.

After Continuous TX mode has been entered, the SEQ_STATE register can be monitored to verify the Sequence Manager has reached the TX_PKT state, and remains in this state.

Continuous RX Mode

Continuous RX mode instructs the Sequence Manager to stay in a receiving state forever (RX_PKT state), rather than shutting down the sequence after a 802.15.4 packet has been received. For continuous RX mode, it is assumed that another device has been deployed as a continuous transmitter. The length of the continuous transmitter’s TX buffer, will also be the length of the continuous receiver’s RX buffer. For the receiver, in Continuous RX mode, the length of the buffer is determined by the DUAL_PAN_DWELL register. (Needless to say, Continuous RX mode should not be engaged simultaneously with Dual PAN mode, since DUAL_PAN_DWELL is serving an alternate purpose). For Continuous RX mode, DUAL_PAN_DWELL should be programmed to the same value as the “length” byte of the continuous transmitter’s TX buffer. See step 2) above, for Continuous TX mode. In Continuous RX mode, the first byte received (after Preamble and SFD), will be the “length” byte, which can be read from the RX_FRM_LEN register of the receiving device. (It is up to the user to guarantee that this length byte on the TX device matches the contents of the DUAL_PAN_DWELL register on the RX devices, by correctly programming the 2 devices). Each subsequent byte which is received, will be written to the next sequential address of the Packet Buffer, starting at PB address 0. When the number of received bytes matches the contents of DUAL_PAN_DWELL register, the Packet Buffer’s address generation logic will circle back to PB address 0, and store the next received byte to this location. Then, each subsequent received byte, will be written to the next sequential address of the Packet Buffer. In this way, the contents of TX buffer on the continuous transmitter, will be identical to the contents of the RX buffer on the continuous receiver (assuming no reception errors). To prepare for Continuous RX mode, the following steps should be taken:

1) Program DUAL_PAN_DWELL register with the contents of the “length” byte of the continuously-transmitting device’s TX Buffer

2) Assert the CONTINUOUS_EN bit

3) Start the Receive Sequence by writing Sequence R to XCVSEQ.

After Continuous RX mode has been entered, the SEQ_STATE register can be monitored to verify the Sequence Manager has reached the RX_PKT state, and remains in this state. The contents of the Packet Buffer can also be observed in realtime, to verify they match the contents of the TX device’s TX buffer. The Sequence Manager will remain in the RX_PKT state until one of the following events occurs:

1) CONTINUOUS_EN is deasserted by SW

2) SW aborts the Sequence R by writing Sequence I to XCVSEQ

3) An PLL Unlock occurs, aborting the Sequence

55.4.9.1.2.3 Packet Storage

The 802.15.4 Link Layer has access to a Packet Buffer to store data to be transmitted, and to receive incoming packet data. The 802.15.4 Link Layer software prepares data to be transmitted, by loading the octets, in order, into the Packet Buffer (Zigbee LL

TX_RAM Buffer). After reception, the octets received over the air are stored into the Packet Buffer (Zigbee LL RX_RAM Buffer), in the order they are received. The Packet Buffer contains separate spaces for TX and RX data, so incoming receive octets don't overwrite previously-loaded content intended for transmission. The Packet Buffer is large enough to accommodate the longest 802.15.4 packet, 127 bytes, for both transmit and receive.

The 802.15.4 Packet Buffer is contained within the multi-protocol Packet RAM. In the Packet RAM, space has been allocated for 802.15.4 packets, with separate spaces for TX and RX. The Packet RAM is 64 bits wide, so the 802.15.4 Link Layer controller hardware will address the RAM in an interleaved fashion as it stores octets to, or fetches octets from, the RAM-based packet buffer. The 802.15.4 TX buffer space begins at TX_RAM_BASE_ADDR, and 802.15.4 RX buffer space begins at RX_RAM_BASE_ADDR. Separate TX and RX spaces means that incoming RX packets don't overwrite TX packet data.

In addition, the 802.15.4 Link Layer controller includes acceleration for Source Address Matching (see Section Source Address Management). The hardware acceleration includes a Source Address Table, which is 128 entries by 16 bits wide. This table begins at TX_RAM_BASE_ADDR+0x100.

During packet reception, the MCU is allowed to access the RX packet buffer, in order to download the packet while it's being received, if so desired. In cases where both the MCU and the 802.15.4 Link Layer controller attempt to access RX packet buffer during reception, internal hardware arbitration is provided to delay the MCU access until 802.15.4 Link Layer completes its access.

During packet reception, the MCU is allowed to access the Source Address Table. Access to he Source Address Table is indirect, and is performed through the 802.15.4 Register Interface, specifically by programming SAM_TABLE register with the index of the table entry to be written or read, and the desired checksum in the case of write accesses. Because accesses to Source Address Table consist of discrete writes to 802.15.4 registers, and not long burst accesses, arbitration shall be able to accommodate Link Layer read accesses to the table while the Host access is underway, by delaying the Host access until the Link Layer completes.

During packet transmission MCU access to TX_RAM Buffer is also allowed. Arbitration shall be able to accommodate Link Layer read accesses to the TX RAM Buffer while the Host access is underway, by delaying the Host access until the Link Layer completes. This is very useful when sending enhanced ACK frames. But the 802.15.4 softwaremust not alter the Source Address Table once transmission has begun.

Arbitration

Arbitration hardware has been provided to allow the MCU to download data from the RX RAM Buffer while the packet is being received, in order to allow a headstart on software packet processing, if required by the application. Arbitration hardware has also been provided to allow the MCU to write data to the TX RAM Buffer while the packet is being sent, in order to allow software packet preparation, if required by the application. In the SoC, there is no provisions to wait the MCU on IPS or APB bus accesses to the radio. Therefore, in case of simultaneous access attempts by the MCU and the 802.15.4 Link Layer to the TX/RX RAM Buffer, arbitration hardware delays the MCU access to the TX/RX RAM Buffer until Link Layer completes its access. In the intended scenario, 802.15.4 software sets a RX watermark interrupt, and on this interrupt, proceeds to download the contents of the Packet RX buffer in RX RAM Buffer.

55.4.9.1.2.4 Memory Mapping

In the SoC memory map, space has been allocated to 802.15.4. Within the SoC, peripherals are allocated memory space in 4KB blocks, called IPS slots. 802.15.4 occupies one such slot. 802.15.4 address space consists of the 802.15.4 registers, as well as the Packet Buffer (RAM), which is memory-mapped into 802.15.4 memory space. The partitioning of address space for 802.15.4 is shown in the diagram below.

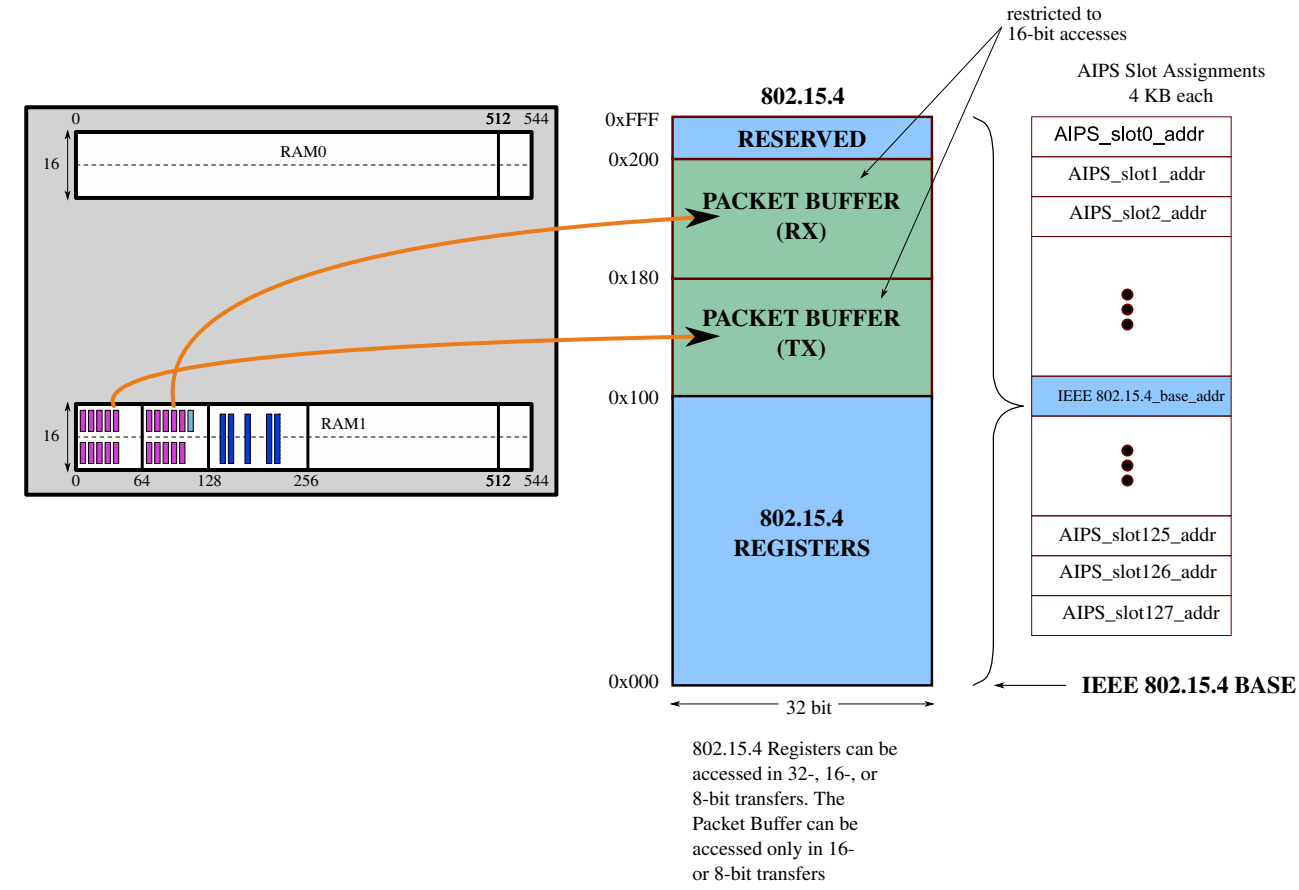


Figure 384. 802.15.4 MEMORY MAPPING

NOTE

See AIPS Memory Map for specific memory assignments details.

The Packet Buffer (RAM), is accessible by the MCU in 16-, or 8-bit accesses. The 802.15.4 Registers are accessible in 32-, 16-, or 8-bit accesses. The Source Address Table partition in RAM, is not accessible directly by the MCU, but via a prescribed protocol using the 802.15.4 SAM_TABLE register (see Section Source Address Management). Therefore, the table is not memory-mapped directly into 802.15.4 address space.

55.4.9.1.2.5 Event Timer

OVERVIEW

The 802.15.4 Link Layer Controller contains an internal Event Timer block that manages link layer timing.

The Event Timer consists of a prescaler and a 24-bit counter which increment whenever the 32MHz reference oscillator is operating (i.e., the controller is not in Deep Sleep Mode). Interrupts to the MCU may be generated when the current time of the counter matches several pre-determined values set in programmable registers. The current time is accessible at any time via a read operation, as well as, programmable via a write operation.

The Event Timer provides the following functions:

- Timer to generate current system time
- Interrupt generation at pre-determined system times

- Abort an RX sequence at pre-determined system time
- Latches timestamp value during packet reception
- Initiates timer-triggered sequences

Note: The Event Timer is actually a 28-bit value, with 24 integer bits and 4 fractional bits. For most operations involving the Event Timer, such as interrupt-generation and sequence-triggering, only the integer portion need be considered, and the fractional component can be ignored. Extended-precision (28-bit) Event Timer operations are required only to realize the increased accuracy required for entering and exiting Deep Sleep Mode. See Section Extended Precision Event Timer below.

EVENT TIMER TIME BASE

The Event Timer's base clock (tmr_clk) is derived from a prescaler which is clocked by the 32MHz crystal source. The prescaler provides a counter input frequency of 62.5KHz, which sets the granularity and resolution of the current time. The 24-bit counter automatically rolls over upon reaching its maximum value, yielding a maximum possible Event Timer duration of 268.436 seconds.

READING CURRENT TIME

"Current Time" is defined as the value of the Event Timer internal counter. This 24-bit value can be read as the upper 3 bytes of the EVENT_TMR register. The EVENT_TMR register resides at at address ZLL_BASE + 0x8, so the upper 3 bytes begin at ZLL_BASE + 0x9. The 24-bit value represents the integer component of the 28-bit Event Timer, which is all that is needed for most Event Timer operations. (For accessing the extended precision Event Timer, see Section Extended Precision Event Timer below.)

SETTING CURRENT TIME

"Current Time" is defined as the value of the Event Timer internal counter. This 24-bit value is available as the upper 3 bytes of the EVENT_TMR register. The EVENT_TMR register resides at at address ZLL_BASE + 0x8, so the upper 3 bytes begin at ZLL_BASE + 0x9. The current time is programmable, but does not have to be programmed. In the reset condition, the current time is set to zero. Current time advances from zero at the tmr_clk clock rate and rolls over to zero after reaching its maximum value.

The Event Timer can be updated in one of 2 ways:

EVENT_TMR_LD: Write the desired new EVENT_TMR value into the EVENT_TMR field of the EVENT_TMR register (upper 3 bytes), with the EVENT_TMR_LD bit set.

EVENT_TMR_ADD: Write the desired *increment* to the EVENT_TMR value into the EVENT_TMR field of the EVENT_TMR register (upper 3 bytes), with the EVENT_TMR_ADD bit set. The increment will be added to the current value of EVENT_TMR to become the new EVENT_TMR value. Since the Event Timer value is 24-bits, the addition is modulo 2^{24}.

Needless to say, EVENT_TMR_LD and EVENT_TMR_ADD are mutually exclusive, and no more than 1 of these bits should be set.

The 24-bit value represents the integer component of the 28-bit Event Timer, which is all that is needed for most Event Timer operations. (For accessing and updating the extended precision Event Timer, see Section Extended Precision Event Timer below.)

LATCHING THE TIMESTAMP

During packet reception, the 802.15.4 link layer controller has the ability generate a timestamp, or to latch a copy of the "current time" while continuing to increment its internal counter. This timestamp value latched within the Event Timer corresponds to the beginning of a receive packet where the actual packet data begins after the SFD (Start-of-Frame-Delimiter) has been received. The timestamp[23:0] can be accessed by reading the read-only register of the same name, TIMESTAMP, which resides at address ZLL_BASE + 0xC. Once the timestamp has been latched, it shall remain valid until the *next* SFD detection.

EVENT TIMER COMPARATORS

There are four 24-bit timer compare fields:

1. T1CMP[23:0] at address ZLL_BASE + 0x10
2. T2CMP[23:0] at address ZLL_BASE + 0x14
3. T3CMP[23:0] at address ZLL_BASE + 0x1C
4. T4CMP[23:0] at address ZLL_BASE + 0x20

And a special 16-bit timer compare field, for situations where the upper byte of EVENT_TMR is a don't-care.

- T2PRIMECMP[15:0] at address ZLL_BASE + 0x18

The TMR2IRQ status bit, can be set by a match to T2CMP or T2PRIMECMP. If register bit TC2PRIME_EN=1 (PHY_CTRL register, address ZLL_BASE + 0x04), then TMR2IRQ becomes set by a match to T2PRIMECMP (16-bit compare); if TC2PRIME_EN=0, then TMR2IRQ becomes set by a match to T2CMP (24-bit compare).

Each timer comparator has a enable bit that enables or disables the compare function. The enable bit is written to a 1 to enable the corresponding comparator, and the default condition is the timer compare disabled (reset to 0):

1. TMR1CMP_EN (for T1CMP)
2. TMP2CMP_EN (for T2CMP and T2PRIMECMP)
3. TMR3CMP_EN (for T3CMP)
4. TMR4CMP_EN (for T4CMP)

If a timer comparator is enabled using its associated bit, the corresponding interrupt status bit (TMRxIRQ) will be set by a timer compare match to its respective TxCMP register. If a timer comparator is disabled using its associated bit, the corresponding interrupt status bit (TMRxIRQ) will not be set by a timer compare match to its respective TxCMP register.

When enabled, all four fields can be continuously compared to the current value of the Event Timer counter. When a match occurs, the following corresponding internal status flags assert:

1. TMR1IRQ (T1CMP match)
2. TMR2IRQ (T2CMP or T2PRIMECMP match)
3. TMR3IRQ (T3CMP match)
4. TMR4IRQ (T4CMP match)

These 4 status bits all reside in the IRQSTS register at address ZLL_BASE + 0x0. The 4 interrupt status bits are write-1-to-clear. The status bit remains set until a 1 is written to the bit location in the IRQSTS register. Writing 0 to the bit, or reading the bit, will not change its state.

When a comparator match occurs and the internal status flag asserts, the following interrupt masks can enable an interrupt on the ipi_int_zigbee line to the MCU:

1. TMR1MSK, applies to TMRQ1IRQ
2. TMR2MSK, applies to TMRQ2IRQ
3. TMR3MSK, applies to TMRQ3IRQ
4. TMR4MSK, applies to TMRQ4IRQ

If the interrupt mask is set to 1 (masked), the respective interrupt status bit, TMRxIRQ, will not cause an interrupt on the ipi_int_zigbee line. If the interrupt mask is set to 0 (enabled), the respective interrupt status bit, TMRxIRQ, will cause an interrupt on ipi_int_zigbee. Timer mask bits, TMRxMSK, have no effect on the state of the TMRxIRQ interrupt status bits; they only allow (or prevent) the respective TMRxIRQ bit from generating an interrupt to the MCU on ipi_int_zigbee.

INTENDED EVENT TIMER USAGE

It is intended that the system utilize the current time value and the timer compare functions of the Event Timer to schedule system events, including:

1. Generating time-based interrupts
2. Aborting an RX operation
3. Triggering transceiver operations

GENERATING TIME-BASED INTERRUPTS

Generating time-based interrupts is accomplished by setting timer compare values relative to the current time, allowing the Event Timer counter to increment until a timer compare match is generated, and using this match to generate an interrupt to the host. The general procedure is as follows:

1. Disable the timer compare: set TMRxCMP_EN=0

2. Enable the timer compare interrupt mask: set TMRxMSK=1
3. Read the current time value from EVENT_TMR register
4. Determine an offset to this value to equal desired future time.
5. Write to EVENT_TMR register with the EVENT_TMR field set to this offset, and set EVENT_TMR_ADD=1
6. Re-enable the timer compare: set TMRxCMP_EN=1
7. Unmask the timer compare interrupt mask: set TMRxMSK=0
8. Wait for TMRxIRQ
9. After TMRxIRQ, program the appropriate TMRxCMP_EN bit to disable the compare function. If this is not done, the compare function will continue to run and generate another interrupt every time the counter rolls over and again matches the comparator.

USING T3CMP TO ABORT AN RX OPERATION

The Event Timer provides a timer-based mechanism to automatically abort an RX operation. The 802.15.4 link layer is put into an RX operation when XCVSEQ[2:0] (PHY_CTRL register, address ZLL_BASE + 0x4) is set to 0x01. An RX autosequence will commence, and will begin a search for preamble. Software can program in advance, a hardware timeout to occur, such that the RX autosequence can be automatically aborted by hardware after a certain amount of time elapses without a packet being detected. The general procedure is as follows:

1. Read the current time value from EVENT_TMR register
2. Add an offset to this value to equal desired future time to abort the RX operation.
3. Program register T3CMP[23:0] to value future time.
4. Program TC3TMOUT=1, (PHY_CTRL Register, address ZLL_BASE + 0x4) to 1, to enable hardware aborting of the RX operation.
5. Program XCVSEQ=1, to start the RX autosequence
6. When current time equals T3CMP[23:0], the 802.15.4 Sequence Manager aborts the RX autosequence, and the Sequence Manager returns to SEQ_IDLE state. The SEQIRQ interrupt status bit (IRQSTS Register, address ZLL_BASE + 0x0) will be set, and the TMR3IRQ interrupt status bit will also be set. The RXIRQ interrupt status bit will not be set, since no packet was received before the timeout. In the SEQ_CTRL_STATUS register (address ZLL_BASE + 0x78), the TC3_ABORTED status bit will also be asserted to indicate that the type of abort that occurred was a TMR3 timeout. This bit will self-clear at the start of the next autosequence.

As mentioned above, TMR3 with TC3TMOUT=1 can be used to abort an RX operation. Such an RX operation can be entered by launching a Sequence R (XCVSEQ=1), and can also be entered by launching a Sequence TR (XCVSEQ=4); after the TX operation of a Sequence TR completes, an RX operation begins, and can be aborted using this method. In addition, a Continuous CCA autosequence (XCVSEQ=5) can be aborted using this method. When a TMR3 timeout occurs with TC3TMOUT=1 during transceiver operations other than those listed above, TMR3 will not abort the autosequence, and TMR3 becomes merely a software timer, capable of setting TMR3IRQ and interrupting the MCU. The table below shows the effect of TMR3 timeouts on the 802.15.4 autosequences:

Table 477. TMR3 Hardware Abort Functions

SEQUENCE	TC3TMOUT=0	TC3TMOUT=1
I	S/W timer	S/W timer
R	S/W timer	Abort sequence
T	S/W timer	S/W timer
C	S/W timer	S/W timer
TR	S/W timer	Abort sequence, (during receive operation only)
CCCA	S/W timer	Abort sequence

USING T2CMP OR T2PRIMECMP TO TRIGGER TRANSCEIVER OPERATIONS

The Event Timer provides a timer-based mechanism to automatically launch a transceiver autosequence. Using this method, a sequence can be scheduled for time in the future, and the MCU can be put into a reduced power state, without the need to wake up at the appropriate time to start the autosequence. The general procedure is as follows.

1. Read the current time value from EVENT_TMR register.
2. Add an offset to this value to equal desired future time to launch the autosequence.
3. For a 24-bit compare, program register T2CMP[23:0] to value future time, and set TC2PRIME_EN=0. For a 16-bit compare, program register T2PRIMECMP[15:0] to value future time, and set TC2PRIME_EN=1. Set TMRTRIGEN=1 and program the desired autosequence into XCVSEQ[2:0]. The Sequence Manager will not launch the sequence immediately because it sees TMRTRIGEN=1.
4. When current time equals T2CMP[23:0] (for a 24-bit compare), or T2PRIMECMP[15:0] (for a 16-bit compare), the Sequence Manager will launch the scheduled autosequence in accordance with the XCVSEQ[2:0] field. The TMR2IRQ interrupt status bit will become set simultaneous with the autosequence launch.
5. The MCU can wake up at this point (if TMR2MSK=0), or can elect to continue in low-power state until the autosequence completes, at which time SEQIRQ will become set. If SEQMSK=0, then SEQIRQ will interrupt the MCU at the completion of the transceiver operation.

EXTENDED PRECISION EVENT TIMER

For most 802.15.4 transceiver operations referencing the Event Timer, the 24-bit precision and the 62.5KHz update rate (16µs period) are sufficient. However, the required precision for maintaining the 802.15.4 timebase is tighter: 1 microsecond, instead of 16. As long as the 32MHz reference oscillator is running there is no difficulty maintaining that precision. However, when the link layer has to enter low-power state, and the reference oscillator must be switched off, and alternate clock source must be temporarily employed to keep the timebase running. This low power state is known as Deep Sleep Mode, or DSM. The alternate clock source, a 32KHz crystal oscillator, is equally as precise as the reference oscillator, but with a much longer period. At the instant the reference oscillator is turned off and the switchover to the 32KHz is made, the EVENT_TMR becomes frozen at its current state. After a period of DSM, when the precisely-programmed wakeup point is reached, the reference oscillator, which has been previously restarted, is re-engaged by the link layer, and the EVENT_TMR resumes counting where it left off. The precise duration of time for which the EVENT_TMR was frozen, is known to software, based on DSM programming parameters. The EVENT_TMR must be corrected for this "time slept", by adding to it an amount equal to "time slept", in microseconds (hence the requirement for microsecond precision). After the precise correction is made, the EVENT_TMR is now up-to-date, as if the DSM entry/exit had never happened, and all previously scheduled events will occur on time, within 1µs precision.

To realize the required microsecond precision, the actual Event Timer has been expanded to 28 bits, with 24 integer bits and 4 fractional bits, as shown in the diagram below.

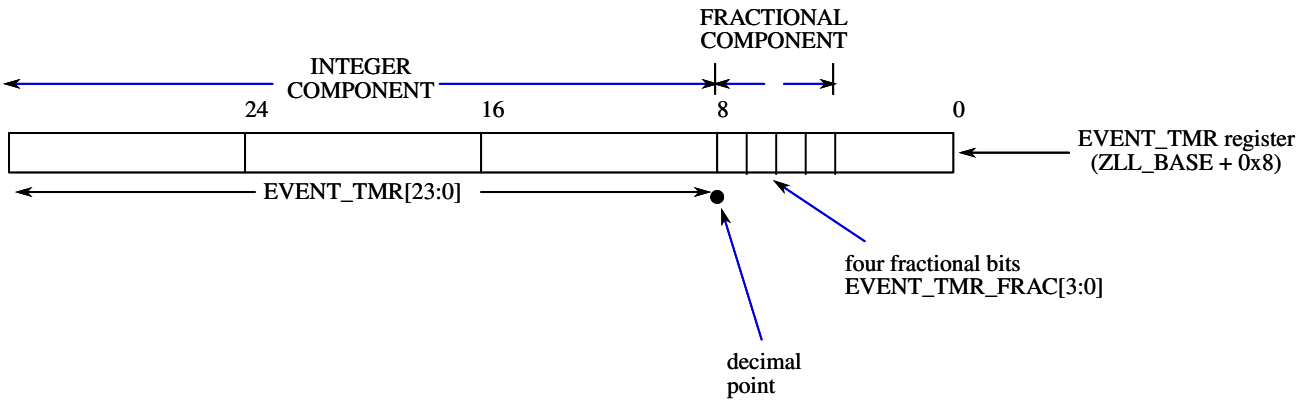


Figure 385. Event Timer Integer and Fractional Components

The 24-bit integer component is the EVENT_TMR value that has been referenced earlier in this section; most 802.15.4 operations require only the integer component, and the fractional component can be ignored. When the fractional bits are not required, writes

to EVENT_TMR register (read-modify-write operations) with EVENT_TMR_LD=1 should leave the fractional bits unchanged. Transceiver operations which reference the EVENT_TMR (i.e., automatic sequence launching, and RX sequence aborting), reference only the integer component, as shown below.

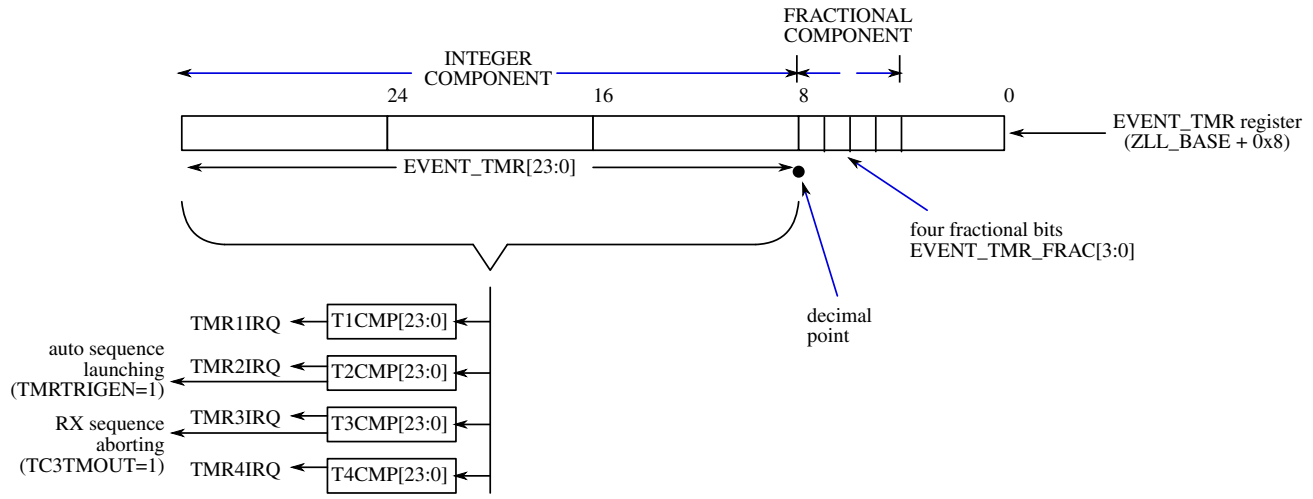


Figure 386. Event Timer Hardware Referencing

However, software EVENT_TMR updates to correct for "time slept" after a DSM cycle must make use of the full precision, including the fractional bits, in order to maintain microsecond accuracy. For example, say a DSM cycle was just completed in which the EVENT_TMR was frozen for exactly 163µs. The 163µs must be "added back" to the EVENT_TMR.

In this case the integer component is:

integer(163µs / 16µs) = **10**

And the fractional component is the remainder x 16:

(163µs / 16µs) - integer(163µs / 16µs) = 3/16 x 16 = **3**

So the value written to the EVENT_TMR with the EVENT_TMR_ADD bit set, accounting for the bit shifts to align to the integer and fractional components of the EVENT_TMR register, is:

(10 $lt;< 8) | (3 $lt;< 4), *or,*

EVENT_TMR = (10 $lt;< 8) | (3 $lt;< 4) | EVENT_TMR_ADD;

This operation adds 163µs to the EVENT_TMR current time, and after the update has occurred, the 802.15.4 timebase has been restored, as if no DSM cycle had occurred, within 1µs accuracy.

55.4.9.1.2.6 Simultaneous CCA/RX

The 802.15.4 Link Layer controller features an autosequence that precedes a packet transmission with a CCA (Clear Channel Assessment), to ensure the channel is idle before transmitting. If the CCA indicates a busy channel, the autosequence ends after the CCA, without transmitting. In slotted mode, 2 CCA measurements take place, with the start points for the measurements 320us apart. Both must indicate an idle channel for transmission to proceed.

During intervals of severe network congestion, high node density, and packets which are heavily fragmented, the situation can arise where Device A is preparing to transmit a packet, or fragment, and performing pre-transmit CCA, and during the CCA, Device B attempts to transmit another packet or fragment to Device A. Previous NXP 802.15.4 transceivers were unable to receive a packet during CCA, so Device B's packet will be missed by Device A. For the Gen3 2.4GHz radio, the capability to receive-during-CCA has been added to 802.15.4. This capability is referred to as "Simultaneous CCA/RX".

To enable this capability, 3 enhancements have been made to the digital receiver and the 802.15.4 sequence manager:

1. CCA Mode 2 Algorithm Separated from 802.15.4 Demodulator

2. New 802.15.4 State Machine Transitions
3. New TSM Fast RX-to-TX Capability

1. Separated CCA Mode 2 Algorithm

One reason for the lack of prior support, is that the CCA Mode 2 algorithm has been tightly integrated with the 15.4 demodulator, so that both functions cannot simultaneously co-exist. The Gen3 digital receiver separates these functions, removing the CCA Mode 2 impediment to Simultaneous CCA/RX. See Chapter *Receiver Digital Module*, Section *IEEE 802.15.4 CCA mode 2*. (CCA Mode 1 does not utilize the demodulator, so such impediment exists.)

The legacy CCA Mode 2 hardware has been retained as a back-up. The control bit USE_DEMOD_CCA2 selects between legacy (1) and new CCA Mode 2 (0). This bit must be set to 0 to utilize Simultaneous CCA/RX. The following table summarizes the programmability associated with CCA Mode 2. The third column references table *IEEE 802.15.4 CCA mode 2 parameters* in Chapter *Receiver Digital Module*, Section *IEEE 802.15.4 CCA mode 2*

REGISTER NAME	ADDRESS SPACE	ASSOCIATED CCA MODE 2 PARAMETER	DESCRIPTION
USE_DEMOD_CCA2	XCVR	n/a	CCA Mode 2 hardware selector. • 0: use new CCA Mode 2 • 1: use demodulator-based (legacy) CCA Mode 2
CCA2_CORR_THRESH[7:0]	802.15.4	zb_cca2_thresh_corr[7:0]	Threshold used to compare correlation products. **Note:** this register is shared between the new and legacy CCA Mode 2 hardware
CCA2_CNT_THRESH[9:0]	XCVR	zb_cca2_thresh_cnt[9:0]	Threshold to compare count of correlation products exceeding CCA2_CORR_THRESH[7:0]
CCA2_SYM_THRESH[9:0]	XCVR	zb_cca2_thresh_sym[9:0]	Threshold used to detect repetition sequence addresses corresponding to correlation products that exceed CCA2_CORR_THRESH[7:0]
CCA2_REF_SEQ[7:0]	XCVR	zb_cca2_ref_seq[7:0]	Selection of the sequence addresses to be used.
CCA2_INTERVAL[1:0]	XCVR	zb_cca2_duration[1:0]	Specifies the CCA2 measurement window duration: • 0x00 - 64 µs • 0x01 - 128 µs • 0x02 - 256 µs • 0x03 - 512 µs

Note: Set CCA2_DURATION=0x01 (128us) for 802.15.4 operation.

2. New 802.15.4 State Machine Transitions

Using Simultaneous CCA/RX, the 802.15.4 Link Layer controller needs to be able to anticipate preamble, and even SFD detection, during the states in which it is taking CCA measurement. Those states are RX_CCA1, for non-slotted, and RX_CCA2, for slotted operation. If preamble and/or SFD is detected in those states, the 802.15.4 sequence manager needs to transition to a packet reception state, and then receive the packet in its entirety as if the called sequence was Sequence R. To address this, these new state transitions have been added to the 802.15.4 Sequence Manager (ZSM) state machine:

- RX_CCA1 -> RX_PRE (preamble detected during 1st CCA but no SFD)
- RX_CCA1 -> RX_PKT (preamble + SFD detected during 1st CCA)
- RX_CCA2 -> RX_PRE (preamble detected during 2nd CCA but no SFD)
- RX_CCA2 -> RX_PKT (preamble + SFD detected during 1st CCA)

The following diagram highlights the new ZSM state transitions, in red:

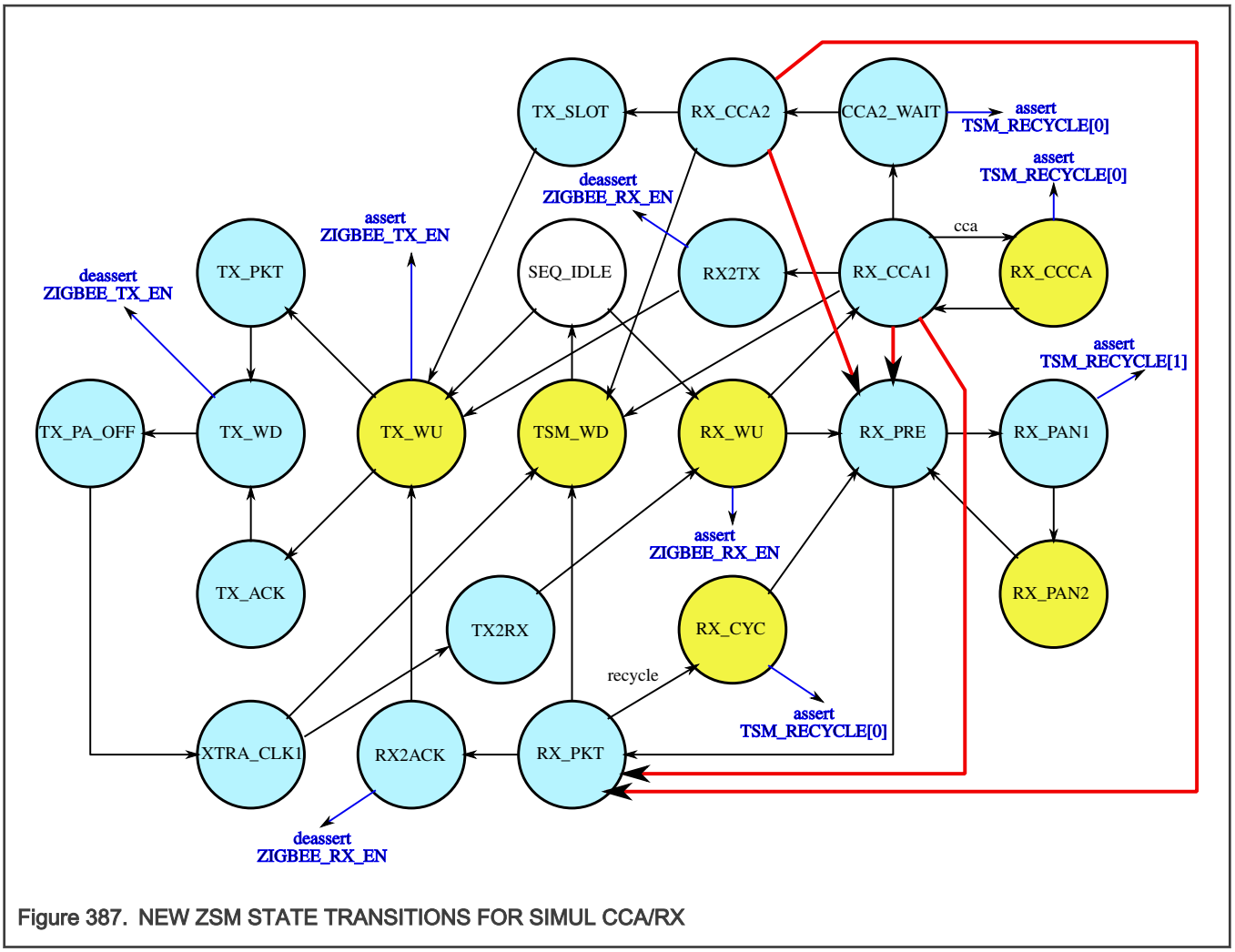


Figure 387. NEW ZSM STATE TRANSITIONS FOR SIMUL CCA/RX

False preambles may occur during the CCA states as well. False preambles occur when the 802.15.4 demodulator detects a sufficient number of consecutive 0 symbols to meet the preamble threshold, but the symbol which follows is neither a '0' nor a '7'. False preambles result in the 802.15.4 demodulator signal sfd_rst asserting either during, or after, the CCA state. If a false preamble is declared *within* the 128µs CCA window, then the fact that there was a false preamble has no bearing on the channel status, the CCA measurement determines channel status, as usual. If the false preamble is declared *after* the CCA window, the

channel is assessed to be busy and the autosequence terminates without transmission, because the scheduled transmission time has been missed.

In defining the Simultaneous CCA/RX operation, 9 "scenarios", or cases, were developed to help ensure correct hardware response to various permutations of preamble and SFD arrival times, preamble falsing, channel status, and RX packet goodness. These scenarios were used to guide the hardware design, for each of the 3 CCA modes, and for mode 3, both boolean combinations. The 9 scenarios are listed in the table below, with the correct hardware response indicated. The hardware response consists of a sequence of state transitions, as well as assertion (or non-assertion) of the various 802.15.4 interrupts. The final combination of asserted interrupts assists software in determining what transpired during the autosequence, and therefore what must be done next.

SCENARIO	#1	#2	#3	#4	#5	#6	#7	#8	#9
CHANNEL IDLE @ 128us	1	0	0	0	0	0	0	0	0
CHANNEL BUSY @ 128us	0	1	0	0	0	0	0	0	0
PREAMBLE ASSERTED DURING CCA SFD_RST OCCURS DURING CCA CHANNEL OTHERWISE IDLE @ 128us	0	0	1	0	0	0	0	0	0
PREAMBLE ASSERTED DURING CCA SFD_RST OCCURS DURING CCA CHANNEL OTHERWISE BUSY @ 128us	0	0	0	1	0	0	0	0	0
PREAMBLE ASSERTED @ 128us SFD NOT ASSERTED @ 128us SFD_RST ASSERTS LATER (FALSE PRE)	0	0	0	0	1	0	0	0	0
PREAMBLE ASSERTED @ 128us SFD NOT ASSERTED @ 128us SFD ASSERTS LATER (BAD PKT)	0	0	0	0	0	1	0	0	0
PREAMBLE ASSERTED @ 128us SFD ASSERTED @ 128us (BAD PKT)	0	0	0	0	0	0	1	0	0
PREAMBLE ASSERTED @ 128us SFD NOT ASSERTED @ 128us SFD ASSERTS LATER (GOOD PKT)	0	0	0	0	0	0	0	1	0
PREAMBLE ASSERTED @ 128us SFD ASSERTED @ 128us (GOOD PKT)	0	0	0	0	0	0	0	0	1
OUTCOME:									
AUTOSEQUENCE ENDS AFTER:	TX	CCA	TX	CCA	SFD_RST	RX	RX	RX	RX
MODE SWITCH TO SFD SEARCH OCCURS AT:	--	--	--	--	128us	128us	--	128us	--
MODE SWITCH TO PKT RECEPTION OCCURS AT:	--	--	--	--	--	--	SFD DET	--	SFD DET
CCAIRQ?	yes (CCA=0)	yes (CCA=1)	yes (CCA=0)	yes (CCA=1)	yes (CCA=1)	yes (CCA=1)	yes (CCA=1)	yes (CCA=1)	yes (CCA=1)
RXIRQ?	no	no	no	no	no	no	no	yes	yes
TXIRQ?	yes	no	yes	no	no	no	no	no	no

Figure 388. SIMUL CCA/RX SCENARIOS

A single register bit enables Simultaneous CCA/RX mode: SIMUL_CCA_RX. This bit resides in the CCA_CTRL register in 802.15.4 address space. By default SIMUL_CCA_RX=0, meaning that the new ZSM transitions are inhibited. Set this bit to 1, to enable the new transitions. Simultaneous CCA/RX works with any CCA Mode (1, 2, or 3), and works for both non-slotted (single CCA before TX) and slotted (2 CCA before TX) modes.

Note that to use the Simultaneous CCA/RX feature, software (not just hardware) needs to anticipate packet reception during a Sequence T (not just a Sequeunce TR). This may not exist today, since prior to Gen3, the 802.15.4 controller did not have this capability.

3. New TSM Fast RX-to-TX Capability

The 802.15.4 Sequence Manager (ZSM), makes use of the radio's all-protocol Transceiver Sequence Manager (TSM) to execute TX and RX warmup and warmdown sequences; TSM handles low-level commanding, with precise, programmable timing, of the analog and digital blocks in the transmit and receive chains, absolving the higher-level sequence managers from this responsibility. TSM sequences have, in the past, been discrete TX or RX operations, with a required software command to initiate the sequence, and a return-to-idle after the sequence is complete, before a new sequence-start command can be issued. For 802.15.4 Simultaneous CCA/RX mode, it is highly desirable to be able to jump directly from the end of a CCA measurement, which is essentially an RX operation, to a packet transmit operation, without returning the TSM to IDLE in between, if CCA deems the channel to be free. A new TSM transition called Fast RX2TX accomplishes this. The 802.15.4 ZSM will signal to TSM when to perform such a transition, and TSM will respond by jumping from the RX ON phase, to a programmable point within the TX warmup. See Chapter *Transceiver Sequence Manager*, Section *FAST TSM Warmup and FAST RX2TX Transition* for more details on this TSM feature, and a description of the ZSM -> TSM signaling required to execute such a transition.

A single register bit enables TSM Fast RX-to-TX: FAST_RX2TX_EN. This bit resides in the TSM_FAST_CTRL1 register in XCVR address space. By default FAST_RX2TX_EN=0, meaning that this TSM capability is inhibited, and RX-to-TX transitions remain discrete RX and TX sequences (legacy).

Once enabled with FAST_RX2TX_EN=1, the ZSM will request a TSM Fast RX-to-TX warmup, by asserting fast_rx2tx_wu, when all of the following conditions are met:

- 802.15.4 autosequence is T or TR
- Pre-TX CCA is enabled with CCABFRTX=1
- ZSM state is RX_CCA1 (non-slotted or slotted) or RX_CCA2 (slotted mode only)
- 128µs measurement timeout expires
- Neither preamble-detect nor SFD-detect are *currently* asserted
- Channel is deemed idle (CCA=0)

The TSM will respond to fast_rx2tx_wu by jumping from the "ON" phase of the RX operation to the "WARMUP" phase of the TX operation at the programmed point. Simultaneously, ZSM must jump to its TX_WU state to wait for the TX warmup to complete. This requires 2 new state transitions, to enable the Simultaneous CCA/RX feature, shown in red in the diagram below:

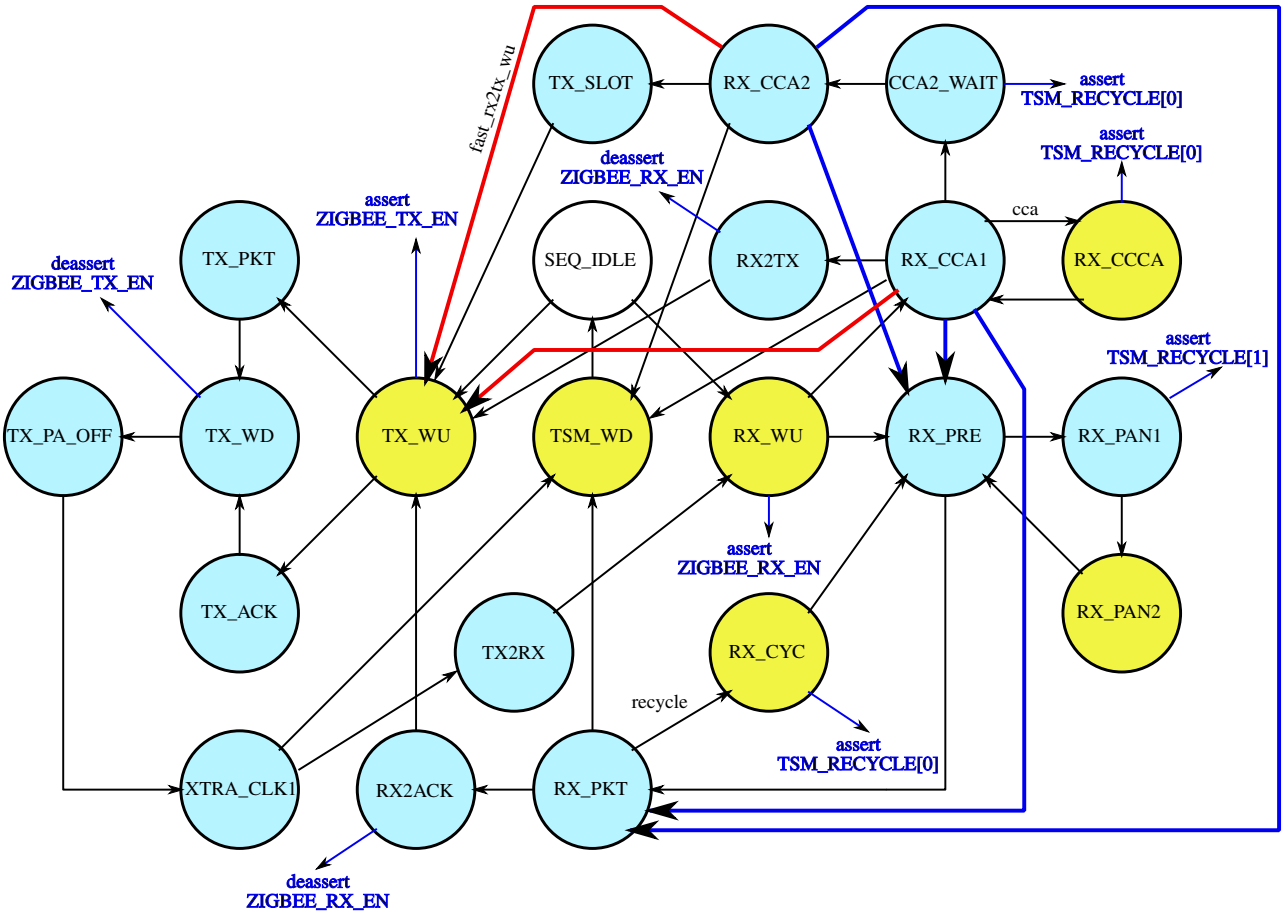


Figure 389. ZSM STATE TRANSITIONS FOR FAST RX2TX

Once the TX warmup completes, the 802.15.4 controller will transmit the packet as per the Sequence T command.

55.4.9.1.2.7 Manual DSM

When the 802.15.4 Link Layer anticipates long periods of inactivity, the Link Layer controller can be put into a Deep Sleep Mode (DSM), where all of its clocks are turned off. In addition, the 802.15.4 Link Layer can be configured to optionally turn off the RF Oscillator, if it is not otherwise needed by the SoC. Deep Sleep Mode results in dramatic power savings. The procedure described below, is called Manual DSM.

To make Manual DSM possible, the transceiver incorporates a low-frequency, high-precision Deep Sleep Timer, TIMER. This timer is clocked by a crystal-referenced 32.768KHz oscillator. The timer is 24-bit wide, yielding a 8.5 minute rollover (the maximum length of a deep sleep period).

The hardware to implement DSM within the SoC, is partitioned across several modules, and several power domains. This is to allow the maximum achievable power savings, by retaining full voltage for only those modules which need to be active in DSM, and placing the remaining modules in “state retention” mode. The diagram below depicts the partitioning of DSM hardware across the various modules, including the 802.15.4 Link Layer Controller.

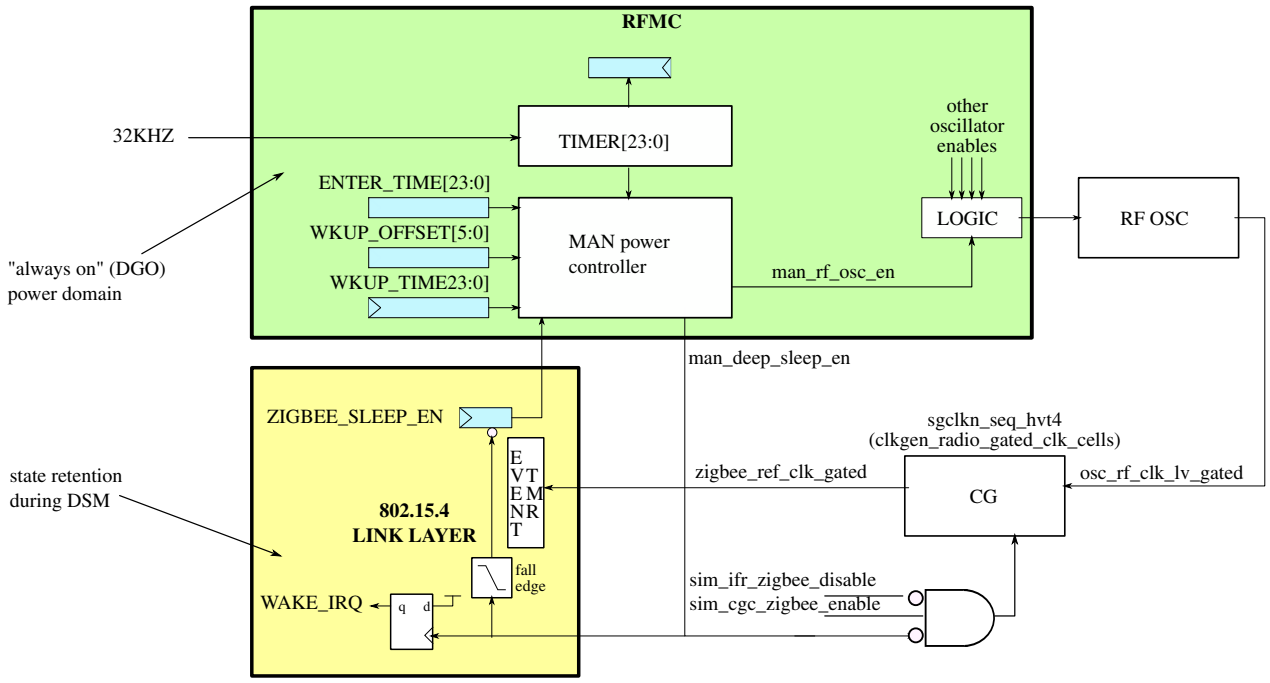


Figure 390. MANUAL DSM BLOCK DIAGRAM

The concept behind Manual DSM, as it is implemented for 802.15.4, is to allow the 802.15.4 timebase to be maintained for a period of time, while the RF Oscillator is turned off and much of the SoC is placed in state-retention. During DSM, the timebase management is temporarily transferred from the 802.15.4 EVENT_TMR to the low-power TIMER, and after DSM exit, timebase management is transferred back to the EVENT_TMR, under software control. During DSM, the EVENT_TMR is frozen. The total time spent in DSM is known to software, so that upon exiting DSM, the EVENT_TMR can be updated to correct for the precise amount of time it was frozen.

DSM entry and exit, and RF Oscillator re-start during Manual DSM, are governed by 5 registers, as described in the table below.

Field	R/W	Description
ENTER_TIME[23:0]	rw	If ZIG_SLEEP_REQUEST=1, enter DSM & freeze EVENT_TMR when ENTER_TIME matches TIMER. **NOTE** This register resides in WOR Address Space
WKUP_OFFSET[5:0]	rw	Determines the power and RF oscillator re-start times relative to WKUP_TIME **NOTE** This register resides in RFMC Address Space

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
WKUP_TIME[23:0]	rw	Exit Deep Sleep Mode, and resume EVENT_TMR, when WKUP_TIME matches TIMER **NOTE** This register resides in WOR Address Space
ZIG_SLEEP_REQUEST	w	Enable a match on ENTER_TIME[23:0] to TIMER[23:0], to enter Deep Sleep Mode, by writing a 1 to this bit. Cancel a pending sleep request by writing a 0 to it, if already set, at least 2 32.768KHz clock cycles prior to ENTER_TIME[23:0]
TIMER[23:0]	r	Current State of the 32KHz Sleep Timer **NOTE** This register resides in RFMC Address Space

Procedure

The procedure to schedule a Manual DSM cycle (entry/exit) as well as to program the desired RF Oscillator re-start time, and the calculation to restore the 802.15.4 EVENT_TMR after a DSM cycle, is described below.

1. Software determines the future time at which it would like to enter DSM, by reading the TIMER, computing the number of 32KHz clock cycles remaining until the desired DSM start-time, and writing this value into ENTER_TIME register. The value programmed into ENTER_TIME should be no fewer than 4 clocks greater (in the future) than the current time as read from TIMER.
2. Software determines the future time at which it would like to exit DSM, by computing the number of 32KHz clock cycles remaining until the desired DSM exit-time, and writing this value into WKUP_TIME register.
3. Software writes a 1 to DSM_CTRL[ZIG_SLEEP_REQUEST]. This bit resides in 802.15.4 address space. This bit is write-only and always reads back 0. Writing this bit to 1 enables a DSM cycle to commence using the values programmed into ENTER_TIME and WKUP_TIME.
4. MCU can go into a low power state.
5. When TIMER = ENTER_TIME, the EVENT_TMR will be clock-gated so that it will remain frozen at its current count. The RF Oscillator will be turned off (unless overridden due to a non-802.15.4-related SoC requirement)
6. All hardware in the low-voltage domain will be placed into state-retention, or other low-power state.
7. When TIMER = WKUP_TIME, clocking of the 802.15.4 Link Layer Controller will resume and the EVENT_TMR will be ungated (allowed to resume counting).
8. Software computes the time that the SoC was in DSM (and hence the time the EVENT_TMR was frozen), in microseconds, with the equation:

   (WKUP_TIME – ENTER_TIME) / 32768 * 1000000

9. Software increments the 802.15.4 EVENT_TMR by this amount, by writing this value to the EVENT_TMR register with EVENT_TMR_ADD=1. **Note:** The 802.15.4 EVENT_TMR has an integer component (24 bits) and a fractional component (4 bits). The fractional component provides the microsecond precision to the Event Timer, so both components need to be accounted for when compensating for the DSM "time slept".
10. On the next 1MHz clock edge, the EVENT_TMR has been restored to where it would have been, had no DSM occurred, with 1.0-microsecond accuracy.

Note that the 1.0-microsecond accuracy quoted above, assumes an ideal 32.768KHz source. Any ppm error in the 32.768KHz source will degrade the microsecond accuracy proportionally to the magnitude of the error.

Once ZIG_SLEEP_REQUEST has been set to 1 to request Manual DSM, the request can be cancelled by writing a 0 to the bit. The cancellation must be performed at least 2 32.768KHz before the ENTER_TIME[23:0] match to TIMER[23:0] to avoid a race condition to enter DSM.

The following timing diagram depicts an 802.15.4-driven DSM entry/exit cycle.

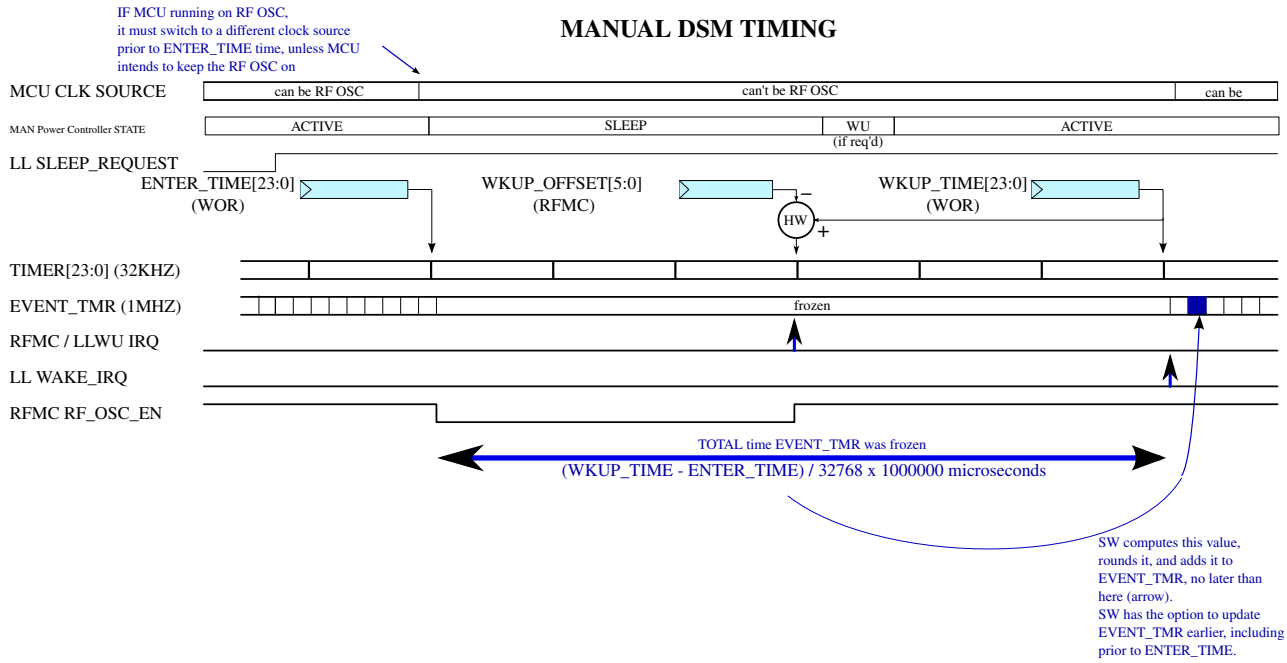


Figure 391. 802.15.4 DEEP SLEEP TIMING

DSM Early Exit

Circumstances may arise where the Manual DSM cycle may occasionally need to be cut short, due to circumstances beyond the control of the radio. When this happens, the SoC, starting with the MCU, is awakened out of its low power state, long before the RFMC TIMER reaches WKUP_TIME. Therefore, software handling the event which triggered the wakeup, will determine if the radio needs to be awakened as well. If this is the case, software will advance the WKUP_TIME setting to a point in the near future, but no fewer than 4 32KHz clock cycles beyond the current time ascertained by reading TIMER. So, to execute a Manual DSM Early Exit, software shall read TIMER, add +4 to the value, and write the sum to the WOR WKUP_TIME register. Software shall then wait for the 802.15.4 WAKE_IRQ interrupt, which will assert once the full radio wakeup from the low power state has completed. Software should then take whatever steps are necessary to configure the radio to respond to the conditions which triggered the early exit. Servicing of a DSM early exit interrupt is beyond the scope of this document.

55.4.9.1.2.8 Wake-On-Radio

The 802.15.4 Link Layer can take advantage of the automated sequence- and DSM-scheduling provided by Wake-On-Radio. Wake-On-Radio is one of two DSM control schemes that can be selected for 802.15.4, as an alternative to Manual DSM. See Chapter *Wake-On-Radio*.

Several changes to the 802.15.4 Link Layer Controller have been made to support Wake-On-Radio.

Interrupt WAKE_IRQ has been made dual-purpose. If Wake-On-Radio is disabled, i.e., WOR_CTRL[WOR_EN]=0, or Wake-On-Radio not assigned to 802.15.4, WAKE_IRQ performs its legacy function for Manual DSM. (See Section Manual DSM). If Wake-On-Radio is enabled, i.e., WOR_EN=1, and assigned to 802.15.4, this interrupt becomes WOR_IRQ, and has new functionality specific to Wake-On-Radio. See Chapter *Wake-On-Radio*, Section *Interrupt Management*, for more details on the meaning, and handling of WOR_IRQ.

In addition to WOR_IRQ, numerous interface changes have been made to support Wake-On-Radio. The changes allow, for example, WOR to manage 802.15.4's RF interrupts, inhibit timestamp updates, etc. The added port signals are listed in the table below, along with a description or each:

WAKE-ON-RADIO INTERFACE SIGNAL	DIRECTION	DESCRIPTION
wor_event_tmr_add[27:0]	WOR -> 802.15.4	The Event Timer "Addback" value computed by WOR to compensate for the amount of time Event Timer was frozen during DSM
wor_addback	WOR -> 802.15.4	A 1µs-long pulse commanding the 802.15.4 Event Timer to auto-increment by the the amount appearing on wor_event_tmr_add[27:0]
wor_force_wake_irq	WOR -> 802.15.4	Force WOR_IRQ to assert, due to an occurrence of one of the 3 WOR_IRQ interrupt triggers
wor_block_wake_irq	WOR -> 802.15.4	Inhibit the legacy WAKE_IRQ assertion when WOR_EN=1 (this interrupt becomes WOR_IRQ)
tx_start_ext	WOR -> 802.15.4	A 1µs-long pulse to the 802.15.4 command decoder to initiate a WOR-driven TX sequence
rx_start_ext	WOR -> 802.15.4	A 1µs-long pulse to the 802.15.4 command decoder to initiate a WOR-driven RX sequence
seq_stop_ext	WOR -> 802.15.4	A 1µs-long pulse to the 802.15.4 command decoder to stop (abort) a WOR-driven RX sequence
wor_allow_irq	WOR -> 802.15.4	Inhibits RF interrupts, so that interrupt assertions can be managed based on WOR descriptor settings. Controlled dynamically by the WOR state machine.
timestamp_inhibit	WOR -> 802.15.4	In Wake-On-Radio, the legacy 802.15.4 TIMESTAMP register becomes WOR TIMESTAMP0 (timestamp for SLOT0).

Table continues on the next page...

Table continued from the previous page...

WAKE-ON-RADIO INTERFACE SIGNAL	DIRECTION	DESCRIPTION
		This signal blocks this register from updating on SLOTS 1-3
wake_irq_trig	WOR <- 802.15.4	WOR uses this signal to cancel ZIG_SLEEP_REQUEST and transition from IN_DSM -> COMP_ADDBACK state
event_tmr[27:0]	WOR <- 802.15.4	802.15.4 Event Timer, used by WOR for timestamp capture
event_tmr_plus_1[27:0]	WOR <- 802.15.4	802.15.4 Event Timer + 1, used by WOR to schedule RF operation starts and stops
tx_irq_trig	WOR <- 802.15.4	TXIRQ trigger, indicates the completion of a packet transmission, used by WOR to transition out of WAKE_SEQ_END state
rx_irq_trig	WOR <- 802.15.4	RXIRQ trigger, indicates the completion of a good packet reception, used by WOR to transition out of WAKE_SEQ_END state
rx_status_for_wor[2:0]	WOR <- 802.15.4	Internal RX status signals used by WOR to formulate TIMESTAMPx_STS[2:0]
timestamp_clk_en	WOR <- CGM <- 802.15.4	Timestamp clock gate enable, the resultant clock is used by WOR to update the TIMESTAMP1,2, and 3 registers

55.4.9.1.2.9 CCA/ED/LQI

55.4.9.1.2.9.1 Introduction

CCA/ED/LQI block resides inside 802.15.4 demodulator top. The purpose of this block is to perform the following operations: 1) Clear Channel Assessment: Modes 1, 2 and 3, 2) Energy Detection and 3) Link Quality Indicator.

This block takes RSSI value, as an input, for computations related to CCA modes 1 and 3, Energy Detection and LQI. For CCA2, this block takes output of 802.15.4 correlators for detecting an 802.15.4 signal in the channel of interest.

At the output, it provides a signal called (cca) to indicate if the channel of interest is busy (cca=1) or idle (cca=0), the final computed energy value (cca1_ed_fnl) and Link Quality Indicator (LQI) value. The cca signal is further used to generate cca interrupt, according to the set interrupt mask. The cca1_ed_fnl is a signed 8 bit value that can be read back as a register. LQI is an unsigned 8 bit value that is written at the end of each valid packet that is received and is available to be read in the packet buffer.

55.4.9.1.2.9.1.1 Block diagram

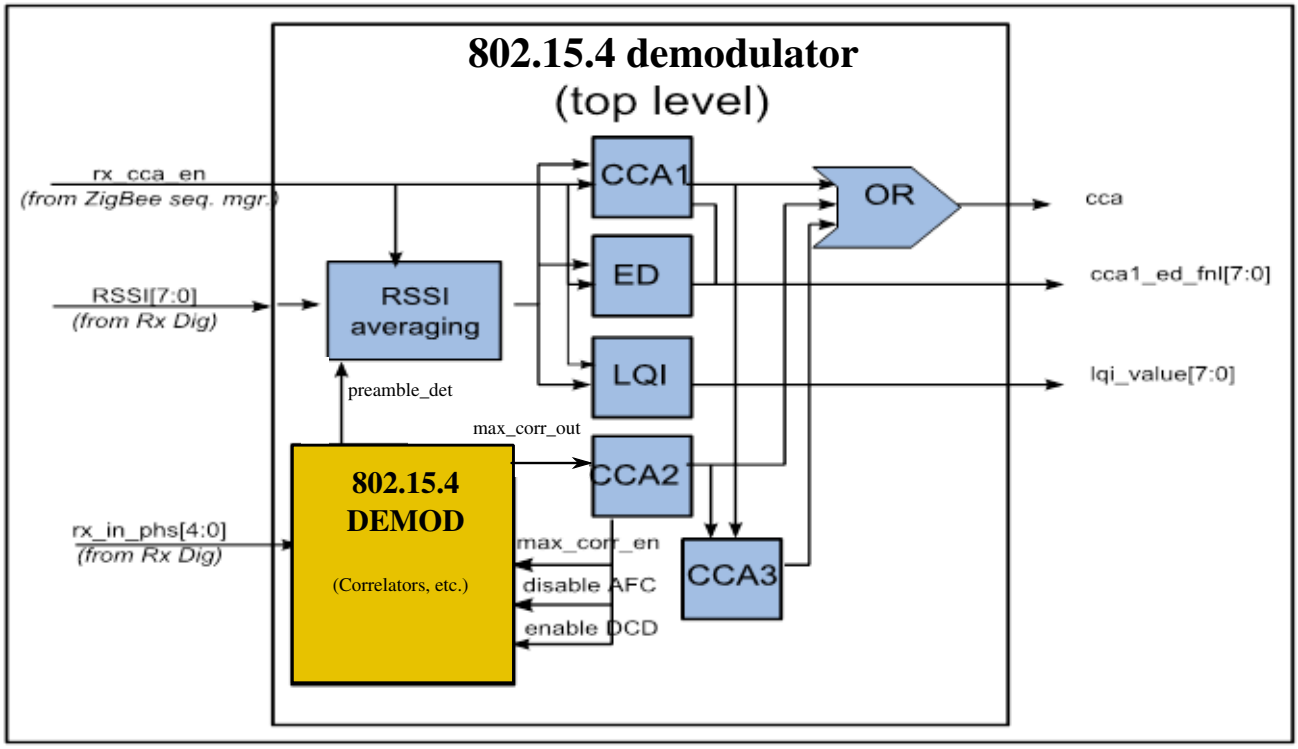


Figure 392. Block diagram

55.4.9.1.2.9.2 Memory Map and register definition

Field	R/W	Description
LQI_VALUE[7:0]	r	8-bit LQI value of each Rx packet
CCA1_ED_FNL[7:0]	r	8-bit signed value to indicate last energy detected
CCA1_THRESH[7:0]	rw	8-bit signed value to be compared against averaged signed RSSI value from Rx DIG block, to determine channel busy or idle
LQI_OFFSET_COMP[7:0]	rw	8-bit value to offset average of signed RSSI value coming from Rx DIG, in order to compute 8-bit LQI value.
CCA3_AND_NOT_OR	rw	This register bit controls how CCA1 and CCA2 are combined to assert CCA3 mode: 1 : CCA1 AND CCA2 0: CCA1 OR CCA2
CCA2_MIN_NUM_CORR_THRESH[2:0]	rw	3-bit value to compare against number of correlation peaks within CCA2 time.
CCA2_NUM_CORR_PEAKS[3:0]	r	4-bit value indicating number of correlation peaks detected in last CCA2 run

Table continues on the next page...

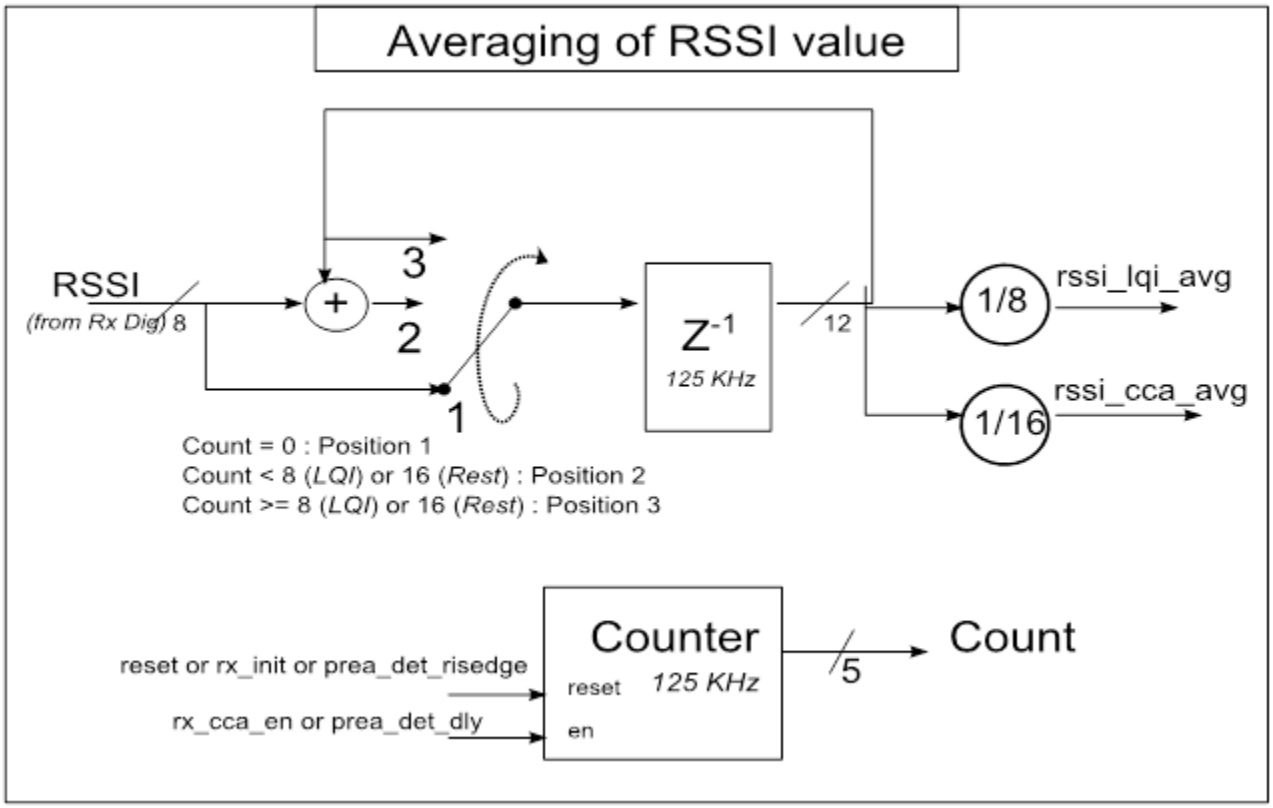
Table continued from the previous page...

Field	R/W	Description
CCA2_CORR_THRESH[7:0]	rw	8-bit value to be compared against correlation peak value during CCA2 operation.
LQI_START_AT_SFD	rw	Defines starting point for LQI computation during packet reception 1: Begin computing LQI at SFD detect (new for KW41) 0: Begin computing LQI at preamble detect (legacy)

55.4.9.1.2.9.3 Functional description

The following sections describe functional details of the module.

55.4.9.1.2.9.3.1 RSSI Averaging



The above picture shows how RSSI is averaged. 8-bit signed RSSI value from Rx Dig block is averaged for 128 us for CCA1/2/3/ED sequences. For LQI, it is averaged for 64 us. In both cases, the RSSI value is sampled at 125 KHz rate.

55.4.9.1.2.9.3.2 CCA1

When the 802.15.4 link layer is set up to perform CCA1 operation, rx_cca_en signal is asserted at the end of Rx warm up by the 802.15.4 Sequence Manager (ZSM). After rx_cca_en is asserted high, at the end of 128 us, i.e. count = 16, rssi_cca_avg is compared against 8-bit signed register cca1_thresh to assess CCA as follows:

If rssi_cca_avg >= cca1_thresh, CCA1 = 1, i.e. channel is busy.

Else CCA1 = 0, i.e. channel is idle.

55.4.9.1.2.9.3.3 CCA2

When the 802.15.4 link layer is set up to perform CCA2 operation, rx_cca_en signal is asserted at the end of Rx warm up by the 802.15.4 Sequence Manager (ZSM). During CCA2 operation, differential chip detection is enabled and automatic frequency correction is disabled, in order to receive correlation peaks related to all available 16 802.1.5.4 symbols. The signal max_corr_en is held high, in order to refresh maximum correlator value after each symbol period. 128 us after rx_cca_en is asserted high, i.e. count = 16, the number of maximum correlation peaks are counted which exceeded the value cca2_corr_thresh register. If the number of correlation peaks exceeds minimum desired number of correlation peaks (programmable register: cca2_num_corr_peaks), cca2 is asserted high to indicate the channel is busy and low otherwise, to indicate channel is idle.

55.4.9.1.2.9.3.4 CCA3

CCA mode 3 is a combination of CCA1 and CCA2. There are 2 combinations in which CCA mode 3 operates, based on the register setting of CCA3_AND_NOT_OR: 1) CCA1 OR CCA2 2) CCA1 AND CCA2. When the 802.15.4 link layer is set to perform CCA3, both CCA1 and CCA2 operations, as mentioned above and decides if the channel is busy or idle, according to the desired combination (i.e. AND/OR).

55.4.9.1.2.9.3.5 Energy Detection

During Energy Detection operation, the demodulator is expected to average RSSI value (from Rx_DIG) for 128 us, as mentioned above in the block diagram. At the end of 128 us, the register CCA1_ED_FNL is updated with this average energy value. Energy detection is similar to CCA1. Unlike CCA1, busy/idle signal is not generated.

55.4.9.1.2.9.3.6 Link Quality Indicator

LQI maps RSSI value (from Rx_DIG) to values between 0x00 and 0xFF, as shown in the following block diagram. Upon preamble detect, RSSI value is averaged for 64 us and mapped to LQI value. This latched LQI value is available in the register LQI_VALUE. Packet processor also appends this LQI value at the end of packet buffer.

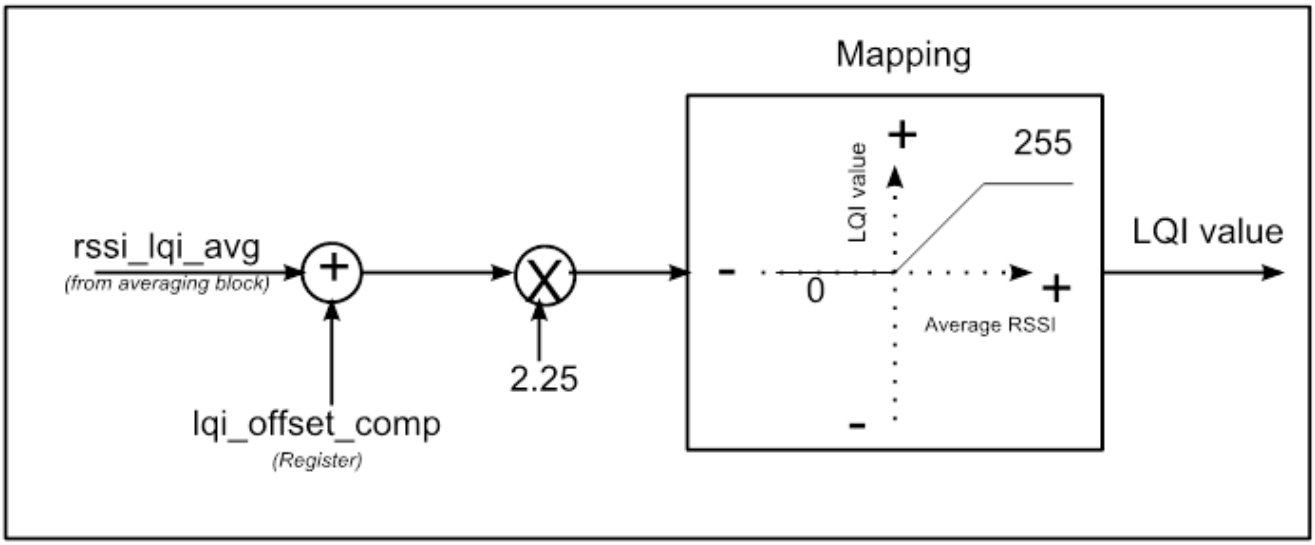


Figure 393. LQI Computation

The following picture shows an example of mapped LQI, when the register LQI_OFFSET_COMP is programmed to decimal value of 102.

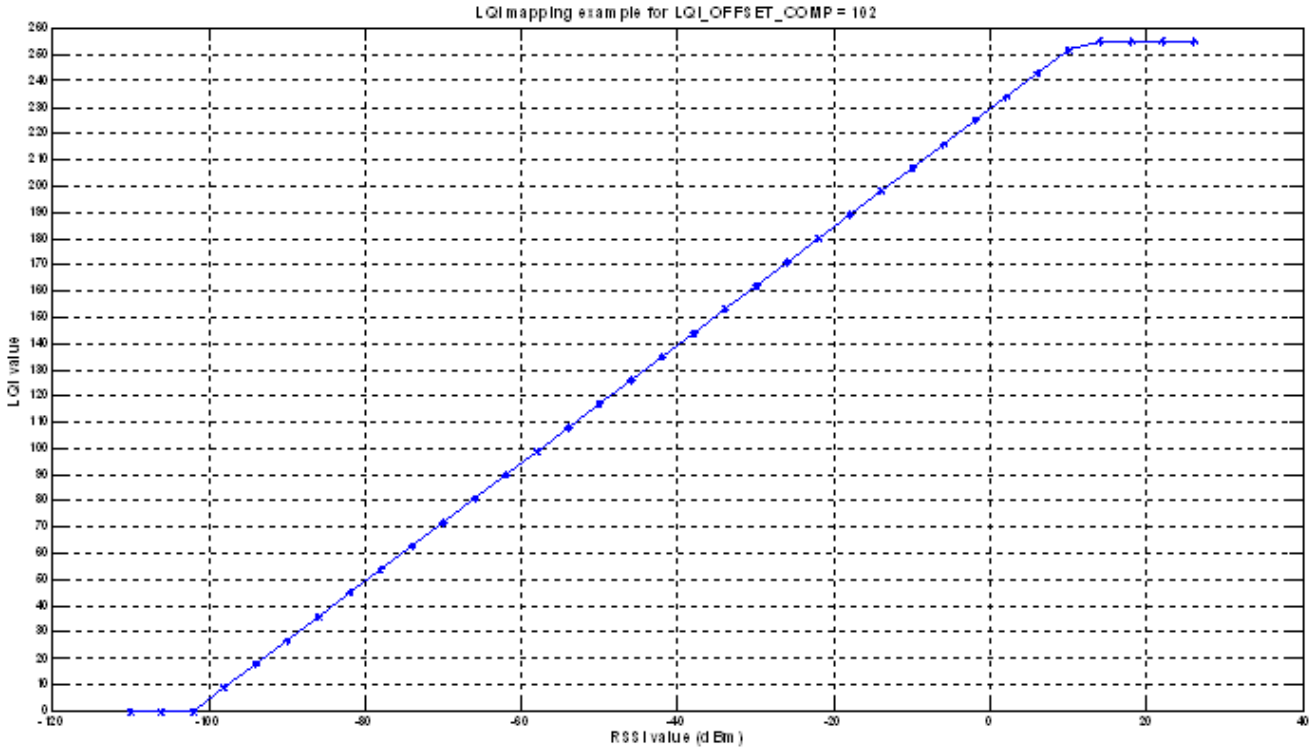


Figure 394. Plot of LQI vs. RSSI

55.4.9.1.2.9.3.7 Clocks

The CCA/ED/LQI module is a fully synchronous module. It has a single clock input **clk32m**. This clock input is driven by the RF Oscillator. The CCA/ED/LQI module has no asynchronous interfaces or clock domain crossings.

55.4.9.1.2.9.3.8 Reset

The CCA/ED/LQI module has a single, active-low, asynchronous reset input: **resetb** . At integration, this reset should be tied to the master reset for the entire transceiver. There are no special reset requirements.

55.4.9.1.2.9.3.9 Interrupts

The CCA/ED/LQI module does not generate any interrupts.

55.4.9.1.2.10 Fast Antenna Diversity (FAD)

FAD ALGORITHM

The Fast Antenna Diversity (FAD) mode, when enabled, will allow the choice between two antennae to be selected during the preamble search. By continually monitoring the received signal the FAD block will select the first antenna on which the received signal has a correlation factor above a predefined, programmable, threshold. This threshold is register programmable. The FAD accomplishes the antenna selection by sequentially switching between the two antenna testing for the presence of a suitably strong s0 symbol, the first antenna to reach this condition is then selected for the reception of the packet.

The first antenna is monitored for a period equal to 1 symbol, T_s = 16µs, then the antenna monitoring is switched to the second antenna, T_a = 8µs. T_a is required to allow for the external pin diodes to turn on/off to select the antenna. $T_s + T_a$ = 24µs which allows enough time to be able to test both antenna within the first 4 preamble symbols, $T_{fad} = 3 \times T_a + 2 \times T_s$ = 56µs, thus T_{fad} $lt; 4 x T_s $lt; 64µs.

The FAD will continually switch between the two antenna until one is found which a sufficiently strong s0 detection. By virtue of the fact that the FAD operation covers less than four s0 symbols before the antenna is selected still allows the symbol demodulator to then detect at least four s0 symbols before declaring "Preamble Detect".

The symbol correlator runs during the T_a period to calculate the max correlation factor for the antenna Cx. However since T_c is not guaranteed to coincide with 1 unique symbol T_s the correlation must be able to work across symbols. To accomplish this the correlator will rotate the s0 PN chip code through its 32 combinations in order to find the s0 symbol split across two adjacent symbols.

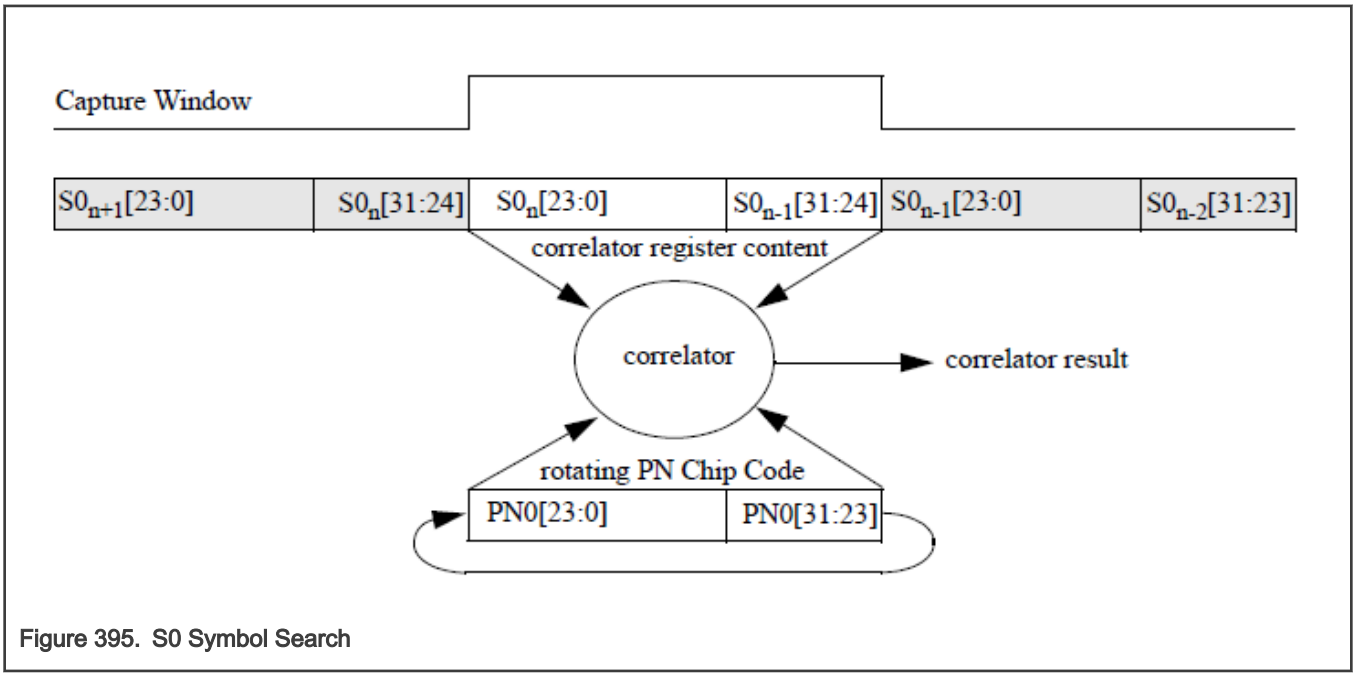


Figure 395. S0 Symbol Search

When the S0 PN Chip code becomes aligned to the Symbol data within the correlator register a correlation peak is produced that is used to determine the Antenna has a sufficiently strong signal to be selected.

It is not possible to choose between Antenna with the highest correlation factor, since indeed 1 antenna could receive no signal and thus is not distinguishable from a clear channel. This being the case, the first antenna that meets the requirements, correlation factor above the defined threshold, is selected.

When a correlator peak is detected, a second check is done ½ symbol later. A new peak should be detected with the rotation equal to ½ symbols ±2 chips. If it is the case the channel is validated and the classical *CORR_NVAL* consecutive S0 is used to certify a correct preamble, where *CORR_NVAL* is the programmable number of consecutive S0 symbols required to declare preamble detected.

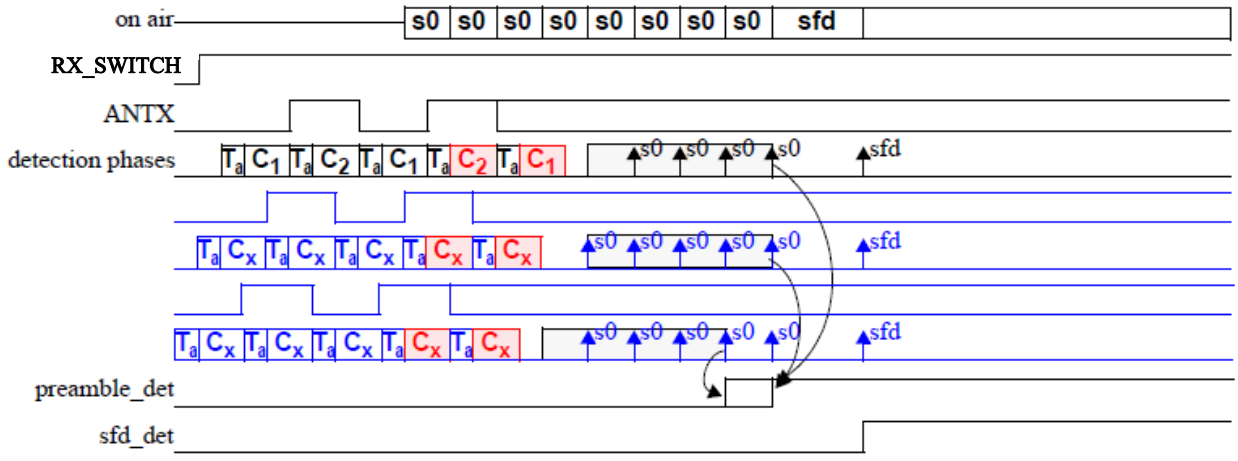


Figure 396. FAD Switching

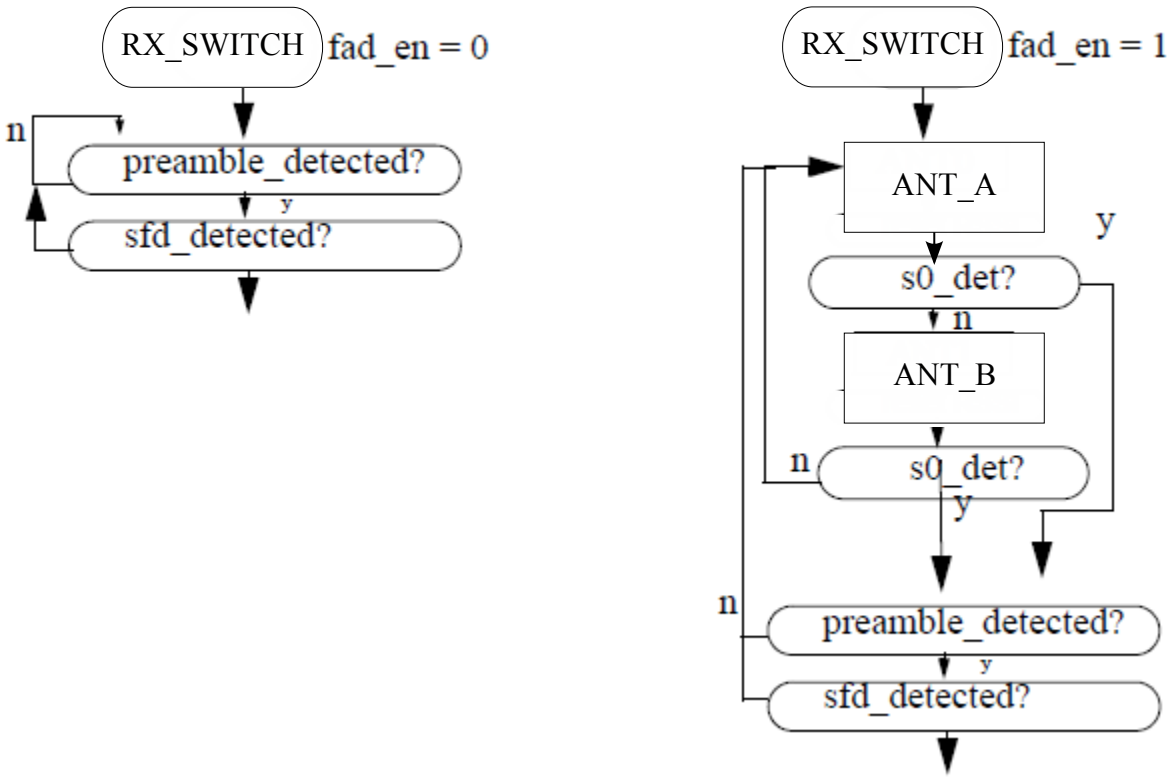


Figure 397. FAD Decision Flow Diagram

The choice of the antenna is also used for the transmit. Since the antx_out value (the last antenna selected by the FAD algorithm) is kept after the end of the receive, the following TX will use the last best antenna found during the last receive. This is true also for any ensuing CCA or ED RX frame; the antenna selected will be the one selected by the FAD algorithm during the last successful preamble search. So a CCA will search for an empty channel on the best antenna chosen at the last RX. When FAD is enabled (FAD_CTRL[FAD_EN]=1), the state of the antenna can be ascertained at any time by reading the FAD_CTRL[ANTX] bit: 0 = ANT_A, 1 = ANT_B. When FAD_CTRL[FAD_EN]=0, antenna selection is placed under direct software control, and the value written to the FAD_CTRL[ANTX] bit becomes the selected antenna: 0 = ANT_A, 1 = ANT_B. The FAD_CTRL register resides in XCVR space.

FAD PIN INTERFACE

This content has been moved to: Chapter *Transceiver Sequence Manager*, Section *Fast Antenna Diversity (FAD)*

55.4.9.1.2.11 Interrupts

55.4.9.1.2.11.1 Introduction

This section describes the 802.15.4 Interrupt Architecture.

Revision	Date	Author	Changes
0.1	04-28-2014	D. Brown	Copy-over from Coconino
0.2	10-24-2015	D. Brown	Updated for KW41

The 802.15.4 Link Layer has a single, active high, interrupt line (ipi_int_zigbee) to the MCU. Internally, the Link Layer has 13 interrupt sources, as shown in the following table:

802.15.4 Interrupt Sources
SEQIRQ
TXIRQ
RXIRQ
CCAIRQ
RXWTRMRKIRQ
FILTERFAIL_IRQ
PLL_UNLOCK_IRQ
TMR1IRQ
TMR2IRQ
TMR3IRQ
TMR4IRQ
WAKE_IRQ
TSM_IRQ

Any or all of the interrupt sources, can be enabled to cause an assertion on ipi_int_zigbee. Each interrupt source has its own interrupt status bit in 802.15.4 register space. Each interrupt source is individually maskable (each has its own MASK bit), so that each interrupt source can be individual enabled to trigger an assertion on ipi_int_zigbee. There is also a global interrupt mask, TRCV_MSK, which can enable/disable all ipi_int_zigbee assertions by programming a single masking bit. All interrupt status bits (except TSM_IRQ) use a write-1-to-clear protocol. Interrupt status bits are not affected by reads. The ipi_int_zigbee pin is a level-sensitive interrupt indicator to the MCU; on an interrupt, ipi_int_zigbee will remain asserted until all active interrupt sources are cleared or masked.

55.4.9.1.2.11.1.1 Features

- 13 interrupt sources (interrupt status bits)
- each interrupt source individually maskable
- each interrupt source is individually write-1-to-clear.
- interrupts remain asserted until cleared.

55.4.9.1.2.11.2 Memory Map and register definition

Numerous register bits are provided to control 802.15.4 interrupt behaviour. The registers referred to below (IRQSTS1, IRQSTS2, IRQSTS3, PHY_CTRL2, PHY_CTRL3, and PHY_CTRL4) all reside in the 802.15.4 Address space.

Field	R/W	Description	Default
SEQIRQ	r/w1tc	1: A Sequencer Interrupt has occurred 0: A Sequencer Interrupt has not occurred	-
TXIRQ	r/w1tc	1: A TX Interrupt has occurred 0: A TX Interrupt has not occurred	-
RXIRQ	r/w1tc	1: A RX Interrupt has occurred 0: A RX Interrupt has not occurred	-
CCAIRQ	r/w1tc	1: A CCA Interrupt has occurred 0: A CCA Interrupt has not occurred	-
RXWTRMRKIRQ	r/w1tc	1: A RX Watermark Interrupt has occurred 0: A RX Watermark Interrupt has not occurred	-
FILTERFAIL_IRQ	r/w1tc	1: A Filter Fail Interrupt has occurred 0: A Filter Fail Interrupt has not occurred	-
PLL_UNLOCK_IRQ	r/w1tc	1: A PLL Unlock Interrupt has occurred 0: A PLL Unlock Interrupt has not occurred	-
TMR1IRQ	r/w1tc	1: A TMR1 Interrupt has occurred 0: A TMR1 Interrupt has not occurred	-
TMR2IRQ	r/w1tc	1: A TMR2 Interrupt has occurred 0: A TMR2 Interrupt has not occurred	-
TMR3IRQ	r/w1tc	1: A TMR3 Interrupt has occurred 0: A TMR3 Interrupt has not occurred	-
TMR4IRQ	r/w1tc	1: A TMR4 Interrupt has occurred 0: A TMR4 Interrupt has not occurred	-
WAKE_IRQ	r	1: A Wake Interrupt has occurred 0: A Wake Interrupt has not occurred	-
TSM_IRQ	r	1: A TSM Interrupt has occurred 0: A TSM Interrupt has not occurred	-
TMR1MSK	rw	1: Mask TMR1 Interrupt from asserting ipi_int_zigbee	1

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description	Default
		0: Enable TMR1 Interrupt to assert ipi_int_zigbee	
TMR2MSK	rw	1: Mask TMR2 Interrupt from asserting ipi_int_zigbee 0: Enable TMR2 Interrupt to assert ipi_int_zigbee	1
TMR3MSK	rw	1: Mask TMR3 Interrupt from asserting ipi_int_zigbee 0: Enable TMR3 Interrupt to assert ipi_int_zigbee	1
TMR4MSK	rw	1: Mask TMR4 Interrupt from asserting ipi_int_zigbee 0: Enable TMR4 Interrupt to assert ipi_int_zigbee	1
SEQMSK	rw	1: Mask Sequencer Interrupt from asserting ipi_int_zigbee 0: Enable Sequencer Interrupt to assert ipi_int_zigbee	1
TXMSK	rw	1: Mask TX Interrupt from asserting ipi_int_zigbee 0: Enable TX Interrupt to assert ipi_int_zigbee	1
RXMSK	rw	1: Mask RX Interrupt from asserting ipi_int_zigbee 0: Enable RX Interrupt to assert ipi_int_zigbee	1
CCAMSK	rw	1: Mask CCA Interrupt from asserting ipi_int_zigbee 0: Enable CCA Interrupt to assert ipi_int_zigbee	1
RX_WMRK_MSK	rw	1: Mask RX Watermark Interrupt from asserting ipi_int_zigbee 0: Enable RX Watermark Interrupt to assert ipi_int_zigbee	1
FILTERFAIL_MSK	rw	1: Mask Filter Fail Interrupt from asserting ipi_int_zigbee 0: Enable Filter Fail Interrupt to assert ipi_int_zigbee	1
PLL_UNLOCK_MSK	rw	1: Mask PLL Unlock Interrupt from asserting ipi_int_zigbee 0: Enable PLL Unlock Interrupt to assert ipi_int_zigbee	1
WAKE_MSK	rw	1: Mask Wake Interrupt from asserting ipi_int_zigbee 0: Enable Wake Interrupt to assert ipi_int_zigbee	1
TSM_MSK	rw	1: Mask TSM Interrupt from asserting ipi_int_zigbee 0: Enable TSM Interrupt to assert ipi_int_zigbee	1
TRCV_MSK	rw	1: Mask all Interrupts from asserting ipi_int_zigbee 0: Enable any Interrupt to assert ipi_int_zigbee	0
T1CMP[23:0]	rw	TMR1 compare value. If TMR1CMP_EN=1 and the Event Timer matches this value, TMR1IRQ is set.	0xFFF

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description	Default
T2CMP[23:0]	rw	TMR2 compare value. If TMR2CMP_EN=1 and the Event Timer matches this value, TMR2IRQ is set.	0xFFF
T2PRIMECMP[15:0]	rw	TMR2-prime compare value. If TMR2CMP_EN=1 and TC2PRIME_EN=1 and the lower 16 bits of Event Timer matches this value, TMR2IRQ is set.	0xFF
T3CMP[23:0]	rw	TMR3 compare value. If TMR3CMP_EN=1 and the Event Timer matches this value, TMR3IRQ is set.	0xFFF
T4CMP[23:0]	rw	TMR4 compare value. If TMR4CMP_EN=1 and the Event Timer matches this value, TMR4IRQ is set.	0xFFF
TMR1CMP_EN	rw	1: Allow an Event Timer Match to T1CMP to set TMR1IRQ 0: Don't allow an Event Timer Match to T1CMP to set TMR1IRQ	0
TMR2CMP_EN	rw	1: Allow an Event Timer Match to T2CMP or T2PRIMECMP to set TMR2IRQ 0: Don't allow an Event Timer Match to T2CMP or T2PRIMECMP to set TMR2IRQ	0
TMR3CMP_EN	rw	1: Allow an Event Timer Match to T3CMP to set TMR3IRQ 0: Don't allow an Event Timer Match to T3CMP to set TMR3IRQ	0
TMR4CMP_EN	rw	1: Allow an Event Timer Match to T4CMP to set TMR4IRQ 0: Don't allow an Event Timer Match to T4CMP to set TMR4IRQ	0
TC2PRIME_EN	rw	1: Allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ 0: Don't allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ	0
RX_WTR_MARK[7:0]	rw	Receive byte count (octets) needed to trigger a RXWTRMRKIRQ interrupt . A setting of 0 generates an interrupt at end of the Frame Length field (first byte after SFD). A setting of 1 generates an interrupt after the first byte of Frame Control Field, etc.	0xFF

NOTE

For the standalone Block Guide shell project the DIL reference is commented out so that register tables will not show up in the project until they are explicitly included (see the .ditamap file).

55.4.9.1.2.11.3 Functional description

55.4.9.1.2.11.3.1 Interrupt Status Bit Structure

The 802.15.4 Link Layer has 13 interrupt sources, each represented in the register map by an interrupt status bit. All interrupt status bits share a common, generic structure. If a particular interrupt source is enabled, a "TRIGGERING EVENT" for that interrupt source, will always set the status bit. A write-1-to-clear input from the IPS bus, will clear the status bit, but only if there is not a simultaneous triggering event. The triggering event prevails over a software clear attempt, if both events coincide. The clock gate guarantees that the status bit only sees a clock when either a triggering event occurs, or a write-1-to-clear pulse arrives

from the IPS bus. This reduces interrupt status bit power consumption to an absolute minimum. The common interrupt status bit structure is shown in the following diagram.

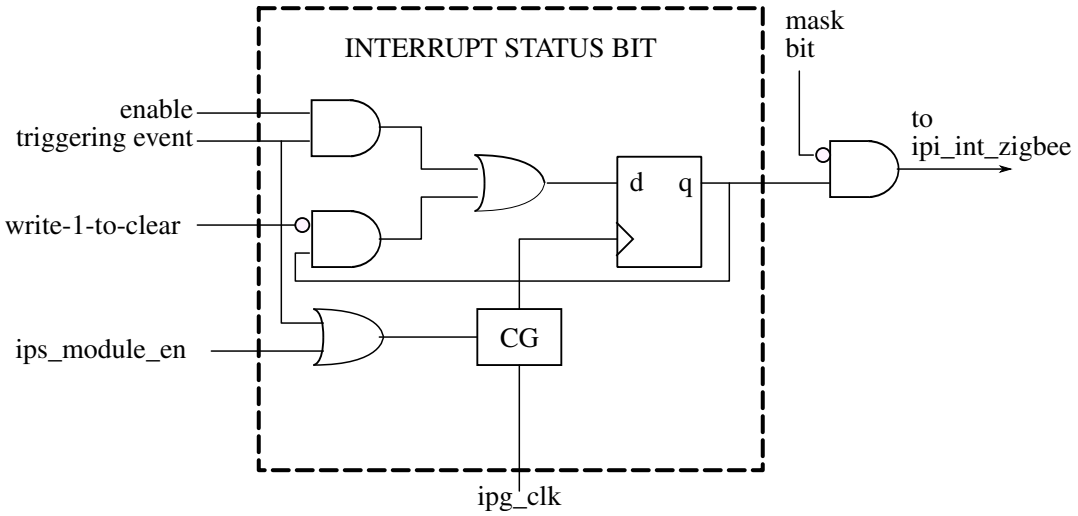


Figure 398. Interrupt Status Bit

Note that for any 802.15.4 interrupt source, if the triggering event occurs, the Interrupt Status Bit will be set regardless of the state of the corresponding MASK bit. If any of the 13 interrupts is to be ignored, software should set the corresponding MASK bit, and apply the appropriate bit mask when reading the IRQSTS1, IRQSTS2, or IRQSTS3 register, to mask out the unwanted status bit.

55.4.9.1.2.11.3.2 802.15.4 Interrupt Architecture

The 13 802.15.4 interrupt sources (status bits) are combined with their individual MASK bits, and then logically OR'ed together, in sum-of-products fashion, to generate single line to the MCU: ipi_int_zigbee. A global mask bit, TRCV_MSK, can enable or disable ipi_int_zigbee altogether. There is no prioritization of interrupt sources, they all have equal weight. The ipi_int_zigbee output is a level-sensitive indicator and will remain asserted until all interrupt status bits are cleared (or the corresponding mask bits set). The following diagram depicts the 802.15.4 Interrupt Architecture.

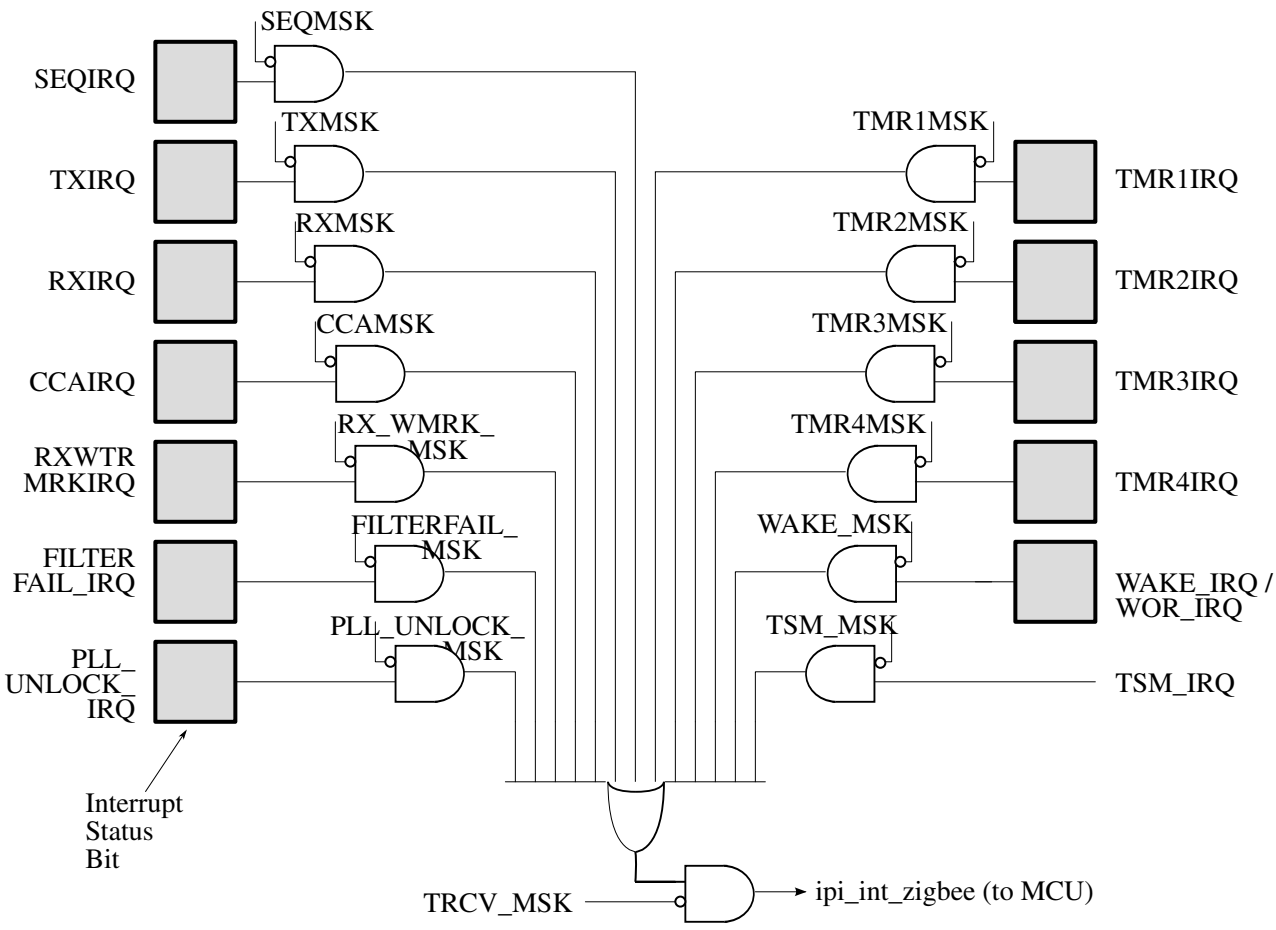


Figure 399. 802.15.4 Interrupt Architecture

55.4.9.1.2.11.3.3 Clearing Interrupts

All interrupt status bit use a write-1-to-clear protocol. Writing a '1' to the interrupt status bit, in the IRQSTS1, IRQSTS2, or IRQSTS3 register, clears the offending interrupt. Writing a '0' to an interrupt status bit has no effect on the bit. Interrupt status bits are not affected by reads.

Note: The TSM interrupt is an exception from the interrupt-clearing rules stated above. See Section TSM Interrupt below.

55.4.9.1.2.11.3.4 Timer Interrupts

The 802.15.4 Link Layer features a 24-bit Event Timer (see Section Event Timer). The Link Layer has 4 Timer interrupts (TMR1IRQ, TMR2IRQ, TMR3IRQ, and TMR4IRQ), each with its own 24-bit compare register (T1CMP, T2CMP, T3CMP, and T4CMP), and each with its own compare enable (TMR1CMP_EN, TMR2CMP_EN, TMR3CMP_EN, and TMR4CMP_EN). For each timer compare enable, if the bit is set, a match on the respective 24-bit compare value to the Event Timer will cause the corresponding interrupt status bit to become set. If the compare enable is low, Event Timer matches won't cause the corresponding interrupt status bit to become set.

In addition, a 16-bit T2PRIMECMP compare value is provided, along with a compare-enable bit: TC2PRIME_EN. When TC2PRIME_EN is set (and TMR2CMP_EN is also set), a match on T2PRIMECMP with the *lower 16 bits of Event Timer*, will cause TMR2IRQ to become set, rather than a full 24-bit compare.

55.4.9.1.2.11.3.5 PLL Unlock Interrupt

When an PLL unlock event occurs during a 802.15.4 autosequence, the PLL_UNLOCK_IRQ status bit will become set. The ZSM Sequence Manager will begin monitoring for PLL unlock only after a complete transceiver warmup has occurred; unlocks which occur during warmup will not cause a PLL_UNLOCK_IRQ. A PLL unlock which occurs after the warmup period, can be enabled to cause a sequence abort. (See Section Sequence Aborting for more details).

55.4.9.1.2.11.3.6 Filterfail Interrupt

Filterfail interrupt will occur during packet reception, when a packet fails filtering rules. On a Filterfail interrupt, the FILTERFAIL_CODE register can be read to ascertain more information about the nature of the filter failure. Filterfail interrupt is not expected to be widely used in mission modes; it has been provided primarily for debug purposes.

55.4.9.1.2.11.3.7 RX Watermark Interrupt

RX Watermark interrupt will occur during packet reception, when the number of received bytes matches the contents of the RX_WTR_MARK register, minus 1. For the purpose of defining RX_WTR_MARK, the first byte received is the Frame Length field (PHR). The second byte received is the least-significant byte of Frame Control Field, etc. For example, to cause an RX Watermark Interrupt to occur after the 10th received byte, set RX_WTR_MARK=11.

55.4.9.1.2.11.3.8 CCA Interrupt

CCA interrupts occur at the end of CCA and ED measurement intervals. The presence, and timing, of CCA interrupts varies based on which autosequence is engaged. See Chapter ZSM Sequence Manager, Section Supported Sequences for more details on when to expect a CCAIRQ for each of the relevant autosequences (Sequences C, T, TR, and CCCA).

55.4.9.1.2.11.3.9 RX Interrupt

RX interrupts occur at the end of an RX operation. An RX operation can be a standalone Sequence R, or the “R” portion of a Sequence TR. An RXIRQ, in combination with a SEQIRQ, is considered a “Data Indication”. RX interrupts are not generated on packets which fail FCS (CRC check), or fail packet filtering. Either of these two conditions results in an RX recycle back to the preamble-search state. To prevent RX recycling on packets which fail filtering or FCS, set the NO_RX_RECYCLE bit of the SEQ_MGR_CTRL register to 1. To receive a Data Indication (including RXIRQ) on packets which fail CRC, clear the CRC_MSK bit of the PHY_CTR2 register to 0. To inhibit packet filtering and enable Data Indication on packets which would otherwise fail packet filtering rules, set the PROMISCUOUS bit of the PHY_CTRL4 register to 1.

55.4.9.1.2.11.3.10 TX Interrupt

TX interrupts occur at the end of a TX operation. An TX operation can be a standalone Sequence T, the “T” portion of a Sequence TR, or the auto-TXACK portion of a Sequence R with AUTOACK=1.

55.4.9.1.2.11.3.11 Sequencer Interrupt

The Sequencer Interrupt (SEQIRQ), indicates that an autosequence has completed, and the 802.15.4 Sequence Manager has returned to its idle state. A SEQIRQ will *always* occur at the end of an autosequence, even if the autosequence terminated abnormally (such as a Software Abort, a TC3 Timeout, or a PLL Unlock Abort). **Note:** the ABORT_STS register can be read to determine which of these abnormal terminations occurred, if any. The SEQIRQ always occurs whenever the Sequence Manager transitions from non-idle to idle state. When SEQIRQ occurs, software can be sure that the Sequence Manager is in its idle state, and a new sequence can be programmed immediately.

55.4.9.1.2.11.3.12 TSM Interrupt

TSM_IRQ is a debug feature, enabling the Transceiver Sequence Manager (TSM) to generate an interrupt at any point in a TX or RX Warmup. TSM is a multipurpose hardware resource shared by all of the protocol engines in the SoC. Thus, TSM does not have its own interrupts; its interrupts are assigned to whichever link layer controller is currently executing an RF operation. The 802.15.4 interrupt TSM_IRQ is asserted when the TSM interrupt is enabled in the transceiver's TSM_CTRL register, and the TSM interrupt asserts during an 802.15.4 TX or RX Warmup. The TSM Interrupt status bit in the IRQSTS register of 802.15.4 space is read-only. It is a logical OR of the 2 TSM_IRQ interrupt status bits of the XCVR_STATUS register in XCVR address space. and the asserted status bit(s) should be cleared there. There is no intended mission-mode use for TSM_IRQ. See the TSM Chapter for more details.

55.4.9.1.2.11.3.13 Wake Interrupt

For Manual DSM: This is WAKE_IRQ. A WAKE_IRQ will be triggered when the 802.15.4 Link Layer Controller has awoken from a Manual DSM (Deep Sleep Mode) cycle. WAKE_IRQ indicates that the RF Oscillator has been restarted, and the 802.15.4 EVENT_TMR has resumed counting.

For Wake-On-Radio:: This is WOR_IRQ. This interrupt will become set when 802.15.4 has been assigned to Wake-On-Radio, and one or more of the following Wake-On-Radio conditions occurs:

1. A Wake-On-Radio slot of type "NO_RF" has completed, and MCU wakeup is requested in the descriptor
2. A Wake-On-Radio maximum slot count has been reached
3. A Wake-On-Radio Early Exit condition has been encountered

See Chapter *Wake-On-Radio* for more details

55.4.9.1.2.12 Wifi Coexistence

Provisions have been made to allow for the 2.4GHz transceiver to coexist in the same space as a WiFi transceiver IC, which shares the same frequency band. The coexistence scheme designates the WiFi transceiver as the master, and the 2.4GHz transceiver as the slave. The objective of the coexistence strategy is to prevent both the WiFi and 2.4GHz transceivers transmitting simultaneously; a configuration option exists to also prevent the 2.4GHz transceiver from receiving when the WiFi transceiver owns the RF channel.

A very basic mechanism is used in cases where 2.4GHz radio does not generate requests, but instead defers to the WiFi transceiver to grant permission to use the medium. The WiFi IC generates a signal, RF_NOT_ALLOWED. If this signal is asserted, then 2.4GHz radio does not perform any communication. When this signal is de-asserted, then 2.4GHz radio is free to perform communications. In case the signal RF_NOT_ALLOWED is de-asserted when 2.4GHz radio has already initiated the transmission/reception of a packet then the 2.4GHz radio must stop its activity immediately .

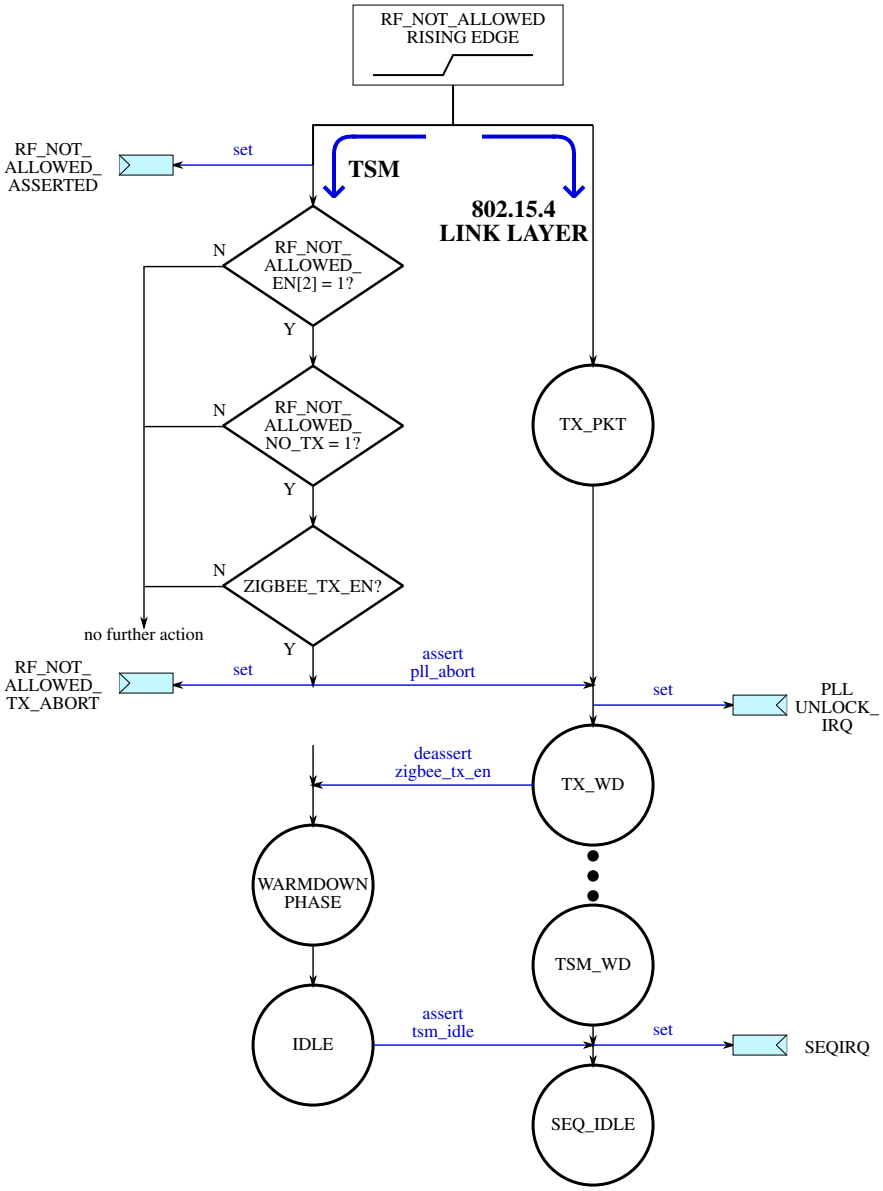
When RF_NOT_ALLOWED aborting is enabled, the 802.15.4 Link Layer hardware must respond to RF_NOT_ALLOWED assertions, by aborting any autosequence which is currently underway, and Link Layer software must not initiate any new autosequences until RF_NOT_ALLOWED has been deasserted. The hardware response must be handled autonomously by the Link Layer, with no MCU intervention required, because the RF_NOT_ALLOWED assertions can occur during a Wake-On-Radio timeslot which has been designated as "no-MCU-wakeup" in the slot's descriptor. In this scenario, the aborting of the autosequence occurs without any assistance from, or notification of, the MCU.

In the multi-protocol 2.4GHz radio, RF_NOT_ALLOWED aborting can be individually enabled/disabled for each protocol engine. For 802.15.4, RF_NOT_ALLOWED_EN[2] is the associated enable control bit. This bit resides in the COEX_CTRL register in XCVR address space. When this bit is 0, transitions on RF_NOT_ALLOWED are ignored by the 802.15.4 Link Layer hardware; when this bit is 1, the 802.15.4 Link Layer hardware will monitor RF_NOT_ALLOWED at all times, and abort any active autosequence which is underway when an assertion on the pin occurs. The complete hardware response to RF_NOT_ALLOWED assertions is described below.

Additional control over RF_NOT_ALLOWED aborting is provided by the RF_NOT_ALLOWED_NO_TX and RF_NOT_ALLOWED_NO_RX control bits. If RF_NOT_ALLOWED_NO_TX=1, then an RF_NOT_ALLOWED abort will occur only if an 802.15.4 autosequence is active, and currently executing a TX operation (zigbee_tx_en=1); 802.15.4 autosequence TX operations will not be aborted if RF_NOT_ALLOWED_NO_TX=0. If RF_NOT_ALLOWED_NO_RX=1, then an RF_NOT_ALLOWED abort will occur only if an 802.15.4 autosequence is active, and currently executing an RX operation (zigbee_rx_en=1); 802.15.4 autosequence RX operatons will not be aborted if RF_NOT_ALLOWED_NO_RX=0. The RF_NOT_ALLOWED_NO_TX and RF_NOT_ALLOWED_NO_RX control bits reside in the COEX_CTRL register.

For the purposes of triggering a hardware abort, the *pll_abort* input to the 802.15.4 Sequence Manager (ZSM) is used. This is because the hardware response to the RF_NOT_ALLOWED assertion is identical to that of a PLL unlock event. This also means that, when RF_NOT_ALLOWED aborting is enabled for 802.15.4, the PLL_UNLOCK_IRQ interrupt status bit, will be dual purpose: it will not only indicate a PLL unlock condition, but also a RF_NOT_ALLOWED abort. PLL aborting and RF_NOT_ALLOWED aborting are enabled separately, with the COEX_CTRL register maintaining the control bits required for the latter. The status bits for RF_NOT_ALLOWED aborting are also available in COEX_CTRL, so that software will be able to distinguish the source of the PLL_UNLOCK_IRQ, in case both PLL and RF_NOT_ALLOWED aborting are enabled.

The sequence of events which results in a hardware abort of an 802.15.4 TX operation triggered by an assertion on RF_NOT_ALLOWED, is a collaboration between the TSM (Transceiver Sequence Manager) and the 802.15.4 Sequence Manager (ZSM), as shown in the following diagram.



Upon assertion on RF_NOT_ALLOWED, the TSM sets the RF_NOT_ALLOWED_ASSERTED status bit in COEX_CTRL, then checks that the conditions for a hardware abort are all met:

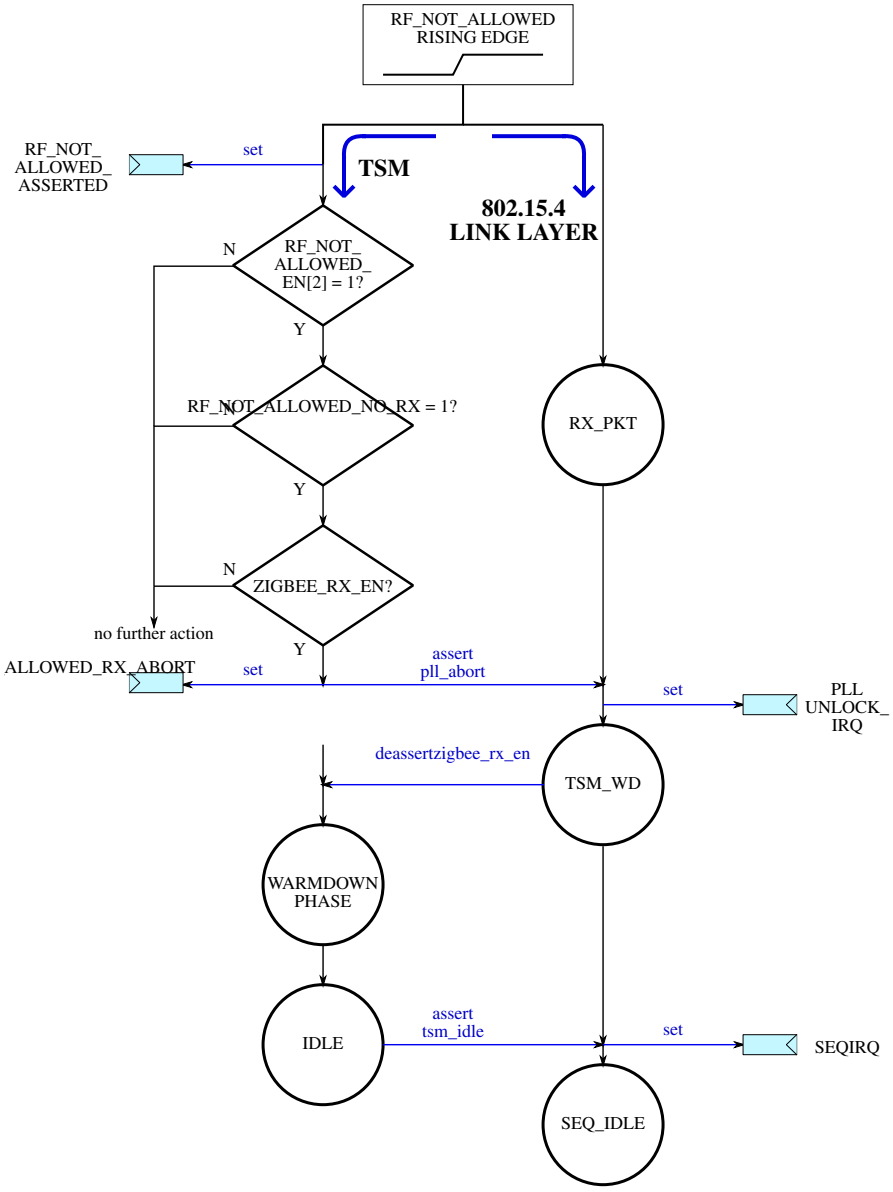
- RF_NOT_ALLOWED_EN[2] = 1, which enables 802.15.4 to respond to RF_NOT_ALLOWED events
- RF_NOT_ALLOWED_NO_TX = 1, which enables TX operations to be aborted

- zigbee_tx_en = 1, TX request to TSM from 802.15.4 ZSM, indicating TX operation in progress

If all conditions are not met, no further action is taken. Otherwise, the TSM sets COEX_CTRL[RF_NOT_ALLOWED_TX_ABORT], and also asserts *pll_unlock* to the 802.15.4 ZSM Sequence Manager. The ZSM responds by asserting IRQSTS1[PLL_UNLOCK_IRQ] in 802.15.4 space. ZSM enters TX_WD state, which deasserts zigbee_tx_en to the TSM. This initiates the TSM TX warmdown, while the ZSM simultaneously steps through its TX warmdown sequence of states. ZSM enters TSM_WD to wait for TSM to return to idle. Once this occurs, ZSM asserts IRQSTS1[SEQIRQ] and returns to its SEQ_IDLE state. Three status bits are now set to indicate to software that the source of the abort was an RF_NOT_ALLOWED assertion:

- IRQSTS1[PLL_UNLOCK_IRQ]
- COEX_CTRL[RF_NOT_ALLOWED_ASSERTED]
- COEX_CTRL[RF_NOT_ALLOWED_TX_ABORT]

The sequence of events which results in a hardware abort of an 802.15.4 RX operation triggered by an assertion on RF_NOT_ALLOWED, is shown in the following diagram.



Upon assertion on RF_NOT_ALLOWED, the TSM sets the RF_NOT_ALLOWED_ASSERTED status bit in COEX_CTRL, then checks that the conditions for a hardware abort are all met:

- RF_NOT_ALLOWED_EN[2] = 1, which enables 802.15.4 to respond to RF_NOT_ALLOWED events
- RF_NOT_ALLOWED_NO_RX = 1, which enables RX operations to be aborted
- zigbee_rx_en = 1, RX request to TSM from 802.15.4 ZSM, indicating RX operation in progress

If all conditions are not met, no further action is taken. Otherwise, the TSM sets COEX_CTRL[RF_NOT_ALLOWED_RX_ABORT], and also asserts *pll_unlock* to the 802.15.4 ZSM Sequence Manager. The ZSM responds by asserting IRQSTS1[PLL_UNLOCK_IRQ] in 802.15.4 space. ZSM enters TSM_WD state, which deasserts zigbee_rx_en to the TSM. This initiates the TSM RX warmdown. ZSM waits in the TSM_WD to wait for TSM to return to idle. Once this occurs, ZSM asserts IRQSTS1[SEQIRQ] and returns to its SEQ_IDLE state. Three status bits are now set to indicate to software that the source of the abort was an RF_NOT_ALLOWED assertion:

- IRQSTS1[PLL_UNLOCK_IRQ]
- COEX_CTRL[RF_NOT_ALLOWED_ASSERTED]
- COEX_CTRL[RF_NOT_ALLOWED_RX_ABORT]

The complete set of control and status bits in the COEX_CTRL register is described in the following table:

Field	R/W	Description
RF_NOT_ALLOWED_EN	RW	The coexistence input RF_NOT_ALLOWED can be enabled to selectively abort TX or RX sequences, with individual enables for each supported protocol, subject also to the state of the RF_NOT_ALLOWED_NO_TX and RF_NOT_ALLOWED_NO_RX control bits, according to the following table: **xxx1**: RF_NOT_ALLOWED assertions are enabled to abort Bluetooth LE TX and RX sequences **xxx0**: RF_NOT_ALLOWED assertions are not enabled to abort Bluetooth LE TX and RX sequences **xx1x**: RF_NOT_ALLOWED assertions are enabled to abort ANT TX and RX sequences **xx0x**: RF_NOT_ALLOWED assertions are not enabled to abort ANT TX and RX sequences **x1xx**: RF_NOT_ALLOWED assertions are enabled to abort 802.15.4 TX and RX sequences **x0xx**: RF_NOT_ALLOWED assertions are not enabled to abort 802.15.4 TX and RX sequences **1xxx**: RF_NOT_ALLOWED assertions are enabled to abort GENERIC_FSK TX and RX sequences **0xxx**: RF_NOT_ALLOWED assertions are not enabled to abort GENERIC_FSK TX and RX sequences
RF_NOT_ALLOWED_NO_TX	RW	TX Sequences will be aborted by RF_NOT_ALLOWED assertions, if the appropriate RF_NOT_ALLOWED_EN[x] bit is set.
RF_NOT_ALLOWED_NO_RX	RW	RX Sequences will be aborted by RF_NOT_ALLOWED assertions, if the appropriate RF_NOT_ALLOWED_EN[x] bit is set.
RF_NOT_ALLOWED_ASSERTED	R/W1TC	RF_NOT_ALLOWED has gone from not asserted, to asserted, since the last time this bit was cleared

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
RF_NOT_ALLOWED_TX_ABORT	R/W1TC	An TX sequence has been aborted since the last time this bit was cleared
RF_NOT_ALLOWED_TX_ABORT	R/W1TC	An TX sequence has been aborted since the last time this bit was cleared
RF_NOT_ALLOWED	R	Reflects the instantaneous state of the RF_NOT_ALLOWED pin,

55.4.9.1.2.13 Enhanced Acknowledgment (Enh-Ack) Support

This device supports to send Enh-Ack frame, acknowledging to a frame version 2 of DATA or MAC Data Request Command packet. By the combination of Destination Address Mode, Source Address Mode and PAN ID Compression, the 802.15.4-2015 protocol specifies 14 possible packet structures in its Table 7-2. According to specific application scenarios, this device defines case 5 and 9 ~ 14 are legal, as shown in the Figure 400. Other cases will be filtered out by default unless LENIENCY[40] is set.

CASE #	Destination Address Mode	Source Address Mode	Destination Address	Source Address	Destination PAN ID	Source PAN ID	PAN ID Compression	SAM (checksum)	ACK
#1	0	0	Not Present	Not Present	Not Present	Not Present	0	NO	NO
#2	0	0	Not Present	Not Present	Present	Not Present	1	NO	NO
#3A	2	0	Short	Not Present	Present	Not Present	0	NO	NO
#3B	3	0	Extended	Not Present	Present	Not Present	0	NO	NO
#4A	2	0	Short	Not Present	Not Present	Not Present	1	NO	NO
#4B	3	0	Extended	Not Present	Not Present	Not Present	1	NO	NO
#5A	0	2	Not Present	Short	Not Present	Present	0	SAM(SP+SA)	YES
#5B	0	3	Not Present	Extended	Not Present	Present	0	SAM(SP+SA)	YES
#6A	0	2	Not Present	Short	Not Present	Not Present	1	NO	NO
#6B	0	3	Not Present	Extended	Not Present	Not Present	1	NO	NO
#7	3	3	Extended	Extended	Present	Not Present	0	NO	NO
#8	3	3	Extended	Extended	Not Present	Not Present	1	NO	NO
#9	2	2	Short	Short	Present	Present	0	SAM(SP+SA)	YES
#10	2	3	Short	Extended	Present	Present	0	SAM(SP+SA)	YES
#11	3	2	Extended	Short	Present	Present	0	SAM(SP+SA)	YES
#12	2	3	Short	Extended	Present	Not Present	1	SAM(DP+SA)	YES
#13	3	2	Extended	Short	Present	Not Present	1	SAM(DP+SA)	YES
#14	2	2	Short	Short	Present	Not Present	1	SAM(DP+SA)	YES

Figure 400. Supported Address Modes of Beacon, MAC Command and DATA Frame

In response to these supported packets, the format of the ACK packet may be in case #7~14, as shown in the table. This rule is used in the TR sequence with RXACKRQD is set.

CASE #	Destination Address Mode	Source Address Mode	Destination Address	Source Address	Destination PAN ID	Source PAN ID	PAN ID Compression
#1	0	0	Not Present	Not Present	Not Present	Not Present	0
#2	0	0	Not Present	Not Present	Present	Not Present	1
#3A	2	0	Short	Not Present	Present	Not Present	0
#3B	3	0	Extended	Not Present	Present	Not Present	0
#4A	2	0	Short	Not Present	Not Present	Not Present	1
#4B	3	0	Extended	Not Present	Not Present	Not Present	1
#5A	0	2	Not Present	Short	Not Present	Present	0
#5B	0	3	Not Present	Extended	Not Present	Present	0
#6A	0	2	Not Present	Short	Not Present	Not Present	1
#6B	0	3	Not Present	Extended	Not Present	Not Present	1
#7	3	3	Extended	Extended	Present	Not Present	0
#8	3	3	Extended	Extended	Not Present	Not Present	1
#9	2	2	Short	Short	Present	Present	0
#10	2	3	Short	Extended	Present	Present	0
#11	3	2	Extended	Short	Present	Present	0
#12	2	3	Short	Extended	Present	Not Present	1
#13	3	2	Extended	Short	Present	Not Present	1
#14	2	2	Short	Short	Present	Not Present	1

Figure 401. Supported Address Modes of ACK Frame

The Enh-Ack frame is formatted as illustrated in Figure 402, also can be found in Figure 7-16 in IEEE Std 802.15.4-2015.

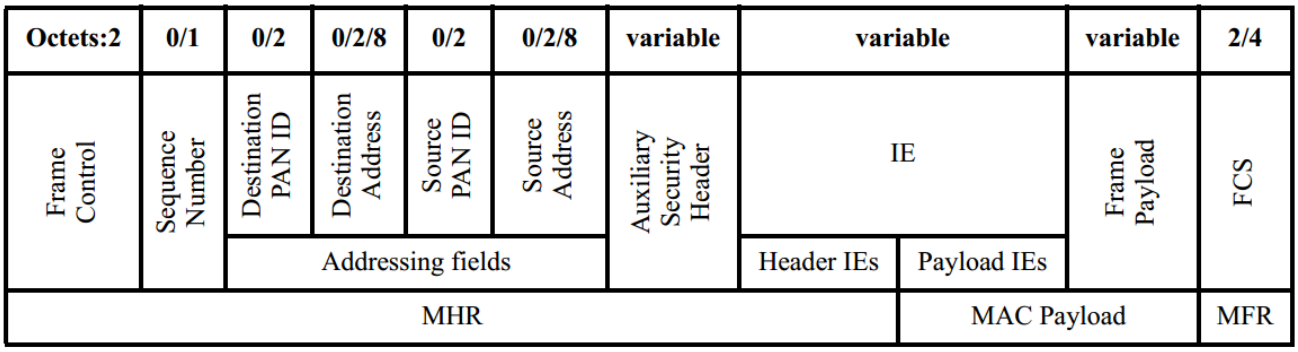


Figure 402. Enh-Ack frame format

The Frame Control field of the Enh-Ack frame is formatted as illustrated in Figure 403, also can be found in Figure 7-2 in IEEE Std 802.15.4-2015.

Bits: 0–2	3	4	5	6	7	8	9	10–11	12–13	14–15
Frame Type	Security Enabled	Frame Pending	AR	PAN ID Compression	Reserved	Sequence Number Suppression	IE Present	Destination Addressing Mode	Frame Version	Source Addressing Mode

Figure 403. Frame Control Field of Enh-Ack

Enh-Ack is much more complicated than Imm-Ack because it may contain IEs and Frame Payload Field, and it also introduces data decryption and encryption. This makes it very difficult to achieve a turnaround time of 192us in pure software or pure hardware. So the entire Enh-Ack process is completed through software and hardware cooperation.

The main idea is that the software is responsible for decrypting, analyzing received packets and forming the Enh-Ack packet(Full-Enh-Ack). At the turnaround time, the hardware automatically sends the Full-Enh-Ack. If the software cannot prepare the Full-Enh-Ack in time, the hardware will send an Empty-Enh-Ack frame. After the hardware sends Empty-Enh-Ack, the software will send a complete Enh-Ack packet, which is Full-Enh-Ack.

According to the requirements of 802.15.4 -2015 spec for Enh-Ack frame format, the fields in Full-Enh-Ack and Empty-Enh-Ack frame are as follows:

In the Full-Enh-Ack:

- The Frame Control field:
  - The Frame Type is b010, Acknowledgment type
  - The Frame Version is b10.
  - Security Enabled field shall be set if security feature is enabled.
  - If the Enh-Ack frame is being sent in response to a received either a Data frame or Data Request command the Frame Pending field is calculated by the Source Address Management process; If the Enh-Ack frame is being sent in response to another type of MAC command, the Frame Pending field is zero.
  - The AR field is zero.
  - The PAN ID compression field and Sequence Number Compression field shall be set to be the same value as the corresponding fields in the Frame Control field of the frame that is being acknowledged.
  - The IEs present field shall be set to one if IEs are included in the Enh-Ack frame.
  - The Destination Addressing Mode field shall contain the value of the Source Addressing Mode field in the Frame Control field of the frame that is being acknowledged.
  - The Source Addressing mode field shall be set as appropriate for the address of the device transmitting the Enh-Ack frame.
- The Sequence Number field, when present shall contain the value of the sequence number received in the frame for which the acknowledgment is to be sent.
- The Source Address field, when present, is from MACSHORTADDRS0(if CURRENT_NETWORK is 0) or MACSHORTADDRS1(if CURRENT_NETWORK is 1) when EMPTY_SRC_ADDR_MODE is 2; or is from MACLONGADDRS0(if CURRENT_NETWORK is 0) or MACLONGADDRS1(if CURRENT_NETWORK is 1) when EMPTY_SRC_ADDR_MODE is 3.
- The Source PAN ID field, when present, is from MACPANID0 (if CURRENT_NETWORK is 0) or MACPANID1(if CURRENT_NETWORK is 1).

- The Destination Address field, when present, shall contain the value of the Source Address field from the frame that is being acknowledged. The Destination PAN ID, when present, contains the value from the Source PAN ID field from the frame that is being acknowledged.
- The Auxiliary Security Header, IE and Frame Payload may be included.

Because Empty-Enh-Ack is a simplified format, compared with Full-Enh-Ack, there are some *differences*.

- The Frame Control field:
  - The Frame Type is b010, Acknowledgment type.
  - The Frame Version is b10.
  - *Security Enabled field is 0, indicating the Enh-Ack frame is not protected. (when EMPTY_SECURITY_ENABLED_OVRD_EN is 1, this field tracks the EMPTY_SECURITY_ENABLED_OVRD bit)*
  - *The Frame Pending field is 1, indicating that there will be a Full-Enh-Ack frame after the Empty-Enh-Ack frame. (when ACK_FRM_PND_CTRL is 1, this field tracks the ACK_FRM_PND bit)*
  - The AR field is zero
  - The PAN ID compression field and Sequence Number Compression field shall be set to be the same value as the corresponding fields in the Frame Control field of the frame that is being acknowledged.
  - *The IEs present field is 0, indicating there is no IEs field in the Empty-Enh-Ack*
  - The Destination Addressing Mode field shall contain the value of the Source Addressing Mode field in the Frame Control field of the frame that is being acknowledged
  - *The Source Addressing mode field is from register EMPTY_SRC_ADDR_MODE, which should be 2 or 3 only.*
- The Sequence Number field, when present shall contain the value of the sequence number received in the frame for which the acknowledgment is to be sent.
- The Source Address field, when present, is from MACSHORTADDRS0(if CURRENT_NETWORK is 0) or MACSHORTADDRS1(if CURRENT_NETWORK is 1) when EMPTY_SRC_ADDR_MODE is 2; or is from MACLONGADDRS0(if CURRENT_NETWORK is 0) or MACLONGADDRS1(if CURRENT_NETWORK is 1) when EMPTY_SRC_ADDR_MODE is 3.
- The Source PAN ID field, when present, is from MACPANID0 (if CURRENT_NETWORK is 0) or MACPANID1(if CURRENT_NETWORK is 1).
- The Destination Address field, when present, shall contain the value of the Source Address field from the frame that is being acknowledged. The Destination PAN ID, when present, contains the value from the Source PAN ID field from the frame that is being acknowledged.
- *The Auxiliary Security Header, IE and Frame Payload are not exist.*

The key steps/points of the Enhanced Acknowledgment flow are as follows.

- The hardware provides a mechanism to ensure that the software can read and write RAM during the process of receiving and sending packets.
- The hardware provides RXWTRMRKIRQ, which can provide an interrupt indication when receiving any number bytes of packets, by software configuration.
- During the receiving process, by dynamically configuring the RX_WTR_MARK register, software no longer waits until the entire packet is received, but after a part of the packet is written into RAM by the hardware. The purpose of this is to enable the software to start decrypting and parsing the package in advance.
- After received the unencrypted MHR and parsed the Frame Version 2, the hardware checks whether the packet is sent to this device and whether it complies with the 802.15.4 -2015 spec. The detailed parsing flow is show in Figure 404. For stage "Address Mode Checking(Valid case in Table 7-2)", it checks if the frame type is Beacon, MAC Command or DATA, check if it is a valid case in Figure 400; if the frame type is ACK, check if it is a valid case in Figure 401. Other rule-checking for each

stage is described in Packet Filtering Detail. If the packet does not pass the parsing flow, the hardware may abort process and set FILTERFAIL_IRQ. Software should also abort its process when received FILTERFAIL_IRQ.

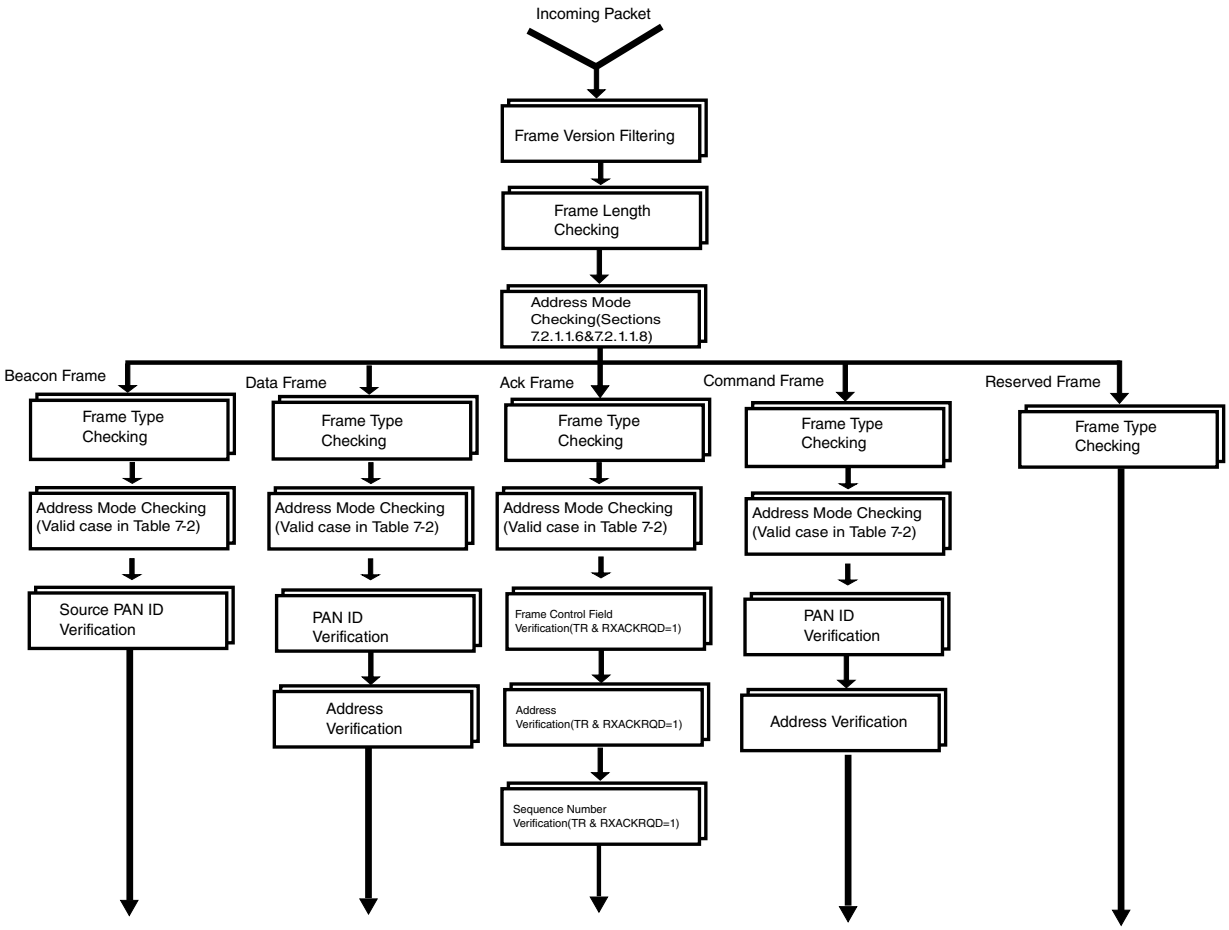


Figure 404. Packet Filtering Flow Diagram of Frame Version 2

- If received DATA or MAC Command packet, the hardware will automatically perform the Source Address Management process if it is enabled, and calculate Frame Pending field, put it in register HW_FRAME_PENDING. This bit will be placed in the Frame Pending field of Enh-Ack, if the software further parses the packet is a MAC Data Request Command or DATA type packet. Otherwise, Enh-Ack's Frame Pending field is 0 and HW_FRAME_PENDING is not used.
- Software may also do some additional rule checks and can abort the process on fail.
- If data authenticity is enabled, software also checkes the data authenticity by message integrity code(MIC). If fail, the software can abort the process.
- At the end of packet, the CRC check is performed by hardware. If CRC fails, the auto acknowledge process is aborted. There are two cases if CRC fails. For each case, the software will be notified to be aborted: Case 1. if NO_RX_RECY CLE is 0 (by default), hardware automaticly recycles and RECYC_IRQ is asserted; Case 2. if NO_RX_RECY CLE is 1, RXIRQ and SEQIRQ are asserted. Software aborts if register CRCVALID is 0.
- If the received packet meets the following conditions, the software starts to prepare the Enh-Ack packet and hardware starts to schedule the acknowledge time.
  - the Frame Version is 2
  - the AR field is 1
  - the Frame Type is DATA or MAC Command

  - filtering flow check pass
  - MIC check pass
  - CRC check pass
- At the 192us turnaround time, if no software abort, hardware starts sending Preamble(4 bytes) and SFD(1 byte), which lasts for 160us. So from the receipt of the last CRC bit, there is a total of 352us for the software to prepare the Enh-Ack packet.
- The software needs to prepare the Length Field first, because this is the first field to be sent after Preamble and SFD. After writing Length Field to RAM, the software needs to set register SW_LEN_RDY to 1.
- After the SFD is sent, the SW_LEN_RDY is used by the hardware to check whether the software Enh-Ack package has been assembled. If SW_LEN_RDY is 1, the hardware continues to read the Full-Enh-Ack from RAM and send it to the air; else, hardware will send an unencrypted Empty-Enh-Ack packet with the correct MHR, but no IEs and Frame Payload. At the same time, the hardware will generate an EMPTY_ACK_IRQ if enabled.
- While the hardware is sending Full-Enh-Ack's Length Field, the software can continue to calculate and write the rest of the packet into RAM.
- Due to possible software and hardware race conditions, before sending the Header IE field(HIE), the hardware has a checkpoint. If the HIE exists in the Enh-Ack packet, after writing the HIE into the RAM, the software needs to set the register SW_HIE_RDY to 1. This SW_HIE_RDY is used by the hardware to check whether the software is capable of completing the HIE and other reset Fields.
- The register SW_MHR_LENGTH is used by the software to tell the hardware the specific location of the HIE in the packet.
- If SW_HIE_RDY is checked as 1, the hardware continues to read bytes from RAM and send them to air; Otherwise, hardware aborts the sequence and generates an interrupt ACK_ABORT_IRQ if enabled.
- When the last byte of the Full-Enh-Ack or Empty-Enh-Ack is sent according to the length specified by the Length field, the sequence completes with the TXIRQ set.

Just like the usage of RXACKRQD bit in Imm-Ack, it can also be used in Enh-Ack. If RXACKRQD=1, then Sequence TR is executed as a Sequence T followed by a Receive-Acknowledge-Only frame. This type of receive operation is special, and not the same as a standalone Sequence R. The Ack-only receive operation filters all incoming frames, looking only for an Enh-Ack frame if the frame version is 2. And the Enh-Ack frame should pass the following checks which is also shown in Figure 404.

- Frame Control field check (can be overridden by leniency[41])
  - The received PAN ID Compression matches the transmitted PAN ID Compression
  - The received Sequence Number Suppression matches the transmitted Sequence Number Suppression
  - The received Destination Address Mode matches the transmitted Source Address Mode
- Address Field check (can be overridden by leniency[42])
  - The Destination Address field, when present, shall contain the value of the Source Address field from the frame that is being acknowledged.
  - The Destination PAN ID, when present, contains the value from the Source PAN ID field from the frame that is being acknowledged
- Sequence Number Check(same as Imm-Ack, can be overridden by leniency[9])

All non-matching frames are discarded, and after each non-matching frame, the sequence manager will return the receiver to preamble-detect mode. This receive operation will continue until the matching Enh-Ack frame is received. Unlike Imm-Ack, the hardware will put the entire Enh-Ack frame in RAM. Because Enh-Ack may contain other information required by the software.

55.4.9.1.3 Memory Map and Registers

55.4.9.1.3.1 ZLL register descriptions

55.4.9.1.3.1.1 ZLL memory map

ZLL base address: 48A0_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	INTERRUPT REQUEST STATUS (IRQSTS)	32	RW	See section
4h	PHY CONTROL (PHY_CTRL)	32	RW	0807_FF00h
8h	EVENT TIMER (EVENT_TMR)	32	RW	See section
Ch	TIMESTAMP (TIMESTAMP)	32	R	See section
10h	T1 COMPARE (T1CMP)	32	RW	00FF_FFFFh
14h	T2 COMPARE (T2CMP)	32	RW	00FF_FFFFh
18h	T2 PRIME COMPARE (T2PRIMECMP)	32	RW	0000_FFFFh
1Ch	T3 COMPARE (T3CMP)	32	RW	00FF_FFFFh
20h	T4 COMPARE (T4CMP)	32	RW	00FF_FFFFh
24h	PA POWER (PA_PWR)	32	RW	0000_0000h
28h	CHANNEL NUMBER 0 (CHANNEL_NUM0)	32	RW	0000_0012h
2Ch	LQI AND RSSI (LQI_AND_RSSI)	32	R	See section
30h	MAC SHORT ADDRESS 0 (MACSHORTADDRS0)	32	RW	FFFF_FFFFh
34h	MAC LONG ADDRESS 0 LSB (MACLONGADDRS0_LSB)	32	RW	FFFF_FFFFh
38h	MAC LONG ADDRESS 0 MSB (MACLONGADDRS0_MSB)	32	RW	FFFF_FFFFh
3Ch	RECEIVE FRAME FILTER (RX_FRAME_FILTER)	32	RW	See section
40h	CCA AND LQI CONTROL (CCA_LQI_CTRL)	32	RW	0866_004Bh
44h	CCA2 CONTROL (CCA2_CTRL)	32	RW	See section
4Ch	DSM CONTROL (DSM_CTRL)	32	RW	0000_0000h
54h	MAC SHORT ADDRESS FOR PAN1 (MACSHORTADDRS1)	32	RW	FFFF_FFFFh
58h	MAC LONG ADDRESS 1 LSB (MACLONGADDRS1_LSB)	32	RW	FFFF_FFFFh
5Ch	MAC LONG ADDRESS 1 MSB (MACLONGADDRS1_MSB)	32	RW	FFFF_FFFFh
60h	DUAL PAN CONTROL (DUAL_PAN_CTRL)	32	RW	See section
64h	CHANNEL NUMBER 1 (CHANNEL_NUM1)	32	RW	0000_007Fh
68h	SAM CONTROL (SAM_CTRL)	32	RW	8080_4000h
6Ch	SOURCE ADDRESS MANAGEMENT TABLE (SAM_TABLE)	32	RW	See section
70h	SOURCE ADDRESS MANAGEMENT MATCH (SAM_MATCH)	32	R	See section
74h	SAM FREE INDEX (SAM_FREE_IDX)	32	R	See section
78h	SEQUENCE CONTROL AND STATUS (SEQ_CTRL_STS)	32	RW	See section
7Ch	ACK DELAY (ACKDELAY)	32	RW	002D_0002h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
80h	FILTER FAIL CODE (FILTERFAIL_CODE)	32	RW	See section
84h	RECEIVE WATER MARK (RX_WTR_MARK)	32	RW	0000_00FFh
8Ch	SLOT PRELOAD (SLOT_PRELOAD)	32	RW	0000_0074h
90h	802.15.4 SEQUENCE STATE (SEQ_STATE)	32	R	See section
94h	TIMER PRESCALER (TMR_PRESCALE)	32	RW	0000_0005h
98h	LENIENCY LSB (LENIENCY_LSB)	32	RW	0000_0000h
9Ch	LENIENCY MSB (LENIENCY_MSB)	32	RW	0000_0000h
A0h	PART ID (PART_ID)	32	R	0000_0004h
A4h	COEXISTENCE CONTROL (COEX_CTRL)	32	RW	0000_0304h
A8h	COEXISTENCE PRIORITY (COEX_PRIORITY)	32	RW	0000_0000h
ACh	ENHACK_CTRL 0 (ENHACK_CTRL0)	32	RW	0000_FF00h

55.4.9.1.3.1.2 INTERRUPT REQUEST STATUS (IRQSTS)

Offset

Register	Offset
IRQSTS	0h

Function

The IRQSTS register indicates the status of interrupt requests in the 802.15.4 module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RX_FRAME_LENGTH							TMR4 MSK	TMR3 MSK	TMR2 MSK	TMR1 MSK	TMR4IRQ	TMR3IRQ	TMR2IRQ	TMR1IRQ
W													W1C	W1C	W1C	W1C
Reset	0	u	u	u	u	u	u	u	1	1	1	1	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CRCVALID	CCA	SRCADDR	PI	ENH_PKT...	TSM_IRQ	ARB_GRA...	WAKE_IRQ	RX_FRM_...	PLL_UNL...	FILTERF...	RXWTRMR...	CCAIRQ	RXIRQ	TXIRQ	SEQIRQ
W							W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 —	Reserved
30-24 RX_FRAME_LENGTH	Receive Frame Length Contents of the PHR (PHY header), or FrameLength field, of the most recently received packet. Read-only.
23 TMR4MSK	Timer Comparator 4 Interrupt Mask bit 0b - allows interrupt when comparator matches event timer count 1b - Interrupt generation is disabled, but a TMR4IRQ flag can be set
22 TMR3MSK	Timer Comparator 3 Interrupt Mask bit 0b - allows interrupt when comparator matches event timer count 1b - Interrupt generation is disabled, but a TMR3IRQ flag can be set
21 TMR2MSK	Timer Comparator 2 Interrupt Mask bit 0b - allows interrupt when comparator matches event timer count 1b - Interrupt generation is disabled, but a TMR2IRQ flag can be set
20 TMR1MSK	Timer Comparator 1 Interrupt Mask bit 0b - allows interrupt when comparator matches event timer count 1b - Interrupt generation is disabled, but a TMR1IRQ flag can be set
19 TMR4IRQ	Timer 4 IRQ Timer Comparator 4 Interrupt Status bit: Indicates T4CMP comparator value matched event timer counter. This is write '1' to clear bit
18 TMR3IRQ	Timer 3 IRQ Timer Comparator 3 Interrupt Status bit: Indicates T3CMP comparator value matched event timer counter. This is write '1' to clear bit
17 TMR2IRQ	Timer 2 IRQ Timer Comparator 2 Interrupt Status bit: Indicates comparator value matched event timer counter. This flag is shared between the T2CMP (24-bit) and T2PRIMECMP (16-bit) compare registers. This is write '1' to clear bit
16 TMR1IRQ	Timer 1 IRQ Timer Comparator 1 Interrupt Status bit: Indicates T1CMP comparator value matched event timer counter. This is write '1' to clear bit
15 CRCVALID	CRC Valid Status Code Redundancy Check Valid: This flag indicates the compare result between the FCS field, in the most-recently received frame, and the internally calculated CRC value. This flag is cleared at next receiver warm up.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Rx FCS != calculated CRC (incorrect) 1b - Rx FCS = calculated CRC (correct)
14 CCA	CCA Status Channel IDLE/BUSY indicator. This indicator is valid at CCAIRQ and also at SEQIRQ. This flag is cleared at next receiver warm up. 0b - IDLE 1b - BUSY
13 SRCADDR	Source Address Match Status If Source Address Management is engaged, meaning at least one of the following bits is set: SAP0_EN SAA0_EN SAP1_EN SAA1_EN Then SRCADDR will be set to 1 if the packet just received is a poll request (PI=1), *and* at least one of the following conditions is met: SAP0_EN and SAP0_ADDR_PRESENT SAA0_EN and SAA0_ADDR_ABSENT SAP1_EN and SAP1_ADDR_PRESENT SAA1_EN and SAA1_ADDR_ABSENT If SRCADDR=1, this indicates to SW that the Packet Processor has determined that an auto-TxACK frame must be transmitted with the FramePending subfield of the FrameControlField set to 1. HW will assemble and transmit this Ack packet. If the above conditions are not met, SRCADDR will be cleared to 0.
12 PI	Poll Indication 0b - the received packet was not a data request 1b - the received packet was a data request, regardless of whether a Source Address table match occurred, or whether Source Address Management is enabled or not
11 ENH_PKT_STATUS	Enhanced Packet Status 0b - The last packet received was neither 4e- nor 2015-compliant 1b - The last packet received was 4e- or 2015-compliant (RX_FRAME_FILTER register should be queried for additional status bits)
10 TSM_IRQ	TSM IRQ TSM Interrupt Request 0b - A TSM Interrupt has not occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - A TSM Interrupt has occurred
9 ARB_GRANT_DEASSERTION_IRQ	arb_grant Deassertion IRQ arb_grant(!RF_NOT_ALLOWED) Deassertion Interrupt Status bit. A '1' indicates an arb_grant deassertion event has occurred. This is write a '1' to clear bit. 0b - An arb_grant Deassertion Interrupt has not occurred 1b - An arb_grant Deassertion Interrupt has occurred
8 WAKE_IRQ	WAKE Interrupt Request Wake Interrupt. The RFMC TIMER has matched the WOR WKUP_TIME register, and DSM has exited. The 802.15.4 EVENT_TMR has resumed counting. 0b - A Wake Interrupt has not occurred 1b - A Wake Interrupt has occurred
7 RX_FRM_PEND	RX Frame Pending Status of the frame pending bit of the frame control field for the most-recently received packet. Read-only.
6 PLL_UNLOCK_IRQ	PLL Unlock IRQ PLL Un-lock Interrupt Status bit. A '1' indicates an unlock event has occurred in the PLL. This is write a '1' to clear bit. 0b - A PLL Unlock Interrupt has not occurred 1b - A PLL Unlock Interrupt has occurred
5 FILTERFAIL_IRQ	Filter Fail IRQ Receiver Packet Filter Fail Interrupt Status bit. A '1' indicates that the most-recently received packet has been rejected due to elements within the packet. This is write a '1' to clear bit. In Dual PAN mode, FILTERFAIL_IRQ applies to either or both networks, as follows: **A:** If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_IRQ applies to PAN0. **B:** If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_IRQ applies to PAN1. **C:** If PAN0 and PAN1 occupy the same channel, FILTERFAIL_IRQ is the logical 'AND' of the individual PANs' Filter Fail status. 0b - A Filter Fail Interrupt has not occurred 1b - A Filter Fail Interrupt has occurred
4 RXWTRMRKIRQ	Receive Watermark IRQ Receiver Byte Count Water Mark Interrupt Status bit. A '1' indicates that the number of bytes specified in the RX_WTR_MARK register has been reached. This is write a '1' to clear bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - A Receive Watermark Interrupt has not occurred 1b - A Receive Watermark Interrupt has occurred
3 CCAIRQ	CCA IRQ Clear Channel Assessment Interrupt Status bit. A '1' indicates completion of CCA operation. This is write '1' to clear bit. 0b - A CCA Interrupt has not occurred 1b - A CCA Interrupt has occurred
2 RXIRQ	RX IRQ Receiver Interrupt Status bit. A '1' indicates the completion of a receive operation. This is write a '1' to clear bit. 0b - A RX Interrupt has not occurred 1b - A RX Interrupt has occurred
1 TXIRQ	TX IRQ Transmitter Interrupt Status bit. A '1' indicates the completion of a transmit operation. This is write a '1' to clear bit. 0b - A TX Interrupt has not occurred 1b - A TX Interrupt has occurred
0 SEQIRQ	Sequencer IRQ Sequence-end Interrupt Status bit. A '1' indicates the completion of an autosequence. This interrupt will assert whenever the Sequence Manager transitions from non-idle to idle state, for any reason. This is write a '1' to clear bit. 0b - A Sequencer Interrupt has not occurred 1b - A Sequencer Interrupt has occurred

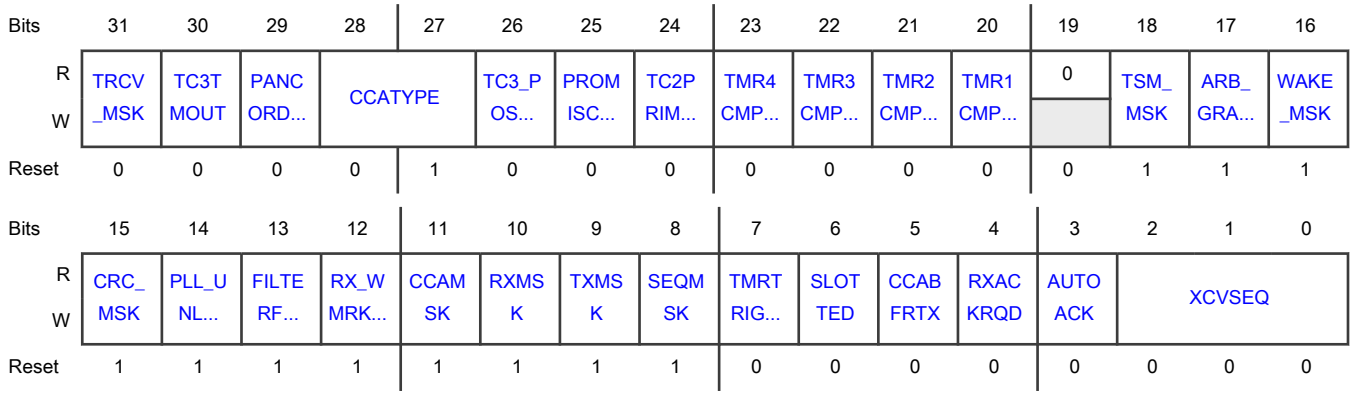
55.4.9.1.3.1.3 PHY CONTROL (PHY_CTRL)

Offset

Register	Offset
PHY_CTRL	4h

Function
PHY Control Register

Diagram



Fields

Field	Function
31 TRCV_MSK	Transceiver Global Interrupt Mask Transceiver Global Interrupt Mask 0b - Enable any unmasked interrupt source to assert zigbee interrupt 1b - Mask all interrupt sources from asserting zigbee interrupt
30 TC3TMOUT	TMR3 Timeout Enable TMR3 Timeout Enable 0b - TMR3 is a software timer only 1b - Enable TMR3 to abort Rx or CCCA operations.
29 PANCORDNTR0	Device is a PAN Coordinator on PAN0 Device is a PAN Coordinator on PAN0. Allows device to receive packets with no destination address, if Source PAN ID matches.
28-27 CCATYPE	Clear Channel Assessment Type Clear Channel Assessment Type. Selects one of four possible functions for CCA or Energy Detect, per below. 00b - ENERGY DETECT 01b - CCA MODE 1 10b - CCA MODE 2 11b - CCA MODE 3
26 TC3_POSTPONE_ON_SFD	Postpone TC3 Timeout On SFD Enable If this bit is set, if a TMR3 timeout occurs while a packet is being received (*rx_sfd_detect* = 1), the timeout-related abort will be deferred until the packet reception is completed. If the packet is good, HW will signal Data Indication; otherwise the 802.15.4 ZSM (Sequence Manager) will return to SEQ_IDLE and await further commanding.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - TC3 Abort will occur on TMR3 timeout, regardless of rx_sfd_detect 1b - TC3 Abort will be deferred on TMR3 timeout if rx_sfd_detect is asserted; otherwise the TC3 Abort will occur immediately
25 PROMISCUOUS	Promiscuous Mode Enable Bypasses most packet filtering. 0b - normal mode 1b - all packet filtering except frame length checking (FrameLength>=5 and FrameLength<=127) is bypassed.
24 TC2PRIME_EN	Timer 2 Prime Compare Enable 0b - Don't allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ 1b - Allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ
23 TMR4CMP_EN	Timer 4 Compare Enable 0b - Don't allow an Event Timer Match to T4CMP to set TMR4IRQ 1b - Allow an Event Timer Match to T4CMP to set TMR4IRQ
22 TMR3CMP_EN	Timer 3 Compare Enable 0b - Don't allow an Event Timer Match to T3CMP to set TMR3IRQ 1b - Allow an Event Timer Match to T3CMP to set TMR3IRQ
21 TMR2CMP_EN	Timer 2 Compare Enable 0b - Don't allow an Event Timer Match to T2CMP or T2PRIMECMP to set TMR2IRQ 1b - Allow an Event Timer Match to T2CMP or T2PRIMECMP to set TMR2IRQ
20 TMR1CMP_EN	Timer 1 Compare Enable 0b - Don't allow an Event Timer Match to T1CMP to set TMR1IRQ 1b - Allow an Event Timer Match to T1CMP to set TMR1IRQ
19 —	Reserved
18 TSM_MSK	Mask generating interrupt from TSM 0b - allows assertion of a TSM interrupt to generate a zigbee interrupt 1b - Assertion of a TSM interrupt will set the TSM_IRQ status bit, but a zigbee interrupt is not generated
17 ARB_GRANT_DEASSERTION_MSK	arb_grant Deassertion Interrupt Mask 0b - allows arb_grant deassertion event to generate a zigbee interrupt 1b - An arb_grant deassertion event will set the ARB_GRANT_DEASSERTION_IRQ status bit, but a zigbee interrupt is not generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 WAKE_MSK	Mask wakeup from DSM 0b - Allows a wakeup from DSM to generate a zigbee interrupt 1b - Wakeup from DSM will set the WAKE_IRQ status bit, but a zigbee interrupt is not generated
15 CRC_MSK	CRC Mask CRC Mask 0b - sequence manager ignores CRCVALID and considers the receive operation complete after the last octet of the frame has been received. 1b - sequence manager requires CRCVALID=1 at the end of the received frame in order for the receive operation to complete successfully; if CRCVALID=0, sequence manager will return to preamble-detect mode after the last octet of the frame has been received.
14 PLL_UNLOCK_MSK	PLL Unlock Interrupt Mask 0b - allows PLL unlock event to generate a zigbee interrupt 1b - A PLL unlock event will set the PLL_UNLOCK_IRQ status bit, but a zigbee interrupt is not generated
13 FILTERFAIL_MSK	FilterFail Interrupt Mask FilterFail Interrupt Mask 0b - allows Packet Processor Filtering Failure to generate a zigbee interrupt 1b - A Packet Processor Filtering Failure will set the FILTERFAIL_IRQ status bit, but a zigbee interrupt is not generated
12 RX_WMRK_MSK	RX Watermark Interrupt Mask RX Watermark Interrupt Mask 0b - allows a Received Byte Count match to the RX_WTR_MARK threshold register to generate a zigbee interrupt 1b - A Received Byte Count match to the RX_WTR_MARK threshold register will set the RXWTRMRKIRQ status bit, but a zigbee interrupt is not generated
11 CCAMSK	CCA Interrupt Mask CCA Interrupt Mask 0b - allows completion of a CCA operation to generate a zigbee interrupt 1b - Completion of a CCA operation will set the CCA status bit, but a zigbee interrupt is not generated
10 RXMSK	RX Interrupt Mask 0b - allows completion of a RX operation to generate a zigbee interrupt 1b - Completion of a RX operation will set the RXIRQ status bit, but a zigbee interrupt is not generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 TXMSK	TX Interrupt Mask 0b - allows completion of a TX operation to generate a zigbee interrupt 1b - Completion of a TX operation will set the TXIRQ status bit, but a zigbee interrupt is not generated
8 SEQMSK	Sequencer Interrupt Mask 0b - allows completion of an autosequence to generate a zigbee interrupt 1b - Completion of an autosequence will set the SEQIRQ status bit, but a zigbee interrupt is not generated
7 TMRTRIGEN	Timer2 Trigger Enable 0b - programmed sequence initiates immediately upon write to XCVSEQ. 1b - allow timer TC2 (or TC2') to initiate a preprogrammed sequence (see XCVSEQ register).
6 SLOTTED	Slotted Mode Slotted Mode, for beacon-enabled networks. Applies only to Sequences T, TR, and R, ignored during all other sequences. Used, in concert with CCABFRTX, to determine how many CCA measurements are required prior to a transmit operation. Also used during R sequence to determine whether the ensuing transmit acknowledge frame (if any) needs to be synchronized to a backoff slot boundary.
5 CCABFRTX	CCA Before TX Applies only to Sequences T and TR, ignored during all other sequences. 0b - no CCA required, transmit operation begins immediately. 1b - at least one CCA measurement is required prior to the transmit operation (see also SLOTTED).
4 RXACKRQD	Receive Acknowledge Frame required Applies only to Sequence TR, ignored during all other sequences. 0b - An ordinary receive frame (any type of frame) follows the transmit frame. 1b - A receive Ack frame is expected to follow the transmit frame (non-Ack frames are rejected).
3 AUTOACK	Auto Acknowledge Enable Applies only to Sequence R and Sequence TR, ignored during other sequences 0b - sequence manager will not follow a receive frame with a Tx Ack frame, under any conditions; the autosequence will terminate after the receive frame. 1b - sequence manager will follow a receive frame with an automatic hardware-generated Tx Ack frame, assuming other necessary conditions are met.
2-0 XCVSEQ	802.15.4 Transceiver Sequence Selector The Transceiver Sequence Selector register selects an autosequence for the sequence manager to execute. Sequence initiation can be immediate, or scheduled (see TMRTRIGEN). A write of XCVSEQ=IDLE will abort any ongoing sequence. A write of XCVSEQ=IDLE must always be performed

Table continues on the next page...

Table continued from the previous page...

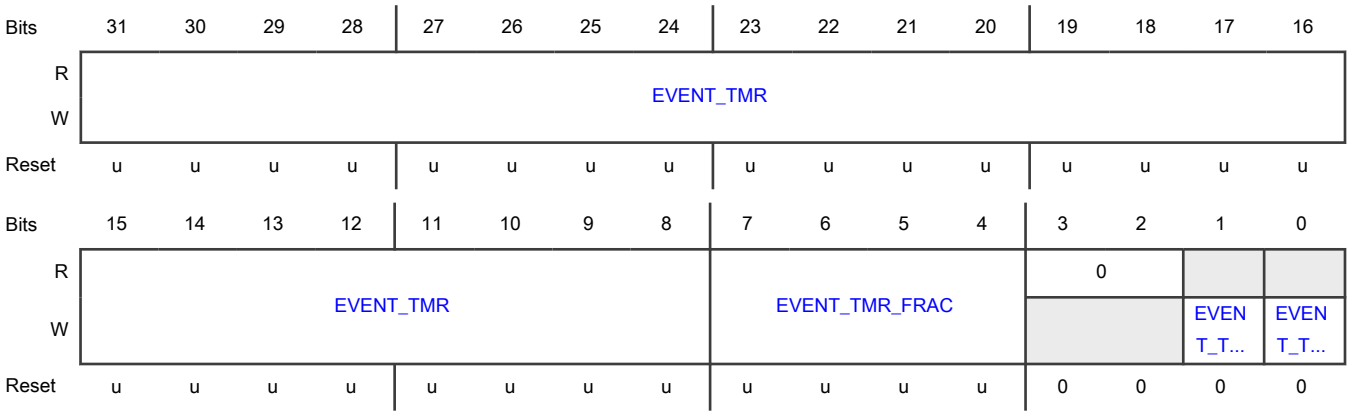
Field	Function
	after a sequence is complete, and before a new sequence is programmed. Any write to XCVSEQ other than XCVSEQ=IDLE during an ongoing sequence, shall be ignored. The mapping of XCVSEQ to sequence types is as follows: 000b - I (IDLE) 001b - R (RECEIVE) 010b - T (TRANSMIT) 011b - C (CCA) 100b - TR (TRANSMIT/RECEIVE) 101b - CCCA (CONTINUOUS CCA) 110b - Reserved 111b - Reserved

55.4.9.1.3.1.4 EVENT TIMER (EVENT_TMR)

Offset

Register	Offset
EVENT_TMR	8h

Diagram



Fields

Field	Function
31-8 EVENT_TMR	Event Timer Integer Component

Table continues on the next page...

Table continued from the previous page...

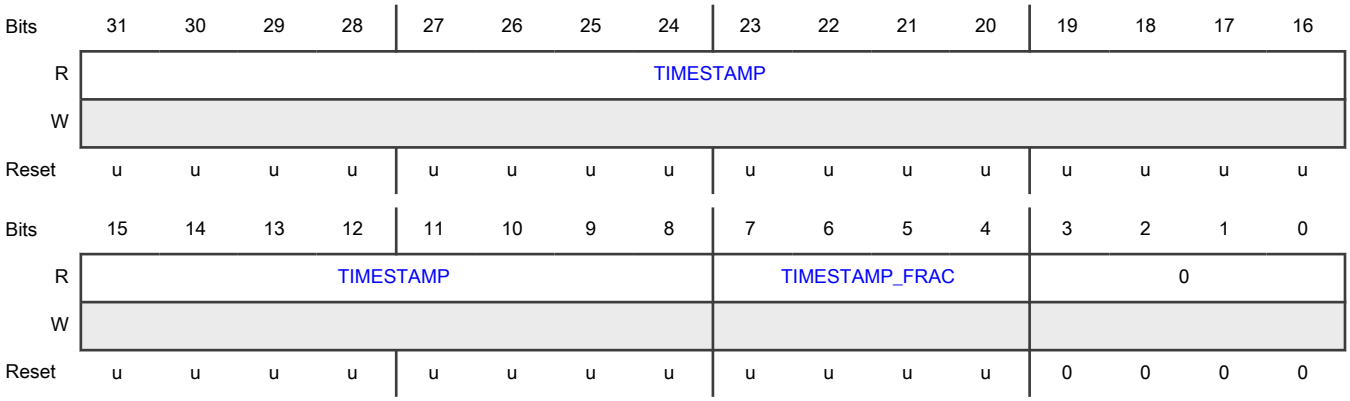
Field	Function
	The EVENT_TMR represents a 28-bit number with 24 Integer bits and 4 Fractional bits. This field is the integer component, which appears to the left of the decimal point. Most 802.15.4 operations which relate to the Event Timer require only the integer component.
7-4 EVENT_TMR_FRAC	Event Timer Fractional Component The EVENT_TMR represents a 28-bit number with 24 Integer bits and 4 Fractional bits. This field is the fractional component, which appears to the right of the decimal point. The fractional component can be ignored for most 802.15.4 operations which relate to the Event Timer; it has been provided only to allow for microsecond accuracy when software updates the Event Timer after a long period of DSM (sleep) mode
3-2 —	Reserved
1 EVENT_TMR_ADD	Event Timer Add Enable Increments the Event Timer by the values written to the EVENT_TMR[23:0] and EVENT_TMR_FRAC[3:0] fields of this register.
0 EVENT_TMR_LD	Event Timer Load Enable Loads the Event Timer with the values written to the EVENT_TMR[23:0] and EVENT_TMR_FRAC[3:0] fields of this register.

55.4.9.1.3.1.5 TIMESTAMP (TIMESTAMP)

Offset

Register	Offset
TIMESTAMP	Ch

Diagram



Fields

Field	Function
31-8 TIMESTAMP	Timestamp Holds the latched value of the integer portion of the Event Timer current time corresponding to the beginning of the most recently received packet, at the point of SFD detection
7-4 TIMESTAMP_FRAC	Timestamp Fractional Holds the latched value of the fractional portion of the Event Timer current time corresponding to the beginning of the most recently received packet, at the point of SFD detection
3-0 —	Reserved

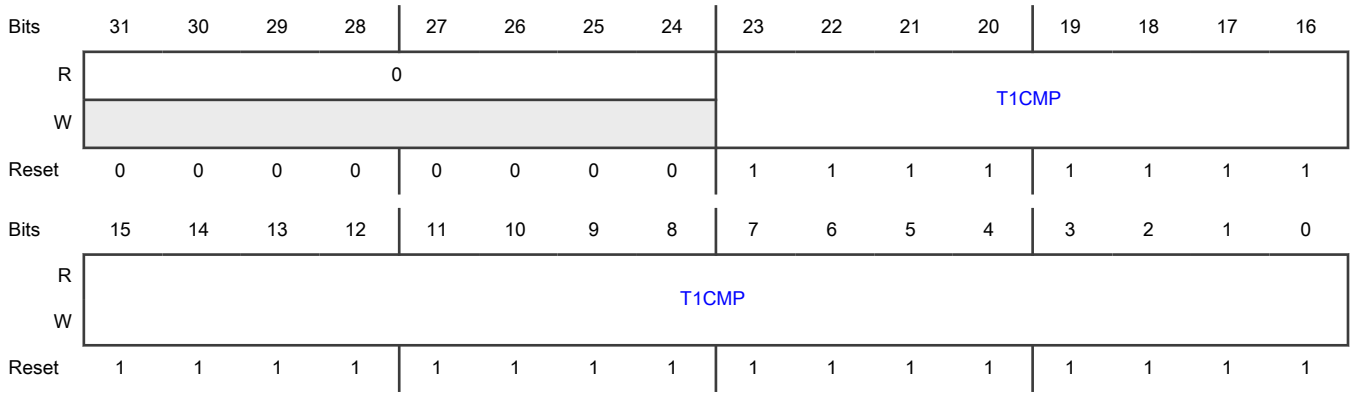
55.4.9.1.3.1.6 T1 COMPARE (T1CMP)

Offset

Register	Offset
T1CMP	10h

Function
TMR1 Compare Value

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0	TMR1 Compare Value

Table continues on the next page...

Table continued from the previous page...

Field	Function
T1CMP	TMR1 compare value. If TMR1CMP_EN=1 and the Event Timer matches this value, TMR1IRQ is set.

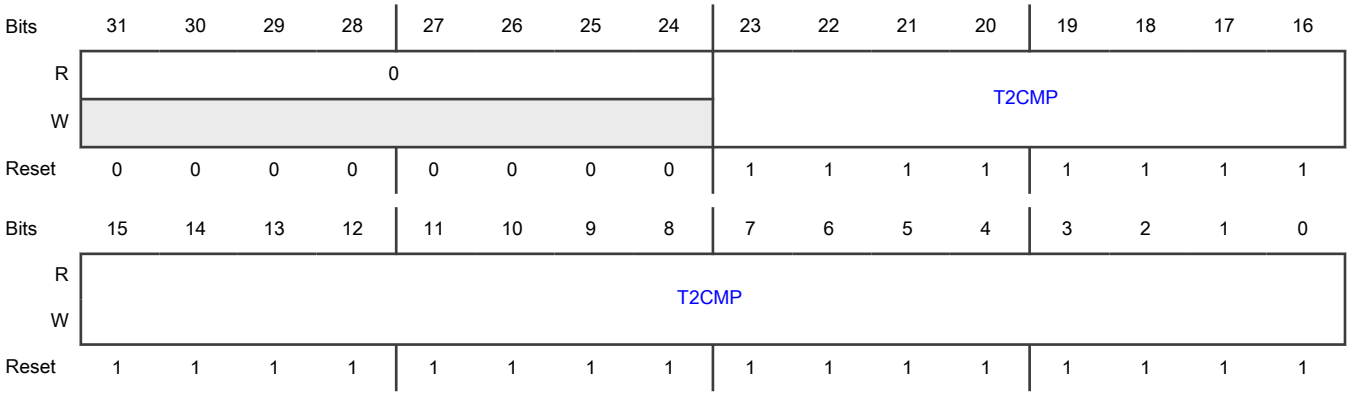
55.4.9.1.3.1.7 T2 COMPARE (T2CMP)

Offset

Register	Offset
T2CMP	14h

Function
TMR2 Compare Value

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 T2CMP	TMR2 Compare Value TMR2 compare value. If TMR2CMP_EN=1 and TC2PRIME_EN=0 and the Event Timer matches this value, TMR2IRQ is set.

55.4.9.1.3.1.8 T2 PRIME COMPARE (T2PRIMECMP)

Offset

Register	Offset
T2PRIMECMP	18h

Function

TMR2 Prime Compare Value

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T2PRIMECMP															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-16 —	Reserved
15-0 T2PRIMECMP	TMR2 Prime Compare Value TMR2 compare value. If TMR2CMP_EN=1 and TC2PRIME_EN=1 and the Event Timer matches this value, TMR2IRQ is set.

55.4.9.1.3.1.9 T3 COMPARE (T3CMP)

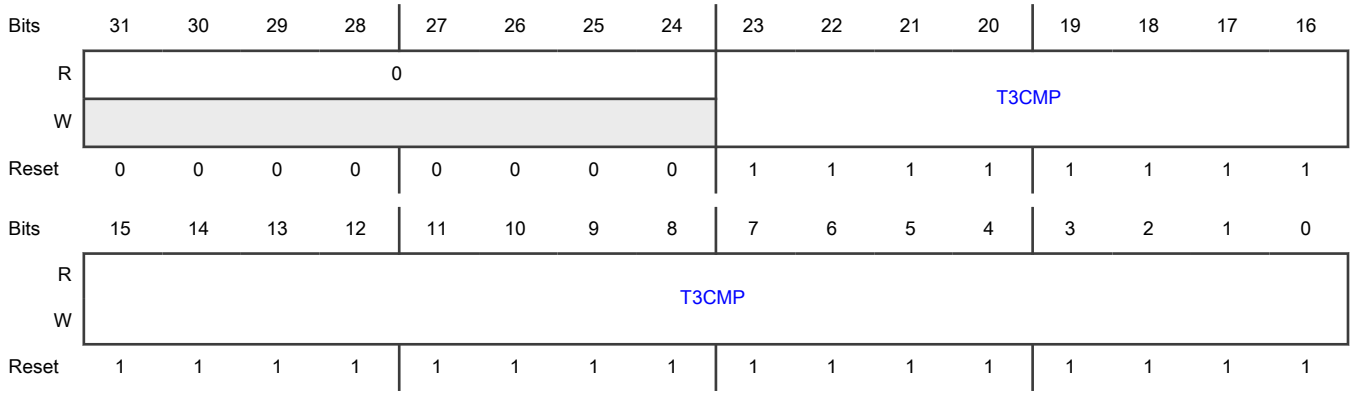
Offset

Register	Offset
T3CMP	1Ch

Function

TMR3 Compare Value

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 T3CMP	TMR3 Compare Value TMR3 compare value. If TMR3CMP_EN=1 and the Event Timer matches this value, TMR3IRQ is set.

55.4.9.1.3.1.10 T4 COMPARE (T4CMP)

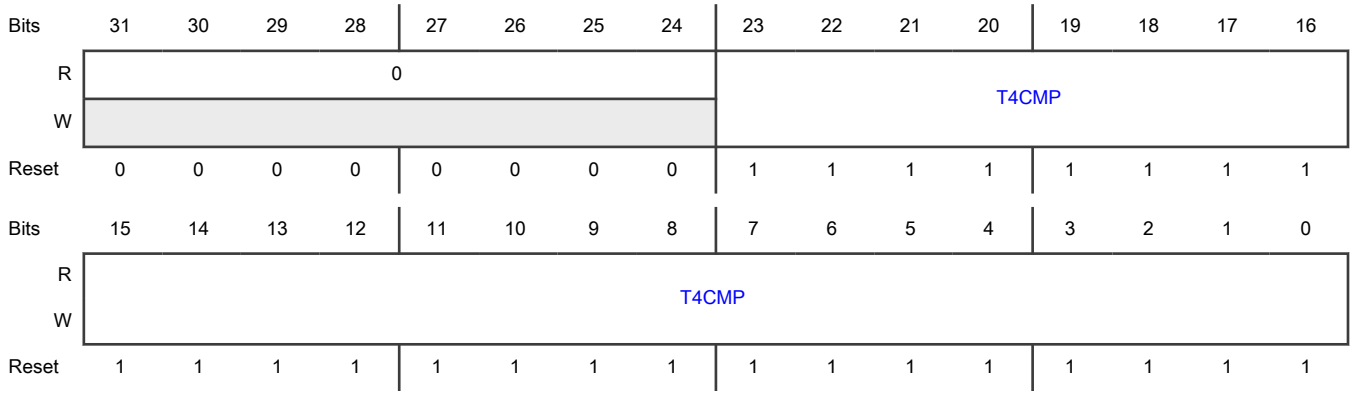
Offset

Register	Offset
T4CMP	20h

Function

TMR4 Compare Value

Diagram



Fields

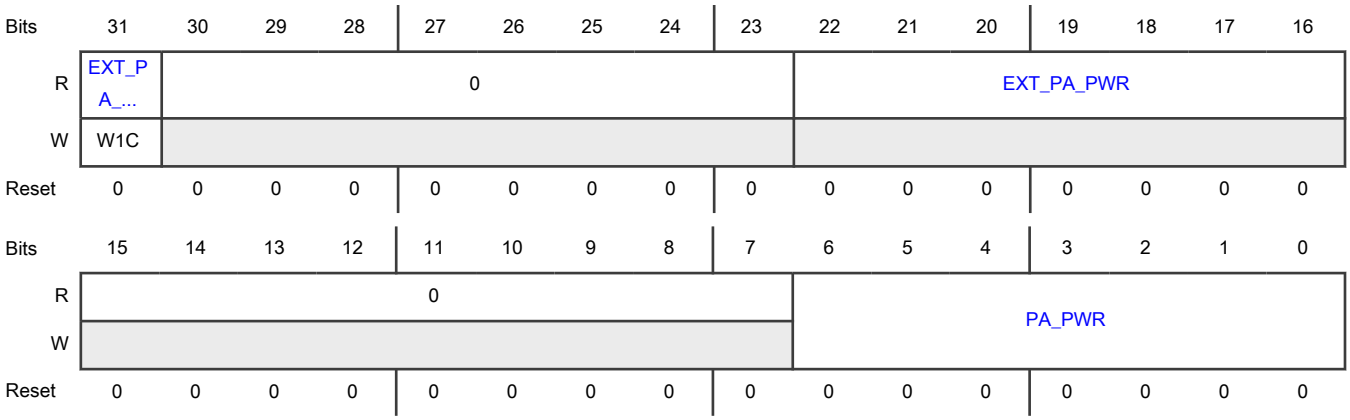
Field	Function
31-24 —	Reserved
23-0 T4CMP	TMR4 Compare Value TMR4 compare value. If TMR4CMP_EN=1 and the Event Timer matches this value, TMR4IRQ is set.

55.4.9.1.3.1.11 PA POWER (PA_PWR)

Offset

Register	Offset
PA_PWR	24h

Diagram



Fields

Field	Function
31 EXT_PA_PWR_CHG	External PA Power Change Flag THis flag is asserted when zfdbk_txpower_ind asserts during transmitting and grant has been asserted. This is write a '1' to clear bit.
30-23 —	Reserved
22-16 EXT_PA_PWR	External PA Power When transmitting and grant has been asserted, if zfdbk_txpower_ind asserts, then zfdbk_tx_power[5:0] is assigned to PA_PWR[6:1] and the status bit EXT_PA_PWR_CHG is set for software to check later. The value recorded in EXT_PA_PWR[6:0] is {zfdbk_tx_power[5:0],PA_PWR[0]} for debug purposes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-7 —	Reserved
6-0 PA_PWR	PA Power

55.4.9.1.3.1.12 CHANNEL NUMBER 0 (CHANNEL_NUM0)

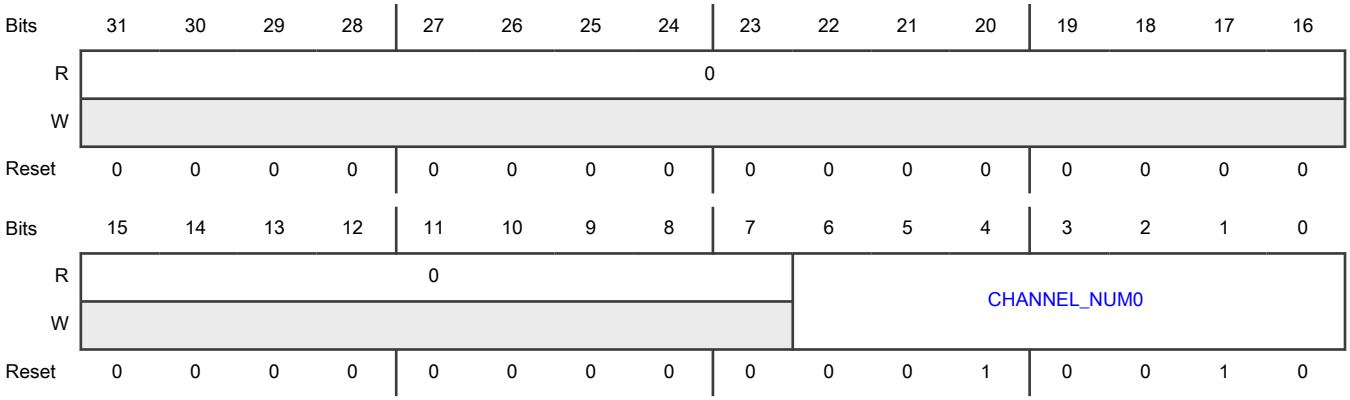
Offset

Register	Offset
CHANNEL_NUM0	28h

Function

Channel Number for PAN0

Diagram



Fields

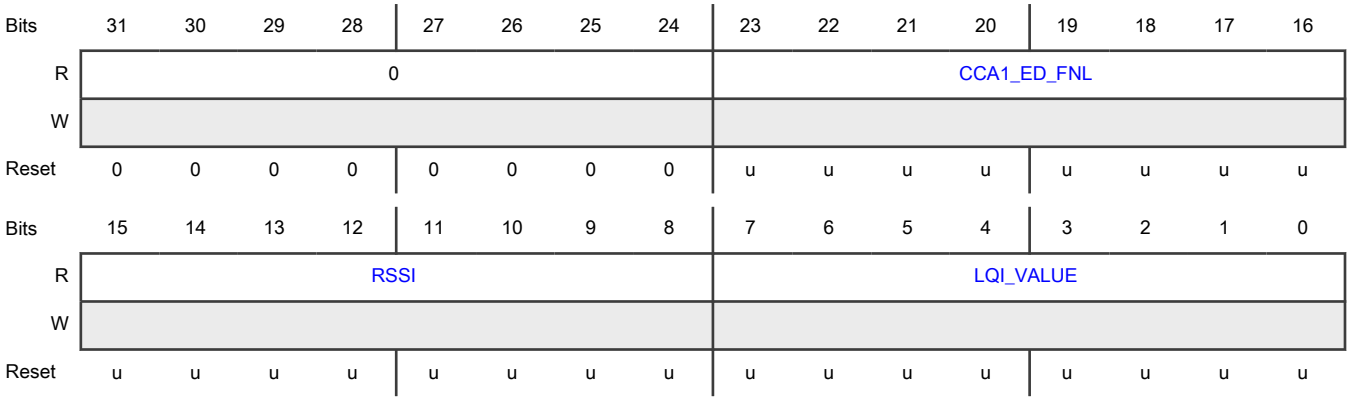
Field	Function
31-7 —	Reserved
6-0 CHANNEL_NUM0	Channel Number for PAN0 This is the mapped channel number used to transmit and receive Zigbee packets. If Dual PAN is engaged, this register applies to PAN0. CHANNEL_NUM0 should be in the range: 11 <= CHANNEL_NUM0 <= 26

55.4.9.1.3.1.13 LQI AND RSSI (LQI_AND_RSSI)

Offset

Register	Offset
LQI_AND_RSSI	2Ch

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 CCA1_ED_FNL	Final Result for CCA Mode 1 and Energy Detect Output register to show final averaged RSSI value or compensated value of the same at the end of a CCA Mode1 or Energy Detect computation.
15-8 RSSI	RSSI Value Received Signal Strength Indicator, in dBm
7-0 LQI_VALUE	LQI Value Link Quality Indicator for the most recently received packet. (LQI is also available in the Packet Buffer, at the end of the received packet data)

55.4.9.1.3.1.14 MAC SHORT ADDRESS 0 (MACSHORTADDRS0)

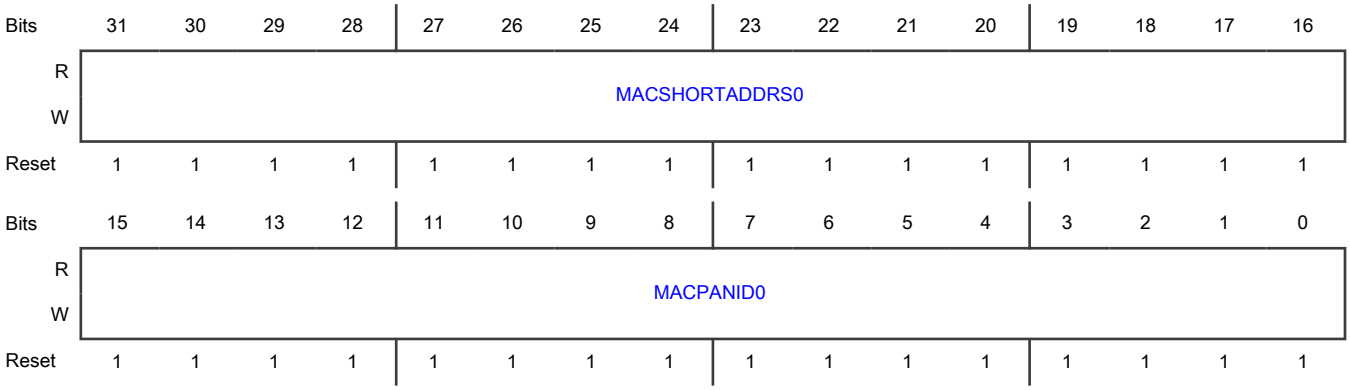
Offset

Register	Offset
MACSHORTADDRS0	30h

Function

MAC SHORT ADDRESS for PAN0

Diagram



Fields

Field	Function
31-16 MACSHORTADDRS0	MAC SHORT ADDRESS FOR PAN0 MAC Short Address for PAN0, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
15-0 MACPANID0	MAC PAN ID for PAN0 MAC PAN ID for PAN0. The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet Source PAN ID against this register. Also, if PANCORDNTR0=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet Source PAN ID against this register.

55.4.9.1.3.1.15 MAC LONG ADDRESS 0 LSB (MACLONGADDRS0_LSB)

Offset

Register	Offset
MACLONGADDRS0_LSB	34h

Function

MAC LONG ADDRESS for PAN0 LSB

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MACLONGADDRS0_LSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACLONGADDRS0_LSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 MACLONGADDRS0_LSB	MAC LONG ADDRESS for PAN0 LSB MAC Long Address for PAN0, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.1.3.1.16 MAC LONG ADDRESS 0 MSB (MACLONGADDRS0_MSB)

Offset

Register	Offset
MACLONGADDRS0_MSB	38h

Function
MAC LONG ADDRESS for PAN0 MSB

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MACLONGADDRS0_MSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACLONGADDRS0_MSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 MACLONGADDRS0_MSB	MAC LONG ADDRESS for PAN0 MSB MAC Long Address for PAN0, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.1.3.1.17 RECEIVE FRAME FILTER (RX_FRAME_FILTER)

Offset

Register	Offset
RX_FRAME_FILTER	3Ch

Function
RECEIVE FRAME FILTER

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								EXTENDE...	0	MULTIPU...	LLDN_RE...	FV2_CMD...	FV2_ACK...	FV2_DAT...	FV2_BEA...
W																
Reset	0	0	0	0	0	0	0	0	u	0	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXTENDE...	ACTIVE_...	0		FRM_VER_FILTER				EXTENDE...	NS_FT	MULTIPU...	LLDN_FT	CMD_FT	ACK_FT	DATA_FT	BEACON_...
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1

Fields

Field	Function
31-24 —	Reserved
23 EXTENDED_RECD	Extended Packet Received 0b - The last packet received was not Frame Type EXTENDED 1b - The last packet received was Frame Type EXTENDED, and EXTENDED_FT=1 to allow such packets.
22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 MULTIPURPOSE_RECD	Multipurpose Packet Received 0b - last packet received was not Frame Type MULTIPURPOSE 1b - The last packet received was Frame Type MULTIPURPOSE, and MULTIPURPOSE_FT=1 to allow such packets.
20 LLDN_RECD	LLDN Packet Received 0b - The last packet received was not Frame Type LLDN 1b - The last packet received was Frame Type LLDN, and LLDN_FT=1 to allow such packets.
19 FV2_CMD_RECD	Frame Version 2 MAC Command Packet Received 0b - The last packet received was not Frame Type MAC Command with Frame Version 2 1b - The last packet received was Frame Type MAC Command with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
18 FV2_ACK_RECD	Frame Version 2 Acknowledge Packet Received 0b - The last packet received was not Frame Type Ack with Frame Version 2 1b - The last packet received was Frame Type Ack with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
17 FV2_DATA_RECD	Frame Version 2 Data Packet Received 0b - The last packet received was not Frame Type Data with Frame Version 2 1b - The last packet received was Frame Type Data with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
16 FV2_BEACON_RECD	Frame Version 2 Beacon Packet Received 0b - The last packet received was not Frame Type Beacon with Frame Version 2 1b - The last packet received was Frame Type Beacon with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
15 EXTENDED_FCS_CHK	Verify FCS on Frame Type Extended 0b - Packet Processor will not check FCS for Frame Type EXTENDED (default) 1b - Packet Processor will check FCS at end-of-packet based on packet length derived from PHR, for Frame Type EXTENDED
14 ACTIVE_PROMISCUOUS	Active Promiscuous 0b - normal operation 1b - Provide Data Indication on all received packets under the same rules which apply in PROMISCUOUS mode, however acknowledge those packets under rules which apply in non-PROMISCUOUS mode
13-12	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
11-8 FRM_VER_FILTER	Frame Version selector. The incoming packet's Frame Control Field is parsed to obtain the FrameVersion subfield, and that value is compared against this register, in accordance with the following: xxx1: Accept received packets with FrameVersion=00 xx1x: Accept received packets with FrameVersion=01 x1xx: Accept received packets with FrameVersion=10 1xxx: Accept received packets with FrameVersion=11 These filtering rules apply to Beacon, Acknowledge, Data, and MAC Command Frame Types, since these frame types require a 2-octet Frame Control Field which embeds a 2-bit FrameVersion subfield. Later frame types introduced in the 802.15.4e addendum (LLDN, Multipurpose) don't guarantee a FrameVersion subfield with the original meaning, so these filtering rules do not apply to these frame types. See registers LLDN_FT and MULTIPURPOSE_FT. For Acknowledge frames, FrameVersion bits are ignored by the Packet Processor, irrespective of FRM_VER_FILTER.
7 EXTENDED_FT	Extended Frame Type Enable 0b - reject all Extended frames 1b - Extended frame type enabled (Frame Type 7).
6 NS_FT	"Not Specified" Frame Type Enable This bit enables reception of all Frame Types not covered by the other _FT bits in this register. 0b - reject all "Not Specified" frames 1b - Not-specified (reserved) frame type enabled. Applies to Frame Type 6. No packet filtering is performed, except for frame length checking (FrameLength>=5 and FrameLength<=127). No AUTOACK is transmitted for this Frame Type
5 MULTIPURPOSE_FT	Multipurpose Frame Type Enable 0b - reject all Multipurpose frames 1b - Multipurpose frame type enabled (Frame Type 5).
4 LLDN_FT	LLDN Frame Type Enable 0b - reject all LLDN frames 1b - LLDN frame type enabled (Frame Type 4).
3 CMD_FT	MAC Command Frame Type Enable 0b - reject all MAC Command frames 1b - MAC Command frame type enabled.
2	Ack Frame Type Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ACK_FT	0b - reject all Acknowledge frames 1b - Acknowledge frame type enabled.
1 DATA_FT	Data Frame Type Enable 0b - reject all Beacon frames 1b - Data frame type enabled.
0 BEACON_FT	Beacon Frame Type Enable 0b - reject all Beacon frames 1b - Beacon frame type enabled.

55.4.9.1.3.1.18 CCA AND LQI CONTROL (CCA_LQI_CTRL)

Offset

Register	Offset
CCA_LQI_CTRL	40h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CCA3_AN...	0		SIMUL_C...	LQI_OFFSET_COMP							
W																
Reset	0	0	0	0	1	0	0	0	0	1	1	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CCA1_THRESH							
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1

Fields

Field	Function
31-28 —	Reserved
27 CCA3_AND_NOT_OR	CCA Mode 3 AND not OR Determines the way CCA3 is required to be detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - CCA1 or CCA2 1b - CCA1 and CCA2
26-25 —	Reserved
24 SIMUL_CCA_RX	Simultaneous CCA and Receive Enable During autosequences of type Sequence T or Sequence TR with CCA before TX, this bit allows a packet to be received if preamble and SFD are detected during the CCA measurement window. 0b - Packets can't be received during CCA measurement 1b - Packet reception is enabled during CCA measurement if preamble and SFD are detected
23-16 LQI_OFFSET_COMP	LQI Offset Compensation Programmable amount to offset RSSI based LQI value
15-8 —	Reserved
7-0 CCA1_THRESH	CCA Mode 1 Threshold Programmable energy threshold register for CCA mode 1.

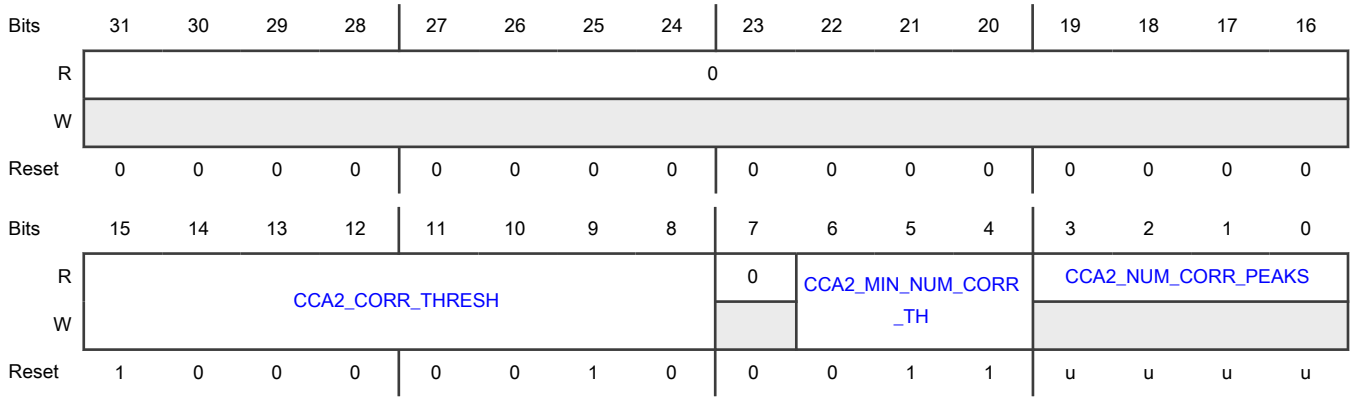
55.4.9.1.3.1.19 CCA2 CONTROL (CCA2_CTRL)

Offset

Register	Offset
CCA2_CTRL	44h

Function
CCA Mode 2 Control

Diagram



Fields

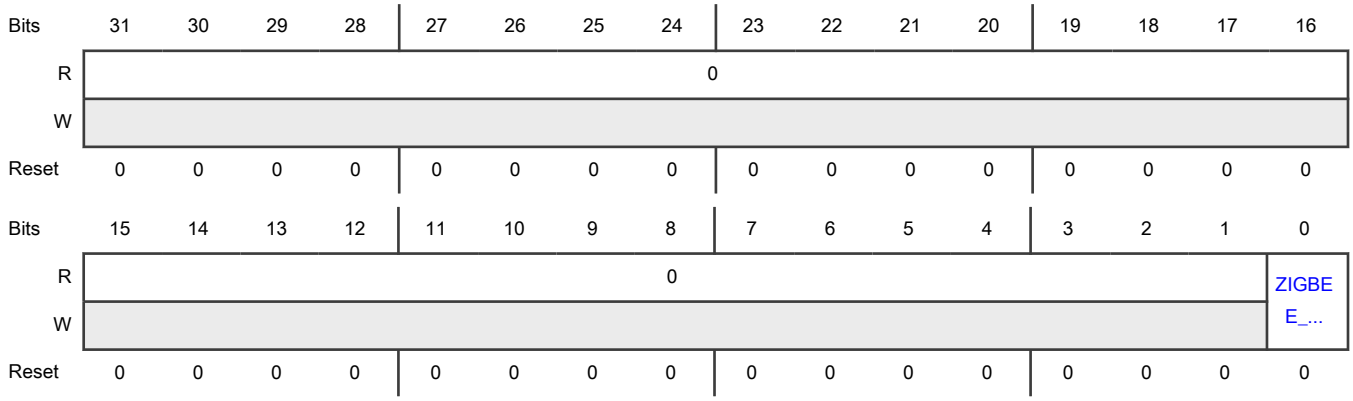
Field	Function
31-16 —	Reserved
15-8 CCA2_CORR_THRESH	CCA Mode 2 Correlation Threshold Correlator threshold used to qualify correlator peaks counted by CCA2_NUM_CORR_PEAKS. Correlator peaks exceeding this value increment CCA2_NUM_CORR_PEAKS.
7 —	Reserved
6-4 CCA2_MIN_NUM_CORR_TH	CCA Mode 2 Threshold Number of Correlation Peaks Programmable threshold to be compared against number of correlation peaks that exceeded CCA2_CORR_THRESH for detecting CCA mode 2. Number of peaks detected = CCA2_MIN_NUM_CORR_TH + 1; Example: If it is programmed to 3, CCA2 logic looks for at least 4 correlation peaks that crossed the threshold, to indicate channel is idle or busy.
3-0 CCA2_NUM_CORR_PEAKS	CCA Mode 2 Number of Correlation Peaks Detected Counts of number of peaks that crossed CCA2_CORR_THRESH in CCA Mode 2 operation

55.4.9.1.3.1.20 DSM CONTROL (DSM_CTRL)

Offset

Register	Offset
DSM_CTRL	4Ch

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ZIGBEE_SLEEP_REQUEST	802.15.4 Deep Sleep Mode Request for Manual DSM If 802.15.4 is assigned to Manual DSM (XCVR_CTRL[MAN_DSM_SEL] = 1), setting ZIGBEE_SLEEP_REQUEST to 1 enables Deep Sleep Mode (DSM) to be entered when the RFMC TIMER matches the WOR ENTER_TIME register.

55.4.9.1.3.1.21 MAC SHORT ADDRESS FOR PAN1 (MACSHORTADDRS1)

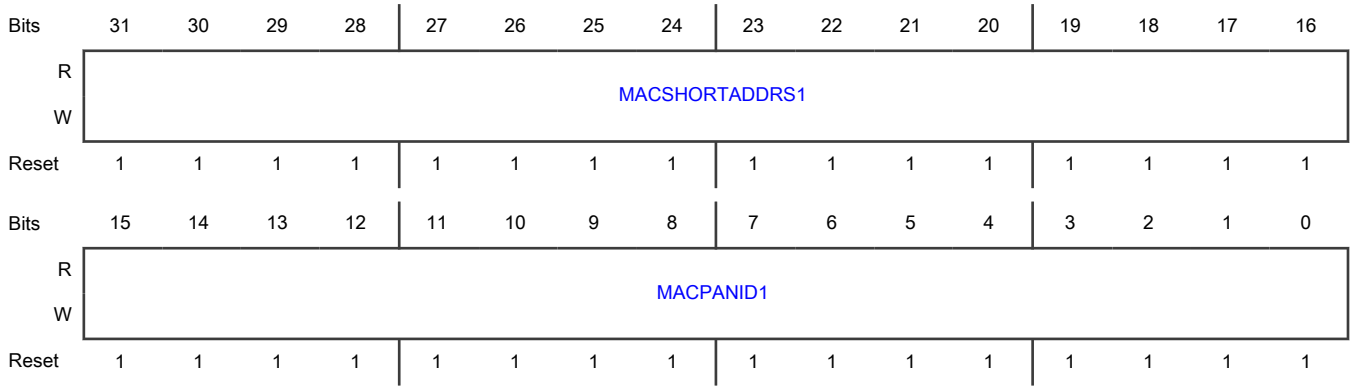
Offset

Register	Offset
MACSHORTADDRS1	54h

Function

The MACSHORTADDRS1 register .

Diagram



Fields

Field	Function
31-16 MACSHORTADDRS1	MAC SHORT ADDRESS for PAN1 MAC Short Address for PAN1, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
15-0 MACPANID1	MAC PAN ID for PAN1 MAC PAN ID for PAN1. The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet Source PAN ID against this register. Also, if PANCORDNTR1=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet Source PAN ID against this register.

55.4.9.1.3.1.22 MAC LONG ADDRESS 1 LSB (MACLONGADDRS1_LSB)

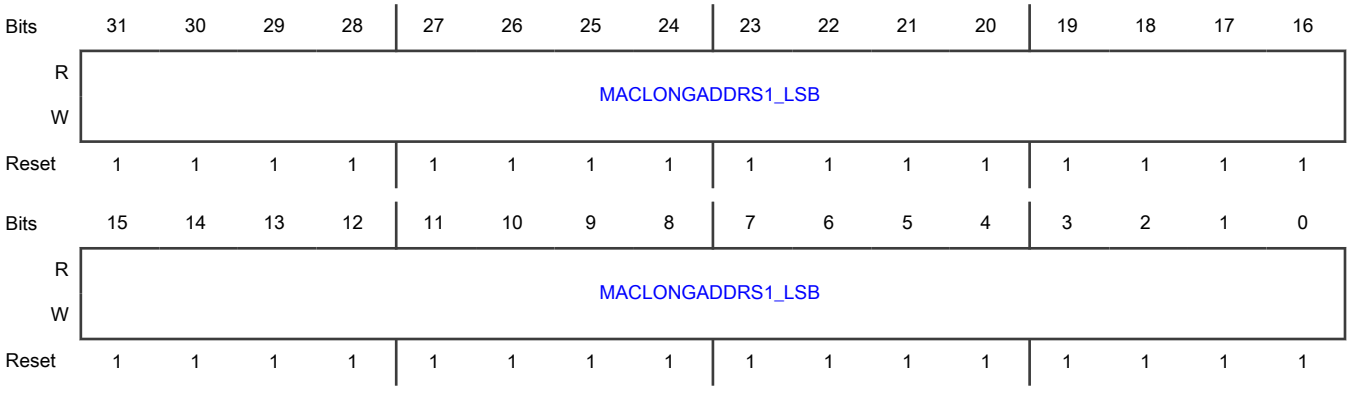
Offset

Register	Offset
MACLONGADDRS1_LSB	58h

Function

MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

Diagram



Fields

Field	Function
31-0	MAC LONG ADDRESS for PAN1 LSB

Table continues on the next page...

Field	Function
MACLONGADDRS1_LSB	MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.1.3.1.23 MAC LONG ADDRESS 1 MSB (MACLONGADDRS1_MSB)

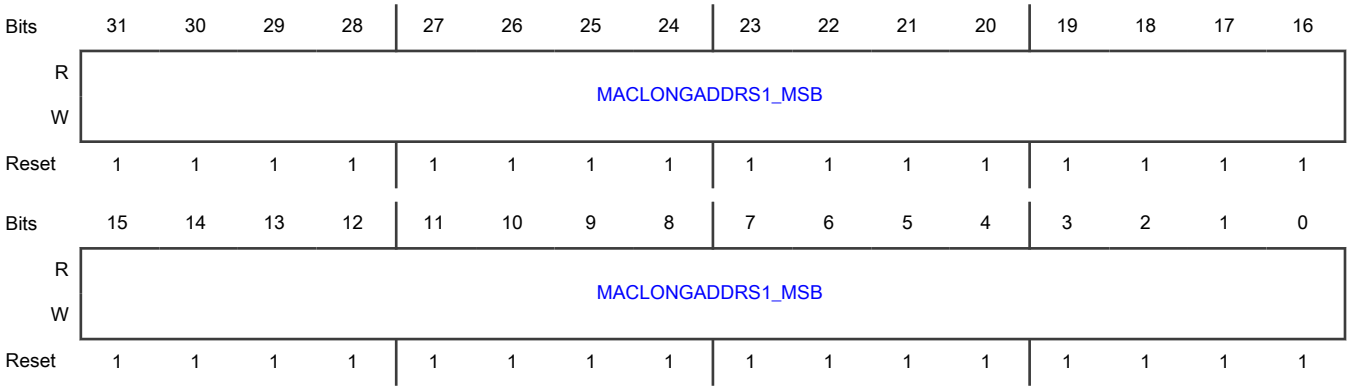
Offset

Register	Offset
MACLONGADDRS1_MSB	5Ch

Function

MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

Diagram



Fields

Field	Function
31-0 MACLONGADDRS1_MSB	MAC LONG ADDRESS for PAN1 MSB MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.1.3.1.24 DUAL PAN CONTROL (DUAL_PAN_CTRL)

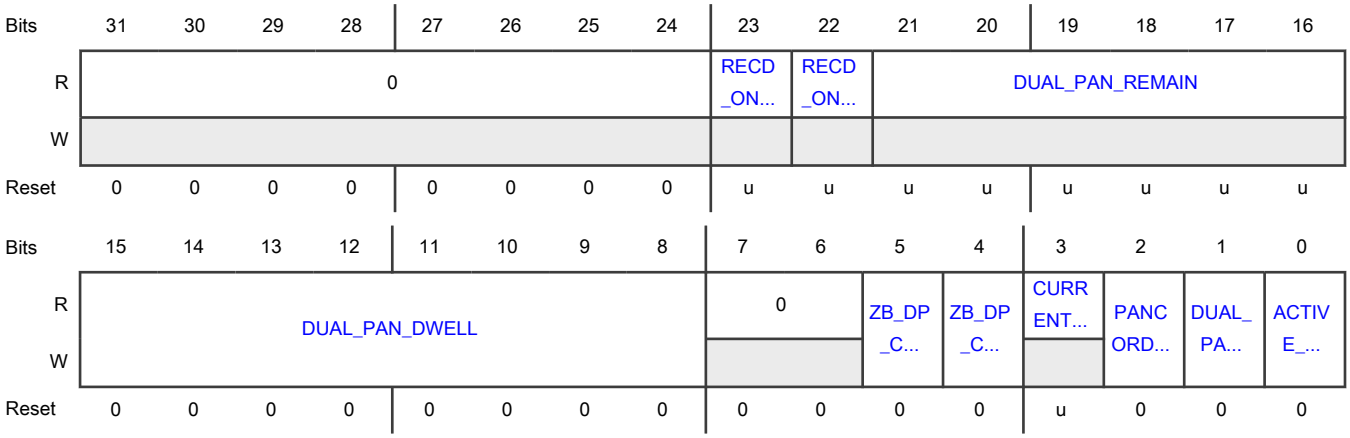
Offset

Register	Offset
DUAL_PAN_CTRL	60h

Function

Dual PAN Control Register

Diagram



Fields

Field	Function
31-24 —	Reserved
23 RECD_ON_PAN1	Last Packet was Received on PAN1 Indicates the packet which was just received, was received on PAN1. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN1 will be set if CURRENT_NETWORK=1 when the packet was received, regardless of FILTERFAIL status. In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN1 will be set only if a valid packet was received on PAN1 (PAN1's FILTERFAIL_FLAG is deasserted). RECD_ON_PAN1 remains valid until the start of the next autoseqeuence.
22 RECD_ON_PAN0	Last Packet was Received on PAN0 Indicates the packet which was just received, was received on PAN0. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN0 will be set if CURRENT_NETWORK=0 when the packet was received, regardless of FILTERFAIL status. In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN0 will be set only if a valid packet was received on PAN0 (PAN0's FILTERFAIL_FLAG is deasserted). RECD_ON_PAN0 remains valid until the start of the next autoseqeuence.
21-16 DUAL_PAN_REMAIN	Time Remaining before next PAN switch in auto Dual PAN mode This read-only register indicates time remaining before next PAN switch in auto Dual PAN mode. The units for this register, depend on the PRESCALER setting (bits [1:0]) in the DUAL_PAN_DWELL register. Tthe units are the same as the TIMEBASE determined by the prescaler, as per the following table: DUAL_PAN_DWELL[1:0]=0b00 (TIMEBASE=0.5ms) DUAL_PAN_REMAIN UNITS=0.5ms DUAL_PAN_DWELL[1:0]=0b01 (TIMEBASE=2.5ms) DUAL_PAN_REMAIN UNITS=2.5ms

Table continues on the next page...

Table continued from the previous page...

Field	Function
	DUAL_PAN_DWELL[1:0]=0b10 (TIMEBASE=10ms) DUAL_PAN_REMAIN UNITS=10ms DUAL_PAN_DWELL[1:0]=0b11 (TIMEBASE=50ms) DUAL_PAN_REMAIN UNITS=50ms The readback value indicates that between N-1 and N timebase units remain until the next PAN switch. For example, a DUAL_PAN_REMAIN readback value of 3, with a DUAL_PAN_DWELL PRESCALER setting of 2 (10ms), indicates that between 20ms (2*10ms) and 30ms (3*10ms), remain until the next automatic PAN switch.
15-8 DUAL_PAN_DWELL	Dual PAN Channel Frequency Dwell Time Channel Frequency Dwell Time. In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to DUAL_PAN_DWELL, always re-initializes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the sequence warmup. DUAL_PAN_DWELL[1:0] select the timebase for the dwell time. There are 4 options: DUAL_PAN_DWELL[1:0]=0b00 --> TIMEBASE=0.5ms DUAL_PAN_DWELL[1:0]=0b01 --> TIMEBASE=2.5ms DUAL_PAN_DWELL[1:0]=0b10 --> TIMEBASE=10ms DUAL_PAN_DWELL[1:0]=0b11 --> TIMEBASE=50ms DUAL_PAN_DWELL[7:2] select how many multiples of the time base to use, from 1 (DUAL_PAN_DWELL[7:2]=0) to 64 (DUAL_PAN_DWELL[7:2]=63). Given those definitions, the table below specifies the minimum and maximum dwell times for each TIMEBASE option: DUAL_PAN_DWELL[1:0]=0b00 --> MIN_DWELL=0.5ms MAX_DWELL=32ms DUAL_PAN_DWELL[1:0]=0b01 --> MIN_DWELL=2.5ms MAX_DWELL=160ms DUAL_PAN_DWELL[1:0]=0b10 --> MIN_DWELL=10ms MAX_DWELL=640ms DUAL_PAN_DWELL[1:0]=0b11 --> MIN_DWELL=50ms MAX_DWELL=3.2seconds A write to DUAL_PAN_DWELL also causes the value of ACTIVE_NETWORK to get latched into the hardware. This latched value will be the starting point for the automatic dual-pan mode (i.e., start on PAN0 or on PAN1). The starting value takes effect immediately (if sequence is underway and DUAL_PAN_AUTO=1), or is otherwise delayed until sequence starts and DUAL_PAN_AUTO=1.
7-6 —	Reserved
5 ZB_DP_CHAN_OVRD_SEL	Dual PAN Channel Override Selector This bit works with **ZB_DP_CHAN_OVRD_EN** to allow one of the two Dual PAN channels to use Direct Frequency programming. See description for **ZB_DP_CHAN_OVRD_EN** .
4	Dual PAN Channel Override Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ZB_DP_CHAN_OVRD_EN	In Dual PAN mode, in case there is a need to generate a frequency which may be offset from the 16 prescribed 5MHz-spaced channels, to, for example, avoid interference on one of the Dual PAN channels, a method has been provided to accomplish that, by designating one of the two PAN channels to use the transceiver's set of direct frequency-programming registers, instead of CHANNEL_NUMx. Programming the direct frequency-programming registers -- integer, numerator, and denominator, allows an RF frequency to be selected with much more precision than the 5MHz granularity of the Zigbee mapped-channel registers, CHANNEL_NUM0 and CHANNEL_NUM1. Two bits have been provided in 802.15.4 space to realize this feature: ZB_DP_CHAN_OVRD_SEL and ZB_DP_CHAN_OVRD_EN. When ZB_DP_CHAN_OVRD_EN=1, this enables one of the Dual PAN channels to use the direct frequency programming. The ZB_DP_CHAN_OVRD_SEL bit determines which channel uses the direct programming, according to the following: If ZB_DP_CHAN_OVRD_EN=0, then PAN0 frequency is determined by register CHANNEL_NUM0[6:0] and PAN1 frequency is determined by register CHANNEL_NUM1[6:0]. If ZB_DP_CHAN_OVRD_EN=1 and ZB_DP_CHAN_OVRD_SEL=0, then PAN0 frequency is determined by DIRECT FREQUENCY PROGRAMMING and PAN1 frequency is determined by register CHANNEL_NUM1[6:0]. If ZB_DP_CHAN_OVRD_EN=1 and ZB_DP_CHAN_OVRD_SEL=1, then PAN0 frequency is determined by register CHANNEL_NUM0[6:0] and PAN1 frequency is determined by DIRECT FREQUENCY PROGRAMMING. **NOTE** Direct Frequency Programming is accomplished by setting the PLL's Integer, Numerator, and Denominator registers to the appropriate values for the desired RF frequency.
3 CURRENT_NETWORK	Indicates which PAN is currently selected by hardware This read-only bit indicates which PAN is currently selected by hardware in automatic Dual PAN mode 0b - PAN0 is selected 1b - PAN1 is selected
2 PANCORDNTR1	Device is a PAN Coordinator on PAN1 Device is a PAN Coordinator on PAN1. Allows device to receive packets with no destination address, if Source PAN ID matches.
1 DUAL_PAN_AUTO	Activates automatic Dual PAN operating mode Activates automatic Dual PAN operating mode. In this mode, PAN-switching is controlled by hardware at a pre-programmed rate, determined by DUAL_PAN_DWELL. 0: Manual Dual PAN mode (or Single PAN mode). 1: Auto Dual PAN Mode Whenever DUAL_PAN_AUTO=0, CURRENT_NETWORK=ACTIVE_NETWORK at all times. In other words, software directly controls which PAN is selected. Whenever DUAL_PAN_AUTO=1, CURRENT_NETWORK is controlled by hardware.
0	Active Network Selector

Table continues on the next page...

Table continued from the previous page...

Field	Function
ACTIVE_NETWORK	Selects the PAN on which to transceive, by activating a PAN parameter set (PAN0 or PAN1). In Manual Dual PAN mode (or Single PAN mode), this bit selects the active PAN parameter set (channel and addressing parameters) which governs all autosequences. In Auto Dual PAN mode, this bit selects the PAN on which to begin transceiving, latched at the point at which DUAL_PAN_DWELL register is written. 0b - Select PAN0 1b - Select PAN1

55.4.9.1.3.1.25 CHANNEL NUMBER 1 (CHANNEL_NUM1)

Offset

Register	Offset
CHANNEL_NUM1	64h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CHANNEL_NUM1							
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Fields

Field	Function
31-7 —	Reserved
6-0 CHANNEL_NUM1	Channel Number for PAN1 This is the mapped channel number used to transmit and receive Zigbee packets. This register applies to PAN1 only. CHANNEL_NUM1 should be in the range: 11 <= CHANNEL_NUM1 <= 26 **Note** : This register should not be programmed, and left in its default state, if Dual PAN mode is not in use.

55.4.9.1.3.1.26 SAM CONTROL (SAM_CTRL)

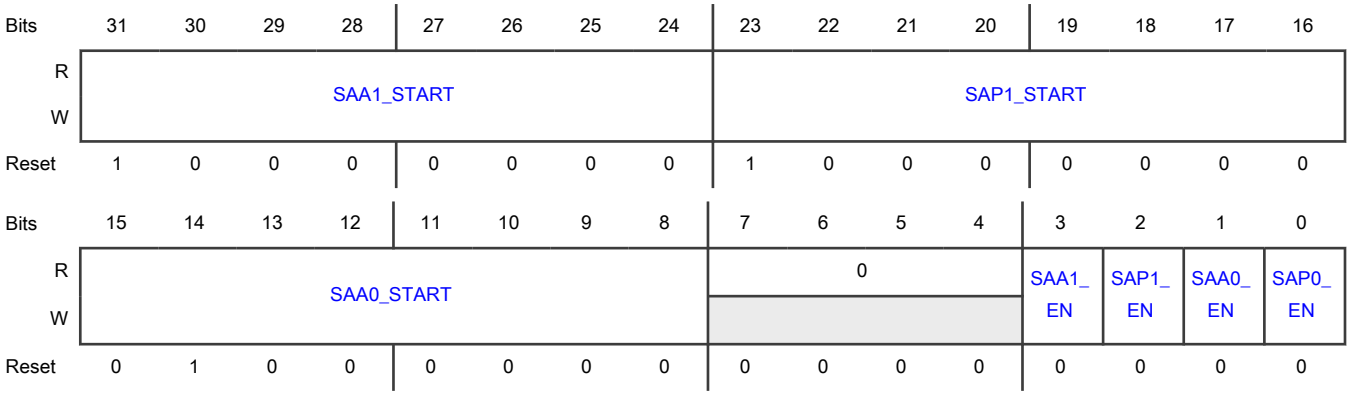
Offset

Register	Offset
SAM_CTRL	68h

Function

Source Address Management Control Register

Diagram



Fields

Field	Function
31-24 SAA1_START	First Index of SAA1 partition
23-16 SAP1_START	First Index of SAP1 partition
15-8 SAA0_START	First Index of SAA0 partition
7-4 —	Reserved
3 SAA1_EN	Enables SAA1 Partition of the SAM Table 0b - Disables SAA1 Partition 1b - Enables SAA1 Partition
2 SAP1_EN	Enables SAP1 Partition of the SAM Table 0b - Disables SAP1 Partition

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables SAP1 Partition
1 SAA0_EN	Enables SAA0 Partition of the SAM Table 0b - Disables SAA0 Partition 1b - Enables SAA0 Partition
0 SAP0_EN	Enables SAP0 Partition of the SAM Table 0b - Disables SAP0 Partition 1b - Enables SAP0 Partition

55.4.9.1.3.1.27 SOURCE ADDRESS MANAGEMENT TABLE (SAM_TABLE)

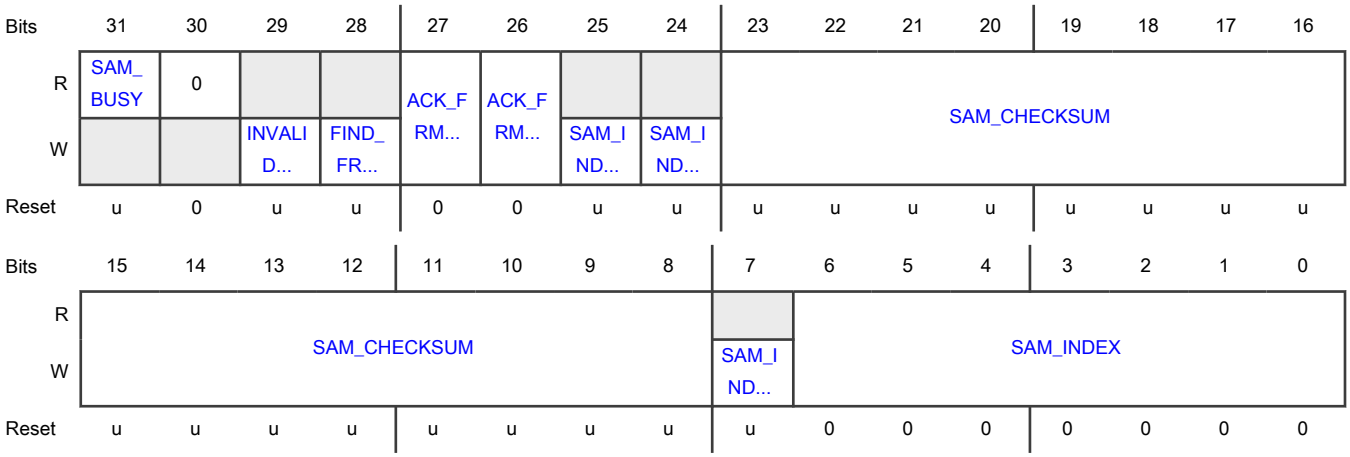
Offset

Register	Offset
SAM_TABLE	6Ch

Function

Source Address Management Table

Diagram



Fields

Field	Function
31 SAM_BUSY	SAM Table Update Status Bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Hardware is in the process of updating the Source Address table, either in response to a poll indication from the packet processor, or due to software setting FIND_FREE_IDX=1. In the latter case, software should poll SAM_BUSY until low before accessing the "First Free Index" registers. Read-only bit.
30 —	Reserved
29 INVALIDATE_ALL	Invalidate Entire SAM Table Writing a 1 to this bit clears all 128 Valid bits. Invalidates the entire table. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
28 FIND_FREE_IDX	Find First Free Index After modifying Valid bits (enabling or invalidating), write this bit to 1 to force hardware to update the "First Free Index" registers to account for the changed Valid bits. This hardware update process takes 4us. Software can poll SAM_BUSY to determine when the table update is complete. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
27 ACK_FRM_PND_CTRL	Manual Control for AutoTxAck FramePending field 0b - the FramePending field of the Frame Control Field of the next automatic TX acknowledge packet is determined by hardware 1b - the FramePending field of the Frame Control Field of the next automatic TX acknowledge packet tracks ACK_FRM_PEND
26 ACK_FRM_PND	State of AutoTxAck FramePending field when SAM Accelleration is Disabled Software can take manual control of the FramePending field of the Frame Control Field of the next automatic TX acknowledge packet, by setting ACK_FRM_PND_CTRL=1; in that case FramePending will track the state of this bit. The FramePending field also tracks this bit if Source Address Management is completely disabled, i.e., SAP0_EN=SAA0_EN=SAP1_EN=SAA1_EN=0 Otherwise, the FramePending field is determined by Source Address Management (SAM) hardware.
25 SAM_INDEX_EN	Enable the SAM table index selected by SAM_INDEX
24 SAM_INDEX_INV	Invalidate the SAM table index selected by SAM_INDEX
23-8 SAM_CHECKSUM	Software-computed source address checksum, to be installed into a table index Software-computed source address checksum, to be installed into a table index. The value on SAM_CHECKSUM[15:0] can be installed into the table with a single, atomic 32-bit write; in that case, the write data would contain the desired SAM_INDEX[6:0] and SAM_CHECKSUM[15:0], and SAM_INDEX_WR=1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If SAM_INDEX_WR=0, then the SAM_INDEX[6:0] register is written, but the checksum is *not* written to the table. The readback value of SAM_CHECKSUM[15:0] is the contents of the SAM Table at the location pointed to by SAM_INDEX[6:0]. To readback from a specific table index, software should first write the desired index to SAM_INDEX[6:0], and then read back the checksum from the table on SAM_CHECKSUM[15:0].
7 SAM_INDEX_WR	Enables SAM Table Contents to be updated For 32-bit writes, SAM_INDEX_WR must be set to indicate that the table entry specified by SAM_INDEX[6:0] is to be written; if SAM_INDEX_WR=0, the table entry is not written, but the SAM_INDEX[6:0] register is updated. For 8-bit writes, this bit is ignored.
6-0 SAM_INDEX	Contains the SAM table index to be enabled or invalidated Contains the table index to be enabled or invalidated. Software must ensure that the index is within the range of the desired partition.

55.4.9.1.3.1.28 SOURCE ADDRESS MANAGEMENT MATCH (SAM_MATCH)

Offset

Register	Offset
SAM_MATCH	70h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SAA1_AD...	SAA1_MATCH							SAP1_AD...	SAP1_MATCH						
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SAA0_AD...	SAA0_MATCH							SAP0_AD...	SAP0_MATCH						
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31	A Checksum Match is Absent in the SAP1 Partition of the SAM Table

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAA1_ADDR_ABSENT	
30-24 SAA1_MATCH	Index in the SAA1 Partition of the SAM Table corresponding to the first checksum match
23 SAP1_ADDR_PRESENT	A Checksum Match is Present in the SAP1 Partition of the SAM Table
22-16 SAP1_MATCH	Index in the SAP1 Partition of the SAM Table corresponding to the first checksum match
15 SAA0_ADDR_ABSENT	A Checksum Match is Absent in the SAA0 Partition of the SAM Table
14-8 SAA0_MATCH	Index in the SAA0 Partition of the SAM Table corresponding to the first checksum match
7 SAP0_ADDR_PRESENT	A Checksum Match is Present in the SAP0 Partition of the SAM Table
6-0 SAP0_MATCH	Index in the SAP0 Partition of the SAM Table corresponding to the first checksum match

55.4.9.1.3.1.29 SAM FREE INDEX (SAM_FREE_IDX)

Offset

Register	Offset
SAM_FREE_IDX	74h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SAA1_1ST_FREE_IDX								SAP1_1ST_FREE_IDX							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SAA0_1ST_FREE_IDX								SAP0_1ST_FREE_IDX							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

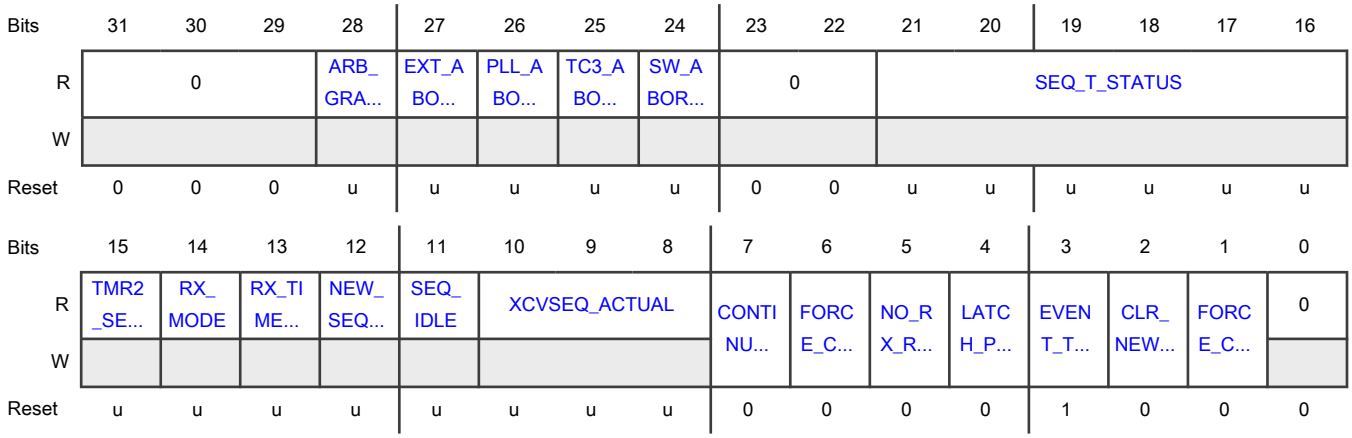
Field	Function
31-24 SAA1_1ST_FREE_IDX	First non-enabled (invalid) index in the SAA1 partition
23-16 SAP1_1ST_FREE_IDX	First non-enabled (invalid) index in the SAP1 partition
15-8 SAA0_1ST_FREE_IDX	First non-enabled (invalid) index in the SAA0 partition
7-0 SAP0_1ST_FREE_IDX	First non-enabled (invalid) index in the SAP0 partition

55.4.9.1.3.1.30 SEQUENCE CONTROL AND STATUS (SEQ_CTRL_STS)

Offset

Register	Offset
SEQ_CTRL_STS	78h

Diagram



Fields

Field	Function
31-29 —	Reserved
28 ARB_GRANT_DEASSERTION_ABORTED	Autosequence has terminated due to an arb_grant deassertion event when asserted, indicates that the autosequence has terminated due to an arb_grant deassertion event. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
27 EXT_ABORTED	Autosequence has terminated due to a Wake-On-Radio command when asserted, indicates that the autosequence has terminated due to an external event, such as a Wake-On-Radio stop command. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
26 PLL_ABORTED	Autosequence has terminated due to an PLL unlock event when asserted, indicates that the autosequence has terminated due to an PLL unlock event. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
25 TC3_ABORTED	autosequence has terminated due to an TMR3 timeout when asserted, indicates that the autosequence has terminated due to an TC3 (TMR3) timeout during a receive operation. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
24 SW_ABORTED	Autosequence has terminated due to a Software abort. when asserted, indicates that the autosequence has terminated due to an Software abort. Software can abort any programmed autosequence by writing Sequence I to XCVSEQ. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21-16 SEQ_T_STATUS	Status of the just-completed or ongoing Sequence T or Sequence TR Status of the just-completed (or ongoing) Sequence T or Sequence TR autosequence. This register is valid at all times during, and after, the Sequence T or Sequence TR. Not valid for other types of autosequences. This is a read-only register. The bits of this register map to status, according to the following table: [0] 1st CCA complete (CCABFRTX=1) [1] 2nd CCA complete (SLOTTED=1) [2] Tx operation complete [3] Rx Recycle occurred (Sequence TR only) [4] Rx operation complete (Sequence TR only) [5] TxAck operation complete(Sequence TR only)
15 TMR2_SEQ_TRIG_ARMED	indicates that TMR2 has been programmed and is armed to trigger a new autosequence when asserted, indicates that TMR2 has been programmed and is "armed" to trigger a new autosequence, when Zigbee Sequence Manager timer-triggering mode is selected (i.e., TMRTRIGEN=1). When timer-triggering mode is selected, TMR2 must be re-programmed (using either T2CMP or T2PRIMECMP), in advance of each new sequence. Once TMR2 is programmed, this bit will be asserted, and will remain asserted until the new sequence commences (at TMR2 match). Hardware will deassert this bit when the new sequence starts. When TMRTRIGEN=0, this bit should be ignored. Read-only bit.
14 RX_MODE	RX Operation in Progress when asserted, this Sequence Manager Output indicates that an RX operation is in progress. An RX operation can be part of a complex transmit autosequence such as a Sequence TR. CCA and ED operations are considered RX operations, during which rx_mode is asserted. Read-only bit.
13 RX_TIMEOUT_PENDING	Indicates a TMR3 RX Timeout is Pending when asserted, indicates that a TMR3 timeout (RX timeout) flag has been set by Hardware, but the Sequence Manager has not yet aborted because an RX operation is not currently underway. This would be the case, for example, during a Sequence TR, if a TMR3 timeout were to occur during the transmit operation of this sequence; the sequence would not be aborted by Hardware until the receive operation begins. This bit will always be 0 if TC3TMOUT=0. Read-only bit.
12 NEW_SEQ_INHIBIT	New Sequence Inhibit When asserted, indicates that a new programmed autosequence has commenced (TMR2 match has occurred if TMRTRIGEN=1). Once this bit is asserted, software is blocked from commanding any "new" autosequences (other than Sequence I to abort the current sequence), until the current sequence completes. Hardware will ignore a sequence-change command from software while this bit is asserted. Hardware will automatically deassert this bit once the sequence completes. Read-only bit.
11 SEQ_IDLE	ZSM Sequence Idle Indicator

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-8 XCVSEQ_ACTUAL	Indicates the programmed sequence that has been recognized by the ZSM Sequence Manager Reflects the programmed sequence that has been recognized by the Zigbee Sequence Manager. Takes into account the fact that sequence-change commands from software are ignored while a sequence is underway (see NEW_SEQ_INHIBIT). Read-only bits.
7 CONTINUOUS_EN	Enable Continuous TX or RX Mode Continuous Mode Enable (Continuous TX or RX). **Note:** Dual PAN mode should not be engaged in Continuous TX or RX modes. 0b - normal operation 1b - Continuous TX or RX mode is enabled (depending on XCVSEQ setting).
6 FORCE_CRC_ERROR	Induce a CRC Error in Transmitted Packets 0b - normal operation 1b - Force the next transmitted packet to have a CRC error
5 NO_RX_RECYCLE	Disable Automatic RX Sequence Recycling when asserted, prevents the Zigbee Sequence Manager (ZSM) from automatically re-starting (recycling) the receiver when a packet is received which results in a FilterFail or CRC failure. Normally, on a RX recycle, the ZSM returns to the RX_WU (warmup) state, and then resumes from there with a new, foreshortened, Rx warmup, in search of a new preamble. When this bit is set, the Sequence Manager will instead return to idle state, and issue a SEQIRQ, after a FilterFail or CRC failure.
4 LATCH_PREAMBLE	Stickiness Control for Preamble Detection 0b - Don't make PREAMBLE_DET and SFD_DET bits of PHY_STS (SEQ_STATE) Register "sticky", i.e, these status bits reflect the realtime, dynamic state of preamble_detect and sfd_detect 1b - Make PREAMBLE_DET and SFD_DET bits of PHY_STS (SEQ_STATE) Register "sticky", i.e., occurrences of preamble and SFD detection are latched and held until the start of the next autosequence
3 EVENT_TMR_DO_NOT_LATCH	Overrides the automatic hardware latching of the Event Timer when asserted, overrides the automatic hardware latching of the Event Timer that prevents the timer from updating while software reads the 3 Event Timer bytes. This allows the Event Timer LS byte to continue to update without reading the upper 2 bytes. Overriding the automatic latching of the Event Timer should be used with caution, as it can allow the Event Timer lower bytes to get out-of-sync with the upper bytes. However, it can be useful when polling the Event Timer LS byte for a value that is just a few counts in the future.
2 CLR_NEW_SEQ_INHIBIT	Overrides the automatic hardware locking of the programmed XCVSEQ while an autosequence is underway when asserted, overrides the automatic hardware locking of the programmed XCVSEQ while an autosequence is underway. Asserting this feature will allow software to change the programmed autosequence "on-the-fly", without aborting and returning to idle between sequences. Overriding the hardware lockout of XCVSEQ should be used with caution, since the Sequence Manager is not designed (or verified) for manual state transitions between one type of autosequence and other (i.e., Sequence T -> Sequence R).

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FORCE_CLK_ON	Force On 802.15.4 phy_gck Force On 802.15.4 phy_gck 0b - Allow TSM to control 802.15.4 phy_gck, for minimum power consumption (default) 1b - Force on 802.15.4 phy_gclk at all times, for debug purposes only
0 —	Reserved

55.4.9.1.3.1.31 ACK DELAY (ACKDELAY)

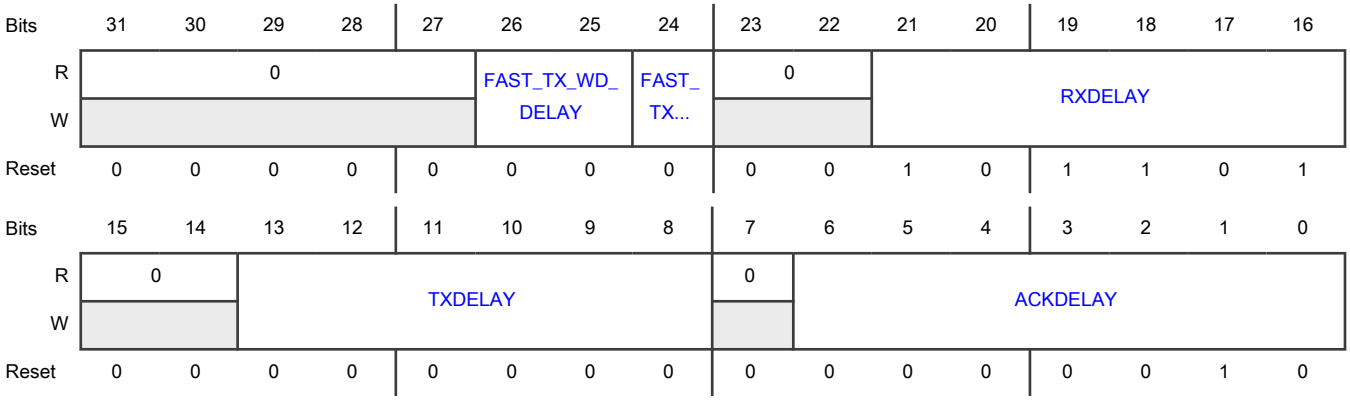
Offset

Register	Offset
ACKDELAY	7Ch

Function

Provides a fine-tune adjustment of the time delay between Rx warmdown and the beginning of Tx warmup for an autoTxAck packet.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-25	FAST_TX_WD_DELAY

Table continues on the next page...

Table continued from the previous page...

Field	Function
FAST_TX_WD_DELAY	Provides a fine-tune adjustment of the time delay from the beginning of Tx warmdown to the end of Tx warmdown when FAST_TX_WD_EN is set to 1.
24 FAST_TX_WD_EN	Fast TX_WD enable/disable When this bit is enabled, the Zigbee squence manager executes the fast TX warmdown sequence by bypassing TX_PA_OFF and XTAL_CLK1 and jumping from TX_WD to TSM_WD directly. 0b - Disable fast Tx warmdown sequence. 1b - Enable fast Tx warmdown sequence.
23-22 —	Reserved
21-16 RXDELAY	RX Delay Provides a fine-tune adjustment of the time delay between Tx warmdown and the beginning of Rx warmup for TR sequence. RXDELAY register will apply to both SLOTTED and UNSLOTTED TR sequence. This is an unsigned value. Resolution = 2us. Range = 3 - 127us. Max RXDELAY = 0x3F. Min RXDELAY = 0x01.
15-14 —	Reserved
13-8 TXDELAY	TX Delay Provides a fine-tune adjustment of the time delay between post-CCA Rx warm-down and the beginning of Tx warm-up for an Tx (non-Ack) packet. TXDELAY register will apply in both SLOTTED and UNSLOTTED modes, but only to T sequences (e.g., T, TR, and T(R)), not TxAck operations. This is a two's complement value. The minimum permissible value is -19 (0x2D). Values less than -19 will lead to unexpected results. Resolution = 2us. Range = +/- 62us. Max TXDELAY = 0x1F. Min TXDELAY = 0x2D.
7 —	Reserved
6-0 ACKDELAY	ACK Delay

Table continues on the next page...

Table continued from the previous page...

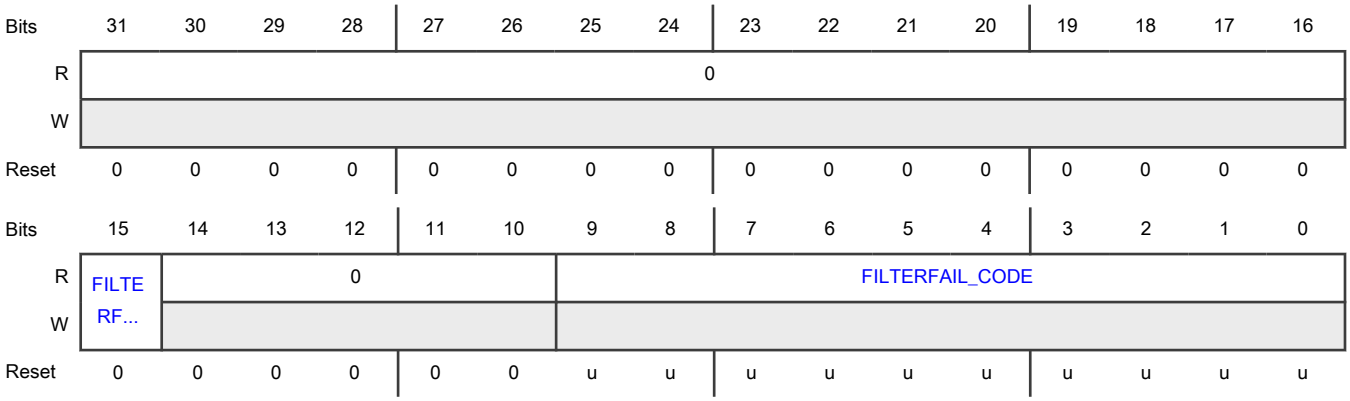
Field	Function
	Provides a fine-tune adjustment of the time delay between Rx warmdown and the beginning of Tx warmup for an Tx Acknowledge packet. ACKDELAY register will apply to both SLOTTED and UNSLOTTED TxAck, but only to TxAck (not T sequences). This is a two's complement value. Resolution = 2us. Range = -128 - 126us. Max ACKDELAY = 0x3F. Min ACKDELAY = 0x40.

55.4.9.1.3.1.32 FILTER FAIL CODE (FILTERFAIL_CODE)

Offset

Register	Offset
FILTERFAIL_CODE	80h

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FILTERFAIL_PAN_SEL	PAN Selector for Filter Fail Code 0b - FILTERFAIL_CODE[9:0] will report the FILTERFAIL status of PAN0 1b - FILTERFAIL_CODE[9:0] will report the FILTERFAIL status of PAN1
14-10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9-0 FILTERFAIL_CODE	Filter Fail Code Code indicating what condition, or conditions, caused the Packet Processor to reject the just-received packet. The bits of FILTERFAIL_CODE indicate the reason for packet rejection. For a description of the individual FILTERFAIL_CODE bits, see the 802.15.4 Link Layer Controller chapter.

55.4.9.1.3.1.33 RECEIVE WATER MARK (RX_WTR_MARK)

Offset

Register	Offset
RX_WTR_MARK	84h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RX_WTR_MARK							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Fields

Field	Function
31-8 —	Reserved
7-0 RX_WTR_MARK	RECEIVE WATER MARK Receive byte count (octets) needed to trigger a RXWTRMRKIRQ interrupt . A setting of 0 generates an interrupt at end of the Frame Length field (first byte after SFD). A setting of 1 generates an interrupt after the first byte of Frame Control Field, etc.

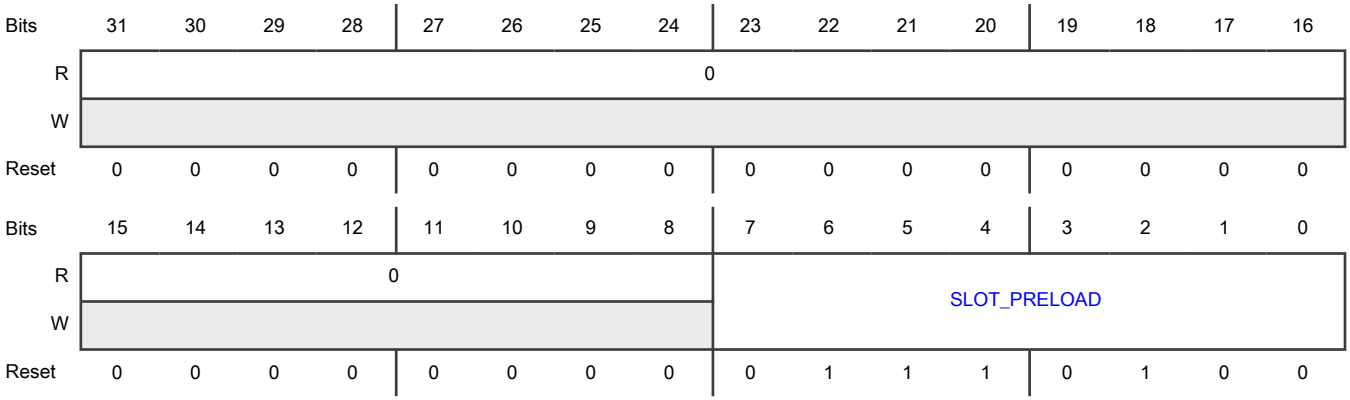
55.4.9.1.3.1.34 SLOT PRELOAD (SLOT_PRELOAD)

Offset

Register	Offset
SLOT_PRELOAD	8Ch

Function
Slotted Mode Preload

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SLOT_PRELOAD	Slotted Mode Preload This register represents the number that gets loaded into the slot_timer at SFD detect, which ultimately determines when the next slot boundary will occur. Due to processing delays within the analog front-end and digital modem, the point at which SFD is detected by the modem, is delayed relative to over-the-air timing. This register setting compensates for that delay. This timing parameter is critical for the Sequence R autosequence in slotted mode, when an automatic TxAck is required.

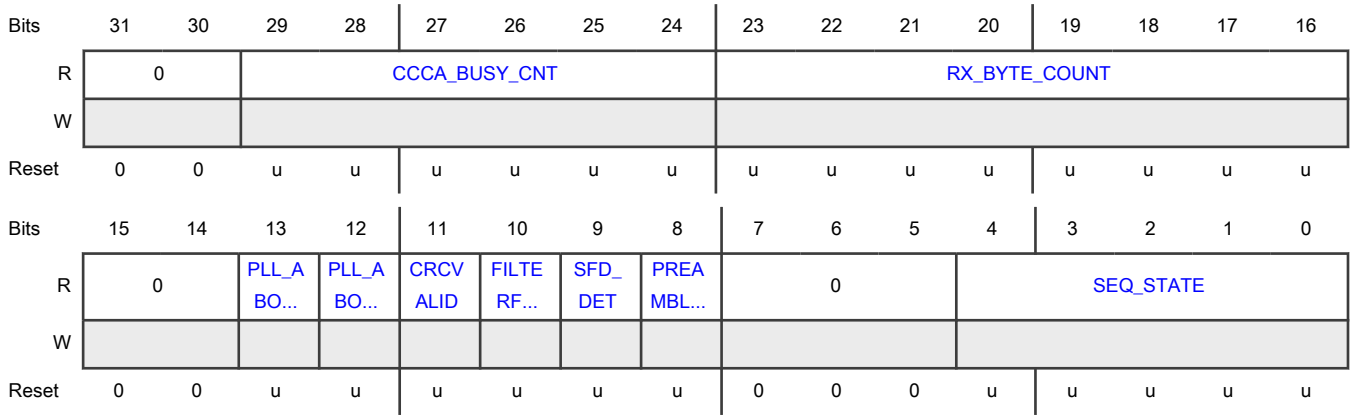
55.4.9.1.3.1.35 802.15.4 SEQUENCE STATE (SEQ_STATE)

Offset

Register	Offset
SEQ_STATE	90h

Function
802.15.4 Sequence State Register

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 CCCA_BUSY_CNT	Number of CCA Measurements resulting in Busy Channel For Sequence CCCA mode only, this register indicates the number of "busy" CCA attempts which occurred during the autosequence, before the channel was detected to be idle. This register can also be read in real-time (during the autosequence) to determine how many busy CCA attempts have occurred to that point. The register saturates at 63 (i.e, if there are more than 63 busy attempts, the register will continue to read 63). This register is automatically cleared to zero by hardware when the next autosequence commences. Read-only register.
23-16 RX_BYTE_COUNT	Realtime Received Byte Count During packet reception, this read-only register is a real-time indicator of the number of bytes that have been received. This register will read 0 until SFD and PHR have been received. It will read 1 after the first byte of Frame Control Field has been received, etc.
15-14 —	Reserved
13 PLL_ABORTED	Autosequence has terminated due to an PLL unlock event when asserted, indicates that the autosequence has terminated due to an PLL unlock event. This bit is valid at the SEQIRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. This bit is a read-only mirror of the register bit of the same name in the ABORT_STS (SEQ_CTRL_STS) register.
12 PLL_ABORT	Raw PLL Abort Signal This bit reflects the instantaneous, consolidated status of the PLL unlock detection circuits; if asserted high, indicates that at least one of the three PLL unlock detect mechanisms is currently reporting an unlocked condition.
11	CRC Valid Indicator

Table continues on the next page...

Table continued from the previous page...

Field	Function
CRCVALID	Cyclic Redundancy Check Valid: This flag indicates the compare result between the FCS field, in the most-recently received frame, and the internally calculated CRC value. This flag is cleared at next receiver warm up. 0b - Rx FCS != calculated CRC (incorrect) 1b - Rx FCS = calculated CRC (correct)
10 FILTERFAIL_FLAG_SEL	Consolidated Filter Fail Flag 0: The incoming, or just-received packet, passed packet filtering rules. 1: The incoming, or just-received packet, failed packet filtering rules When FILTERFAIL_FLAG_SEL=1, a non-zero FILTERFAIL_CODE is present (see FILTERFAIL_CODE registers). In Dual PAN mode, FILTERFAIL_FLAG_SEL applies to either or both networks, as follows: **A:** If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_FLAG_SEL applies to PAN0. **B:** If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_FLAG_SEL applies to PAN1. **C:** If PAN0 and PAN1 occupy the same channel, FILTERFAIL_FLAG_SEL is the logical 'AND' of the individual PANs' FILTERFAIL_FLAG bits.
9 SFD_DET	SFD Detected 0: an 802.15.4 preamble-and-SFD have not been detected. 1: An 802.15.4 preamble-and-SFD have been detected. The function of this read-only bit depends on the setting of the LATCH_PREAMBLE bit of the SEQ_MGR_CTRL register. If LATCH_PREAMBLE=1, any preamble-and-SFD detection during a Sequence R (even false detections), will set this bit, and it will remain set (sticky) until the start of the next autosequence. If LATCH_PREAMBLE=0, this bit is not sticky, and reflects the instantaneous state of the SFD-detection circuit; for false SFD, the bit will clear when the false nature of the SFD is recognized (i.e., an RX recycle). When LATCH_PREAMBLE=0, SFD_DET should be considered valid only while an autosequence is underway.
8 PREAMBLE_DET	Preamble Detected 0: an 802.15.4 preamble has not been detected. 1: An 802.15.4 preamble has been detected. The function of this read-only bit depends on the setting of the LATCH_PREAMBLE bit of the SEQ_MGR_CTRL register. If LATCH_PREAMBLE=1, any preamble detection during a Sequence R (even false detections), will set this bit, and it will remain set (sticky) until the start of the next autosequence. If LATCH_PREAMBLE=0, this bit is not sticky, and reflects the instantaneous state of the preamble-detection circuit; for false preambles, the bit will clear when the false nature of the preamble is recognized. When LATCH_PREAMBLE=0, PREAMBLE_DET should be considered valid only while an autosequence is underway.
7-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-0 SEQ_STATE	ZSM Sequence State This read-only register reflects the instantaneous state of the 802.15.4 Sequence Manager

55.4.9.1.3.1.36 TIMER PRESCALER (TMR_PRESCALE)

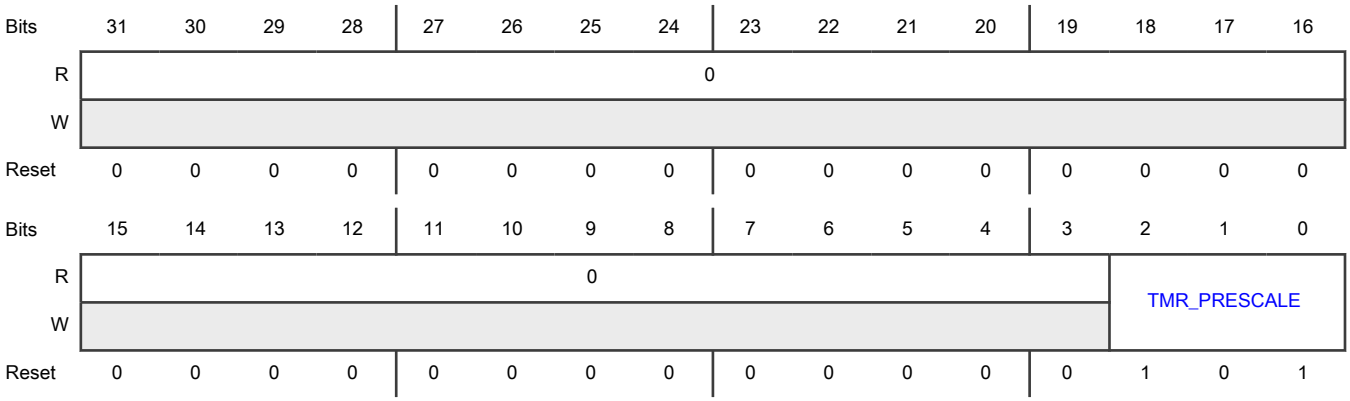
Offset

Register	Offset
TMR_PRESCALE	94h

Function

Timer Prescaler Control Register

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 TMR_PRESCALE	Timer Prescaler Timer Prescaler. Establishes the Event Timer clock rate (and also maximum timer duration) **Note:** To take advantage of the EVENT_TMR Fractional bits for 802.15.4 DSM mode, only the default setting ("5", or 62.5KHz) is allowed. 000b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - Reserved 010b - 500kHz (33.55 S) 011b - 250kHz (67.11 S) 100b - 125kHz (134.22 S) 101b - 62.5kHz (268.44 S) -- default 110b - 31.25kHz (536.87 S) 111b - 15.625kHz (1073.74 S)

55.4.9.1.3.1.37 LENIENCY LSB (LENIENCY_LSB)

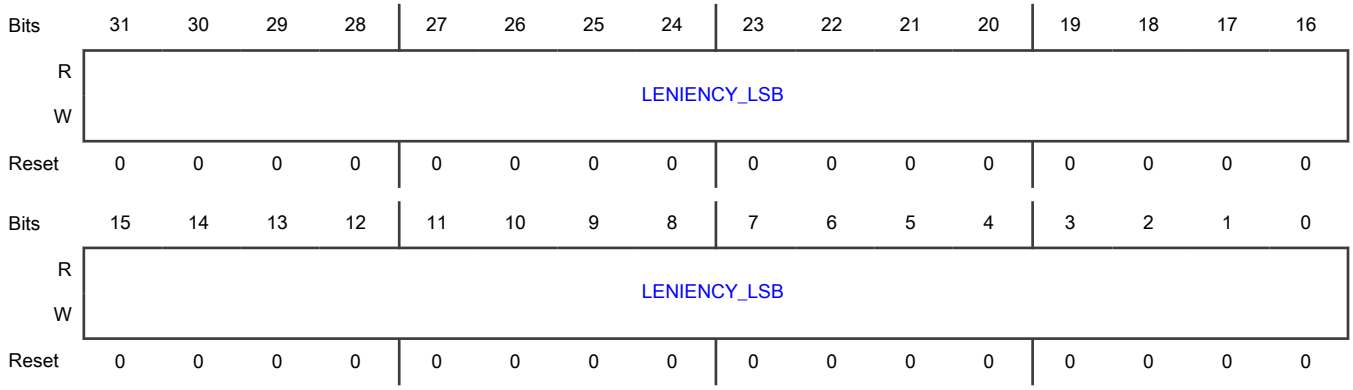
Offset

Register	Offset
LENIENCY_LSB	98h

Function

Packet Processor Leniency Bits (LSB)

Diagram



Fields

Field	Function
31-0 LENIENCY_LSB	Leniency LSB Register The Packet Processor performs filtering on all received packets, in order to determine whether the packet is intended for the device. The packet filtering is based on rules. In case any of the packet filtering rules need to be overridden, a 43-bit "leniency register" has been provided. When the leniency register is programmed to its default value (0), all hardware packet filtering rules are in effect, and if an incoming

Table continues on the next page...

Field	Function
	packet violates any rule, a "Filter Fail" will occur (packet will be rejected). When a given leniency register bit is asserted, the packet filtering rule assigned to that bit will not be in effect, and if any incoming packet violates that rule (but no other rules), then a "Filter Fail" will not occur, the packet will not be rejected, the packet will be treated as "intended for the device", and software will be notified of the incoming packet. For a description of the function of the individual leniency bits, see the 802.15.4 Link Layer Controller chapter.

55.4.9.1.3.1.38 LENIENCY MSB (LENIENCY_MSB)

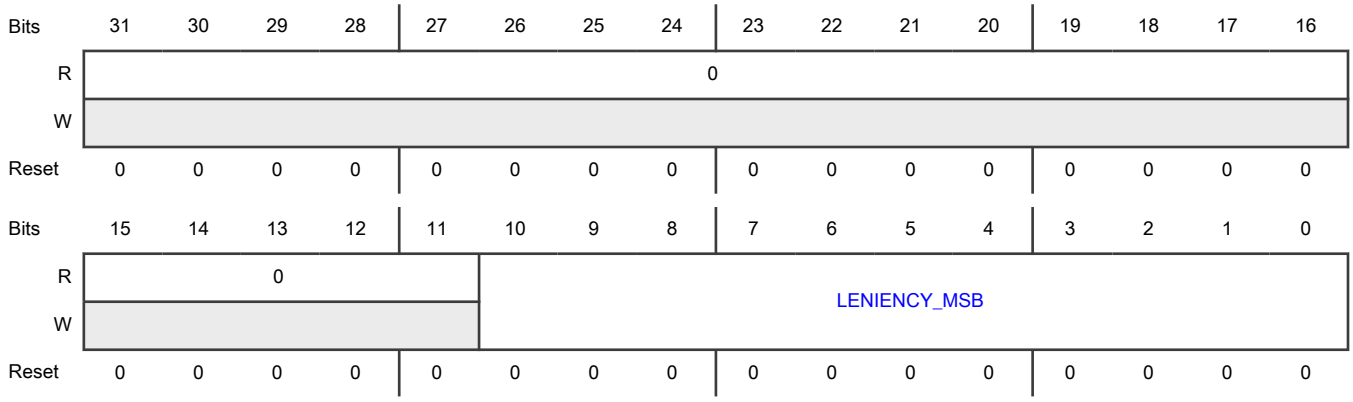
Offset

Register	Offset
LENIENCY_MSB	9Ch

Function

Packet Processor Leniency Bits (MSB)

Diagram



Fields

Field	Function
31-11 —	Reserved
10-0 LENIENCY_MSB	Leniency MSB Register The Packet Processor performs filtering on all received packets, in order to determine whether the packet is intended for the device. The packet filtering is based on rules. In case any of the packet filtering rules need to be overridden, a 43-bit "leniency register" has been provided. When the leniency register is programmed to its default value (0), all hardware packet filtering rules are in effect, and if an incoming packet violates any rule, a "Filter Fail" will occur (packet will be rejected). When a given leniency register bit is asserted, the packet filtering rule assigned to that bit will not be in effect, and if any incoming packet

Table continues on the next page...

Table continued from the previous page...

Field	Function
	violates that rule (but no other rules), then a "Filter Fail" will not occur, the packet will not be rejected, the packet will be treated as "intended for the device", and software will be notified of the incoming packet. For a description of the function of the individual leniency bits, see the 802.15.4 Link Layer Controller chapter.

55.4.9.1.3.1.39 PART ID (PART_ID)

Offset

Register	Offset
PART_ID	A0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								PART_ID							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 PART_ID	802.15.4 Part ID

55.4.9.1.3.1.40 COEXISTENCE CONTROL (COEX_CTRL)

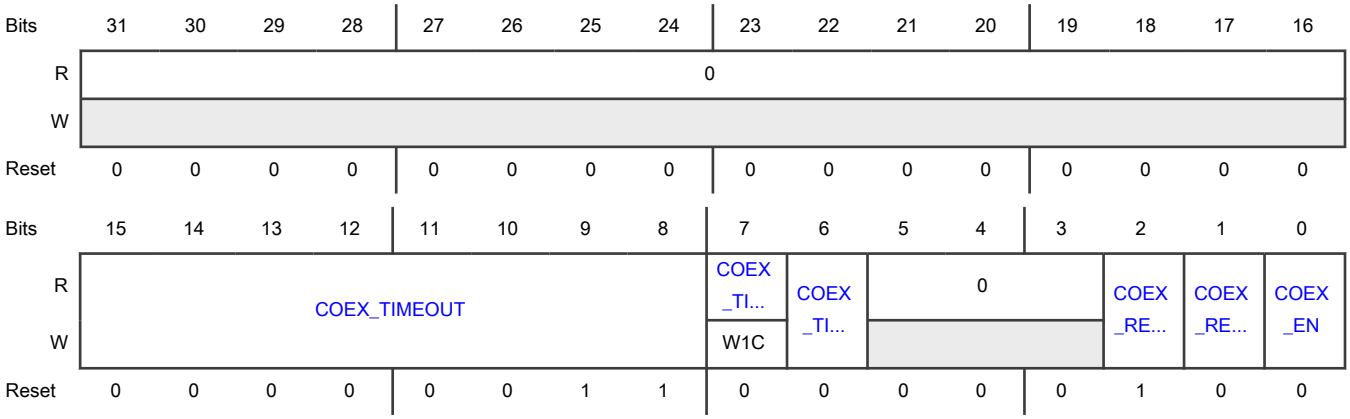
Offset

Register	Offset
COEX_CTRL	A4h

Function

Coexistence Control Registers.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 COEX_TIMEOUT	Coexistence timeout value After arb_request asserts, arb_grant should assert before the timeout value. Otherwise the sequence is abort and COEX_TIMEOUT_IRQ bit is set. The step of the timeout value is 32us.
7 COEX_TIMEOUT_IRQ	Coexistence Timeout Interrupt Indicates coexistence timeout occurs. This is write '1' to clear bit.
6 COEX_TIMEOUT_MSK	Coexistence Timeout Interrupt Mask bit 0b - allows interrupt when coexistence timeout 1b - Interrupt generation is disabled, but a COEX_TIMEOUT_IRQ flag can be set
5-3 —	Reserved
2 COEX_REQ_ON_PD	Coexistence Request on Preamble detected 0b - arb_request is delayed until SFD is detected during R sequence. 1b - arb_request is delayed until preamble is detected during R sequence.
1 COEX_REQ_DELAY_EN	Coexistence Request Delay Enable 0b - arb_request is not delayed during R sequence. 1b - arb_request is delayed until preamble is detected during R sequence.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 COEX_EN	Coexistence Enable 0b - Coexistence function is disabled. 1b - Coexistence function is enabled.

55.4.9.1.3.1.41 COEXISTENCE PRIORITY (COEX_PRIORITY)

Offset

Register	Offset
COEX_PRIORITY	A8h

Function

Programmable 2-bit priority for each RX or TX state in each ZSM.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRIORIT...	PRIORITY_OVRD			0										PRIORITY_RACK_P...	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRIORITY_RACK_P...		PRIORITY_CTX		PRIORITY_CCCA		PRIORITY_CCA		PRIORITY_TACK		PRIORITY_R_PKT		PRIORITY_R_PRE		PRIORITY_T	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PRIORITY_OVRD_EN	PRIORITY_OVRD_EN Enable/Disable overriding PRIORITY value. 0b - Disable overriding PRIORITY value. 1b - Enable overriding PRIORITY value.
30-29 PRIORITY_OVRD	PRIORITY_OVRD When PRIORITY_OVRD_EN is set to 1, PRIORITY_OVRD is used to override PRIORITY value.
28-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 PRIORITY_RACK_PKT	PRIORITY_RACK_PKT PRIORITY value during receiving ACK packet.
15-14 PRIORITY_RACK_PRE	PRIORITY_RACK_PRE PRIORITY value during searching preamble of ACK packet.
13-12 PRIORITY_CTX	PRIORITY_CT PRIORITY value for CCA before TX Packet.
11-10 PRIORITY_CCCA	PRIORITY_CCCA PRIORITY value for Continuous CCA.
9-8 PRIORITY_CCA	PRIORITY_CCA PRIORITY value for CCA.
7-6 PRIORITY_TACK	PRIORITY_TACK PRIORITY value for TX ACK Packet.
5-4 PRIORITY_R_PKT	PRIORITY_R_PKT PRIORITY value for RX Packet.
3-2 PRIORITY_R_PRE	PRIORITY_R_PRE PRIORITY value of searching Preamble for RX Packet.
1-0 PRIORITY_T	PRIORITY_T PRIORITY value for TX Packet.

55.4.9.1.3.1.42 ENHACK_CTRL 0 (ENHACK_CTRL0)

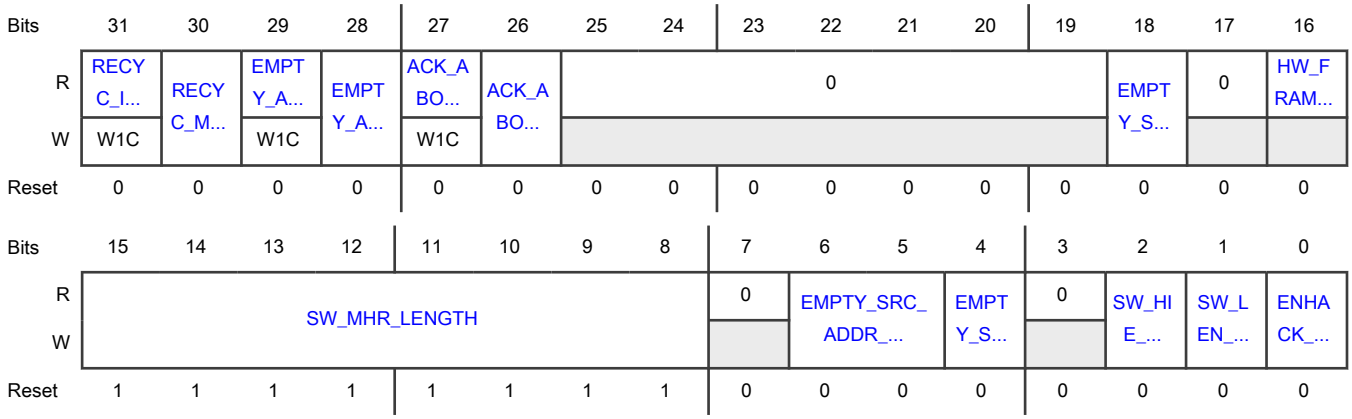
Offset

Register	Offset
ENHACK_CTRL0	ACh

Function

Control Register 0 for Enhanced Acknowledgment Feature

Diagram



Fields

Field	Function
31 RECYC_IRQ	Recycle IRQ This interrupt status bit indicates the hardware recycle on filter fail or CRC invalid.
30 RECYC_MSK	Recycle IRQ Mask bit 0b - allows interrupt when recycle 1b - Interrupt generation is disabled, but a RECYC_IRQ flag can be set
29 EMPTY_ACK_IRQ	Empty Enhanced Acknowledgment IRQ This interrupt status bit indicates the Empty Enhanced Acknowledgment is to be sent.
28 EMPTY_ACK_MSK	Empty Enhanced Acknowledgment IRQ Mask bit 0b - allows interrupt when Empty Enhanced Acknowledgment 1b - Interrupt generation is disabled, but a EMPTY_ACK_IRQ flag can be set
27 ACK_ABORT_IRQ	Enhanced Acknowledgment Abort IRQ This interrupt status bit indicates the HIE field is not ready by software.
26 ACK_ABORT_MSK	Enhanced Acknowledgment Abort IRQ Mask bit 0b - allows interrupt when HIE field is not ready by software. 1b - Interrupt generation is disabled, but a ACK_ABORT_IRQ flag can be set
25-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 EMPTY_SECURITY_ENABLED_OVRD_EN	Override enable of Security Enabled field in Empty Enhanced Acknowledgment 0b - Security Enabled field in Empty Enhanced Acknowledgment frame is 0. 1b - Security Enabled field in Empty Enhanced Acknowledgment frame is from EMPTY_SECURITY_ENABLED_OVRD.
17 —	Reserved
16 HW_FRAME_PENDING	Hardware calculated Frame Pending field When received a DATA or Command Frame, hardware calculates the Frame Pending field by Source Address Management. This bit can be used by software to build the Enhanced Acknowledgment frame.
15-8 SW_MHR_LENGTH	Software calculated MHR(excludes the HIE field) Length in bytes.
7 —	Reserved
6-5 EMPTY_SRC_ADDR_MODE	Source Address Mode field in Empty Enhanced Acknowledgment
4 EMPTY_SECURITY_ENABLED_OVRD	Override value of Security Enabled field in Empty Enhanced Acknowledgment
3 —	Reserved
2 SW_HIE_RDY	Software enhanced acknowledgment frame HIE field ready 0b - Software enhanced acknowledgment frame HIE field is not ready. 1b - Software enhanced acknowledgment frame HIE field is ready in RAM
1 SW_LEN_RDY	Software enhanced acknowledgment frame Length field ready 0b - Software enhanced acknowledgment frame Length field is not ready. 1b - Software enhanced acknowledgment frame Length field is ready in RAM
0 ENHACK_EN	Enhanced Acknowledgment Enable 0b - Enhanced acknowledgment is disabled. 1b - Enhanced acknowledgment is enabled.

55.4.9.2 Generic FSK Link Layer

55.4.9.2.1 Introduction

The Generic FSK protocol enables radio operation using a custom GFSK/GMSK or MSK modulation format achieved by programming a set of PHY variables such as BT product, modulation index and modulation filter co-efficients (such that max frequency deviation <= 500kHz). Generic FSK mode also offers a highly configurable packet structure, variable bit rate transmission and reception, some limited packet processing, and interface to a RAM-based Packet Buffer.

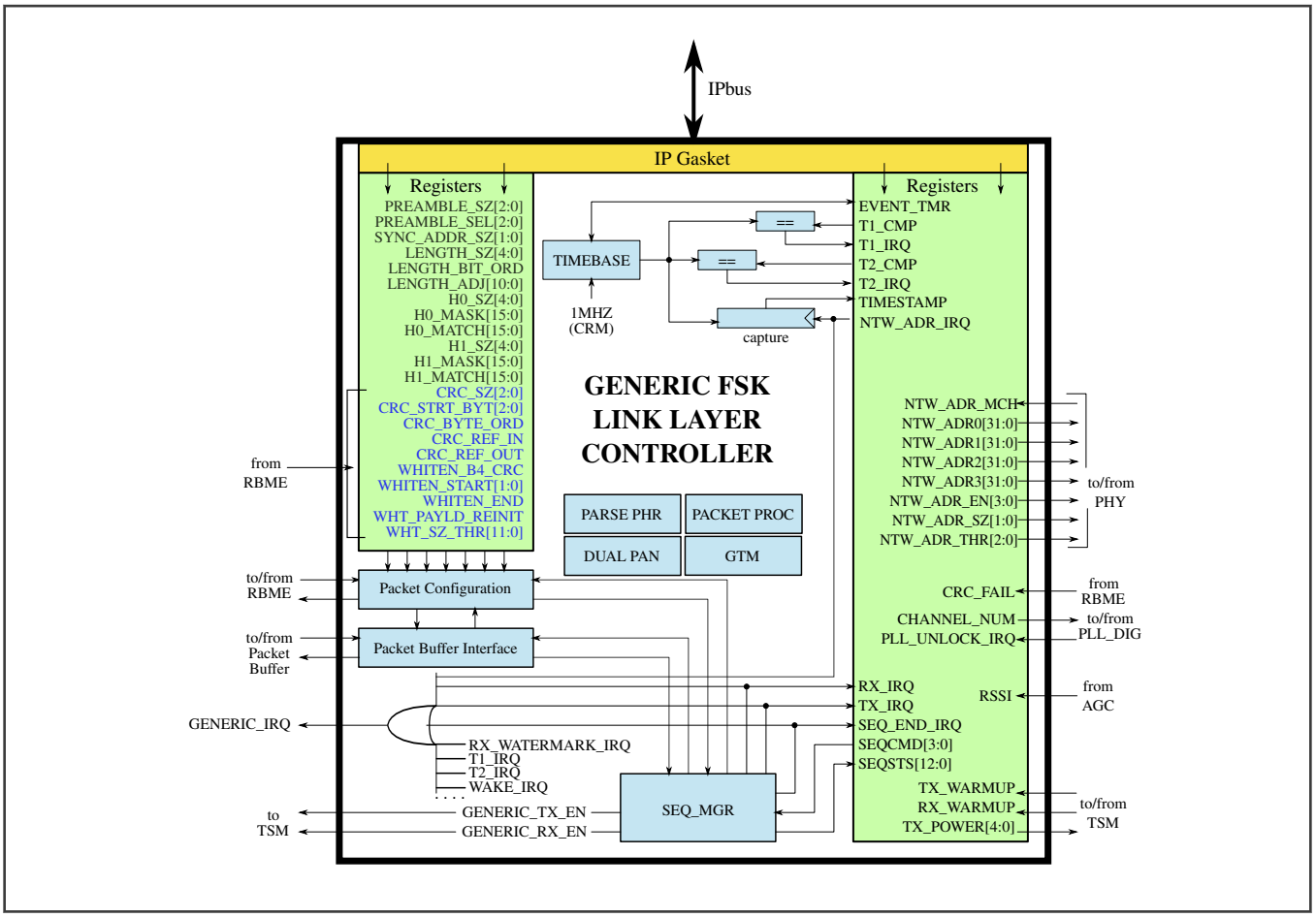
55.4.9.2.1.1 Overview

The Generic FSK Link Layer Controller provides the interface between Generic FSK software, and the transceiver hardware. The interface consists of an IP bus-based set of registers, which provide command, control, and status to Generic FSK software. The Link Layer controller features programmability that allows for a highly configurable packet structure, defining the lengths, bit ordering, and contents of individual packet fields, defining the start and end points for whitening, CRC, and providing for some primitive parsing of the packet header. The Link Layer Controller implements a Sequence Command Set consisting of 20 commands; a command decoder to interpret the commands in realtime as they are received from Generic FSK software, and to provide the necessary control signals to the radio's Transceiver Sequence Manager (TSM); a high-precision timebase based on an external crystal-based oscillator; an Interrupt Control Unit; finite state machines to control the transmission and reception of Generic FSK packets; controls for CRC generation (TX) and verification (RX); controls for data whitening; interface to RAM-based Packet Buffer for packet storage; and provisions for entering and exiting low-power Deep Sleep Mode.

55.4.9.2.1.2 Features

- Highly configurable packet structure
- Optimized Sequence Command Set
- High-precision timebase to maintain network timing
- Two timer-compare mechanisms for Interrupt Generation and Sequence Launching
- Hardware automation for packet transmit and receive, CRC and Whitening
- Up to 4 Network Addresses to synchronize to, can be 8-bit, 16-bit or 32-bit
- Packet Lengths up to 2047 Bytes
- Support complex auto-sequence, like CCA before TX, Auto-ACK, TR.
- Many operating modes can support the sending and receiving of multiple protocol packets, such as 802.15.4, Bluetooth LE, etc.
- Generic test mode(GTM)
- Deep Sleep Mode support

55.4.9.2.1.3 Block Diagram



55.4.9.2.2 Memory Map and Register Definition

55.4.9.2.2.1 GENERIC FSK register descriptions

55.4.9.2.2.1.1 FSK memory map

GENFSK base address: 48A0_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	IRQ CONTROL (IRQ_CTRL)	32	RW	0000_0000h
4h	EVENT TIMER (EVENT_TMR)	32	R	See section
8h	T1 COMPARE (T1_CMP)	32	RW	FFFF_FFFFh
Ch	T2 COMPARE (T2_CMP)	32	RW	FFFF_FFFFh
10h	TIMESTAMP (TIMESTAMP)	32	R	See section
14h	TRANSCEIVER CONTROL (XCVR_CTRL)	32	RW	0007_FF00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	TRANSCEIVER STATUS (XCVR_STS)	32	R	0080_0000h
1Ch	TRANSCEIVER CONFIGURATION (XCVR_CFG)	32	RW	0000_0000h
20h	CHANNEL NUMBER 0 (CHANNEL_NUM0)	32	RW	0000_0000h
24h	TRANSMIT POWER (TX_POWER)	32	RW	0000_0000h
28h	NETWORK ADDRESS CONTROL (NTW_ADR_CTRL)	32	RW	0000_0100h
2Ch	NETWORK ADDRESS 0 (NTW_ADR_0)	32	RW	5555_5555h
30h	NETWORK ADDRESS 1 (NTW_ADR_1)	32	RW	5555_5555h
34h	NETWORK ADDRESS 2 (NTW_ADR_2)	32	RW	5555_5555h
38h	NETWORK ADDRESS 3 (NTW_ADR_3)	32	RW	5555_5555h
3Ch	RECEIVE WATERMARK (RX_WATERMARK)	32	RW	1FFF_0FFFh
40h	DSM CONTROL (DSM_CTRL)	32	RW	0000_0000h
44h	PART ID (PART_ID)	32	R	0000_0002h
48h	SLOT PRELOAD (SLOT_PRELOAD)	32	RW	0000_02A8h
4Ch	SLOT TIME (SLOT_TIME)	32	RW	0000_08E8h
50h	TURNAROUND TIME (TURNAROUND_TIME)	32	RW	0000_03E8h
54h	ACK DELAY (ACKDELAY)	32	RW	0000_0000h
58h	RX DELAY (RXDELAY)	32	RW	0000_0000h
5Ch	TX DELAY (TXDELAY)	32	RW	0000_0000h
60h	PACKET CONFIGURATION (PACKET_CFG)	32	RW	00C0_0040h
64h	H0 CONFIGURATION (H0_CFG)	32	RW	0000_0000h
68h	H1 CONFIGURATION (H1_CFG)	32	RW	0000_0000h
6Ch	CRC CONFIGURATION (CRC_CFG)	32	RW	E000_0000h
70h	LENGTH ADJUSTMENT (LENGTH_ADJ)	32	RW	0000_0000h
74h	TIMESTAMP_RX_DONE (TIMESTAMP_RX_DONE)	32	R	See section
78h	TIMESTAMP_TX_DONE (TIMESTAMP_TX_DONE)	32	R	See section
7Ch	MULT_PKT_CTRL (MULT_PKT_CTRL)	32	RW	0000_0000h
80h	RPA AND WHITE LIST STATUS (RPA_WL_STATUS)	32	RW	0F0F_003Fh
84h	MAXIMUM LENGTH (LENGTH_MAX)	32	RW	0000_0000h
88h	EVENT TIMER LOAD (EVENT_TMR_LD)	32	W	0000_0000h
8Ch	EVENT TIMER ADD (EVENT_TMR_ADD)	32	W	0000_0000h
90h	ENHANCED FEATURES (ENH_FEATURE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
94h	RECEIVE FRAME FILTER (RX_FRAME_FILTER)	32	RW	0000_060Fh
98h	FILTER FAIL CODE (FILTERFAIL_CODE)	32	RW	0000_0000h
9Ch	LENIENCY LSB (LENIENCY_LSB)	32	RW	0000_0000h
9Ch	RPA CONTROL (RPA_CTRL)	32	RW	0000_0000h
A0h	LENIENCY MSB (LENIENCY_MSB)	32	RW	0000_0000h
A0h	WHITE LIST CONTROL (WL_CTRL)	32	RW	0000_0000h
A4h	DUAL PAN CONTROL (DUAL_PAN_CTRL)	32	RW	0001_0000h
A8h	GTM MODE PDU (GTM_PDU)	32	RW	FFFF_FFFFh
A8h	MAC SHORT ADDRESS FOR PAN1 (MACSHORTADDRS1)	32	RW	FFFF_FFFFh
A8h	VALID ENTRY OF WHITE LIST 1 (WL_VALID_ENTRY1)	32	RW	FFFF_FFFFh
ACh	DIRECT_PEER_ADDR[31:0] (DIRECT_PEER_ADDR_LSB)	32	RW	FFFF_FFFFh
ACh	GTM MODE CONFIGURATION (GTM_CFG)	32	RW	FFFF_FFFFh
ACh	MAC LONG ADDRESS 1 LSB (MACLONGADDRS1_LSB)	32	RW	FFFF_FFFFh
B0h	DIRECT_PEER_ADDR[47:32] (DIRECT_PEER_ADDR_MSB)	32	RW	FFFF_FFFFh
B0h	GTM MODE INTER-PACKET DURATION (GTM_IPD)	32	RW	FFFF_FFFFh
B0h	MAC LONG ADDRESS 1 MSB (MACLONGADDRS1_MSB)	32	RW	FFFF_FFFFh
B4h	CHANNEL NUMBER 1 (CHANNEL_NUM1)	32	RW	0000_0000h
B8h	MAC SHORT ADDRESS 0 (MACSHORTADDRS0)	32	RW	FFFF_FFFFh
B8h	VALID ENTRY OF WHITE LIST 0 (WL_VALID_ENTRY0)	32	RW	FFFF_FFFFh
BCh	GTM MODE TIME OF FIRST SFD FOUND TO FORCE RX WARMDOWN (GTM_FIRST_SFD2WD)	32	RW	FFFF_FFFFh
BCh	MAC LONG ADDRESS 0 LSB (MACLONGADDRS0_LSB)	32	RW	FFFF_FFFFh
BCh	WL_SEARCH_ADDR[31:0] (WL_SEARCH_ADDR_LSB)	32	RW	FFFF_FFFFh
C0h	GTM MODE RX RECYCLE TIME (GTM_RX_RECYCLE_TIME)	32	RW	FFFF_FFFFh
C0h	MAC LONG ADDRESS 0 MSB (MACLONGADDRS0_MSB)	32	RW	FFFF_FFFFh
C0h	WL_SEARCH_ADDR[47:32] (WL_SEARCH_ADDR_MSB)	32	RW	FFFF_FFFFh
C4h	CCA AND LQI CONTROL (CCA_LQI_CTRL)	32	RW	0000_4B00h
C8h	TX/RX WARMUP TIME (WARMUP_TIME)	32	R	0070_005Ah
CCh	RX_EN Delay Time (RXEN_DLY)	32	RW	0000_0000h
D4h	SAM CONTROL (SAM_CTRL)	32	RW	8080_4000h
D8h	SOURCE ADDRESS MANAGEMENT TABLE (SAM_TABLE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DCh	SOURCE ADDRESS MANAGEMENT MATCH (SAM_MATCH)	32	R	FF7F_FF7Fh
E0h	SAM FREE INDEX (SAM_FREE_IDX)	32	R	8080_8080h
E4h	MISCELLANEOUS(1) (MISC1)	32	RW	0000_0000h
E8h	SEQUENCE STATUS (SEQ_STS)	32	R	0000_0000h
ECh	PHR MISCELLANEOUS (PHR_MISC)	32	RW	0000_0000h
F0h	GTM CONTROL (GTM_CTRL)	32	RW	0000_0000h
F4h	GTM BAD PACKET COUNTER (GTM_BAD_CNT)	32	R	0000_0000h
F8h	GTM GOOD PACKET COUNTER (GTM_GOOD_CNT)	32	R	0000_0000h
FCh	GTM PACKET COUNTER (GTM_PKT_CNT)	32	R	0000_0000h
100h	COEXISTENCE CONTROL (COEX_CTRL)	32	RW	0000_0300h
104h	COEXISTENCE PRIORITY (COEX_PRIORITY)	32	RW	0000_0000h
108h	IRQ CONTROL 2 (IRQ_CTRL2)	32	RW	0000_0000h

55.4.9.2.2.1.2 IRQ CONTROL (IRQ_CTRL)

Offset

Register	Offset
IRQ_CTRL	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MS_IRQ_...	CCA_IRQ...	FILTERF...	PHRFAIL...	ACK_IRQ...	GENERIC...	TSM_IRQ...	RX_WATE...	WAKE_IR...	PLL_UNL...	T2_IRQ_...	T1_IRQ_...	NTW_ADR...	RX_IRQ_...	TX_IRQ_...	SEQ_END...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MS_IRQ	CCA_IRQ	FILTERF...	PHRFFAI...	ACK_IRQ	CRC_VAL...	TSM_IRQ	RX_WATE...	WAKE_IRQ	PLL_UNL...	T2_IRQ	T1_IRQ	NTW_ADR...	RX_IRQ	TX_IRQ	SEQ_END...
W	W1C	W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MS_IRQ_EN	MS_IRQ Enable 0b - MS Interrupt is not enabled. 1b - MS Interrupt is enabled.
30 CCA_IRQ_EN	CCA_IRQ Enable 0b - CCA Interrupt is not enabled. 1b - CCA Interrupt is enabled.
29 FILTERFAIL_IRQ_EN	FILTERFAIL_IRQ Enable 0b - FILTERFAIL Interrupt is not enabled. 1b - FILTERFAIL Interrupt is enabled.
28 PHRFAIL_IRQ_EN	PHRFAIL_IRQ Enable 0b - PHRFAIL Interrupt is not enabled. 1b - PHRFAIL Interrupt is enabled.
27 ACK_IRQ_EN	ACK_IRQ Enable 0b - Auto ACK Interrupt is not enabled. 1b - Auto ACK Interrupt is enabled.
26 GENERIC_FSK_IRQ_EN	GENERIC_FSK_IRQ Master Enable Master enable for the GENERIC_FSK_IRQ interrupt line to the MCU. 0b - All GENERIC_FSK Interrupts are disabled. 1b - All GENERIC_FSK Interrupts can be enabled.
25 TSM_IRQ_EN	TSM_IRQ Enable 0b - TSM Interrupt is not enabled. 1b - TSM Interrupt is enabled.
24 RX_WATERMARK_IRQ_EN	RX_WATERMARK_IRQ Enable 0b - RX Watermark Interrupt is not enabled. 1b - RX Watermark Interrupt is enabled.
23 WAKE_IRQ_EN	WAKE_IRQ Enable 0b - Wake Interrupt is not enabled. 1b - Wake Interrupt is enabled.
22 PLL_UNLOCK_IRQ_EN	PLL_UNLOCK_IRQ Enable 0b - PLL Unlock Interrupt is not enabled. 1b - PLL Unlock Interrupt is enabled.
21	T2_IRQ Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
T2_IRQ_EN	0b - Timer1 (T2) Compare Interrupt is not enabled. 1b - Timer1 (T2) Compare Interrupt is enabled.
20 T1_IRQ_EN	T1_IRQ Enable 0b - Timer1 (T1) Compare Interrupt is not enabled. 1b - Timer1 (T1) Compare Interrupt is enabled.
19 NTW_ADR_IRQ_EN	NTW_ADR_IRQ Enable 0b - Network Address Match Interrupt is not enabled. 1b - Network Address Match Interrupt is enabled.
18 RX_IRQ_EN	RX_IRQ Enable 0b - RX Interrupt is not enabled. 1b - RX Interrupt is enabled.
17 TX_IRQ_EN	TX_IRQ Enable 0b - TX Interrupt is not enabled. 1b - TX Interrupt is enabled.
16 SEQ_END_IRQ_EN	SEQ_END_IRQ Enable 0b - Sequence End Interrupt is not enabled. 1b - Sequence End Interrupt is enabled.
15 MS_IRQ	Mode Switch Interrupt SUN FSK Mode Switch Interrupt Status bit. A '1' indicates a mode switch frame is received. 0b - A Mode Switch frame is not received 1b - A Mode Switch frame is received
14 CCA_IRQ	CCA Interrupt Clear Channel Assessment Interrupt Status bit. A '1' indicates completion of CCA operation. 0b - A CCA Interrupt has not occurred 1b - A CCA Interrupt has occurred
13 FILTERFAIL_IRQ	Received Frame Filter Fail Interrupt Receiver Packet Filter Fail Interrupt Status bit. A '1' indicates that the most-recently received packet has been rejected due to elements within the packet. This is write a '1' to clear bit. In Dual PAN mode, FILTERFAIL_IRQ applies to either or both networks, as follows: A: If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_IRQ applies to PAN0. B: If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_IRQ applies to PAN1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	C: If PAN0 and PAN1 occupy the same channel, FILTERFAIL_IRQ is the logical 'AND' of the individual PANs' Filter Fail status. 0b - A Filter Fail Interrupt has not occurred. 1b - A Filter Fail Interrupt has occurred.
12 PHRFFAIL_IRQ	Received Frame PHR Fail Interrupt Asserts when the received frame fails at BCH check or Parity check, for LECIM FSK and SUN FSK Mode Switch frame only. 0b - Received frame PHR Fail Interrupt is not asserted. 1b - Received frame PHR Fail Interrupt is asserted.
11 ACK_IRQ	Auto ACK Interrupt Asserts when RX finishes, and an auto ACK frame is needed. Software may need to write packet RAM to prepare the ACK frame. 0b - Auto ACK Interrupt is not asserted. 1b - Auto ACK Interrupt is asserted.
10 CRC_VALID	CRC Valid CRC Valid indicator for RX packets. This bit becomes valid at RX_IRQ, and remains valid until the start of the next RX TSM sequence.
9 TSM_IRQ	TSM Interrupt Indicates TSM0_IRQ or TSM1_IRQ is set in XCVR_STATUS. Clear the bits there. For debug purposes. 0b - TSM0_IRQ and TSM1_IRQ are both clear. 1b - Indicates TSM0_IRQ or TSM1_IRQ is set in XCVR_STATUS.
8 RX_WATERMARK_IRQ	RX Watermark Interrupt Asserts when RX Byte Counter == RX_WATERMARK[12:0] 0b - RX Watermark Interrupt is not asserted. 1b - RX Watermark Interrupt is asserted.
7 WAKE_IRQ	Wake Interrrupt **For Manual DSM:** This is WAKE_IRQ. A WAKE_IRQ will be triggered when the GENERIC_FSK Link Layer Controller has awoken from a Manual DSM (Deep Sleep Mode) cycle. WAKE_IRQ indicates that the RF Oscillator has been restarted, and the GENERIC_FSK EVENT_TMR has resumed counting. **For Wake-On-Radio:** This is WOR_IRQ. See Section **Wake-On-Radio**. 0b - Wake Interrupt is not asserted. 1b - Wake Interrupt is asserted.
6	PLL Unlock Interrupt

Table continues on the next page...

Table continued from the previous page...

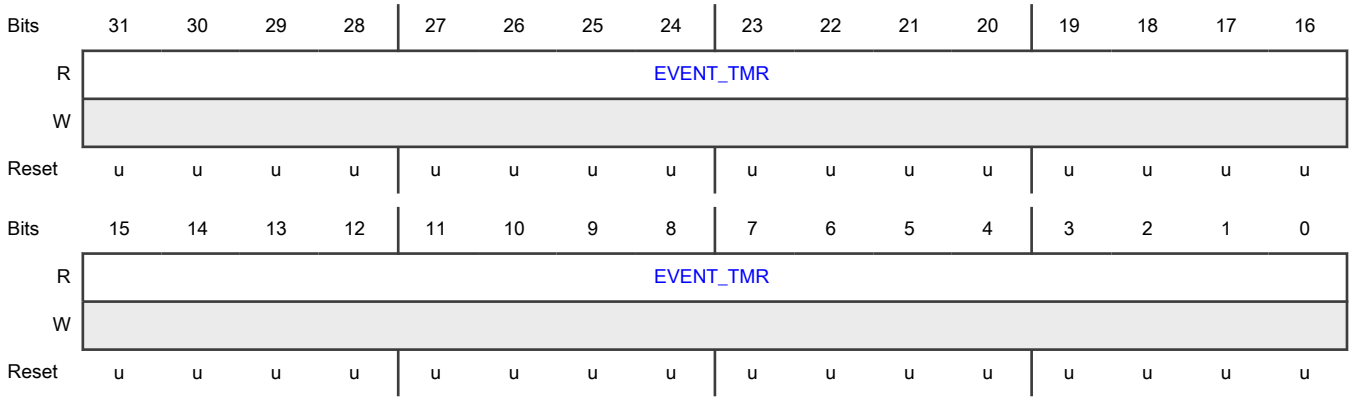
Field	Function
PLL_UNLOCK_IRQ	An unlock event has occurred. 0b - PLL Unlock Interrupt is not asserted. 1b - PLL Unlock Interrupt is asserted.
5 T2_IRQ	Timer2 (T2) Compare Interrupt 0b - Timer2 (T2) Compare Interrupt is not asserted. 1b - Timer2 (T2) Compare Interrupt is asserted.
4 T1_IRQ	Timer1 (T1) Compare Interrupt 0b - Timer1 (T1) Compare Interrupt is not asserted. 1b - Timer1 (T1) Compare Interrupt is asserted.
3 NTW_ADR_IRQ	Network Address Match Interrupt A Network Address Match has occurred. 0b - Network Address Match Interrupt is not asserted. 1b - Network Address Match Interrupt is asserted.
2 RX_IRQ	RX Interrupt The RX sequence has completed with a successful packet reception. 0b - RX Interrupt is not asserted. 1b - RX Interrupt is asserted.
1 TX_IRQ	TX Interrupt The TX sequence has completed with a successful packet transmission. 0b - TX Interrupt is not asserted. 1b - TX Interrupt is asserted.
0 SEQ_END_IRQ	Sequence End Interrupt Will assert when any TX or RX sequence ends for any reason. 0b - Sequence End Interrupt is not asserted. 1b - Sequence End Interrupt is asserted.

55.4.9.2.2.1.3 EVENT TIMER (EVENT_TMR)

Offset

Register	Offset
EVENT_TMR	4h

Diagram



Fields

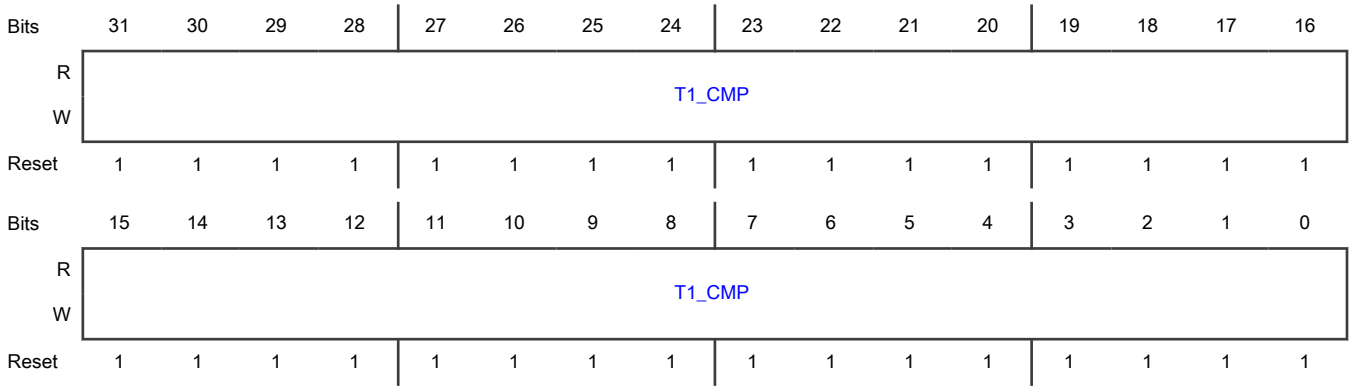
Field	Function
31-0 EVENT_TMR	Event Timer Event Timer can be read in these byte locations. To update the Event Timer, either: 1. Write the desired EVENT_TMR to register EVENT_TMR_LD[31:0], or, 2. Write the desired EVENT_TMR increment amount to regiter EVENT_TMR_ADD[31:0]. **Note:** for EVENT_TMR_ADD, EVENT_TMR[31:0] is a signed, two's-complement value.

55.4.9.2.2.1.4 T1 COMPARE (T1_CMP)

Offset

Register	Offset
T1_CMP	8h

Diagram



Fields

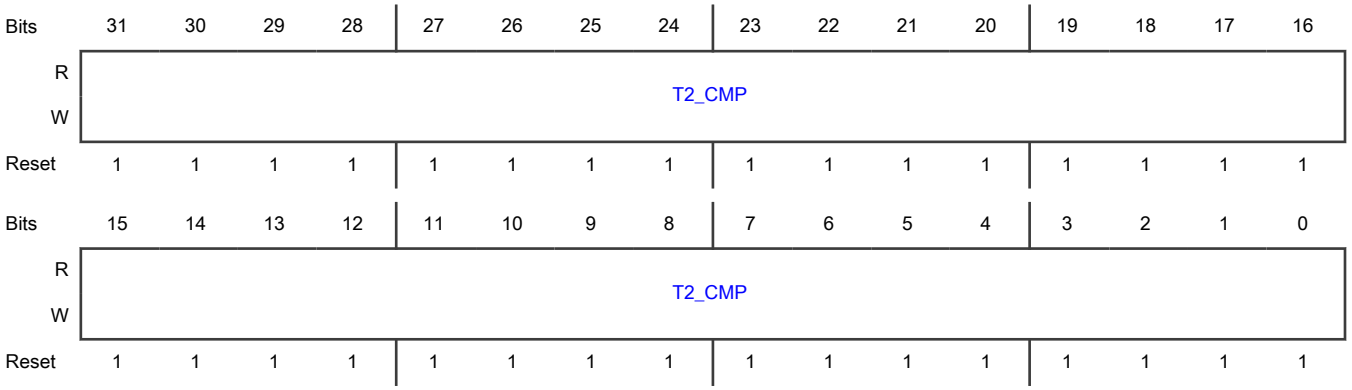
Field	Function
31-0 T1_CMP	Timer1 (T1) Compare Value Timer1 (T1) Compare Value. Can be used to generate T1_IRQ and/or launch Sequence Commands

55.4.9.2.2.1.5 T2 COMPARE (T2_CMP)

Offset

Register	Offset
T2_CMP	Ch

Diagram



Fields

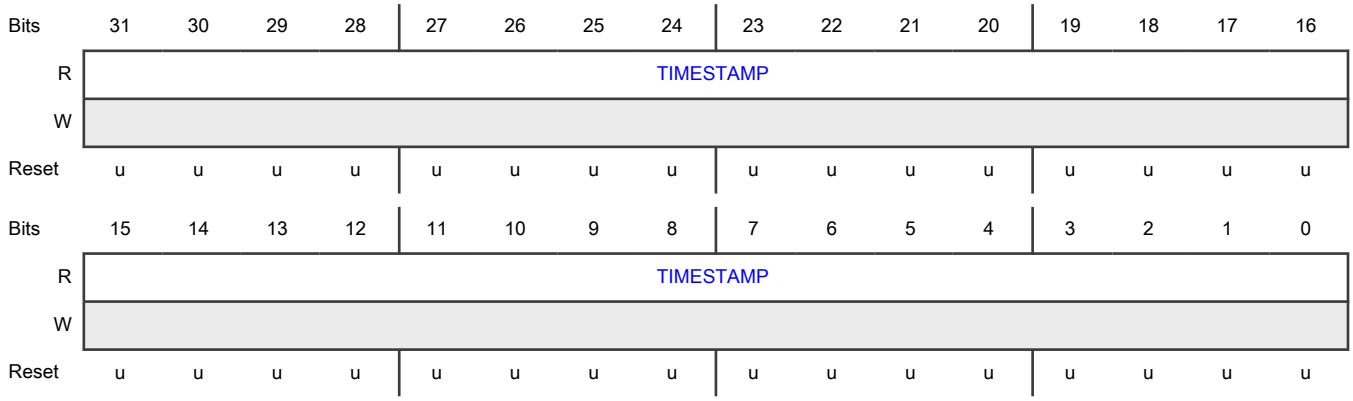
Field	Function
31-0 T2_CMP	Timer2 (T2) Compare Value Timer2 (T2) Compare Value. Can be used to generate T2_IRQ and/or launch Sequence Commands

55.4.9.2.2.1.6 TIMESTAMP (TIMESTAMP)

Offset

Register	Offset
TIMESTAMP	10h

Diagram



Fields

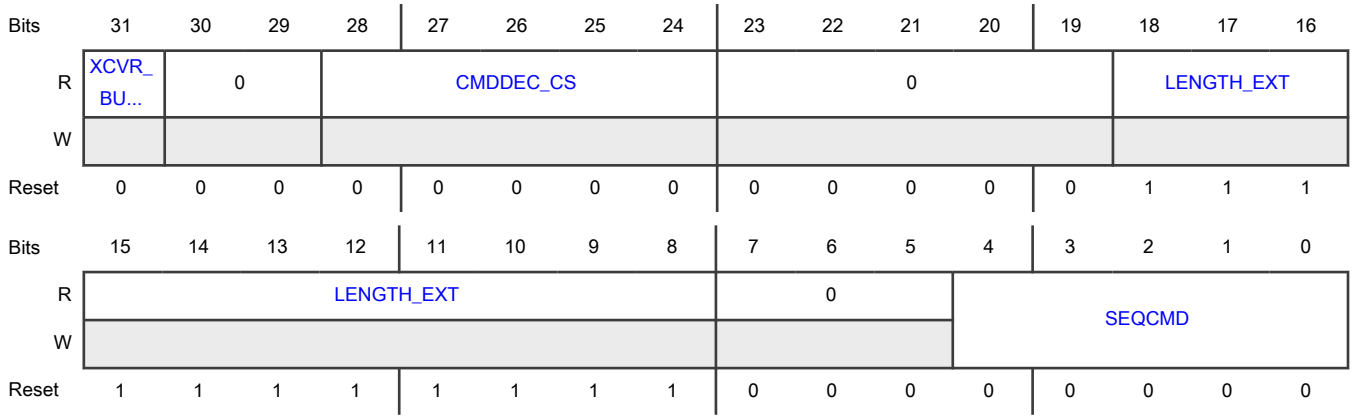
Field	Function
31-0 TIMESTAMP	Received Packet Timestamp Received Packet Timestamp. Captured at NTW_ADR_IRQ.

55.4.9.2.2.1.7 TRANSCEIVER CONTROL (XCVR_CTRL)

Offset

Register	Offset
XCVR_CTRL	14h

Diagram



Fields

Field	Function
31 XCVR_BUSY	Transceiver Busy For multi-protocol arbitration, XCVR_BUSY=1 indicates an RX or TX operation is underway, by either GENERIC_FSK, or some other protocol. 0b - IDLE 1b - BUSY
30-29 —	Reserved
28-24 CMDDEC_CS	Command Decode Current State of the Command Decoder FSM (debug only). Also named as SEQ_STATE.
23-19 —	Reserved
18-8 LENGTH_EXT	Extracted Length Field The read-only register reflects the contents of the LENGTH field of the Header of the most-recently received packet. This is the raw, unmodified LENGTH field, as it appears in the received bit stream, unmodified by LENGTH_ADJ or any other parameter.
7-5 —	Reserved
4-0 SEQCMD	Sequence Commands, also named as "XCVSEQ(Transceiver Sequence)" 0_0000b - Same as command ABORT 0_0001b - TX Start Now 0_0010b - TX Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP) 0_0011b - TX Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP) 0_0100b - TX Cancel -- Cancels pending TX events but do not abort a TX-in-progress 0_0101b - RX Start Now 0_0110b - RX Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP) 0_0111b - RX Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP) 0_1000b - RX Stop @ T1 Timer Compare Match (EVENT_TMR = T1_CMP) 0_1001b - RX Stop @ T2 Timer Compare Match (EVENT_TMR = T2_CMP) 0_1010b - RX Cancel -- Cancels pending RX events but do not abort a RX-in-progress 0_1011b - Abort All - Cancels all pending events and abort any sequence-in-progress 0_1100b - TR Start Now 0_1101b - TR Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP)

Table continues on the next page...

Table continued from the previous page...

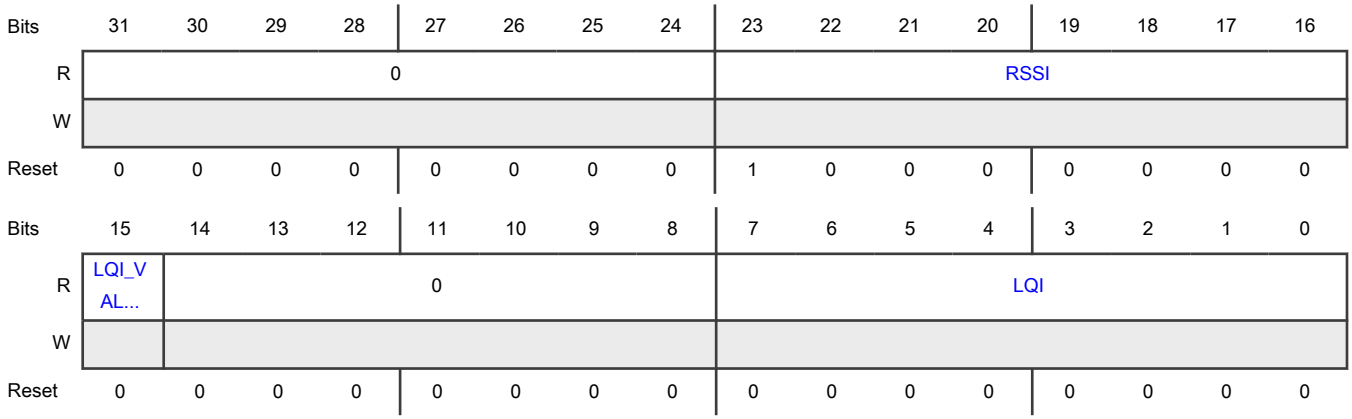
Field	Function
	0_1110b - TR Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP) 0_1111b - TR Cancel -- Cancels pending TR events but do not abort a TR-in-progress 1_0000b - CCA Start Now 1_0001b - CCA Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP) 1_0010b - CCA Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP) 1_0011b - CCA Cancel -- Cancels pending CCA events but do not abort a CCA-in-progress

55.4.9.2.2.1.8 TRANSCEIVER STATUS (XCVR_STS)

Offset

Register	Offset
XCVR_STS	18h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RSSI	RSSI Value Received Signal Strength Indicator, in dBm
15 LQI_VALID	LQI Valid Indicator

Table continues on the next page...

Table continued from the previous page...

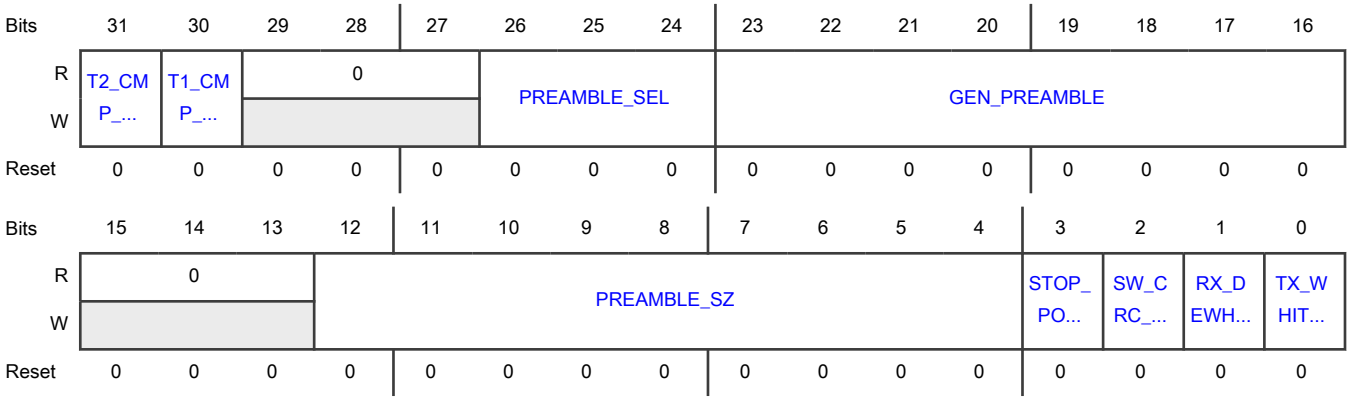
Field	Function
	LQI Valid indicator for RX packets. This bit becomes set when the LQI computation completes for the packet currently being received, and remains set for the remainder of the packet. 0b - LQI is not yet valid for RX packet. 1b - LQI is valid for RX packet.
14-8 —	Reserved
7-0 LQI	Link Quality Indicator This field is valid when LQI_VALID=1. LQI is a unsigned, unitless value.

55.4.9.2.2.1.9 TRANSCEIVER CONFIGURATION (XCVR_CFG)

Offset

Register	Offset
XCVR_CFG	1Ch

Diagram



Fields

Field	Function
31 T2_CMP_EN	Timer2 (T2) Compare Enable Enable Timer Compare #2 (T2_CMP) to generate T2_IRQ and/or execute Sequence Commands.
30 T1_CMP_EN	Timer1 (T1) Compare Enable Enable Timer Compare #1 (T1_CMP) to generate T1_IRQ and/or execute Sequence Commands.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-27 —	Reserved
26-24 PREAMBLE_SEL	Preamble Select Select the Preamble pattern. 000b - The controller hardware selects the preamble pattern based on the first transmitted bit of Network Address, such that the last bit of preamble is the opposite polarity from the first bit of Network Address, forcing a bit transition at this boundary. 001b - Preamble is programmed by register GEN_PREAMBLE[7:0] 010b - Preamble is 0b01 011b - Preamble is 0b10
23-16 GEN_PREAMBLE	Preamble pattern The repeated byte of the preamble when PREAMBLE_SEL[2:0] is 001b.
15-13 —	Reserved
12-4 PREAMBLE_SZ	Preamble Size Number of Octets(PREAMBLE_SEL is 0 or 1) or two-bits(PREAMBLE_SEL is 2 or 3) = PREAMBLE_SZ + 1, where 0 <= PREAMBLE_SZ <= 511
3 STOP_POSTPONE_ON_AA	Postpone Stop Command Timeout On Access Address Match Enable If this bit is set, if a RX STOP condition occurs, due to an Event Timer match to a previously issued RX_STOP_T1 or RX_STOP_T2 sequence command, while a packet is being received (*NTW_ADR_MCH* = 1), the timeout-related abort will be deferred untl the packet reception is completed. If the packet is good, HW will signal Data Indication; otherwise the Sequence Manager will return to IDLE and await further commanding. 0b - STOP Abort will occur on RX_STOP_T1 or RX_STOP_T1 Event Timer match, regardless of NTW_ADR_MCH 1b - STOP Abort will be deferred on RX_STOP_T1 or RX_STOP_T1 Event Timer match, if NTW_ADR_MCH is asserted; otherwise the RX_STOP Abort will occur immediately
2 SW_CRC_EN	Software CRC Enable Software override of the HW-computed CRC for TX. Software must write CRC to Packet Buffer (RAM)
1 RX_DEWHITEN_DIS	RX De-Whitening Disable Disable all de-whitening on RX packets
0	TX Whitening Disable

Table continues on the next page...

Table continued from the previous page...

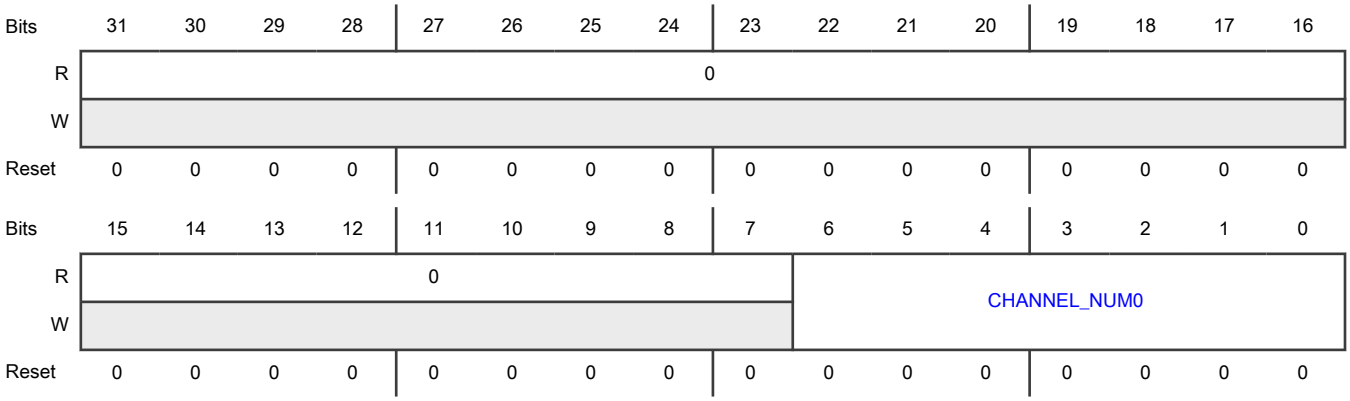
Field	Function
TX_WHITEN_DIS	Disable all whitening on TX packets

55.4.9.2.2.1.10 CHANNEL NUMBER 0 (CHANNEL_NUM0)

Offset

Register	Offset
CHANNEL_NUM0	20h

Diagram



Fields

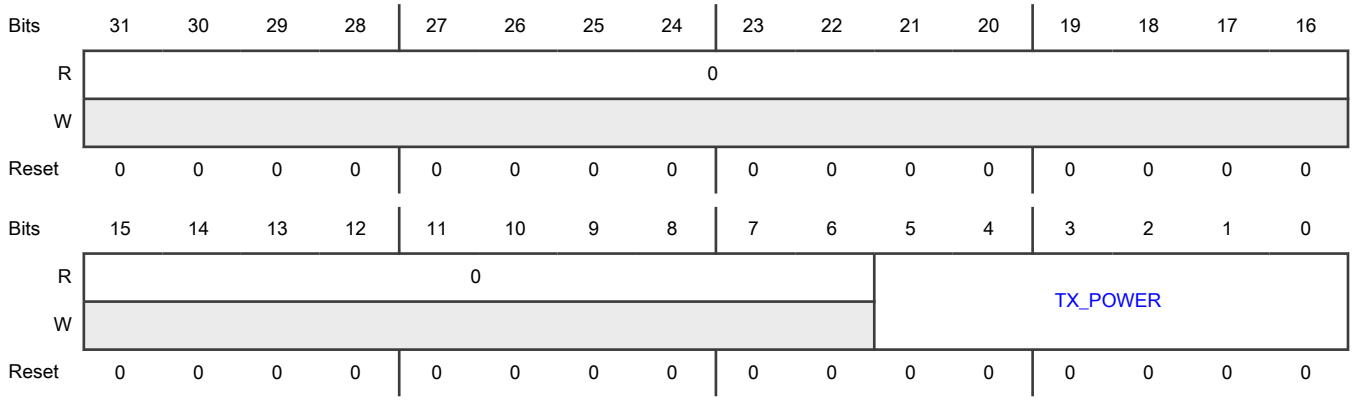
Field	Function
31-7 —	Reserved
6-0 CHANNEL_NUM0	Channel Number for PAN0 This is the mapped channel number used to transmit and receive packets. If Dual PAN is engaged, this register applies to PAN0. Formula: F = (2360 + CHANNEL_NUM0) [in MHz]

55.4.9.2.2.1.11 TRANSMIT POWER (TX_POWER)

Offset

Register	Offset
TX_POWER	24h

Diagram



Fields

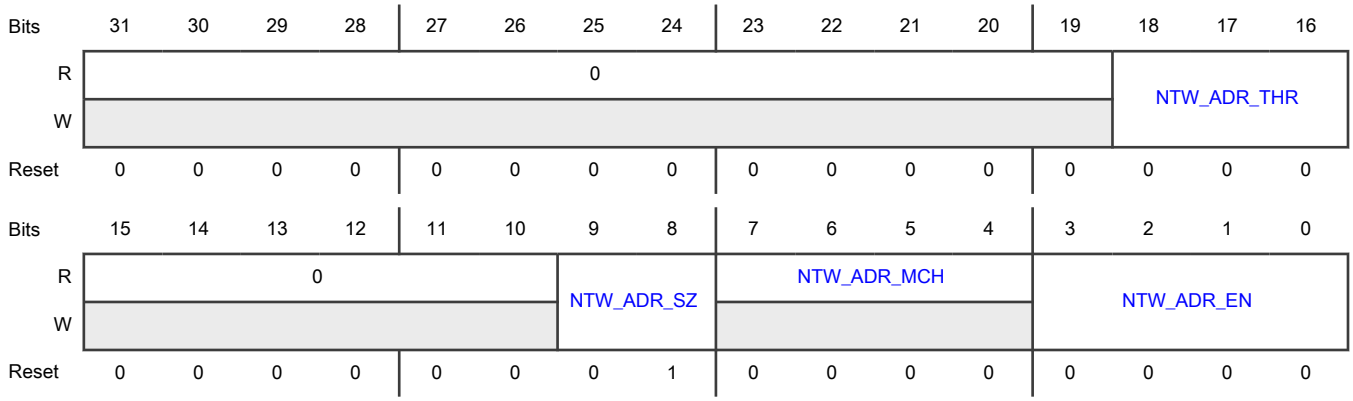
Field	Function
31-6 —	Reserved
5-0 TX_POWER	Transmit Power PA Power Level.

55.4.9.2.2.1.12 NETWORK ADDRESS CONTROL (NTW_ADR_CTRL)

Offset

Register	Offset
NTW_ADR_CTRL	28h

Diagram



Fields

Field	Function
31-19 —	Reserved
18-16 NTW_ADR_THR	Network Address Threshold Number of Tolerated bit errors for Network Address 0/1/2/3
15-10 —	Reserved
9-8 NTW_ADR_SZ	Network Address Size 00b - Network Address 0/1/2/3 requires a 8-bit correlation 01b - Network Address 0/1/2/3 requires a 16-bit correlation 10b - Network Address 0/1/2/3 requires a 24-bit correlation 11b - Network Address 0/1/2/3 requires a 32-bit correlation
7-4 NTW_ADR_MCH	Network Address Match Indicates which of the 4 Network Addresses has matched in the PHY. Valid during an RX sequence at the point of match, and remains asserted until either: 1. The next RX sequence begins (if the current packet passed CRC and header filtering), or, 2. An RX recycle to Network Address search (if the current packet failed CRC or header filtering) 0001b - Network Address 0 has matched 0010b - Network Address 1 has matched 0100b - Network Address 2 has matched 1000b - Network Address 3 has matched
3-0 NTW_ADR_EN	Network Address Enable Enable Network Address N for PHY correlation, where 0 ≤ N ≤ 3. Any bit combination can be set. 0001b - Enable Network Address 0 for correlation 0010b - Enable Network Address 1 for correlation 0100b - Enable Network Address 2 for correlation 1000b - Enable Network Address 3 for correlation

55.4.9.2.2.1.13 NETWORK ADDRESS 0 (NTW_ADR_0)

Offset

Register	Offset
NTW_ADR_0	2Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NTW_ADR_0															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NTW_ADR_0															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Fields

Field	Function
31-0 NTW_ADR_0	Network Address 0 The PHY will search for this Network Address if NTW_ADR_CTRL[NTW_ADR_EN[0]] = 1

55.4.9.2.2.1.14 NETWORK ADDRESS 1 (NTW_ADR_1)

Offset

Register	Offset
NTW_ADR_1	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NTW_ADR_1															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NTW_ADR_1															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Fields

Field	Function
31-0 NTW_ADR_1	Network Address 1 The PHY will search for this Network Address if NTW_ADR_CTRL[NTW_ADR_EN[1]] = 1

55.4.9.2.2.1.15 NETWORK ADDRESS 2 (NTW_ADR_2)

Offset

Register	Offset
NTW_ADR_2	34h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NTW_ADR_2															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NTW_ADR_2															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Fields

Field	Function
31-0 NTW_ADR_2	Network Address 2 The PHY will search for this Network Address if NTW_ADR_CTRL[NTW_ADR_EN[2]] = 1

55.4.9.2.2.1.16 NETWORK ADDRESS 3 (NTW_ADR_3)

Offset

Register	Offset
NTW_ADR_3	38h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NTW_ADR_3															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NTW_ADR_3															
Reset	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Fields

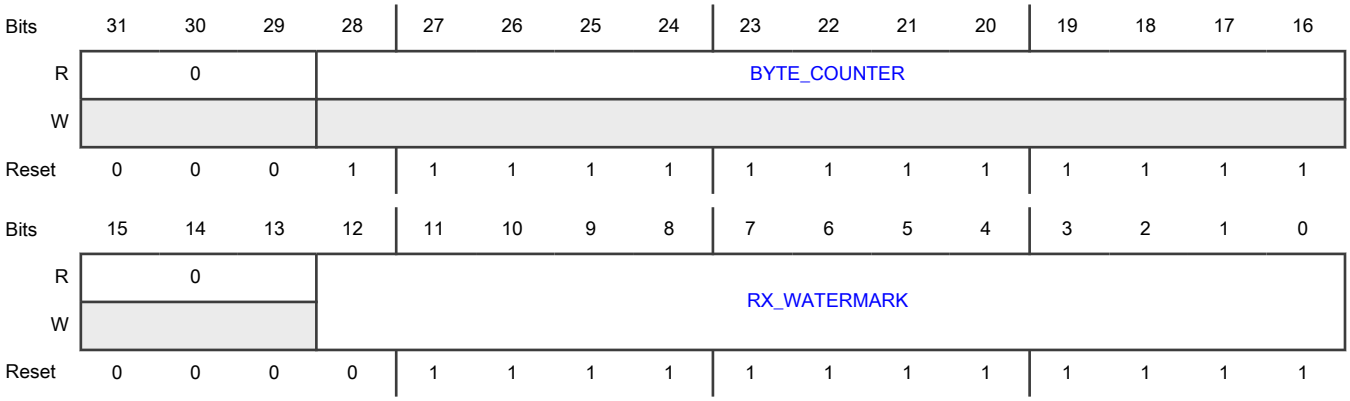
Field	Function
31-0 NTW_ADR_3	Network Address 2 The PHY will search for this Network Address if NTW_ADR_CTRL[NTW_ADR_EN[3]] = 1

55.4.9.2.2.1.17 RECEIVE WATERMARK (RX_WATERMARK)

Offset

Register	Offset
RX_WATERMARK	3Ch

Diagram



Fields

Field	Function
31-29 —	Reserved
28-16 BYTE_COUNTER	Byte Counter Reflects the current Byte Count, for TX and RX. This is a signed, twos-complement value. For values less than zero, indicates the preamble transmission is underway (TX only). A value of 0 indicates the first octet of Network Address is being transmitted or received. A value of 1 indicates the second octet of Network Address is being transmitted or received. Etc. When AA_PLAYBACK_CNT is 0, there is no Network Address sent to link layer, so a value of 0 indicates the first octet of H0 or LENGTH field; for TX, AA_PLAYBACK_CNT does not effect.
15-13 —	Reserved
12-0	Receive Watermark

Table continues on the next page...

Table continued from the previous page...

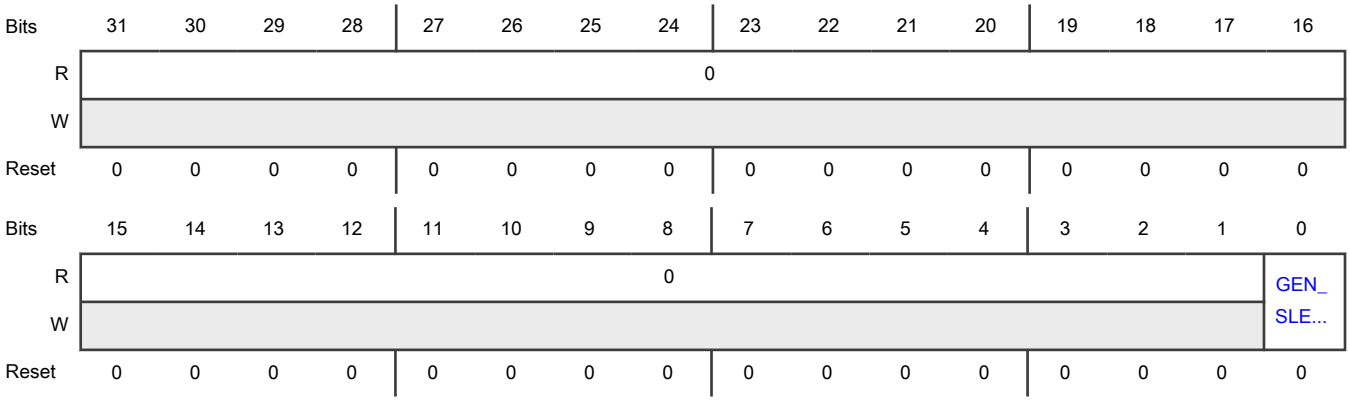
Field	Function
RX_WATERMARK	Sets the trigger for RX_WATERMARK_IRQ. Trigger the RX_WATERMARK_IRQ when: RX Byte Counter == RX_WATERMARK[12:0]

55.4.9.2.2.1.18 DSM CONTROL (DSM_CTRL)

Offset

Register	Offset
DSM_CTRL	40h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEN_SLEEP_REQUEST	GENERIC_FSK Deep Sleep Mode Request If GENERIC_FSK is assigned to Manual DSM, setting GEN_SLEEP_REQUEST to 1 enables Deep Sleep Mode (DSM) to be entered when the RFMC TIMER matches the RFMC ENTER_TIME register.

55.4.9.2.2.1.19 PART ID (PART_ID)

Offset

Register	Offset
PART_ID	44h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								PART_ID							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-8 —	Reserved
7-0 PART_ID	Part ID Part ID to identify HW revision of the Generic FSK Link Layer

55.4.9.2.2.1.20 SLOT PRELOAD (SLOT_PRELOAD)

Offset

Register	Offset
SLOT_PRELOAD	48h

Function
Slotted Mode Preload

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLOT_PRELOAD															
Reset	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0

Fields

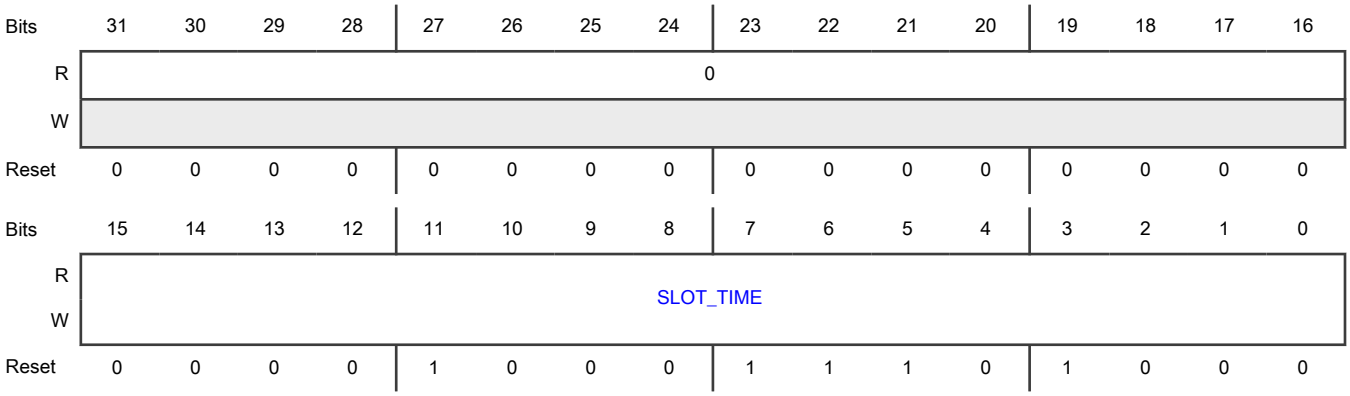
Field	Function
31-16 —	Reserved
15-0 SLOT_PRELOAD	Slotted Mode Preload This register represents the number that gets loaded into the slot_timer at SFD detect, which ultimately determines when the next slot boundary will occur. Due to processing delays within the analog front-end and digital modem, the point at which SFD is detected by the modem, is delayed relative to over-the-air timing. This register setting compensates for that delay. This timing parameter is critical for the Sequence R autosequence in slotted mode, when an automatic TxAck is required.

55.4.9.2.2.1.21 SLOT TIME (SLOT_TIME)

Offset

Register	Offset
SLOT_TIME	4Ch

Diagram



Fields

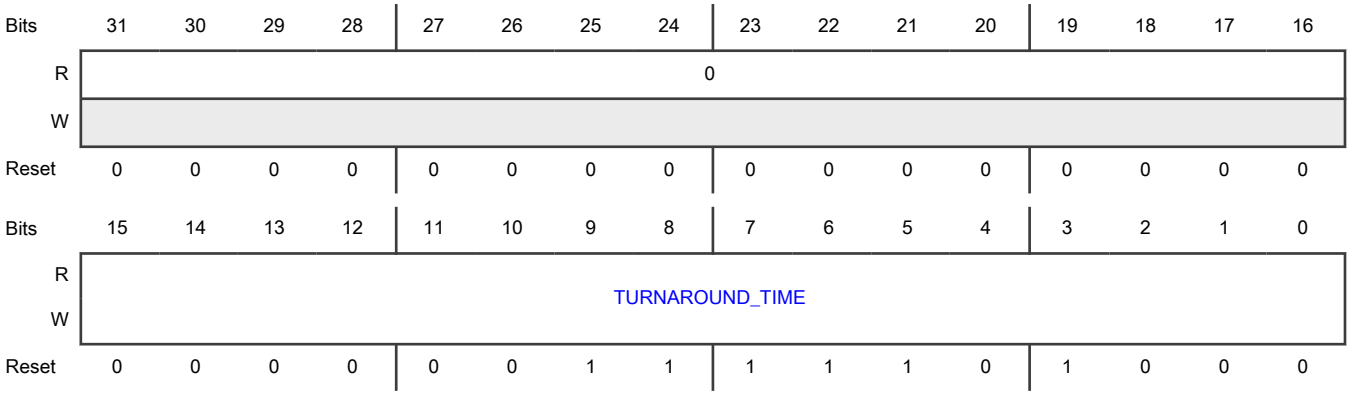
Field	Function
31-16 —	Reserved
15-0 SLOT_TIME	Duration of the Backoff Slot This register represents the time duration of the backoff slot, which is calculated by the constant *aUnitBackoffPeriod* in IEEE Std 802.15.4-2015. For all PHYs except SUN PHYs operating in the 920 MHz band, *aTurnaroundTime* + *aCcaTime*. For SUN PHYs operating in the 920 MHz band, *aTurnaroundTime* + *phyCCADuration*.

55.4.9.2.2.1.22 TURNAROUND TIME (TURNAROUND_TIME)

Offset

Register	Offset
TURNAROUND_TIME	50h

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TURNAROUND_TIME	RX-to-TX or TX-to-RX turnaround time This register represents the RX-to-TX or TX-to-RX turnaround time, which is calculated by the constant *aTurnaroundTime* in IEEE Std 802.15.4-2015.

55.4.9.2.2.1.23 ACK DELAY (ACKDELAY)

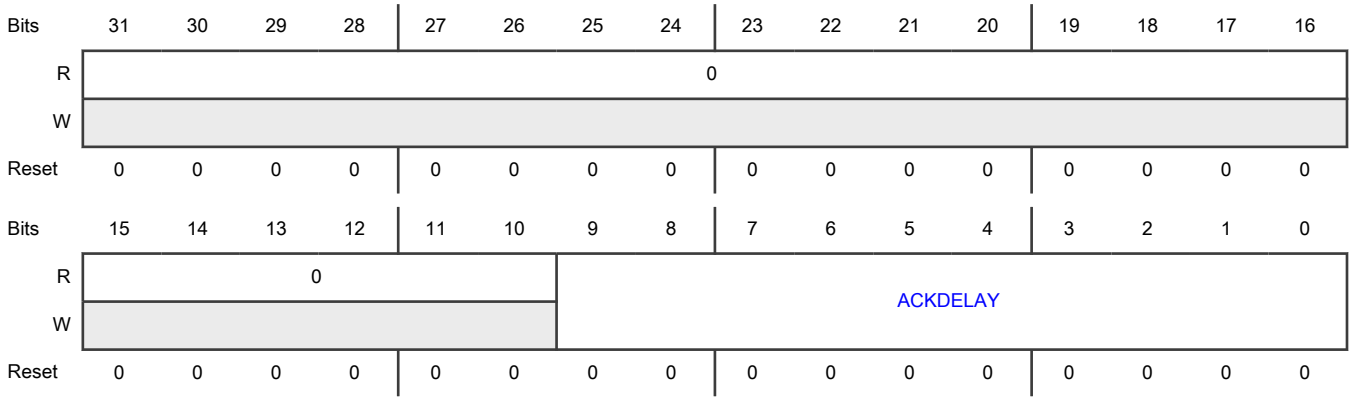
Offset

Register	Offset
ACKDELAY	54h

Function

Provides a fine-tune adjustment of the time delay between Rx warmdown and the beginning of Tx warmup for an autoTxAck packet.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 ACKDELAY	ACK Delay Provides a fine-tune adjustment of the time delay between Rx warmdown and the beginning of Tx warmup for an Tx Acknowledge packet. ACKDELAY register will apply to both SLOTTED and UNSLOTTED TxAck, but only to TxAck (not T sequences). This is a two's complement value. Resolution = 1us.

55.4.9.2.2.1.24 RX DELAY (RXDELAY)

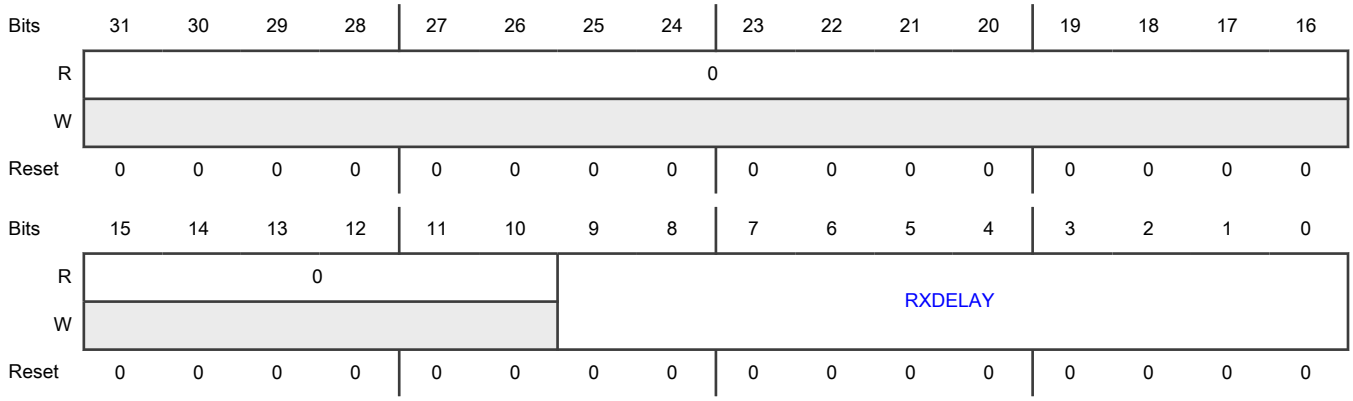
Offset

Register	Offset
RXDELAY	58h

Function

Provides a fine-tune adjustment of the time delay between Tx warmdown and the beginning of Rx warmup in a TR sequence.

Diagram



Fields

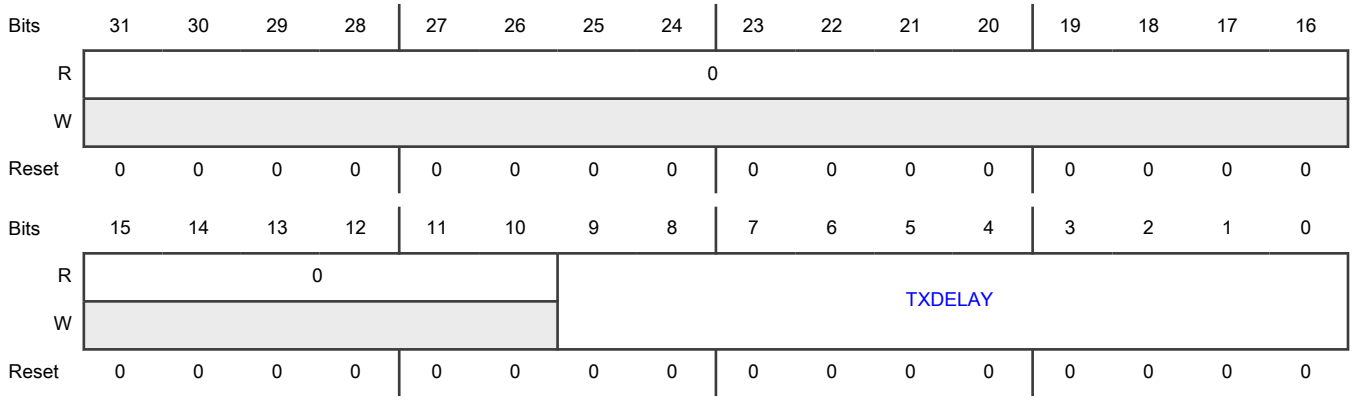
Field	Function
31-10 —	Reserved
9-0 RXDELAY	RX Delay Provides a fine-tune adjustment of the time delay between Tx warm-down and the beginning of Rx warm-up in a TR sequence. RXDELAY register will apply in both SLOTTED and UNSLOTTED modes. This is a two's complement value. Resolution = 1us.

55.4.9.2.2.1.25 TX DELAY (TXDELAY)

Offset

Register	Offset
TXDELAY	5Ch

Diagram



Fields

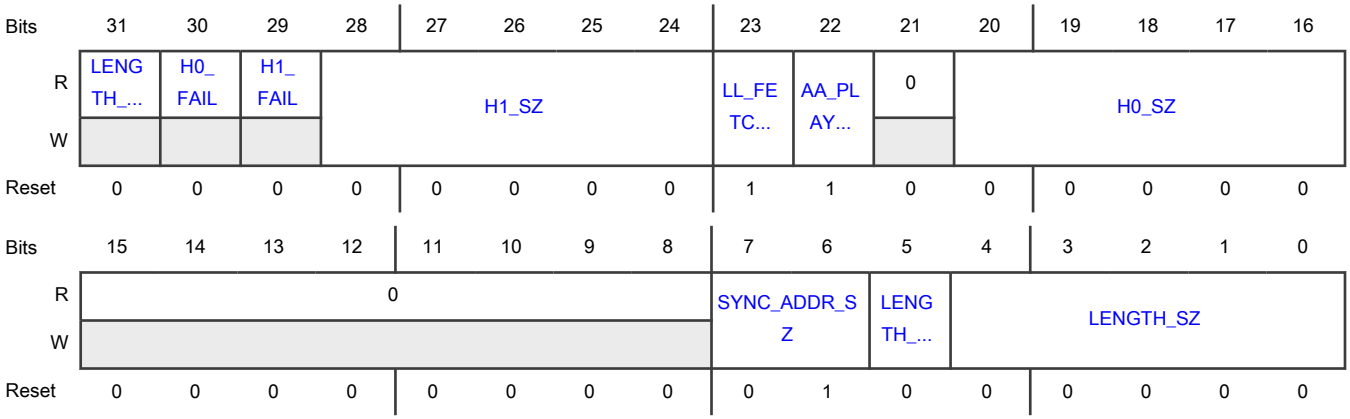
Field	Function
31-10 —	Reserved
9-0 TXDELAY	TX Delay Provides a fine-tune adjustment of the time delay between post-CCA Rx warm-down and the beginning of Tx warm-up for an Tx (non-Ack) packet. TXDELAY register will apply in both SLOTTED and UNSLOTTED modes, but only to T sequences (e.g., T, TR, and T(R)), not TxAck operations. This is a two's complement value. Resolution = 1us.

55.4.9.2.2.1.26 PACKET CONFIGURATION (PACKET_CFG)

Offset

Register	Offset
PACKET_CFG	60h

Diagram



Fields

Field	Function
31 LENGTH_FAIL	Maximum Length Violated Status Bit For packets received with REC_BAD_PKT=1, LENGTH_FAIL indicates the extracted LENGTH header field exceeded LENGTH_MAX
30 H0_FAIL	H0 Violated Status Bit For packets received with REC_BAD_PKT=1, H0_FAIL indicates the received H0 header field violates the H0_MASK/H0_MATCH pattern

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 H1_FAIL	H1 Violated Status Bit For packets received with REC_BAD_PKT=1, H1_FAIL indicates the received H1 header field violates the H1_MASK/H1_MATCH pattern
28-24 H1_SZ	H1 Size Length of H1 in bits; 0 ≤ H1_SZ ≤ 16
23 LL_FETCH_AA	Link layer fetches AA from PHY When AA_PLAYBACK_CNT is 0, link layer can fetch AA from PHY 0b - Link layer does not fetch AA from PHY 1b - Link layer fetches AA from PHY when AA_PLAYBACK_CNT is 0
22 AA_PLAYBACK_CNT	AA PLAYBACK COUNT AA is playback to Link layer 0b - AA is not through CRC and not playback to Link layer. 1b - AA is through CRC and palyback to Link Layer.
21 —	Reserved
20-16 H0_SZ	H0 Size Size of H0 in bits; 0 ≤ H0_SZ ≤ 16
15-8 —	Reserved
7-6 SYNC_ADDR_SZ	Sync Address Size Number of Octets = SYNC_ADDR_SZ + 1, 0 ≤ SYNC_ADDR_SZ ≤ 3. **NOTE** When AA_PLAYBACK_CNT = 0, the Sync Address is not played out when receive. At this case, SYNC_ADDR_SZ is ignored by link layer.
5 LENGTH_BIT_ORD	LENGTH Bit Order Bit order for the LENGTH field of the header 0b - LS Bit First 1b - MS Bit First
4-0 LENGTH_SZ	LENGTH Size Size of LENGTH field of the header, in bits; 0 ≤ LENGTH_SZ ≤ 16

55.4.9.2.2.1.27 H0 CONFIGURATION (H0_CFG)

Offset

Register	Offset
H0_CFG	64h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	H0_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	H0_MATCH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 H0_MASK	H0 Mask Register For each bit that is set to 1, the received H0 field must match the corresponding bit of H0_MATCH[15:0], else the received packet is rejected.
15-0 H0_MATCH	H0 Match Register For each bit of H0_MASK[15:0] that is set to 1, the received H0 field must match this register, else the received packet is rejected.

55.4.9.2.2.1.28 H1 CONFIGURATION (H1_CFG)

Offset

Register	Offset
H1_CFG	68h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	H1_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	H1_MATCH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 H1_MASK	H1 Mask Register For each bit that is set to 1, the received H1 field must match the corresponding bit of H1_MATCH[15:0], else the received packet is rejected.
15-0 H1_MATCH	H1 Match Register For each bit of H1_MASK[15:0] that is set to 1, the received H1 field must match this register, else the received packet is rejected.

55.4.9.2.2.1.29 CRC CONFIGURATION (CRC_CFG)

Offset

Register	Offset
CRC_CFG	6Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	CRC_VAL...	0	0	0	CRC_IGN...	0							
W																
Reset	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

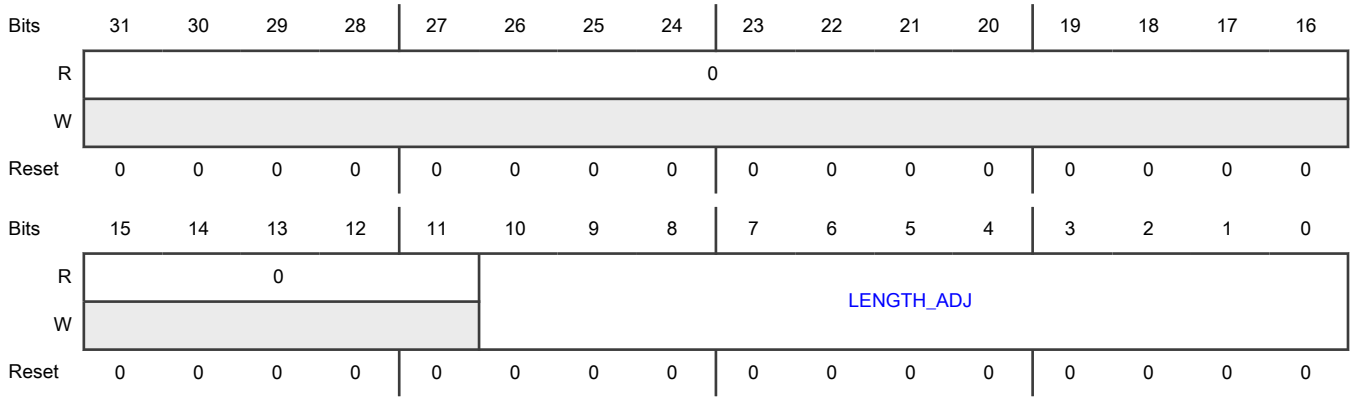
Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 CRC_VALID	CRC Valid CRC Valid indicator for RX packets. This bit becomes valid at RX_IRQ, and remains valid until the start of the next RX TSM sequence. 0b - CRC of RX packet is not valid. 1b - CRC of RX packet is valid.
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 CRC_IGNORE	CRC Ignore If set, assert RX_IRQ even for a received packet which fails CRC verification. 0b - RX_IRQ will not be asserted for a received packet which fails CRC verification. 1b - RX_IRQ will be asserted even for a received packet which fails CRC verification.
23-0 —	Reserved

55.4.9.2.2.1.30 LENGTH ADJUSTMENT (LENGTH_ADJ)

Offset

Register	Offset
LENGTH_ADJ	70h

Diagram



Fields

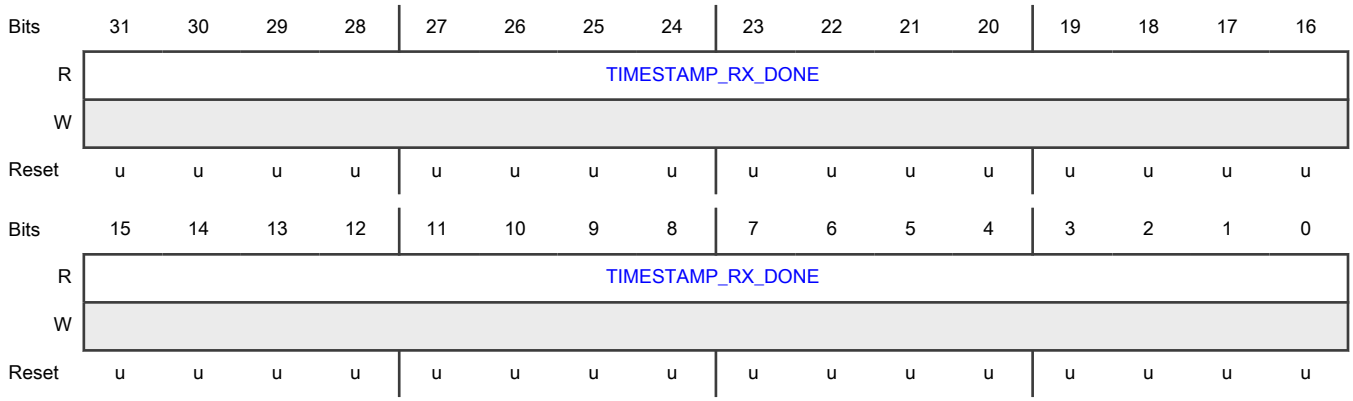
Field	Function
31-11 —	Reserved
10-0 LENGTH_ADJ	Length Adjustment Signed Adjustment to the LENGTH field for TX and RX. A value of 0 (default) means LENGTH is interpreted as PAYLOAD + CRC. **NOTE** When used as negative number, it is a two's complement value. And user should ensure, 0 <= (LENGTH + LENGTH_ADJ) <= 2047

55.4.9.2.2.1.31 TIMESTAMP_RX_DONE (TIMESTAMP_RX_DONE)

Offset

Register	Offset
TIMESTAMP_RX_DONE	74h

Diagram



Fields

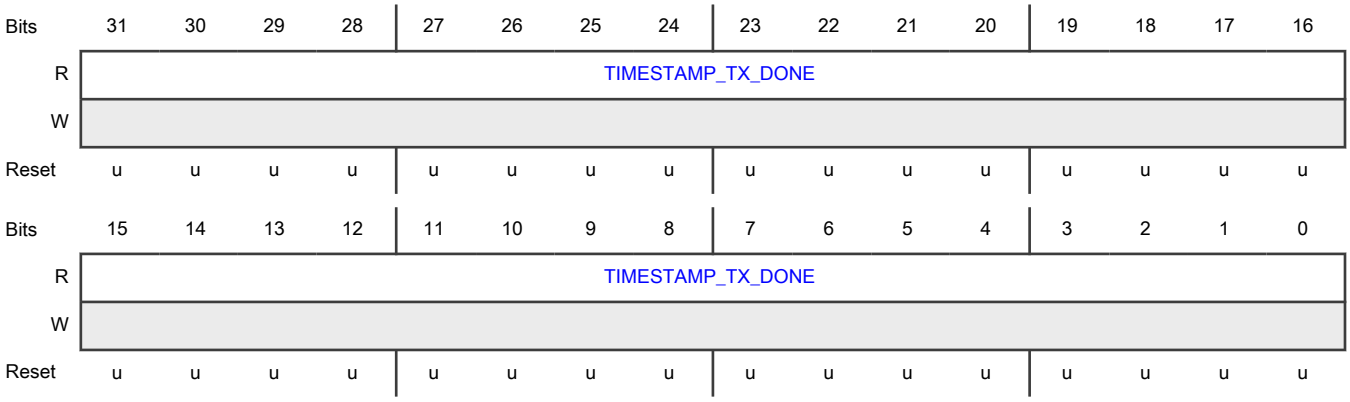
Field	Function
31-0 TIMESTAMP_RX_DONE	Received Packet Timestamp. Captured at Rx done.

55.4.9.2.2.1.32 TIMESTAMP_TX_DONE (TIMESTAMP_TX_DONE)

Offset

Register	Offset
TIMESTAMP_TX_DONE	78h

Diagram



Fields

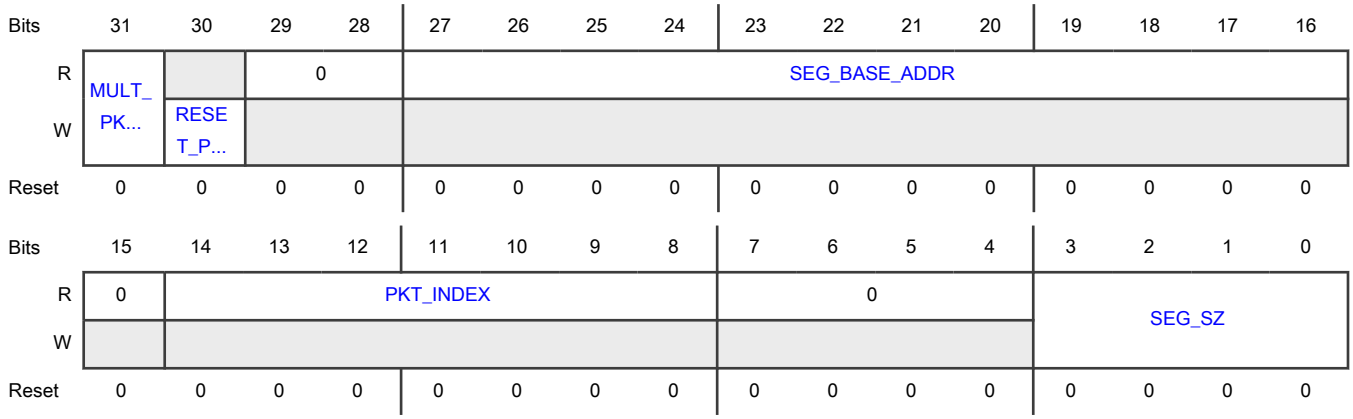
Field	Function
31-0 TIMESTAMP_TX_DONE	Received Packet Timestamp. Captured at Tx done.

55.4.9.2.2.1.33 MULT_PKT_CTRL (MULT_PKT_CTRL)

Offset

Register	Offset
MULT_PKT_CTRL	7Ch

Diagram



Fields

Field	Function
31 MULT_PKT_EN	Enable to send or receive multiple packets 0b - Send or receive multiple packets is not enabled. 1b - Send or receive multiple packets is enabled.
30 RESET_PKT_IDX	Reset the PKT_INDEX to zero This bit is used to reset the PKT_INDEX before a new round of packet reception.
29-28 —	Reserved
27-16 SEG_BASE_ADDR	Segment Offset Address Shows the current RAM segment offset address(regarding to the base address).
15 —	Reserved
14-8 PKT_INDEX	Packet Index Shows the current RAM segment index number.
7-4 —	Reserved
3-0 SEG_SZ	RAM Segment Size Define the RAM segment size for each packet: (SEG_SZ+1)*32 Bytes. For example, when SEG_SZ is 0, the segment size is 32 Bytes; when SEG_SZ is 1, the segment size is 64 Bytes, etc.

55.4.9.2.2.1.34 RPA AND WHITE LIST STATUS (RPA_WL_STATUS)

Offset

Register	Offset
RPA_WL_STATUS	80h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEARCH_...	0			LOCAL_RESOLVED_INDEX				0				PEER_RESOLVED_INDEX			
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										WL_MATCH_INDEX					
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31 SEARCH_WL	Search Identity Address in White List Write 1 to SEARCH_WL will trigger the hardware to search for the identity address in the selected white list. After the search operation completes, SEARCH_WL is cleared by hardware.
30-28 —	Reserved
27-24 LOCAL_RESOLVED_INDEX	The matched RPA index of local address If the value is not 0xF, this register shows the matched RPA index of the local address in the received packet.
23-20 —	Reserved
19-16 PEER_RESOLVED_INDEX	The matched RPA index of peer address If the value is not 0xF, this register shows the matched RPA index of the peer address in the received packet.
15-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

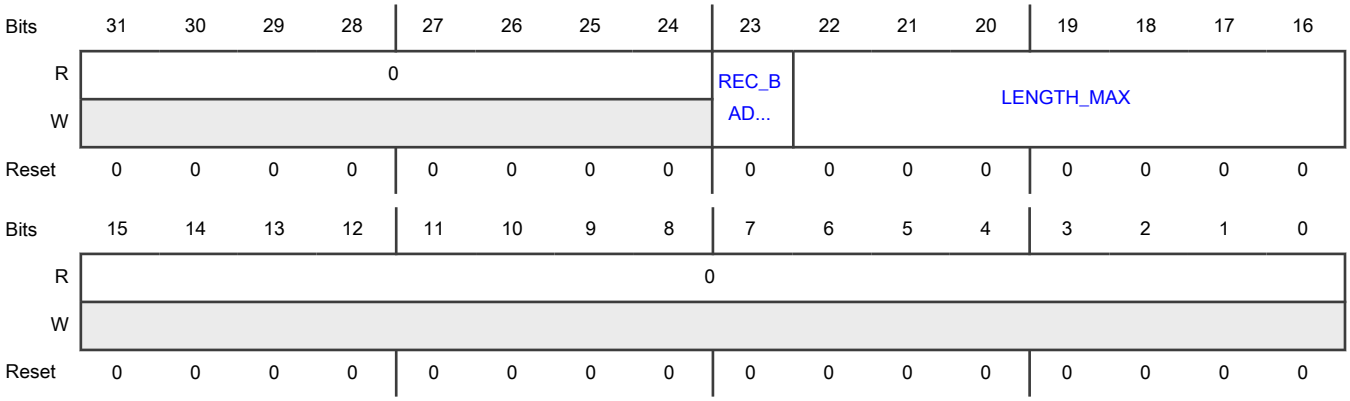
Field	Function
5-0 WL_MATCH_INDEX	The matched white list index of the identity address resolved(RPA is enabled) or peer address received(RPA is not enabled) If the value is not 0x3F, this register shows the matched white list index.

55.4.9.2.2.1.35 MAXIMUM LENGTH (LENGTH_MAX)

Offset

Register	Offset
LENGTH_MAX	84h

Diagram



Fields

Field	Function
31-24 —	Reserved
23 REC_BAD_PKT	Receive Bad Packets Enable Packets which fail H0-filtering, H1-filtering, or Maximum Length-filtering, to be fully received without an RX recycle (intended for debug purposes) 0b - packets which fail H0, H1, or LENGTH_MAX result in an automatic recycle after the header is received and parsed 1b - packets which fail H0, H1, or LENGTH_MAX are received in their entirety
22-16 LENGTH_MAX	Maximum Length for Received Packets

Table continues on the next page...

Table continued from the previous page...

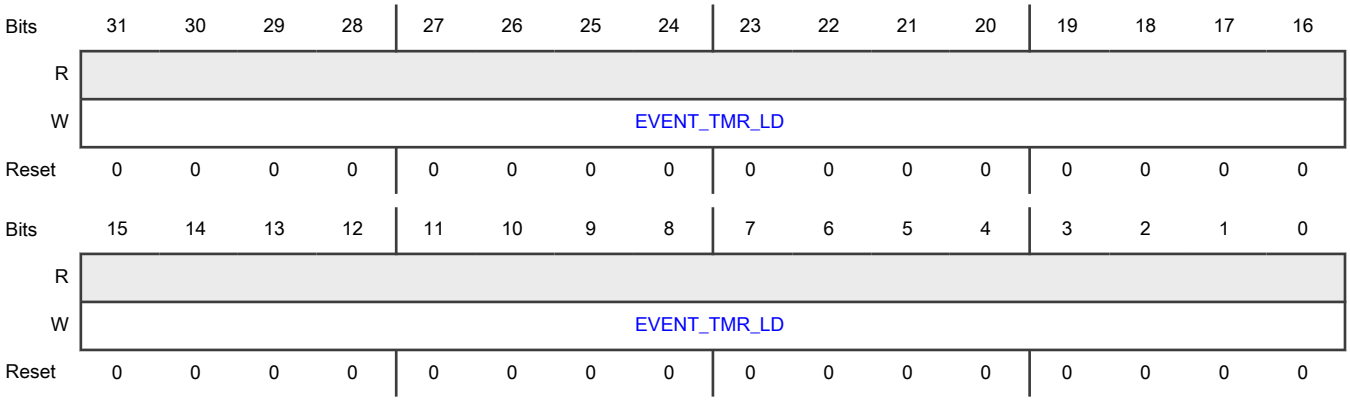
Field	Function
	Sets the Maximum Length Packet that can be received, in multiples of 16 bytes. LENGTH_MAX is compared directly against the extracted LENGTH field of the header (not the adjusted length). LENGTH_MAX=0 (default) is a special case that implies no limit.
15-0 —	Reserved

55.4.9.2.2.1.36 EVENT TIMER LOAD (EVENT_TMR_LD)

Offset

Register	Offset
EVENT_TMR_LD	88h

Diagram



Fields

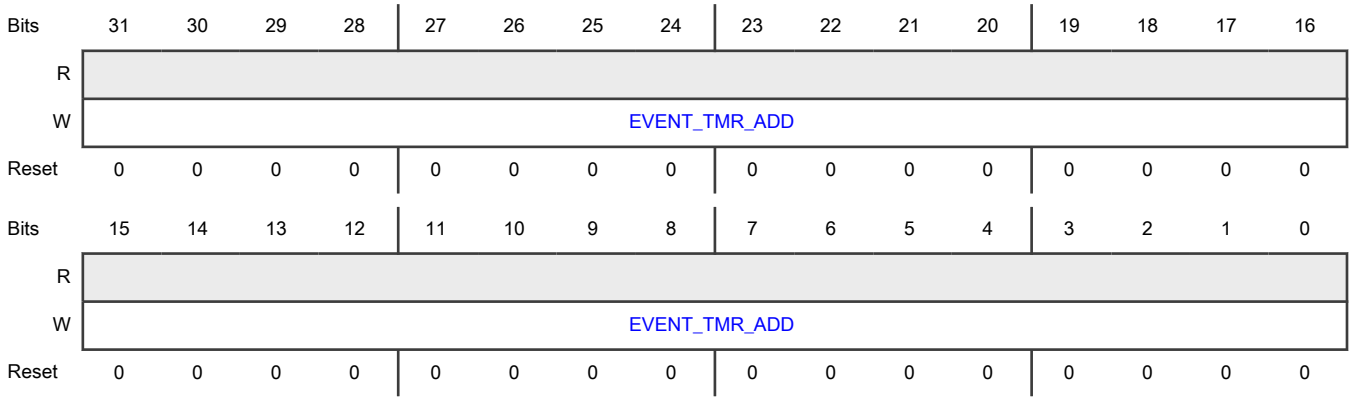
Field	Function
31-0 EVENT_TMR_LD	Event Timer Load A write access to this register loads EVENT_TMR with the contents of EVENT_TMR_LD[31:0]

55.4.9.2.2.1.37 EVENT TIMER ADD (EVENT_TMR_ADD)

Offset

Register	Offset
EVENT_TMR_ADD	8Ch

Diagram



Fields

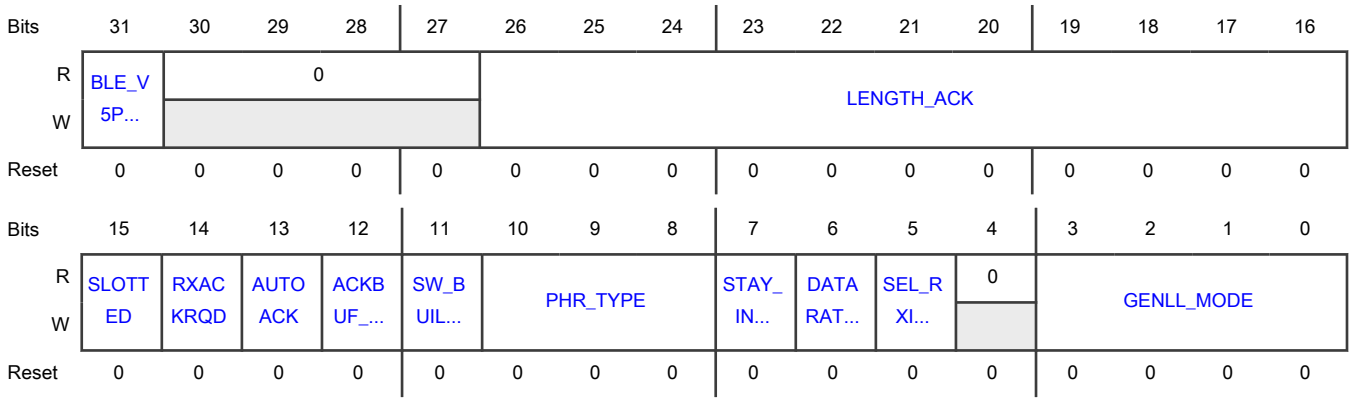
Field	Function
31-0 EVENT_TMR_ADD	Event Timer Add A write access of this register increments EVENT_TMR by the contents of EVENT_TMR_ADD[31:0]. This is a signed addition.

55.4.9.2.2.1.38 ENHANCED FEATURES (ENH_FEATURE)

Offset

Register	Offset
ENH_FEATURE	90h

Diagram



Fields

Field	Function
31 BLE_V5P1_CTE_EN	Bluetooth LE version 5.1 CTE feature enable 0b - Do not support Bluetooth LE version 5.1 CTE feature. 1b - Support Bluetooth LE version 5.1 CTE feature, which means the link layer hardware can parse the CTE field length and extend the RX_EN signal accordingly.
30-27 —	Reserved
26-16 LENGTH_ACK	Length of the ACK frame(or part of the ACK frame) in RAM In GLL mode, it is programed into the LENGTH field in PHR. So, if ACK's payload length is Payload_length, LENGTH_ACK = Payload_length + CRC_SZ - LENGTH_ADJ. In FAN-LL mode, when SW_BUILD_ACK=0, it is the length of ASH and IEs field in PHR. In GTM mode, it is the length of the test packet payload.
15 SLOTTED	Slotted Mode Slotted Mode, for beacon-enabled networks. Applies only to Sequences T, TR, and R, ignored during all other sequences. Used, in concert with CCABFRTX, to determine how many CCA measurements are required prior to a transmit operation. Also used during R sequence to determine whether the ensuing transmit acknowledge frame (if any) needs to be synchronized to a backoff slot boundary.
14 RXACKRQD	Receive Acknowledge Frame required Applies only to Sequence TR, ignored during all other sequences. 0b - An ordinary receive frame (any type of frame) follows the transmit frame. 1b - A receive Ack frame is expected to follow the transmit frame (non-Ack frames are rejected).
13 AUTOACK	Auto Acknowledge Enable Applies only to Sequence R and Sequence TR, ignored during other sequences 0b - sequence manager will not follow a receive frame with a Tx Ack frame, under any conditions; the autosequence will terminate after the receive frame. 1b - sequence manager will follow a receive frame with an automatic hardware-generated Tx Ack frame, assuming other necessary conditions are met.
12 ACKBUF_SEL	ACK frame is in 64-byte dedicated RAM or TX buffer RAM 0b - ACK frame is in 64-byte dedicated RAM 1b - ACK frame is in TX buffer RAM
11 SW_BUILD_ACK	Software builds the ACK packet in RAM 0b - Hardware builds part of or the whole of the auto ACK frame 1b - Software builds the whole auto ACK frame in RAM.
10-8 PHR_TYPE	PHR Type Specify the PHR type in PAN or FAN mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - The packet type is GFSK 001b - The packet type is MSK 010b - The packet type is SUN FSK 011b - The packet type is LECIM FSK
7 STAY_IN_RX	Stay in receive Enable the ability to recycle and stay in receive even after an RX_IRQ, to avoid the full RX re-warmup. 0b - Linklayer will warmdown after an RX_IRQ 1b - Linklayer will recycle and stay in receive even after an RX_IRQ.
6 DATARATE_CONFIG_SEL	Select the data rate configuration bank 0b - Select the data rate as per configuration bank 0 1b - Select the data rate as per configuration bank 1
5 SEL_RXIRQ	Select the RX IRQ assert time **NOTE** This bit must not be changed when RX or TX are active. Unpredictable behavior will ensue. 0b - RX_IRQ is asserted at the end of RX_PKT state. 1b - RX_IRQ is asserted at the end of RXEN_DLY state. This to be used for delaying RX_IRQ to accept TERM2 bits in Bluetooth LE-LR and CTE bits as needed.
4 —	Reserved
3-0 GENLL_MODE	Linklayer Mode Select 0000b - GLL Mode 0001b - PAN Mode 0010b - FAN Mode 0011b - Hybrid Dual PAN Mode 0100b - Reserved 0101b - Reserved 0110b - FCP Mode 0111b - Reserved 1000b - Reserved 1001b - Bluetooth LE Uncoded Mode 1010b - Bluetooth LE LR Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1011b - Bluetooth LE Concurrent Mode (RX configuration only; TX uses either Bluetooth LE UNCODED or Bluetooth LE LR configuration) 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - GTM Mode

55.4.9.2.2.1.39 RECEIVE FRAME FILTER (RX_FRAME_FILTER)

Offset

Register	Offset
RX_FRAME_FILTER	94h

Function
RECEIVE FRAME FILTER

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENH_PKT...	PROMISC...	FILTER_...	RXCYC_S...	0				EXTENDE...	FRAGMEN...	MULTIPU...	LLDN_RE...	FV2_CMD...	FV2_ACK...	FV2_DAT...	FV2_BEA...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXTENDE...	0			FRM_VER_FILTER			NS_FT	EXTENDE...	FRAGMEN...	MULTIPU...	LLDN_FT	CMD_FT	ACK_FT	DATA_FT	BEACON_...
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31 ENH_PKT_STATUS	Enhanced Packet Status 0b - The last packet received was not 2015-compliant 1b - The last packet received was 2015-compliant (RX_FRAME_FILTER register should be queried for additional status bits)
30	Promiscuous Mode Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
PROMISCUOUS	Bypasses most packet filtering. 0b - normal mode 1b - all packet filtering except frame length checking (FrameLength>=5) is bypassed.
29 FILTER_FAIL_IGNORE	Filter Fail Ignore If set, assert RX_IRQ even when filter fail. 0b - RX_IRQ will not be asserted when filter fail. 1b - RX_IRQ will be asserted when filter fail.
28 RXCYC_SEL	Rx Recycle Time Select 0b - Recycle when fail happens. 1b - Recycle when Rx done and fail happens.
27-24 —	Reserved
23 EXTENDED_RECD	Extended Packet Received 0b - The last packet received was not Frame Type EXTENDED 1b - The last packet received was Frame Type EXTENDED, and EXTENDED_FT=1 to allow such packets.
22 FRAGMENT_RECD	Fragment Packet Received 0b - last packet received was not Frame Type FRAGMENT 1b - The last packet received was Frame Type FRAGMENT, and FRAGMENT_FT=1 to allow such packets.
21 MULTIPURPOSE_RECD	Multipurpose Packet Received 0b - last packet received was not Frame Type MULTIPURPOSE 1b - The last packet received was Frame Type MULTIPURPOSE, and MULTIPURPOSE_FT=1 to allow such packets.
20 LLDN_RECD	LLDN Packet Received 0b - The last packet received was not Frame Type LLDN 1b - The last packet received was Frame Type LLDN, and LLDN_FT=1 to allow such packets.
19 FV2_CMD_RECD	Frame Version 2 MAC Command Packet Received 0b - The last packet received was not Frame Type MAC Command with Frame Version 2 1b - The last packet received was Frame Type MAC Command with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
18	Frame Version 2 Acknowledge Packet Received

Table continues on the next page...

Table continued from the previous page...

Field	Function
FV2_ACK_RECD	0b - The last packet received was not Frame Type Ack with Frame Version 2 1b - The last packet received was Frame Type Ack with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
17 FV2_DATA_RECD	Frame Version 2 Data Packet Received 0b - The last packet received was not Frame Type Data with Frame Version 2 1b - The last packet received was Frame Type Data with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
16 FV2_BEACON_RECD	Frame Version 2 Beacon Packet Received 0b - The last packet received was not Frame Type Beacon with Frame Version 2 1b - The last packet received was Frame Type Beacon with Frame Version 2, and FRM_VER_FILTER[2]=1 to allow such packets
15 EXTENDED_FCS_CHK	Verify FCS on Frame Type Extended 0b - Packet Processor will not check FCS for Frame Type EXTENDED (default) 1b - Packet Processor will check FCS at end-of-packet based on packet length derived from PHR, for Frame Type EXTENDED
14-13 —	Reserved
12-9 FRM_VER_FILTER	Frame Version selector. The incoming packet's Frame Control Field is parsed to obtain the FrameVersion subfield, and that value is compared against this register, in accordance with the following: xxx1: Accept received packets with FrameVersion=00 xx1x: Accept received packets with FrameVersion=01 x1xx: Accept received packets with FrameVersion=10 1xxx: Accept received packets with FrameVersion=11 These filtering rules apply to Beacon, Acknowledge, Data, and MAC Command Frame Types only, since these frame types require a 2-octet Frame Control Field which embeds a 2-bit FrameVersion subfield. For Acknowledge frames, FrameVersion bits are ignored by the Packet Processor, irrespective of FRM_VER_FILTER.
8 NS_FT	"Not Specified" Frame Type Enable This bit enables reception of all Frame Types not covered by the other _FT bits in this register. 0b - reject all "Not Specified" frames 1b - Not-specified (reserved) frame type enabled. Applies to Frame Type 6. No packet filtering is performed, except for frame length checking (FrameLength>=5 and FrameLength<=127). No AUTOACK is transmitted for this Frame Type
7	Extended Frame Type Enable

Table continues on the next page...

Table continued from the previous page...

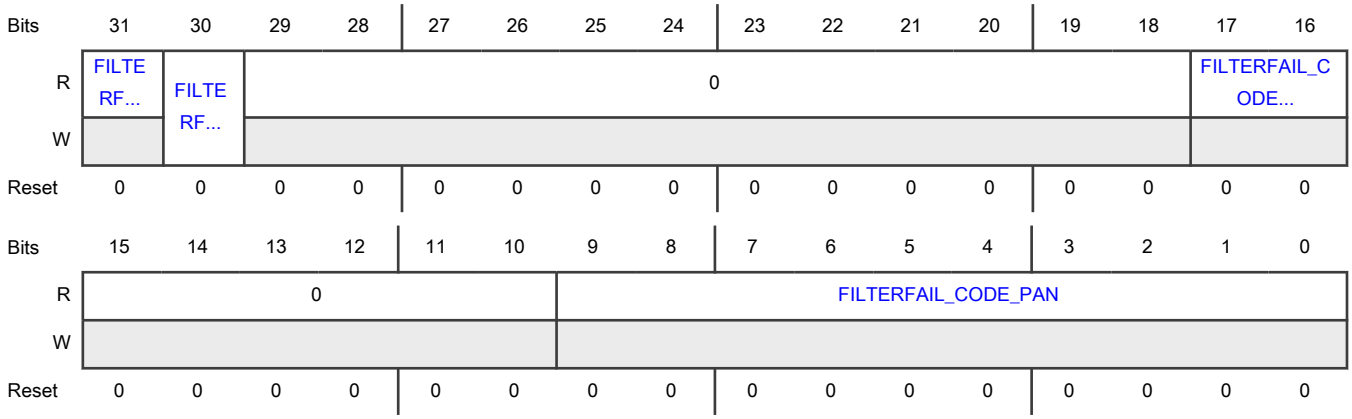
Field	Function
EXTENDED_FT	0b - reject all Extended frames 1b - Extended frame type enabled (Frame Type 7).
6 FRAGMENT_FT	Fragment Frame Type Enable 0b - reject all Fragment frames 1b - Fragment frame type enabled (Frame Type 6).
5 MULTIPURPOSE_FT	Multipurpose Frame Type Enable 0b - reject all Multipurpose frames 1b - Multipurpose frame type enabled (Frame Type 5).
4 LLDN_FT	LLDN Frame Type Enable 0b - reject all LLDN frames 1b - LLDN frame type enabled (Frame Type 4).
3 CMD_FT	MAC Command Frame Type Enable 0b - reject all MAC Command frames 1b - MAC Command frame type enabled.
2 ACK_FT	Ack Frame Type Enable 0b - reject all Acknowledge frames 1b - Acknowledge frame type enabled.
1 DATA_FT	Data Frame Type Enable 0b - reject all Beacon frames 1b - Data frame type enabled.
0 BEACON_FT	Beacon Frame Type Enable 0b - reject all Beacon frames 1b - Beacon frame type enabled.

55.4.9.2.2.1.40 FILTER FAIL CODE (FILTERFAIL_CODE)

Offset

Register	Offset
FILTERFAIL_CODE	98h

Diagram



Fields

Field	Function
31 FILTERFAIL_FLAG_SEL	Consolidated Filter Fail Flag 0: The incoming, or just-received packet, passed packet filtering rules. 1: The incoming, or just-received packet, failed packet filtering rules When FILTERFAIL_FLAG_SEL=1, a non-zero FILTERFAIL_CODE is present (see FILTERFAIL_CODE registers). In Dual PAN mode, FILTERFAIL_FLAG_SEL applies to either or both networks, as follows: **A:** If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_FLAG_SEL applies to PAN0. **B:** If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_FLAG_SEL applies to PAN1. **C:** If PAN0 and PAN1 occupy the same channel, FILTERFAIL_FLAG_SEL is the logical 'AND' of the individual PANs' FILTERFAIL_FLAG bits.
30 FILTERFAIL_PAN_SEL	PAN Selector for Filter Fail Code 0b - FILTERFAIL_CODE_PAN/FILTERFAIL_CODE_FAN will report the FILTERFAIL status of PAN0 1b - FILTERFAIL_CODE_PAN/FILTERFAIL_CODE_FAN will report the FILTERFAIL status of PAN1
29-18 —	Reserved
17-16 FILTERFAIL_CODE_FAN	Filter Fail Code When in FAN Mode Code inidicating what condition, or conditions, caused the Packet Processor to reject the just-received packet in FAN mode. The bits of FILTERFAIL_CODE_FAN indicate the reason for packet rejection.
15-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 FILTERFAIL_CODE_PAN	Filter Fail Code When in PAN Mode Code inidicating what condition, or conditions, caused the Packet Processor to reject the just-received packet in PAN mode. The bits of FILTERFAIL_CODE_PAN indicate the reason for packet rejection.

55.4.9.2.2.1.41 LENIENCY LSB (LENIENCY_LSB)

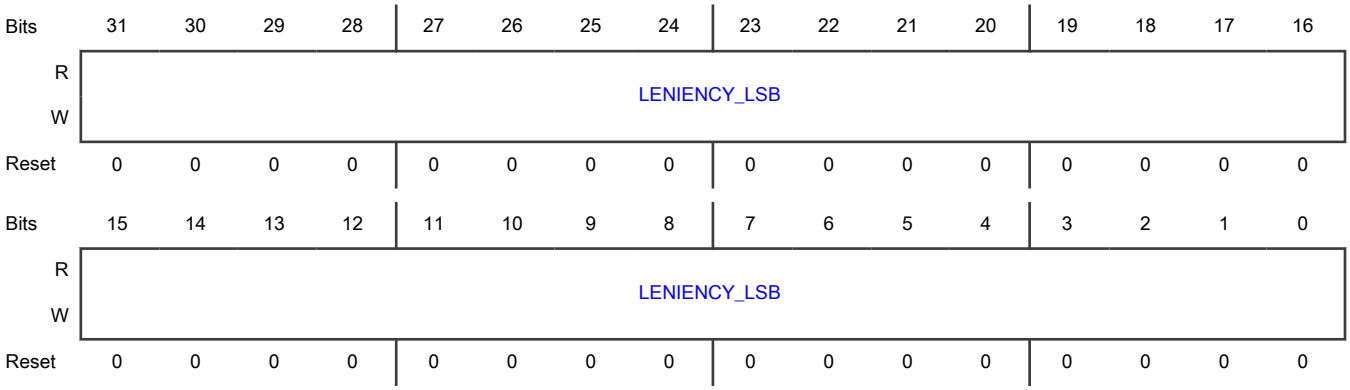
Offset

Register	Offset
LENIENCY_LSB	9Ch

Function

Packet Processor Leniency Bits (LSB)

Diagram



Fields

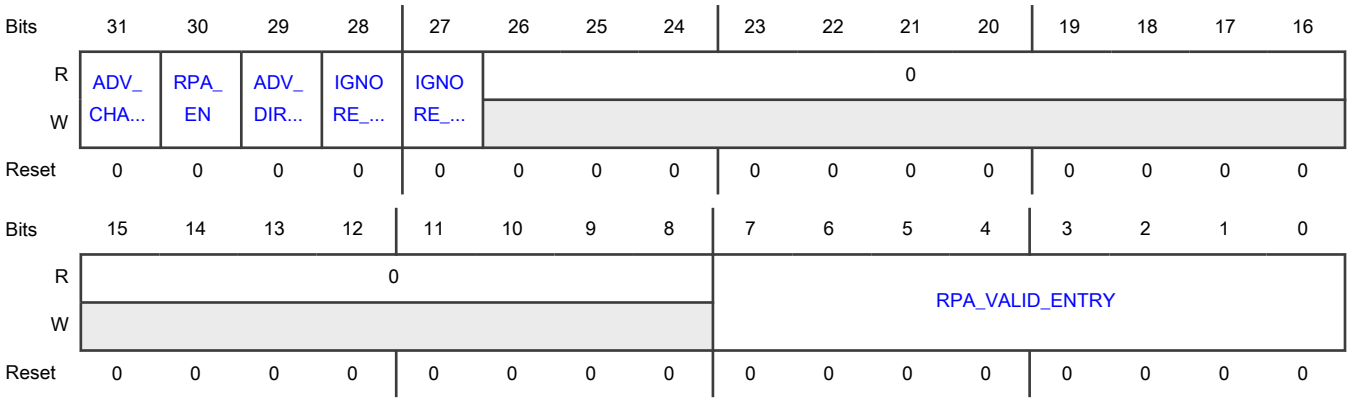
Field	Function
31-0 LENIENCY_LSB	Leniency LSB Register The Packet Processor performs filtering on all received packets, in order to determine whether the packet is intended for the device. The packet filtering is based on rules. In case any of the packet filtering rules need to be overriden, a 45-bit "leniency register" has been provided. When the leniency register is programmed to its default value (0), all hardware packet filtering rules are in effect, and if an incoming packet violates any rule, a "Filter Fail" will occur (packet will be rejected). When a given leniency register bit is asserted, the packet filtering rule assigned to that bit will not be in effect, and if any incoming packet violates that rule (but no other rules), then a "Filter Fail" will not occur, the packet will not be rejected, the packet will be treated as "intended for the device", and software will be notified of the incoming packet.

55.4.9.2.2.1.42 RPA CONTROL (RPA_CTRL)

Offset

Register	Offset
RPA_CTRL	9Ch

Diagram



Fields

Field	Function
31 ADV_CHANNEL_EN	0b - The packet to be received is in Data Channel PDU. 1b - The packet to be received is in Advertising Channel PDU.
30 RPA_EN	0b - The RPA check is disabled. 1b - The RPA check is enabled.
29 ADV_DIRECT_IND_SENT	When set, it means the link layer is advertiser who has sent an ADV_DIRECT_IND. This bit is used by hardware to bypass the white list check when received a CONNECT_IND or SCAN_REQ. In this case, the peer address and peer address type are to be compared with the register DIRECT_PEER_ADDRESS and DIRECT_PEER_ADDRESS_TYPE.
28 IGNORE_DIRECT_FAIL	0b - link layer aborts the Rx process when DIRECT_ID_FAIL_IRQ 1b - link layer ignores DIRECT_ID_FAIL_IRQ.
27 IGNORE_RPA_FAIL	0b - link layer aborts the Rx process when LOCAL_RPA_FAIL_IRQ or PEER_RPA_FAIL_IRQ 1b - link layer ignores LOCAL_RPA_FAIL_IRQ and PEER_RPA_FAIL_IRQ.
26-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 RPA_VALID_ENTRY	Here bits [7:0] corresponds to validity of 8th to 1st entry in RPA list respectively. Value "1" means the entry is valid, "0" means invalid.

55.4.9.2.2.1.43 LENIENCY MSB (LENIENCY_MSB)

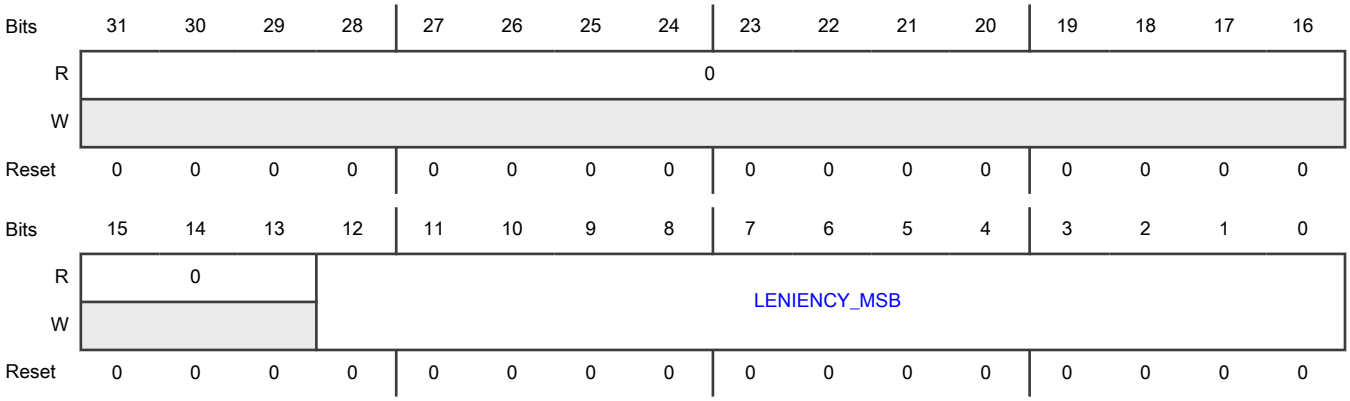
Offset

Register	Offset
LENIENCY_MSB	A0h

Function

Packet Processor Leniency Bits (MSB)

Diagram



Fields

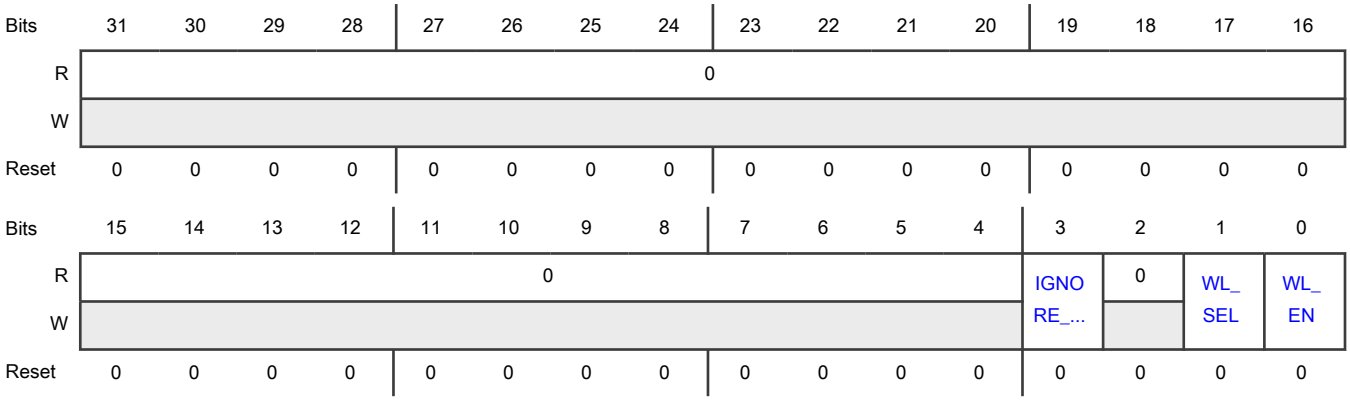
Field	Function
31-13 —	Reserved
12-0 LENIENCY_MSB	Leniency MSB Register The Packet Processor performs filtering on all received packets, in order to determine whether the packet is intended for the device. The packet filtering is based on rules. In case any of the packet filtering rules need to be overriden, a 45-bit "leniency register" has been provided. When the leniency register is programmed to its default value (0), all hardware packet filtering rules are in effect, and if an incoming packet violates any rule, a "Filter Fail" will occur (packet will be rejected). When a given leniency register bit is asserted, the packet filtering rule assigned to that bit will not be in effect, and if any incoming packet violates that rule (but no other rules), then a "Filter Fail" will not occur, the packet will not be rejected, the packet will be treated as "intended for the device", and software will be notified of the incoming packet.

55.4.9.2.2.1.44 WHITE LIST CONTROL (WL_CTRL)

Offset

Register	Offset
WL_CTRL	A0h

Diagram



Fields

Field	Function
31-4 —	Reserved
3 IGNORE_WL_FAIL	0b - link layer aborts the Rx process when WL_FAIL_IRQ 1b - link layer ignores WL_FAIL_IRQ.
2 —	Reserved
1 WL_SEL	0b - Select white list 0 1b - Select white list 1
0 WL_EN	0b - White list search is not enabled 1b - White list search is enabled

55.4.9.2.2.1.45 DUAL PAN CONTROL (DUAL_PAN_CTRL)

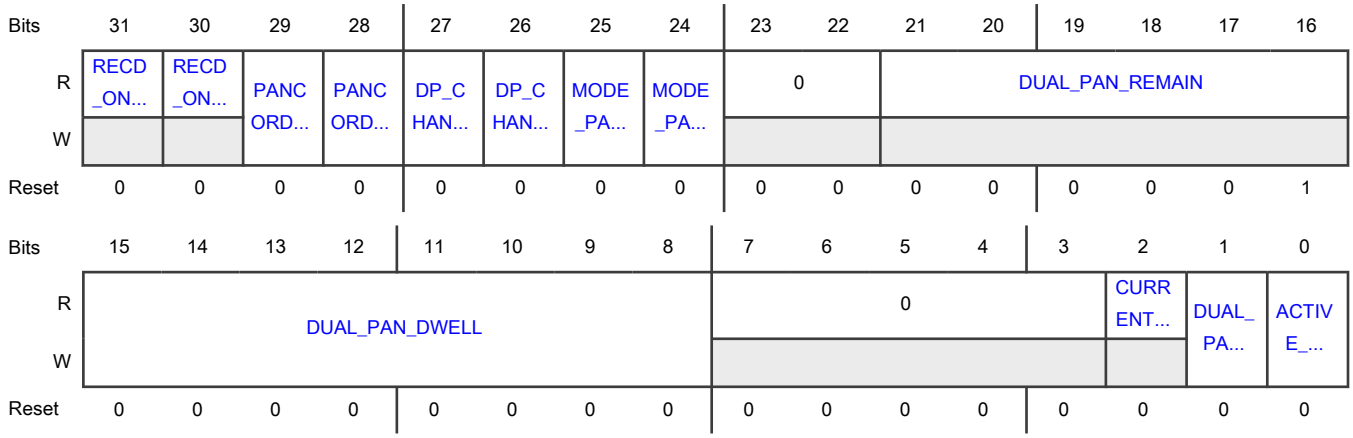
Offset

Register	Offset
DUAL_PAN_CTRL	A4h

Function

Dual PAN Control Register

Diagram



Fields

Field	Function
31 RECD_ON_PAN1	Last Packet was Received on PAN1 Indicates the packet which was just received, was received on PAN1. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN1 will be set if CURRENT_NETWORK=1 when the packet was received, regardless of FILTERFAIL status. In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN1 will be set only if a valid packet was received on PAN1 (PAN1's FILTERFAIL_FLAG is deasserted). RECD_ON_PAN1 remains valid until the start of the next autoseqeuence.
30 RECD_ON_PAN0	Last Packet was Received on PAN0 Indicates the packet which was just received, was received on PAN0. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN0 will be set if CURRENT_NETWORK=0 when the packet was received, regardless of FILTERFAIL status. In DUAL PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN0 will be set only if a valid packet was received on PAN0 (PAN0's FILTERFAIL_FLAG is deasserted). RECD_ON_PAN0 remains valid until the start of the next autoseqeuence.
29 PANCORDNTR1	Device is a PAN Coordinator on PAN1 Device is a PAN Coordinator on PAN1. Allows device to receive packets with no destination address, if Source PAN ID matches.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 PANCORDNTR0	Device is a PAN Coordinator on PAN0 Device is a PAN Coordinator on PAN0. Allows device to receive packets with no destination address, if Source PAN ID matches.
27 DP_CHAN_OVRD_SEL	Dual PAN Channel Override Selector This bit works with **DP_CHAN_OVRD_EN** to allow one of the two Dual PAN channels to use Direct Frequency programming. See description for **DP_CHAN_OVRD_EN**.
26 DP_CHAN_OVRD_EN	Dual PAN Channel Override Enable In Dual PAN mode, in case there is a need to generate a frequency which may be offset from the 16 prescribed 5MHz-spaced channels, to, for example, avoid interference on one of the Dual PAN channels, a method has been provided to accomplish that, by designating one of the two PAN channels to use the transceiver's set of direct frequency-programming registers, instead of CHANNEL_NUMx. Programming the direct frequency-programming registers -- integer, numerator, and denominator, allows an RF frequency to be selected with much more precision than the 5MHz granularity of the Zigbee mapped-channel registers, CHANNEL_NUM0 and CHANNEL_NUM1. Two bits have been provided in 802.15.4 space to realize this feature: DP_CHAN_OVRD_SEL and DP_CHAN_OVRD_EN. When DP_CHAN_OVRD_EN=1, this enables one of the Dual PAN channels to use the direct frequency programming. The DP_CHAN_OVRD_SEL bit determines which channel uses the direct programming, according to the following: If DP_CHAN_OVRD_EN=0, then PAN0 frequency is determined by register CHANNEL_NUM0[6:0] and PAN1 frequency is determined by register CHANNEL_NUM1[6:0]. If DP_CHAN_OVRD_EN=1 and DP_CHAN_OVRD_SEL=0, then PAN0 frequency is determined by DIRECT FREQUENCY PROGRAMMING and PAN1 frequency is determined by register CHANNEL_NUM1[6:0]. If DP_CHAN_OVRD_EN=1 and DP_CHAN_OVRD_SEL=1, then PAN0 frequency is determined by register CHANNEL_NUM0[6:0] and PAN1 frequency is determined by DIRECT FREQUENCY PROGRAMMING. **NOTE** Direct Frequency Programming is accomplished by setting the PLL's Integer, Numerator, and Denominator registers to the appropriate values for the desired RF frequency.
25 MODE_PAN1	PAN1 Mode Select For Hybrid Dual PAN Mode, select the PAN1 Mode 0b - PAN1 is in PAN mode 1b - PAN1 is in FAN mode
24 MODE_PAN0	PAN0 Mode Select For Hybrid Dual PAN Mode, select the PAN0 Mode 0b - PAN0 is in PAN mode 1b - PAN0 is in FAN mode
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21-16 DUAL_PAN_REMAIN	Time Remaining before next PAN switch in auto Dual PAN mode This read-only register indicates time remaining before next PAN switch in auto Dual PAN mode. The units for this register, depend on the PRESCALER setting (bits [1:0]) in the DUAL_PAN_DWELL register. Tthe units are the same as the TIMEBASE determined by the prescaler, as per the following table: DUAL_PAN_DWELL[1:0]=0b00 (TIMEBASE=0.5ms) DUAL_PAN_REMAIN UNITS=0.5ms DUAL_PAN_DWELL[1:0]=0b01 (TIMEBASE=2.5ms) DUAL_PAN_REMAIN UNITS=2.5ms DUAL_PAN_DWELL[1:0]=0b10 (TIMEBASE=10ms) DUAL_PAN_REMAIN UNITS=10ms DUAL_PAN_DWELL[1:0]=0b11 (TIMEBASE=50ms) DUAL_PAN_REMAIN UNITS=50ms The readback value indicates that between N-1 and N timebase units remain until the next PAN switch. For example, a DUAL_PAN_REMAIN readback value of 3, with a DUAL_PAN_DWELL PRESCALER setting of 2 (10ms), indicates that between 20ms (2*10ms) and 30ms (3*10ms), remain until the next automatic PAN switch.
15-8 DUAL_PAN_DWELL	Dual PAN Channel Frequency Dwell Time Channel Frequency Dwell Time. In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to DUAL_PAN_DWELL, always re-initilizes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the sequence warmup. DUAL_PAN_DWELL[1:0] select the timebase for the dwell time. There are 4 options: DUAL_PAN_DWELL[1:0]=0b00 --> TIMEBASE=0.5ms DUAL_PAN_DWELL[1:0]=0b01 --> TIMEBASE=2.5ms DUAL_PAN_DWELL[1:0]=0b10 --> TIMEBASE=10ms DUAL_PAN_DWELL[1:0]=0b11 --> TIMEBASE=50ms DUAL_PAN_DWELL[7:2] select how many multiples of the time base to use, from 1 (DUAL_PAN_DWELL[7:2]=0) to 64 (DUAL_PAN_DWELL[7:2]=63). Given those definitions, the table below specifies the minimum and maximum dwell times for each TIMEBASE option: DUAL_PAN_DWELL[1:0]=0b00 --> MIN_DWELL=0.5ms MAX_DWELL=32ms DUAL_PAN_DWELL[1:0]=0b01 --> MIN_DWELL=2.5ms MAX_DWELL=160ms DUAL_PAN_DWELL[1:0]=0b10 --> MIN_DWELL=10ms MAX_DWELL=640ms DUAL_PAN_DWELL[1:0]=0b11 --> MIN_DWELL=50ms MAX_DWELL=3.2seconds A write to DUAL_PAN_DWELL also causes the value of ACTIVE_NETWORK to get latched into the hardware. This latched value will be the starting point for the automatic dual-pan mode (i.e., start on PAN0 or on PAN1). The starting value takes effect immediately (if sequence is underway and DUAL_PAN_AUTO=1), or is otherwise delayed until sequence starts and DUAL_PAN_AUTO=1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-3 —	Reserved
2 CURRENT_NETWORK	Indicates which PAN is currently selected by hardware This read-only bit indicates which PAN is currently selected by hardware in automatic Dual PAN mode 0b - PAN0 is selected 1b - PAN1 is selected
1 DUAL_PAN_AUTO	Activates automatic Dual PAN operating mode Activates automatic Dual PAN operating mode. In this mode, PAN-switching is controlled by hardware at a pre-programmed rate, determined by DUAL_PAN_DWELL. 0: Manual Dual PAN mode (or Single PAN mode). 1: Auto Dual PAN Mode Whenever DUAL_PAN_AUTO=0, CURRENT_NETWORK=ACTIVE_NETWORK at all times. In other words, software directly controls which PAN is selected. Whenever DUAL_PAN_AUTO=1, CURRENT_NETWORK is controlled by hardware.
0 ACTIVE_NETWORK	Active Network Selector Selects the PAN on which to transceive, by activating a PAN parameter set (PAN0 or PAN1). In Manual Dual PAN mode (or Single PAN mode), this bit selects the active PAN parameter set (channel and addressing parameters) which governs all autosequences. In Auto Dual PAN mode, this bit selects the PAN on which to begin transceiving, latched at the point at which DUAL_PAN_DWELL register is written. 0b - Select PAN0 1b - Select PAN1

55.4.9.2.2.1.46 GTM MODE PDU (GTM_PDU)

Offset

Register	Offset
GTM_PDU	A8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GTM_PDU															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GTM_PDU															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 GTM_PDU	GTM MODE PDU Used in GTM mode, the 8/32-bit Pattern selected by register GTM_PDU_TYPE.

55.4.9.2.2.1.47 MAC SHORT ADDRESS FOR PAN1 (MACSHORTADDRS1)

Offset

Register	Offset
MACSHORTADDRS1	A8h

Function

The MACSHORTADDRS1 register .

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MACSHORTADDRS1															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACPANID1															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

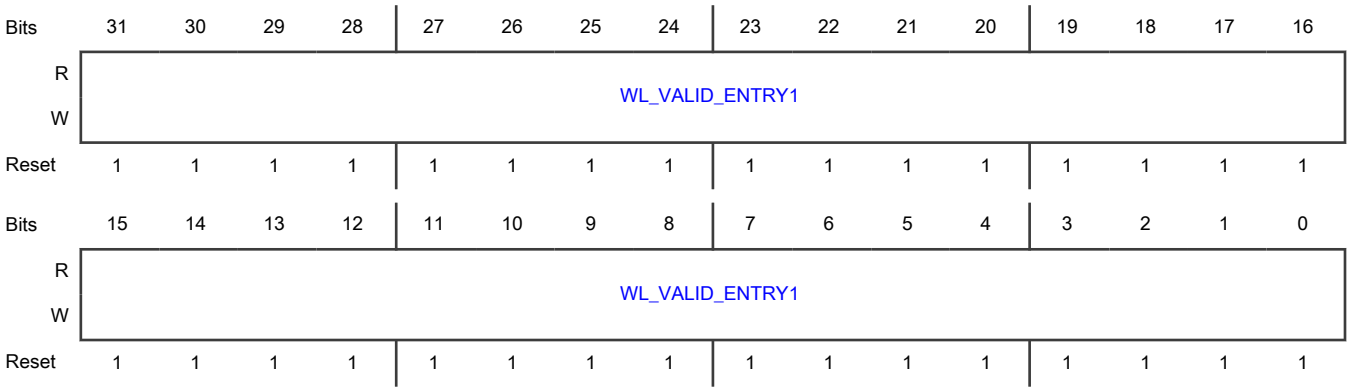
Field	Function
31-16 MACSHORTADDRS1	MAC SHORT ADDRESS for PAN1 MAC Short Address for PAN1, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
15-0 MACPANID1	MAC PAN ID for PAN1 MAC PAN ID for PAN1. The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet Source PAN ID against this register. Also, if PANCORDNTR1=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet Source PAN ID against this register.

55.4.9.2.2.1.48 VALID ENTRY OF WHITE LIST 1 (WL_VALID_ENTRY1)

Offset

Register	Offset
WL_VALID_ENTRY1	A8h

Diagram



Fields

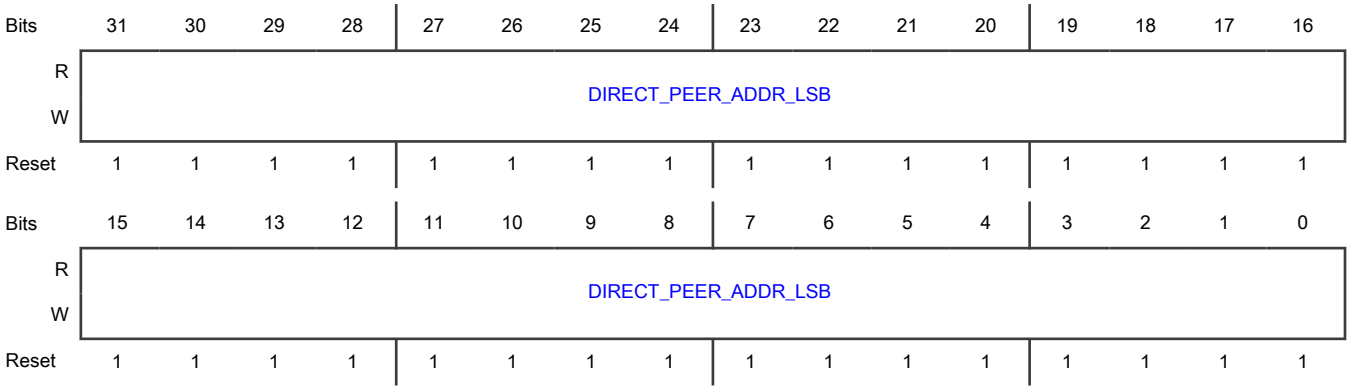
Field	Function
31-0 WL_VALID_ENTRY1	Here bits [31:0] corresponds to validity of 32th to 1st entry in white list 1 respectively. Value "1" means the entry is valid, "0" means invalid.

55.4.9.2.2.1.49 DIRECT_PEER_ADDR[31:0] (DIRECT_PEER_ADDR_LSB)

Offset

Register	Offset
DIRECT_PEER_ADDR_LSB	ACh

Diagram



Fields

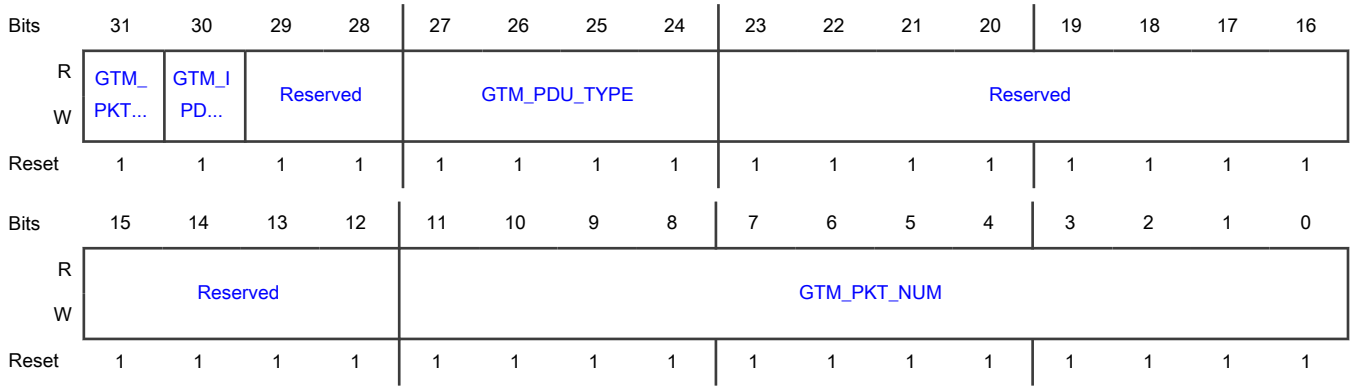
Field	Function
31-0 DIRECT_PEER_ADDR_LSB	Lower 32 bit of 48-bit address of the direct peer device.

55.4.9.2.2.1.50 GTM MODE CONFIGURATION (GTM_CFG)

Offset

Register	Offset
GTM_CFG	ACh

Diagram



Fields

Field	Function
31 GTM_PKT_COUNT_CHECK_DIS	GTM MODE PACKET NUMBER CHECK DISABLE Used in GTM mode, disable the packet number check.
30 GTM_IPD_CHECK_DIS	GTM MODE INTER-PACKET DURATION CHECK DISABLE Used in GTM mode, disable the IPD check.
29-28 —	Reserved
27-24 GTM_PDU_TYPE	GTM MODE PDU TYPE SELECTION 0000b - PRBS9 Sequence 0001b - Programmable 8-bit Pattern (from register GTM_PDU[7:0], reused from MACSHORTADDRS1[7:0]) 0010b - PRBS-13 Sequence 0011b - PRBS-15 Sequence 0100b - Programmable 32-bit Pattern (from register GTM_PDU[31:0], reused from {MACSHORTADDRS1,MACPANID1}) 0101b - Programmable packet from Packet RAM (in this case, PKT_LEN is ignored)
23-12 —	Reserved
11-0 GTM_PKT_NUM	GTM MODE PACKET NUMBER Used in GTM mode, the total number of packets to be sent or received.

55.4.9.2.2.1.51 MAC LONG ADDRESS 1 LSB (MACLONGADDRS1_LSB)

Offset

Register	Offset
MACLONGADDRS1_LSB	ACh

Function

MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MACLONGADDRS1_LSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACLONGADDRS1_LSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

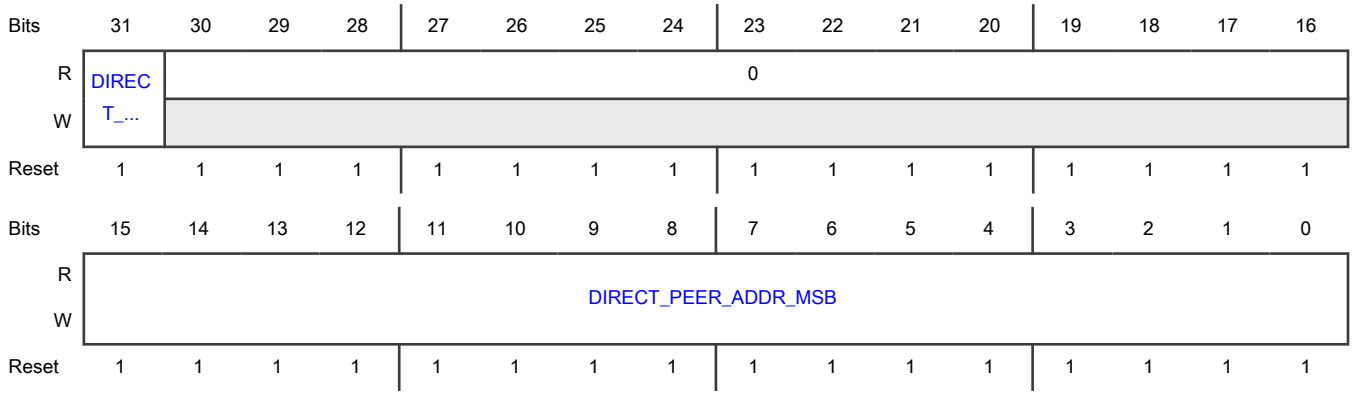
Field	Function
31-0 MACLONGADDRS1_LSB	MAC LONG ADDRESS for PAN1 LSB MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.2.2.1.52 DIRECT_PEER_ADDR[47:32] (DIRECT_PEER_ADDR_MSB)

Offset

Register	Offset
DIRECT_PEER_ADDR_MSB	B0h

Diagram



Fields

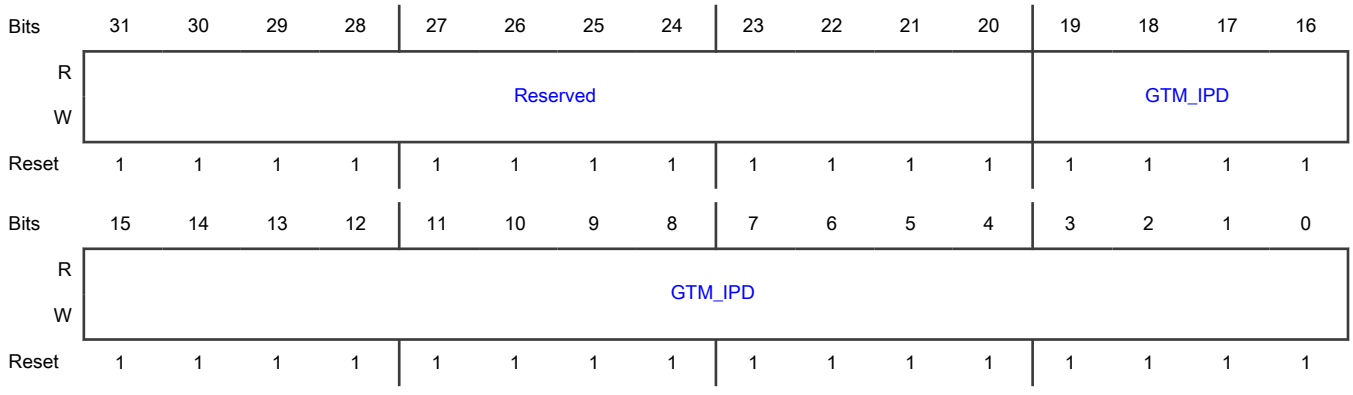
Field	Function
31 DIRECT_PEER_ADDR_TYPE	0b - Direct peer device address type is public. 1b - Direct peer device address type is random.
30-16 —	Reserved
15-0 DIRECT_PEER_ADDR_MSB	Higher 16 bit of 48-bit address of the direct peer device.

55.4.9.2.2.1.53 GTM MODE INTER-PACKET DURATION (GTM_IPD)

Offset

Register	Offset
GTM_IPD	B0h

Diagram



Fields

Field	Function
31-20 —	Reserved
19-0 GTM_IPD	GTM MODE INTER-PACKET DURATION Used in GTM mode, in microseconds unit; indicates the time slot that one packet is sent.

55.4.9.2.2.1.54 MAC LONG ADDRESS 1 MSB (MACLONGADDRS1_MSB)

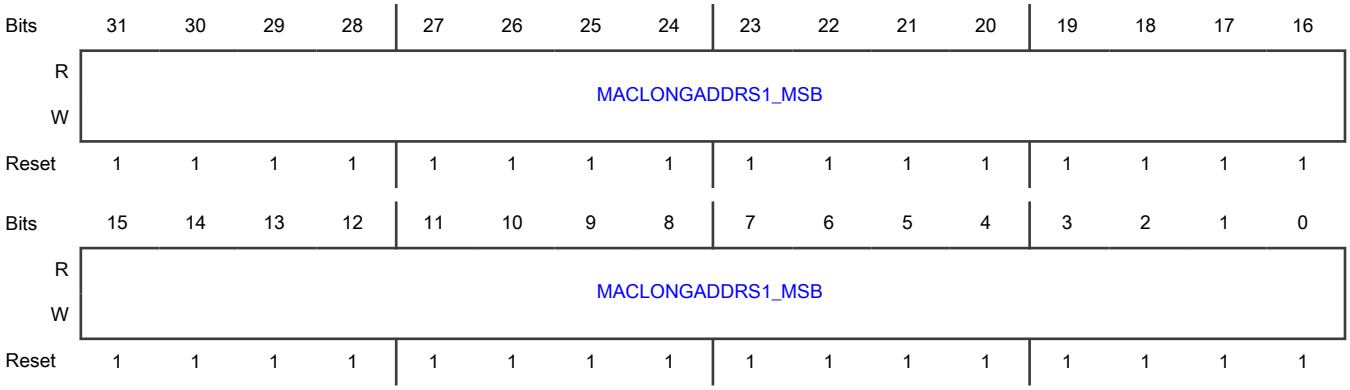
Offset

Register	Offset
MACLONGADDRS1_MSB	B0h

Function

MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

Diagram



Fields

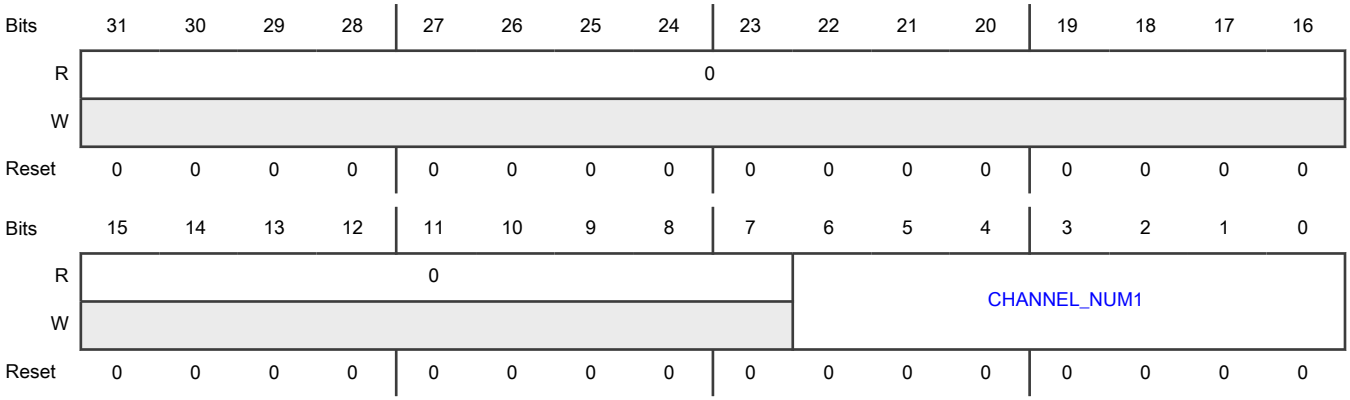
Field	Function
31-0 MACLONGADDRS1_MSB	MAC LONG ADDRESS for PAN1 MSB MAC Long Address for PAN1, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.2.2.1.55 CHANNEL NUMBER 1 (CHANNEL_NUM1)

Offset

Register	Offset
CHANNEL_NUM1	B4h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 CHANNEL_NUM1	Channel Number for PAN1 This is the mapped channel number used to transmit and receive packets. If Dual PAN is engaged, this register applies to PAN1. Formula: F = (2360 + CHANNEL_NUM1) [in MHz]

55.4.9.2.2.1.56 MAC SHORT ADDRESS 0 (MACSHORTADDRS0)

Offset

Register	Offset
MACSHORTADDRS0	B8h

Function

MAC SHORT ADDRESS for PAN0

Diagram

Fields

Field	Function
31-16 MACSHORTADDRS0	MAC SHORT ADDRESS FOR PAN0 MAC Short Address for PAN0, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.
15-0 MACPANID0	MAC PAN ID for PAN0 MAC PAN ID for PAN0. The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet Source PAN ID against this register. Also, if PANCORDNTR0=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet Source PAN ID against this register.

55.4.9.2.2.1.57 VALID ENTRY OF WHITE LIST 0 (WL_VALID_ENTRY0)

Offset

Register	Offset
WL_VALID_ENTRY0	B8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WL_VALID_ENTRY0															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WL_VALID_ENTRY0															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 WL_VALID_ENTRY0	Here bits [31:0] corresponds to validity of 32th to 1st entry in white list 0 respectively. Value "1" means the entry is valid, "0" means invalid.

55.4.9.2.2.1.58 GTM MODE TIME OF FIRST SFD FOUND TO FORCE RX WARMDOWN (GTM_FIRST_SFD2WD)

Offset

Register	Offset
GTM_FIRST_SFD2WD	BCh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												GTM_FIRST_SFD2WD			
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GTM_FIRST_SFD2WD															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-20 —	Reserved
19-0 GTM_FIRST_SFD2WD	GTM MODE TIME OF FIRST SFD FOUND TO FORCE RX WARMDOWN Used in GTM mode, in microseconds unit; Indicates the time between the "force Rx warm-down" and "first SFD found". It is possibly the longest packet duration plus some margin.

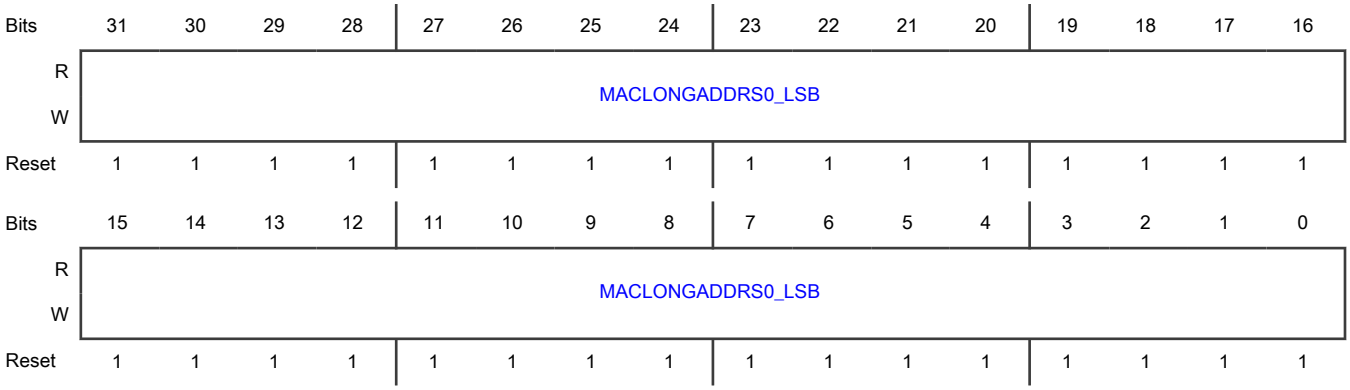
55.4.9.2.2.1.59 MAC LONG ADDRESS 0 LSB (MACLONGADDRS0_LSB)

Offset

Register	Offset
MACLONGADDRS0_LSB	BCh

Function
MAC LONG ADDRESS for PAN0 LSB

Diagram



Fields

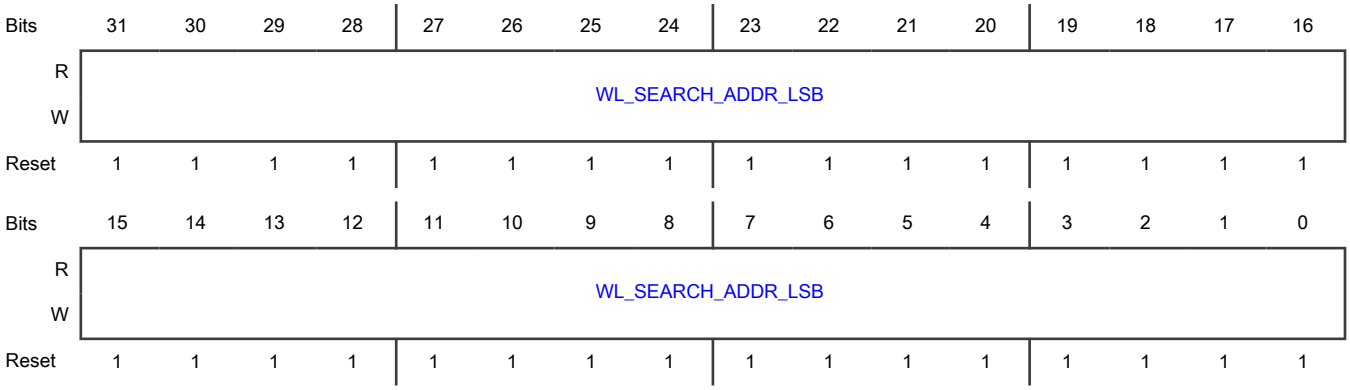
Field	Function
31-0 MACLONGADDRS0_LSB	MAC LONG ADDRESS for PAN0 LSB MAC Long Address for PAN0, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.2.2.1.60 WL_SEARCH_ADDR[31:0] (WL_SEARCH_ADDR_LSB)

Offset

Register	Offset
WL_SEARCH_ADDR_LSB	BCh

Diagram



Fields

Field	Function
31-0 WL_SEARCH_ADDR_LSB	Lower 32 bit of 48-bit address to be searched in white list.

55.4.9.2.2.1.61 GTM MODE RX RECYCLE TIME (GTM_RX_RECYCLE_TIME)

Offset

Register	Offset
GTM_RX_RECYCLE_TIME	C0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												GTM_RX_RECYCLE_TIME			
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GTM_RX_RECYCLE_TIME															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-20 —	Reserved
19-0 GTM_RX_RECYCLE_TIME	GTM MODE RX RECYCLE TIME Used in GTM mode, in microseconds unit; For TX, it means the delay between "first TX warmup" and "GTM_IN_TX is asserted"; For RX, it means the time between "force Rx warm-down" and "next Rx warmup".

55.4.9.2.2.1.62 MAC LONG ADDRESS 0 MSB (MACLONGADDRS0_MSB)

Offset

Register	Offset
MACLONGADDRS0_MSB	C0h

Function
MAC LONG ADDRESS for PAN0 MSB

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MACLONGADDRS0_MSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MACLONGADDRS0_MSB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

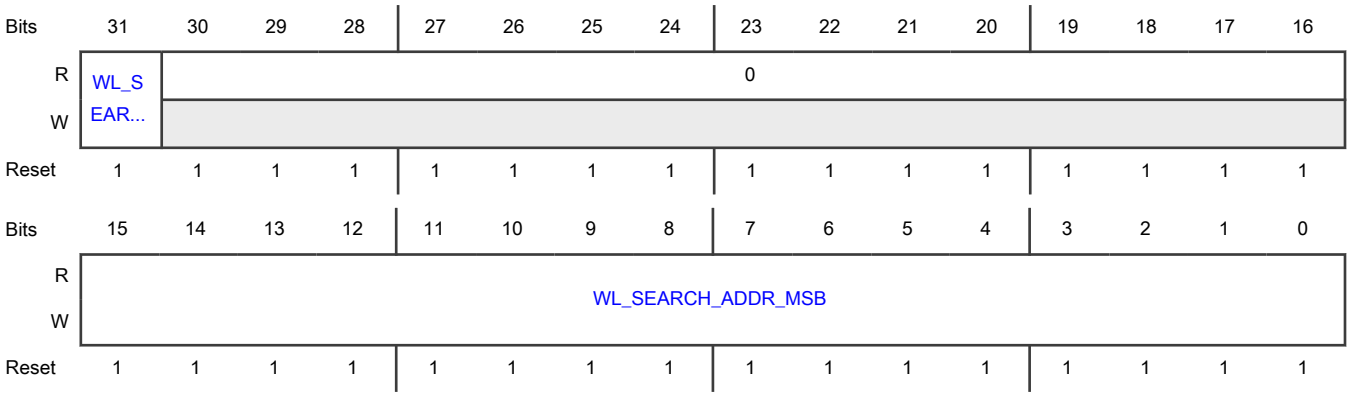
Field	Function
31-0 MACLONGADDRS0_MSB	MAC LONG ADDRESS for PAN0 MSB MAC Long Address for PAN0, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

55.4.9.2.2.1.63 WL_SEARCH_ADDR[47:32] (WL_SEARCH_ADDR_MSB)

Offset

Register	Offset
WL_SEARCH_ADDR_MSB	C0h

Diagram



Fields

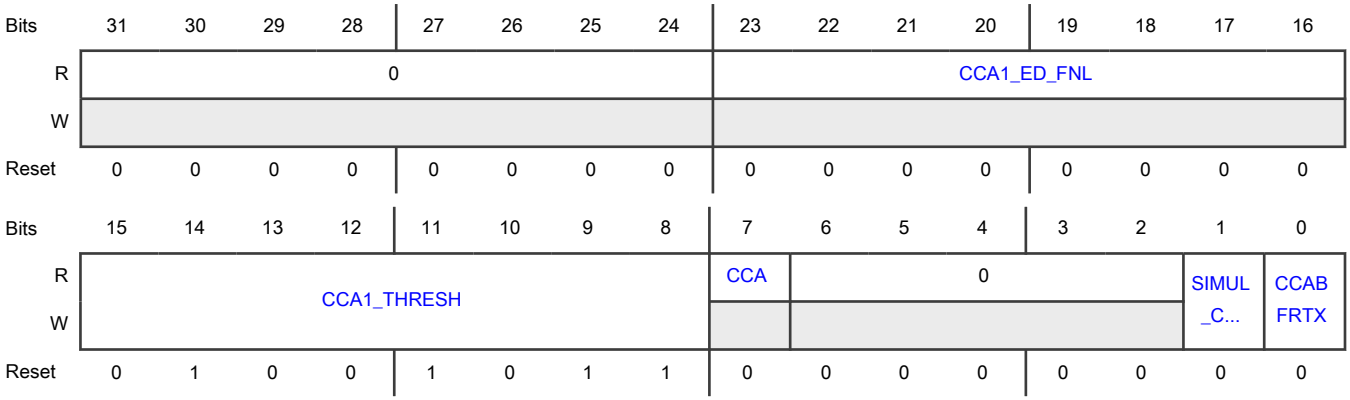
Field	Function
31 WL_SEARCH_ADDR_TYPE	0b - The address type is public. 1b - The address type is random.
30-16 —	Reserved
15-0 WL_SEARCH_ADDR_MSB	Higher 16 bit of 48-bit address to be searched in white list.

55.4.9.2.2.1.64 CCA AND LQI CONTROL (CCA_LQI_CTRL)

Offset

Register	Offset
CCA_LQI_CTRL	C4h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 CCA1_ED_FNL	Final Result for CCA Mode 1 and Energy Detect Output register to show final averaged RSSI value or compensated value of the same at the end of a CCA Mode1 or Energy Detect computation.
15-8 CCA1_THRESH	CCA Mode 1 Threshold Programmable energy threshold register for CCA mode 1.
7 CCA	CCA Status Channel IDLE/BUSY indicator. This indicator is valid at CCA_IRQ and also at SEQ_END_IRQ. This flag is cleared at next receiver warm up. 0b - IDLE 1b - BUSY
6-2 —	Reserved
1	Simultaneous CCA and Receive Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
SIMUL_CCA_RX	During autosequences of type Sequence T or Sequence TR with CCA before TX, this bit allows a packet to be received if preamble and SFD are detected during the CCA measurement window. 0b - Packets can't be received during CCA measurement 1b - Packet reception is enabled during CCA measurement if preamble and SFD are detected
0 CCABFRTX	CCA Before TX Applies only to Sequences T and TR, ignored during all other sequences. 0b - no CCA required, transmit operation begins immediately. 1b - at least one CCA measurement is required prior to the transmit operation (see also SLOTTED).

55.4.9.2.2.1.65 TX/RX WARMUP TIME (WARMUP_TIME)

Offset

Register	Offset
WARMUP_TIME	C8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TX_WARMUP							
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RX_WARMUP							
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	0

Fields

Field	Function
31-24 —	Reserved
23-16 TX_WARMUP	Transmit Warmup Time

Table continues on the next page...

Table continued from the previous page...

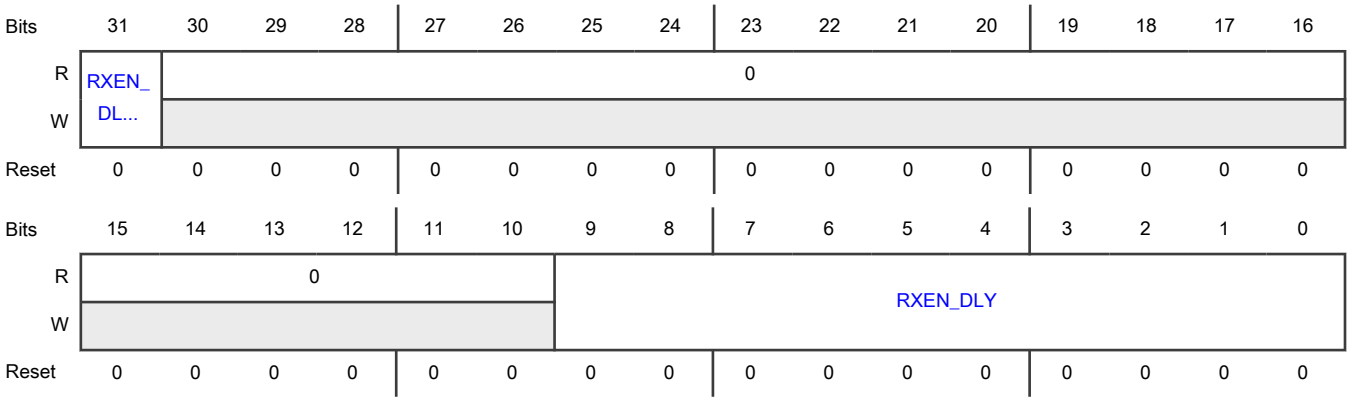
Field	Function
	TX warmup time, in microseconds, provided to Link Layer software, so that warmup time can be accounted for when scheduling over-the-air transactions.
15-8 —	Reserved
7-0 RX_WARMUP	Receive Warmup Time RX warmup time, in microseconds, provided to Link Layer software, so that warmup time can be accounted for when scheduling over-the-air transactions.

55.4.9.2.2.1.66 RX_EN Delay Time (RXEN_DLY)

Offset

Register	Offset
RXEN_DLY	CCh

Diagram



Fields

Field	Function
31 RXEN_DLY_OVERRIDE	RX_EN delay to de-assert time override enable. 0b - For Bluetooth LE case, RX_EN signal will delay to de-assert accroding to the length of TERM2 or CTE(when BLE_V5P1_CTE_EN is enabled) field parsed by hardware 1b - For all receive case, RX_EN signal will delay to de-assert accroding to register RXEN_DLY[9:0].
30-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-0 RXEN_DLY	When RXEN_DLY is not zero, the RX_EN signal will delay (RXEN_DLY +1) microseconds to de-assert after packet is received. The delay time range is from 2~1024 microseconds.

55.4.9.2.2.1.67 SAM CONTROL (SAM_CTRL)

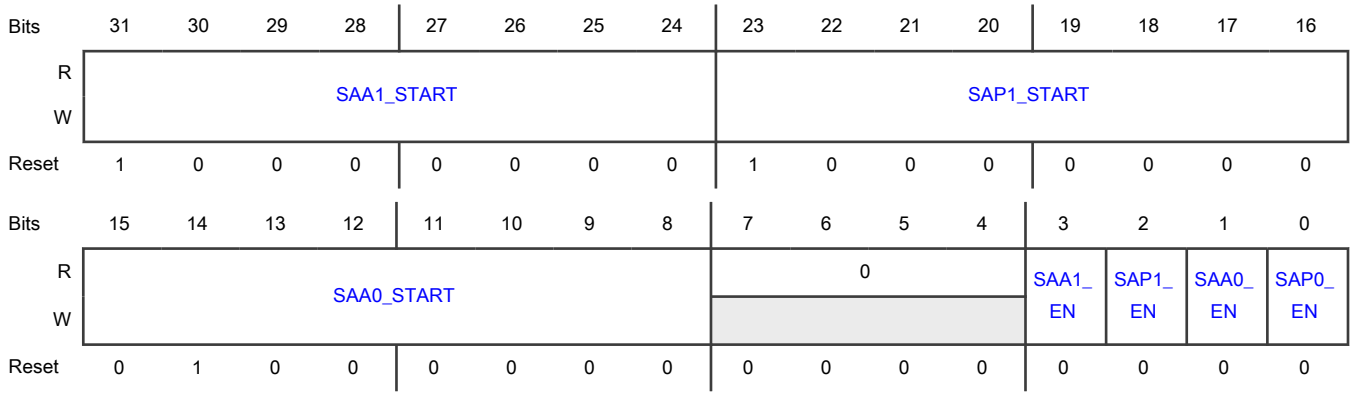
Offset

Register	Offset
SAM_CTRL	D4h

Function

Source Address Management Control Register

Diagram



Fields

Field	Function
31-24 SAA1_START	First Index of SAA1 partition
23-16 SAP1_START	First Index of SAP1 partition
15-8 SAA0_START	First Index of SAA0 partition
7-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SAA1_EN	Enables SAA1 Partition of the SAM Table 0b - Disables SAA1 Partition 1b - Enables SAA1 Partition
2 SAP1_EN	Enables SAP1 Partition of the SAM Table 0b - Disables SAP1 Partition 1b - Enables SAP1 Partition
1 SAA0_EN	Enables SAA0 Partition of the SAM Table 0b - Disables SAA0 Partition 1b - Enables SAA0 Partition
0 SAP0_EN	Enables SAP0 Partition of the SAM Table 0b - Disables SAP0 Partition 1b - Enables SAP0 Partition

55.4.9.2.2.1.68 SOURCE ADDRESS MANAGEMENT TABLE (SAM_TABLE)

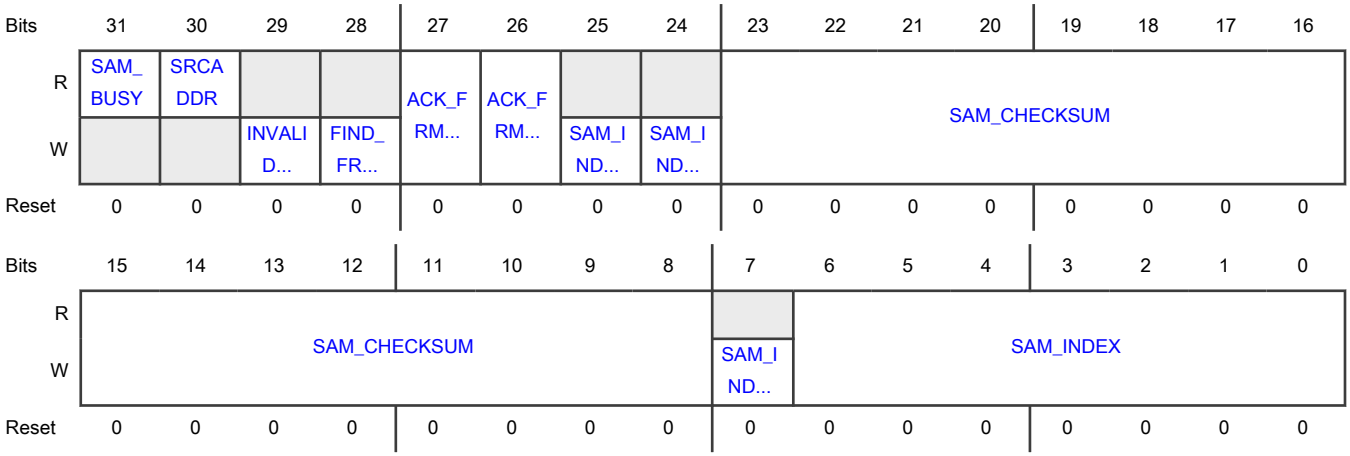
Offset

Register	Offset
SAM_TABLE	D8h

Function

Source Address Management Table

Diagram



Fields

Field	Function
31 SAM_BUSY	SAM Table Update Status Bit Hardware is in the process of updating the Source Address table, either in response to a poll indication from the packet processor, or due to software setting FIND_FREE_IDX=1. In the latter case, software should poll SAM_BUSY until low before accessing the "First Free Index" registers. Read-only bit.
30 SRCADDR	Source Address Match Status If Source Address Management is engaged, meaning at least one of the following bits is set: SAP0_EN SAA0_EN SAP1_EN SAA1_EN Then SRCADDR will be set to 1 if the packet just received is a poll request (PI=1), *and* at least one of the following conditions is met: SAP0_EN and SAP0_ADDR_PRESENT SAA0_EN and SAA0_ADDR_ABSENT SAP1_EN and SAP1_ADDR_PRESENT SAA1_EN and SAA1_ADDR_ABSENT If SRCADDR=1, this indicates to SW that the Packet Processor has determined that an auto-TxACK frame must be transmitted with the FramePending subfield of the FrameControlField set to 1. HW will assemble and transmit this Ack packet. If the above conditions are not met, SRCADDR will be cleared to 0.
29 INVALIDATE_ALL	Invalidate Entire SAM Table Writing a 1 to this bit clears all 128 Valid bits. Invalidates the entire table. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
28 FIND_FREE_IDX	Find First Free Index After modifying Valid bits (enabling or invalidating), write this bit to 1 to force hardware to update the "First Free Index" registers to account for the changed Valid bits. This hardware update process takes 4us. Software can poll SAM_BUSY to determine when the table update is complete. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
27 ACK_FRM_PND_CTRL	Manual Control for AutoTxAck FramePending field 0b - the FramePending field of the Frame Control Field of the next automatic TX acknowledge packet is determined by hardware 1b - the FramePending field of the Frame Control Field of the next automatic TX acknowledge packet tracks ACK_FRM_PEND
26 ACK_FRM_PND	State of AutoTxAck FramePending field when SAM Accelleration is Disabled Software can take manual control of the FramePending field of the Frame Control Field of the next automatic TX acknowledge packet, by setting ACK_FRM_PND_CTRL=1; in that case FramePending will

Table continues on the next page...

Table continued from the previous page...

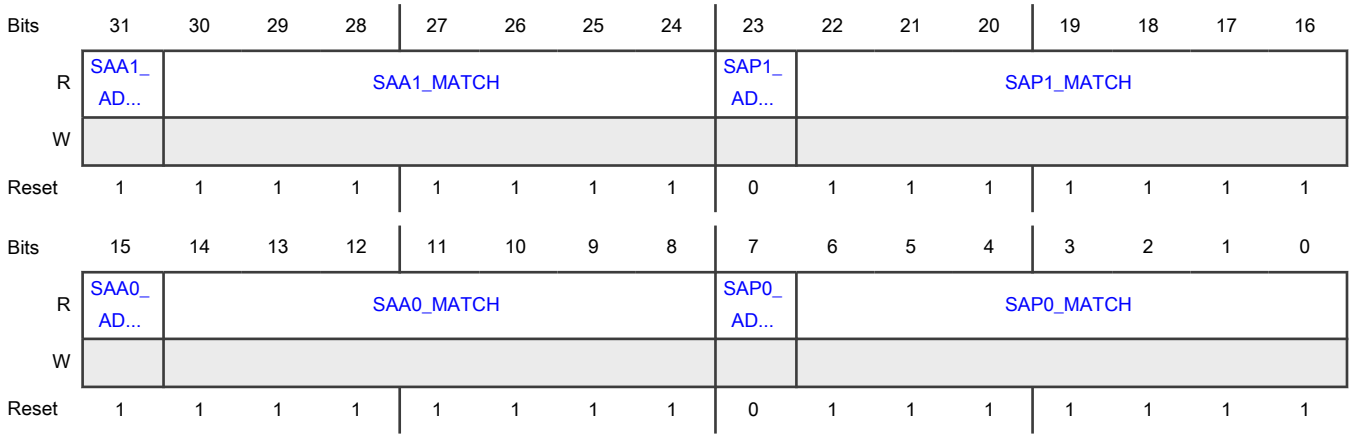
Field	Function
	track the state of this bit. The FramePending field also tracks this bit if Source Address Management is completely disabled, i.e., SAP0_EN=SAA0_EN=SAP1_EN=SAA1_EN=0 Otherwise, the FramePending field is determined by Source Address Management (SAM) hardware.
25 SAM_INDEX_EN	Enable the SAM table index selected by SAM_INDEX
24 SAM_INDEX_INV	Invalidate the SAM table index selected by SAM_INDEX
23-8 SAM_CHECKSUM	Software-computed source address checksum, to be installed into a table index Software-computed source address checksum, to be installed into a table index. The value on SAM_CHECKSUM[15:0] can be installed into the table with a single, atomic 32-bit write; in that case, the write data would contain the desired SAM_INDEX[6:0] and SAM_CHECKSUM[15:0], and SAM_INDEX_WR=1. If SAM_INDEX_WR=0, then the SAM_INDEX[6:0] register is written, but the checksum is *not* written to the table. The readback value of SAM_CHECKSUM[15:0] is the contents of the SAM Table at the location pointed to by SAM_INDEX[6:0]. To readback from a specific table index, software should first write the desired index to SAM_INDEX[6:0], and then read back the checksum from the table on SAM_CHECKSUM[15:0].
7 SAM_INDEX_WR	Enables SAM Table Contents to be updated For 32-bit writes, SAM_INDEX_WR must be set to indicate that the table entry specified by SAM_INDEX[6:0] is to be written; if SAM_INDEX_WR=0, the table entry is not written, but the SAM_INDEX[6:0] register is updated. For 8-bit writes, this bit is ignored.
6-0 SAM_INDEX	Contains the SAM table index to be enabled or invalidated Contains the table index to be enabled or invalidated. Software must ensure that the index is within the range of the desired partition.

55.4.9.2.2.1.69 SOURCE ADDRESS MANAGEMENT MATCH (SAM_MATCH)

Offset

Register	Offset
SAM_MATCH	DCh

Diagram



Fields

Field	Function
31 SAA1_ADDR_ABSENT	A Checksum Match is Absent in the SAP1 Partition of the SAM Table
30-24 SAA1_MATCH	Index in the SAA1 Partition of the SAM Table corresponding to the first checksum match
23 SAP1_ADDR_PRESENT	A Checksum Match is Present in the SAP1 Partition of the SAM Table
22-16 SAP1_MATCH	Index in the SAP1 Partition of the SAM Table corresponding to the first checksum match
15 SAA0_ADDR_ABSENT	A Checksum Match is Absent in the SAA0 Partition of the SAM Table
14-8 SAA0_MATCH	Index in the SAA0 Partition of the SAM Table corresponding to the first checksum match
7 SAP0_ADDR_PRESENT	A Checksum Match is Present in the SAP0 Partition of the SAM Table
6-0 SAP0_MATCH	Index in the SAP0 Partition of the SAM Table corresponding to the first checksum match

55.4.9.2.2.1.70 SAM FREE INDEX (SAM_FREE_IDX)

Offset

Register	Offset
SAM_FREE_IDX	E0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SAA1_1ST_FREE_IDX								SAP1_1ST_FREE_IDX							
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SAA0_1ST_FREE_IDX								SAP0_1ST_FREE_IDX							
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

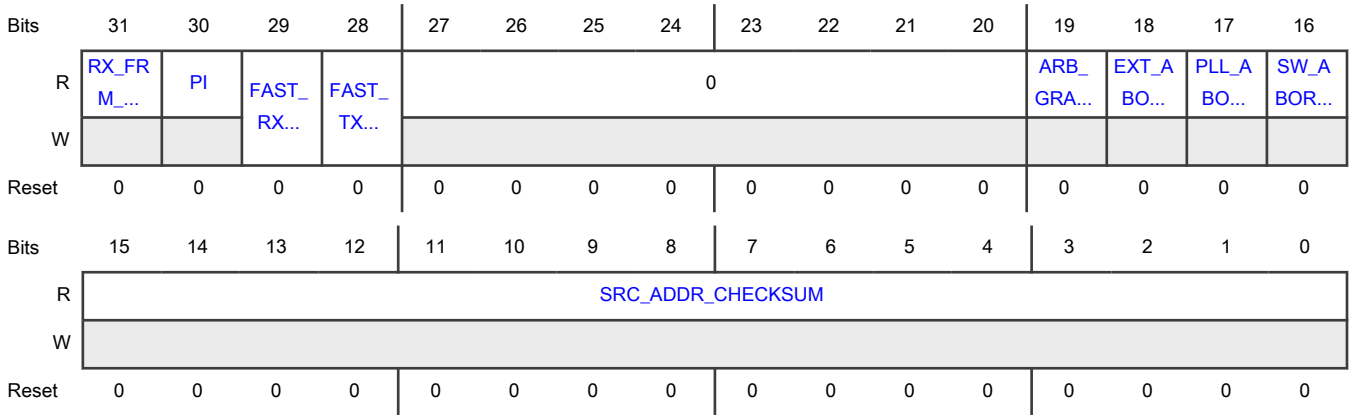
Field	Function
31-24 SAA1_1ST_FREE_IDX	First non-enabled (invalid) index in the SAA1 partition
23-16 SAP1_1ST_FREE_IDX	First non-enabled (invalid) index in the SAP1 partition
15-8 SAA0_1ST_FREE_IDX	First non-enabled (invalid) index in the SAA0 partition
7-0 SAP0_1ST_FREE_IDX	First non-enabled (invalid) index in the SAP0 partition

55.4.9.2.2.1.71 MISCELLANEOUS(1) (MISC1)

Offset

Register	Offset
MISC1	E4h

Diagram



Fields

Field	Function
31 RX_FRM_PEND	RX Frame Pending Status of the frame pending bit of the frame control field for the most-recently received packet. Read-only.
30 PI	Poll Indication 0b - the received packet was not a data request 1b - the received packet was a data request, regardless of whether a Source Address table match occurred, or whether Source Address Management is enabled or not
29 FAST_RX_WU_OVRD	FAST_RX_WU override 0b - If TSM enables Fast Warmup Capability, LL will request it when RX in TR 1b - If TSM enables Fast Warmup Capability, LL will request it at every RX. User should insure channel is not changed since last sequence.
28 FAST_TX_WU_OVRD	FAST_TX_WU override 0b - If TSM enables Fast Warmup Capability, LL will request it when TX in RT or (CCA+TX) 1b - If TSM enables Fast Warmup Capability, LL will request it at every TX. User should insure channel is not changed since last sequence.
27-20 —	Reserved
19 ARB_GRANT_DEASSERTION_ABORTED	Autosequence has terminated due to an arb_grant deassertion event when asserted, indicates that the autosequence has terminated due to an arb_grant deassertion event. This bit is valid at the SEQ_END_IRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
18	Autosequence has terminated due to a Wake-On-Radio command

Table continues on the next page...

Table continued from the previous page...

Field	Function
EXT_ABORTED	when asserted, indicates that the autosequence has terminated due to an external event, such as a Wake-On-Radio stop command. This bit is valid at the SEQ_END_IRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
17 PLL_ABORTED	Autosequence has terminated due to an PLL unlock event. When asserted, indicates that the autosequence has terminated due to an PLL unlock event. This bit is valid at the SEQ_END_IRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
16 SW_ABORTED	Autosequence has terminated due to a Software abort. When asserted, indicates that the autosequence has terminated due to an Software abort. Software can abort any programmed autosequence by writing Sequence I to XCVSEQ. This bit is valid at the SEQ_END_IRQ interrupt. Hardware will maintain this bit asserted until the next autosequence commences. Read-only bit.
15-0 SRC_ADDR_CHECKSUM	Hardware-computed received source address checksum The checksum is computed on-the-fly by hardware as the source address octets are received

55.4.9.2.2.1.72 SEQUENCE STATUS (SEQ_STS)

Offset

Register	Offset
SEQ_STS	E8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SEQ_T_STATUS				0							CCA_STA...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CCA_STA...	TR_STAR...	TR_STAR...	RX_IN_W...	RX_IN_P...	RX_IN_S...	RX_IN_W...	RX_STOP...	RX_STOP...	RX_STAR...	RX_STAR...	TX_IN_W...	TX_IN_P...	TX_IN_W...	TX_STAR...	TX_STAR...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 SEQ_T_STATUS	Status of the just-completed or ongoing Sequence T or Sequence TR Status of the just-completed (or ongoing) Sequence T or Sequence TR autosequence. This register is valid at all times during, and after, the Sequence T or Sequence TR. Not valid for other types of autosequences. This is a read-only register. The bits of this register map to status, according to the following table: [0] 1st CCA complete (CCABFRTX=1) [1] 2nd CCA complete (SLOTTED=1) [2] Tx operation complete [3] Rx Recycle occurred (Sequence TR only) [4] Rx operation complete (Sequence TR only)
23-17 —	Reserved
16 CCA_START_T2_PEND	CCA T2 Start Pending Status CCA Sequence will start @ next T2 Match
15 CCA_START_T1_PEND	CCA T1 Start Pending Status CCA Sequence will start @ next T1 Match
14 TR_START_T2_PEND	TR T2 Start Pending Status TR Sequence will start @ next T2 Match
13 TR_START_T1_PEND	TR T1 Start Pending Status TR Sequence will start @ next T1 Match
12 RX_IN_WARMDN	RX Warmdown Status RX Sequence in TSM Warmdown
11 RX_IN_PROGRESS	RX in Progress Status RX Packet Reception Currently Underway
10	RX Search Status RX Sequence in Network Address Search

Table continues on the next page...

Table continued from the previous page...

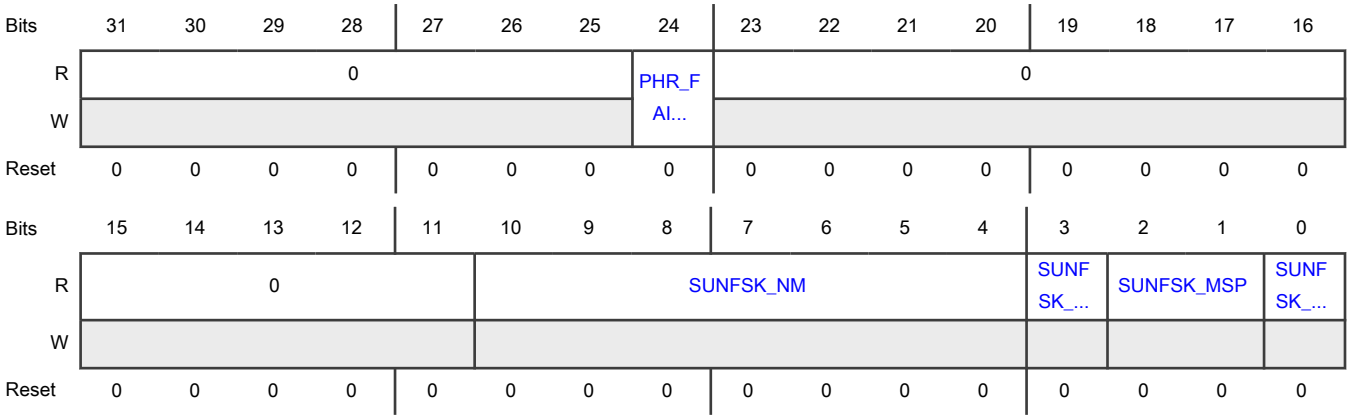
Field	Function
RX_IN_SEARCH	
9 RX_IN_WARMUP	RX Warmup Status RX Sequence in TSM Warmup
8 RX_STOP_T2_PEND	RX T2 Start Pending Status RX Sequence will stop @ next T2 Match
7 RX_STOP_T1_PEND	RX T1 Stop Pending Status RX Sequence will stop @ next T1 Match
6 RX_START_T2_PEND	RX T2 Start Pending Status RX Sequence will start @ next T2 Match
5 RX_START_T1_PEND	RX T1 Start Pending Status RX Sequence will start @ next T1 Match
4 TX_IN_WARMDN	TX Warmdown Status TX Sequence in TSM Warmdown
3 TX_IN_PROGRESS	TX in Progress Status TX Packet Transmission Currently Underway
2 TX_IN_WARMUP	TX Warmup Status TX Sequence in TSM Warmup
1 TX_START_T2_PEND	TX T2 Start Pending Status TX Sequence will start @ next T2 Match
0 TX_START_T1_PEND	TX T1 Start Pending Status TX Sequence will start @ next T1 Match

55.4.9.2.2.1.73 PHR MISCELLANEOUS (PHR_MISC)

Offset

Register	Offset
PHR_MISC	ECh

Diagram



Fields

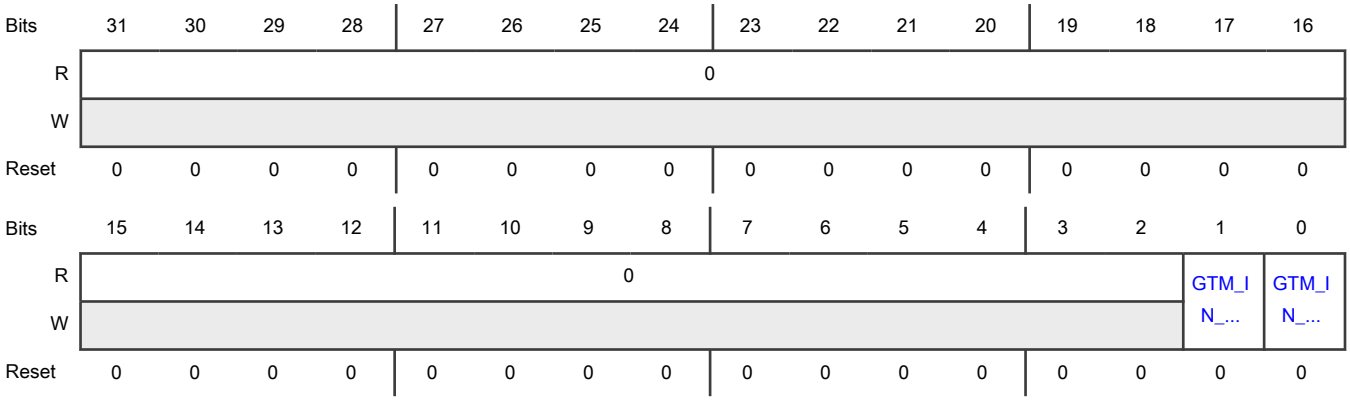
Field	Function
31-25 —	Reserved
24 PHR_FAIL_IGNORE	Ignore PHR Fail If set, assert RX_IRQ even for a received packet which fails PHR verification.
23-11 —	Reserved
10-4 SUNFSK_NM	New Mode Bit SUN FSK Mode Switch PHR New Mode Bit
3 SUNFSK_FEC	New Mode FEC Bit SUN FSK Mode Switch PHR New Mode FEC Bit
2-1 SUNFSK_MSP	Mode Switch Parameter Bit SUN FSK Mode Switch PHR Mode Switch Parameter Bit
0 SUNFSK_MS	Mode Switch Bit SUN FSK Mode Switch PHR Mode Switch Bit

55.4.9.2.2.1.74 GTM CONTROL (GTM_CTRL)

Offset

Register	Offset
GTM_CTRL	F0h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 GTM_IN_TX	Enable GTM Transmit Mode 0b - GTM transmit mode is not enabled. 1b - GTM transmit mode is enabled.
0 GTM_IN_RX	Enable GTM Receive Mode 0b - GTM receive mode is not enabled. 1b - GTM receive mode is enabled.

55.4.9.2.2.1.75 GTM BAD PACKET COUNTER (GTM_BAD_CNT)

Offset

Register	Offset
GTM_BAD_CNT	F4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				GTM_BAD_PKT_COUNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-13 —	Reserved
12-0 GTM_BAD_PKT_COUNT	GTM Bad Packet Counter The Packet is counted as "Bad" for CRC fail or header(H0/Length/H1) fail or there is no AA matched. This register is reset to zero when bit GTM_IN_RX is written to 1.

55.4.9.2.2.1.76 GTM GOOD PACKET COUNTER (GTM_GOOD_CNT)

Offset

Register	Offset
GTM_GOOD_CNT	F8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				GTM_GOOD_PKT_COUNT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

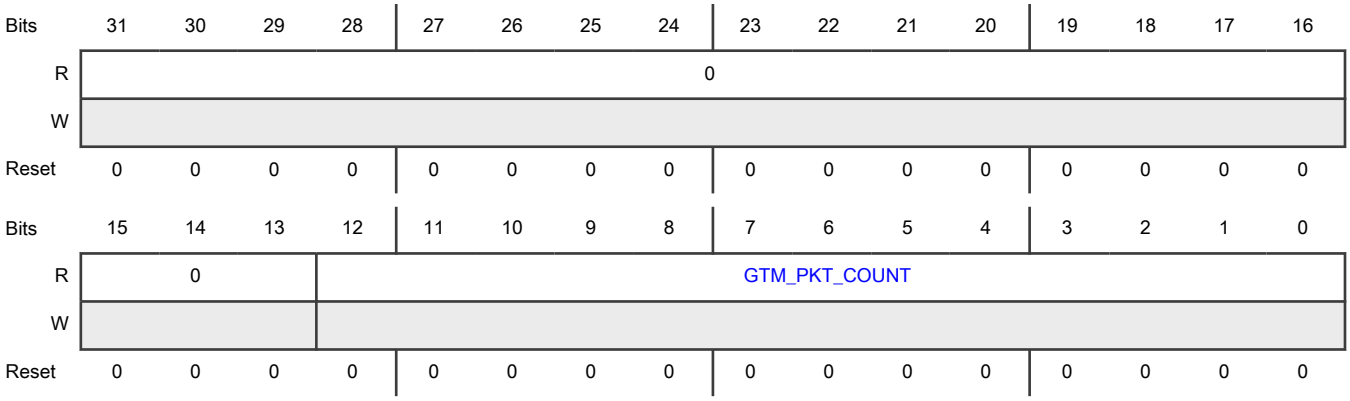
Field	Function
31-13 —	Reserved
12-0 GTM_GOOD_PKT_COUNT	GTM Good Packet Counter The Packet is counted as "Good" when there is no CRC fail or header(H0/Length/H1) fail. This register is reset to zero when bit GTM_IN_RX is written to 1.

55.4.9.2.2.1.77 GTM PACKET COUNTER (GTM_PKT_CNT)

Offset

Register	Offset
GTM_PKT_CNT	FCh

Diagram



Fields

Field	Function
31-13 —	Reserved
12-0 GTM_PKT_COUNT	GTM Packet Counter For Tx process, it counts for the number of packet sent; for Rx, it is same as (GTM_GOOD_PKT_COUNT + GTM_BAD_PKT_COUNT). This register is reset to zero when bit GTM_IN_RX or GTM_IN_TX is written to 1.

55.4.9.2.2.1.78 COEXISTENCE CONTROL (COEX_CTRL)

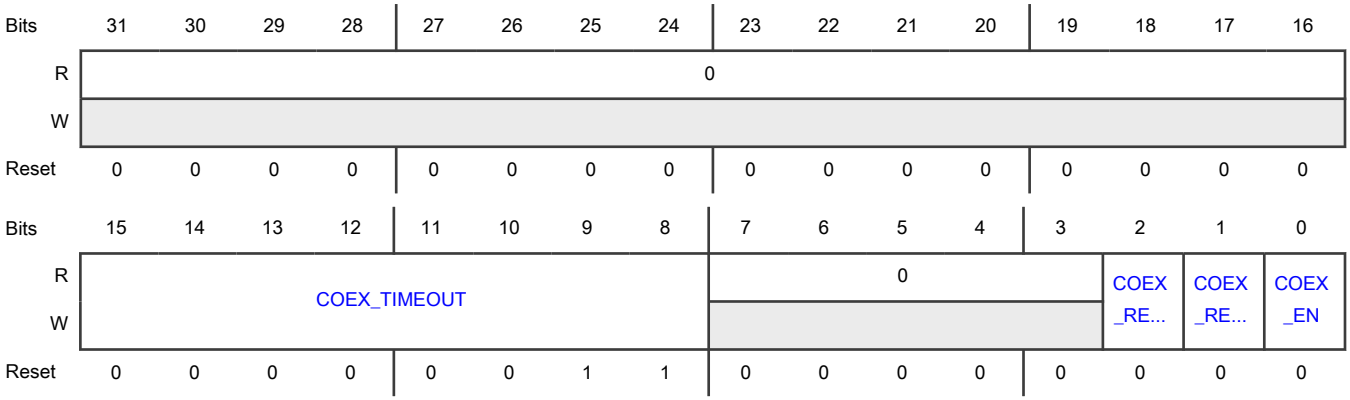
Offset

Register	Offset
COEX_CTRL	100h

Function

Coexistence Control Registers.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 COEX_TIMEOUT	Coexistence timeout value After arb_request asserts, arb_grant should assert before the timeout value. Otherwise the sequence is abort and COEX_TIMEOUT_IRQ bit is set. The step of the timeout value is 32us.
7-3 —	Reserved
2 COEX_REQ_ON_PD	Coexistence Request on Preamble detected 0b - arb_request is delayed until Access Address is detected during R sequence. 1b - arb_request is delayed until preamble is detected during R sequence.
1 COEX_REQ_DELAY_EN	Coexistence Request Delay Enable 0b - arb_request is not delayed during R sequence. 1b - arb_request is delayed until preamble or Access Address is detected during R sequence.
0	Coexistence Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
COEX_EN	0b - Coexistence function is disabled. 1b - Coexistence function is enabled.

55.4.9.2.2.1.79 COEXISTENCE PRIORITY (COEX_PRIORITY)

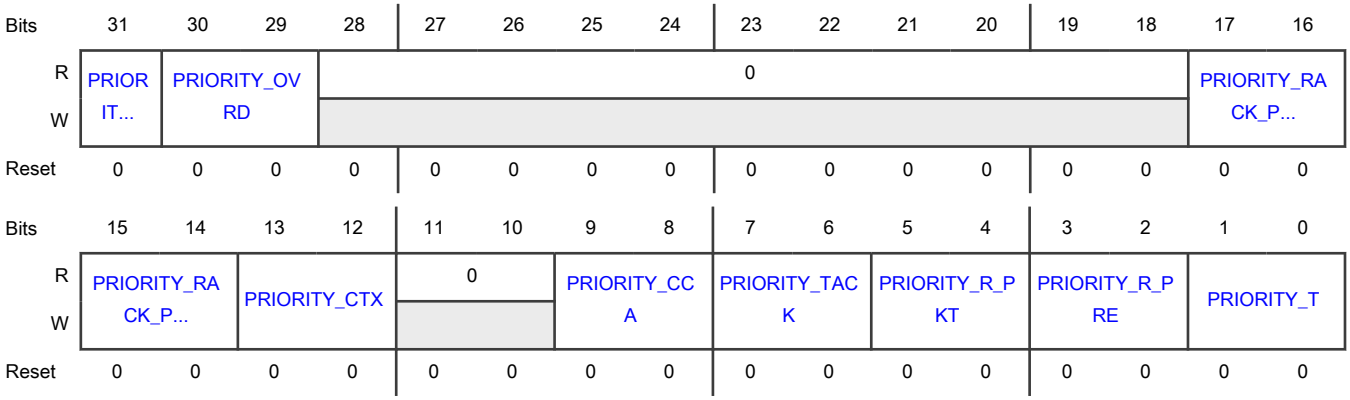
Offset

Register	Offset
COEX_PRIORITY	104h

Function

Programmable 2-bit priority for each RX or TX state.

Diagram



Fields

Field	Function
31 PRIORITY_OVRD_EN	PRIORITY_OVRD_EN Enable/Disable overriding PRIORITY value. 0b - Disable overriding PRIORITY value. 1b - Enable overriding PRIORITY value.
30-29 PRIORITY_OVRD	PRIORITY_OVRD When PRIORITY_OVRD_EN is set to 1, PRIORITY_OVRD is used to override PRIORITY value.
28-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 PRIORITY_RACK_PKT	PRIORITY_RACK_PKT PRIORITY value during receiving ACK packet.
15-14 PRIORITY_RACK_PRE	PRIORITY_RACK_PRE PRIORITY value during searching preamble of ACK packet.
13-12 PRIORITY_CTX	PRIORITY_CT PRIORITY value for CCA before TX Packet.
11-10 —	Reserved
9-8 PRIORITY_CCA	PRIORITY_CCA PRIORITY value for CCA.
7-6 PRIORITY_TACK	PRIORITY_TACK PRIORITY value for TX ACK Packet.
5-4 PRIORITY_R_PKT	PRIORITY_R_PKT PRIORITY value for RX Packet.
3-2 PRIORITY_R_PRE	PRIORITY_R_PRE PRIORITY value of searching Preamble for RX Packet.
1-0 PRIORITY_T	PRIORITY_T PRIORITY value for TX Packet.

55.4.9.2.2.1.80 IRQ CONTROL 2 (IRQ_CTRL2)

Offset

Register	Offset
IRQ_CTRL2	108h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LOCAL_R...	PEER_RP...	DIRECT_...	WL_FAIL...	EVENT_T...	COEX_TI...	ARB_GRA...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									LOCAL_R...	PEER_RP...	DIRECT_...	WL_FAIL...	EVENT_T...	COEX_TI...	ARB_GRA...
W										W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-23 —	Reserved
22 LOCAL_RPA_FAIL_IRQ_EN	0b - LOCAL_RPA_FAIL Interrupt is not enabled. 1b - LOCAL_RPA_FAIL Interrupt is enabled.
21 PEER_RPA_FAIL_IRQ_EN	0b - PEER_RPA_FAIL Interrupt is not enabled. 1b - PEER_RPA_FAIL Interrupt is enabled.
20 DIRECT_ID_FAIL_IRQ_EN	0b - DIRECT_ID_FAIL Interrupt is not enabled. 1b - DIRECT_ID_FAIL Interrupt is enabled.
19 WL_FAIL_IRQ_EN	0b - WL_FAIL Interrupt is not enabled. 1b - WL_FAIL Interrupt is enabled.
18 EVENT_TIMER_OVER_FLOW_IRQ_EN	Event Timer Overflow Interrupt enable bit 0b - Interrupt generation is disabled, but an EVENT_TIMER_OVER_FLOW_IRQ flag can be set 1b - allows interrupt when Event Timer overflow
17 COEX_TIMEOUT_IRQ_EN	Coexistence Timeout Interrupt enable bit 0b - Interrupt generation is disabled, but a COEX_TIMEOUT_IRQ flag can be set 1b - allows interrupt when coexistence timeout
16	arb_grant Deassertion Interrupt enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ARB_GRANT_DEASSERTION_IRQ_EN	0b - An arb_grant deassertion event will set the ARB_GRANT_DEASSERTION_IRQ status bit, but no interrupt is not generated 1b - allows arb_grant deassertion event to generate an interrupt
15-7 —	Reserved
6 LOCAL_RPA_FAIL_IRQ	Local RPA Check Fail Interrupt Interrupt flag to show that local address is not in LRPA list and can not be resolved. This is write '1' to clear bit.
5 PEER_RPA_FAIL_IRQ	Peer RPA Check Fail Interrupt Interrupt flag to show that peer address is not in PRPA list and can not be resolved. This is write '1' to clear bit.
4 DIRECT_ID_FAIL_IRQ	Direct Case Check Fail Interrupt Interrupt flag to show that peer address and address type do not match the DIRECT_PEER_ADDRESS and DIRECT_PEER_ADDRESS_TYPE register. This is write '1' to clear bit.
3 WL_FAIL_IRQ	White List Check Fail Interrupt Interrupt flag to show that identity address resolved(RPA is enabled) or peer address received(RPA is not enabled) and its address type are not in the white list. This is write '1' to clear bit.
2 EVENT_TIMER_OVER_FLOW_IRQ	Event Timer Overflow Interrupt Indicates Event Timer overflow occurs. This is write '1' to clear bit.
1 COEX_TIMEOUT_IRQ	Coexistence Timeout Interrupt Indicates coexistence timeout occurs. This is write '1' to clear bit.
0 ARB_GRANT_DEASSERTION_IRQ	arb_grant Deassertion IRQ arb_grant(!RF_NOT_ALLOWED) Deassertion Interrupt Status bit. A '1' indicates an arb_grant deassertion event has occurred. This is write a '1' to clear bit. 0b - An arb_grant Deassertion Interrupt has not occurred 1b - An arb_grant Deassertion Interrupt has occurred

55.4.9.2.3 Functional Description

55.4.9.2.3.1 Packet Configuration

The Generic FSK Link Layer Controller provides the capability to configure the over-the-air packet structure by way of a set of programmable registers. The Generic FSK packet has a consistent structure, which consists of various elements which are transmitted and received in fixed order. Most of the packet elements have associated configurability, which allows

software to pre-configure, for example, the size of the element (in number of bits, or octets), or bit ordering of the element. In addition, the generic structure defines a variable-length header, for which some (optional) primitive parsing is provided, allowing hardware to differentiate packets based on header-bit settings, into compliant and non-compliant packets. Data Indication is provided on compliant packets only. Non-complaint packets are discarded. By default, all packet elements are transmitted and received LSB-first.

The Generic FSK Link Layer Controller supports a packet structure based on the following template:

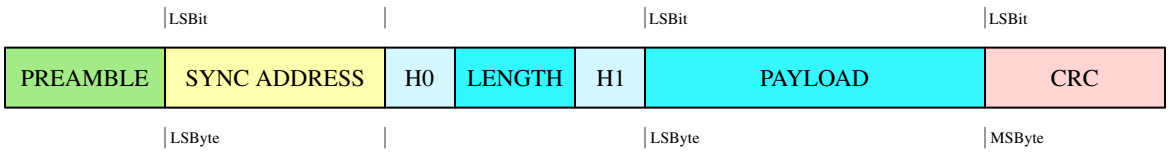


Figure 405. GENERIC_FSK Packet Structure

The Generic_FSK Link Layer controller transmits and receives packets which conform to the format shown above. The elements of the packet are transmitted in received in the order shown, although some of the packet elements may be optionally skipped, depending on the configuration options chosen.

The following diagram depicts the register configurability associated with the Preamble, Sync Address, H0, Length, and H1 elements of the packet structure.

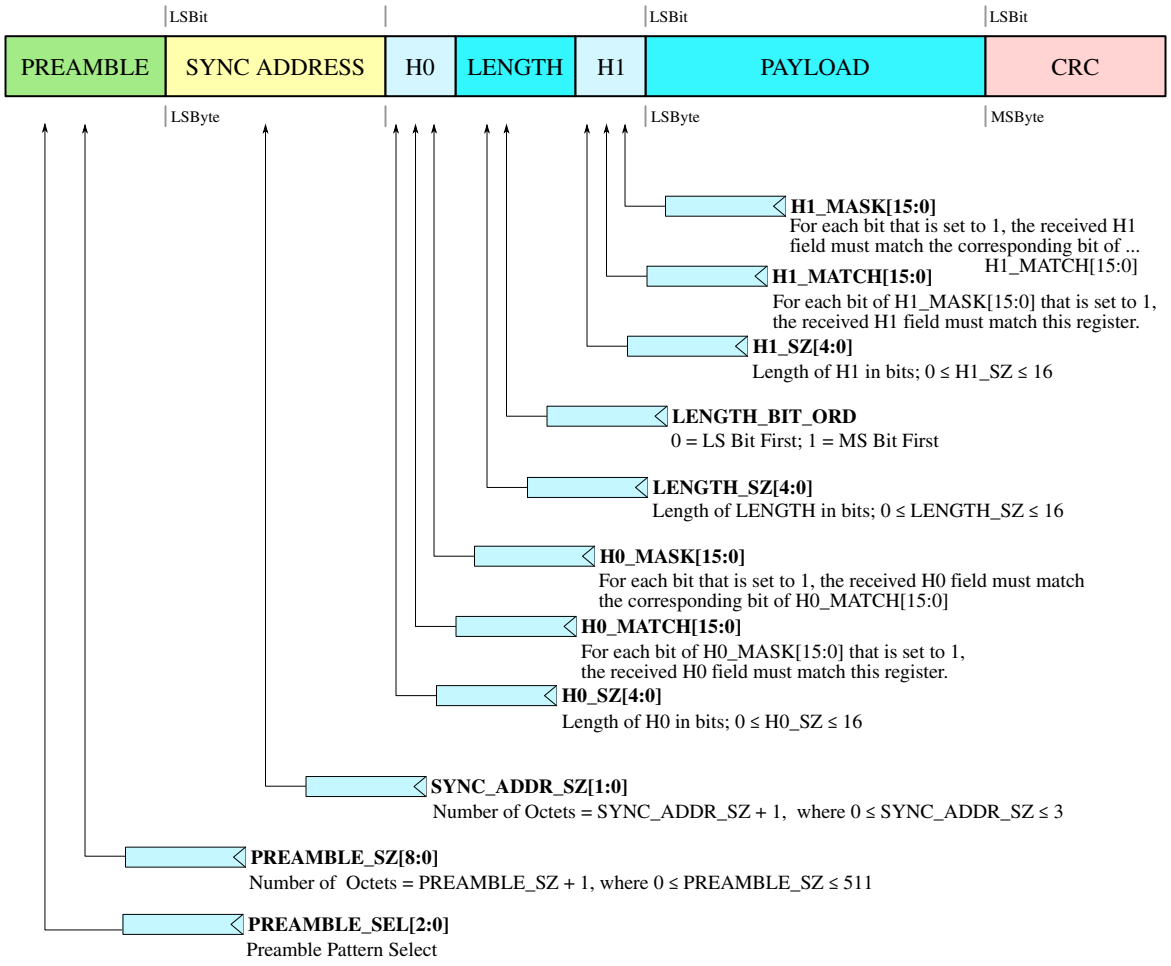


Figure 406. Packet Structure Definitions (Part 1)

The following diagram depicts the register configurability associated with the Payload and CRC elements of the packet structure.

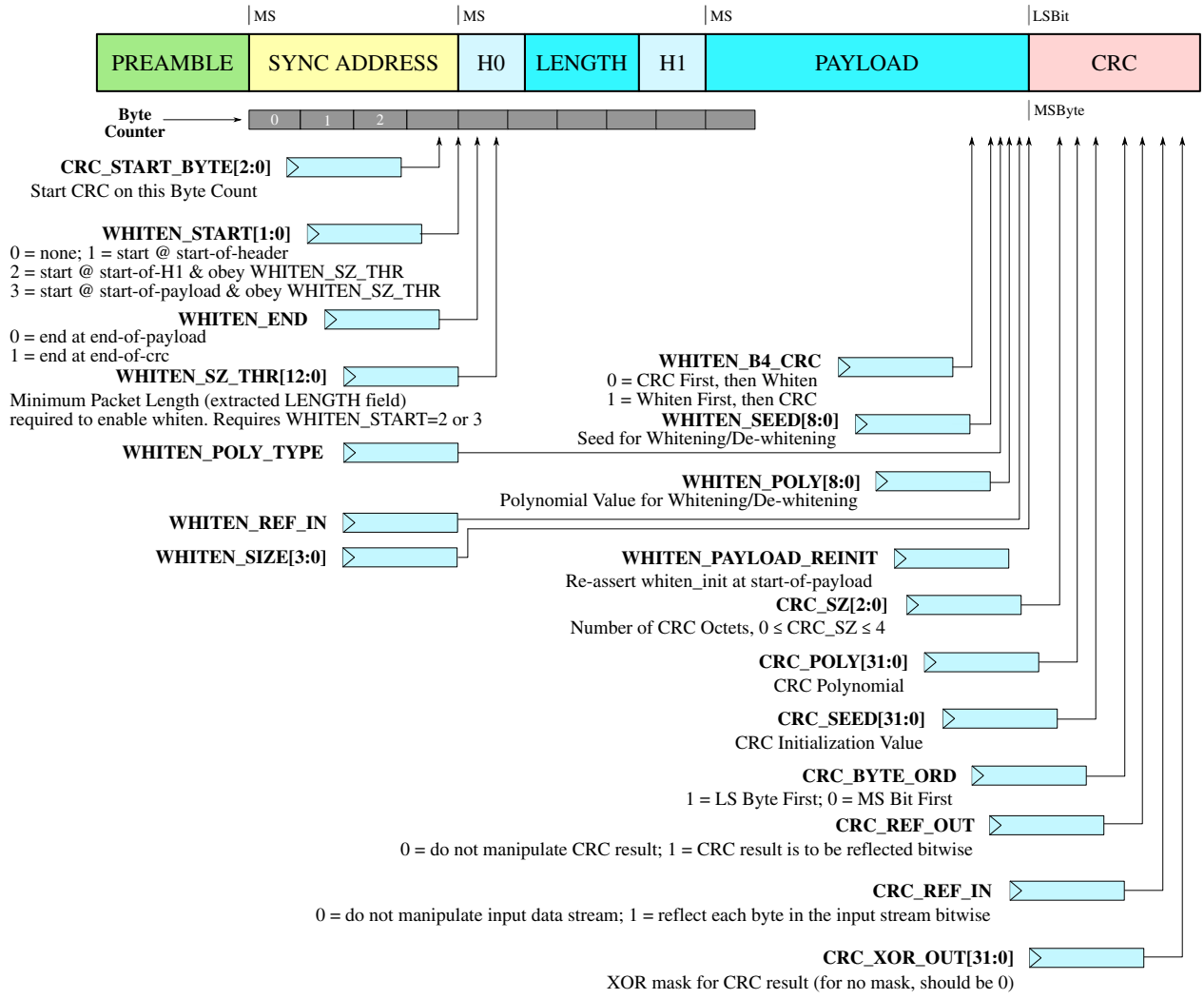


Figure 407. Packet Structure Definitions (Part 2)

The following subsections describe each of the packet elements, and the configurability associated with each.

PREAMBLE

When PREAMBLE_SEL=0x0, the preamble pattern is 0x55 or 0xAA. Like all packet elements, preamble is transmitted LSB first. The Sync Address, which immediately follows the preamble, determines which of the 2 preamble options is selected by hardware: the controller hardware selects the preamble pattern based on the first transmitted bit of Network Address, such that the last bit of preamble is the opposite polarity from the first bit of Network Address, forcing a bit transition at this boundary. The number of octets of preamble is programmable, via the PREAMBLE_SZ[2:0] register, with a range from 1 to 8 octets. A setting of PREAMBLE_SZ=0 yields a single preamble octet, whereas a setting of PREAMBLE_SZ=7 yields 8 octets. For transmission, the Generic FSK BYTE_COUNTER will be negative during preamble transmission. If PREAMBLE_SZ=0 (1 preamble octet), BYTE_COUNTER will be -1 while the single preamble octet is being transmitted. If PREAMBLE_SZ=7 (8 octets), BYTE_COUNTER will be -8 during the first preamble octet, -7 during the next, and so on and so forth, until the eighth and final octet, during which the BYTE_COUNTER will be -1. Unlike most packet elements, the preamble is generated and shifted out by hardware (only preamble and CRC are hardware-generated); there is no software setup required, and preamble does not appear in the Packet Buffer TX buffer. During reception, the Link Layer controller does not receive preamble, since preamble detection is performed by the PHY; preamble does not appear in the Packet Buffer RX buffer.

When PREAMBLE_SEL=0x1, the preamble pattern is fixed to register GEN_PREAMBLE[7:0], regardless of the Sync Address.

When PREAMBLE_SEL=0x2 or 0x3, the preamble pattern is fixed to 0b01 or 0b10, regardless of the Sync Address. At this case, PREAMBLE_SZ is the two-bits number not the byte number. If PREAMBLE_SZ=7 (8 two-bits), BYTE_COUNTER will be -8 during the first preamble two-bits, -7 during the next, and so on and so forth, until the eighth and final two-bits, during which the BYTE_COUNTER will be -1.

SYNC (NETWORK) ADDRESS

Sync Address, known synonymously as Network Address or Access Address depending on the protocol in use, is the second packet element transmitted by the Link Layer controller, and the first element received. In a network of associated devices, all devices generally share a common Network Address. The first bit of Network Address determines the preamble selection for transmission, as noted above. For TX, Network Address is the first packet element programmed into the Packet Buffer TX buffer, in LSB-first format. For RX, Network Address is also the first element stored into the Packet Buffer RX buffer, in LSB-first format. The register NTW_ADR_SZ[1:0] determines the length, in octets, of the Network Address field in the packet structure. A setting of NTW_ADR_SZ=0 yields a single-octet Network Address, whereas a setting of NTW_ADR_SZ=3 yields 4 octets. For reception, up to four unique Network Addresses are supported. Any combination of the 4 can be simultaneously searched for. Network Address correlation is performed in the PHY. Multiple Network Address capability allows GENERIC_FSK devices to be associated to more than one network simultaneously, if the network topology supports it; this feature also enables multi-protocol operation. Each of the 4 Network Address options has its own enable bit, NTW_ADR_EN[x]; when the enable bit for a Network Address is set to 1, the PHY is instructed to search for that Network Address pattern in the demodulated data, and when the enable bit is set to 0, the PHY will disregard that pattern. Each of the 4 Network Address options shares one NTW_ADR_SZ register, to program its octet size during reception. Additionally, each of the 4 Network Address options shares one register NTW_ADR_THR[2:0], which determine the number of bit errors that can be tolerated by the PHY during Network Address correlation to the respective Network Address option. Finally, when the PHY detects a match to one of the 4 Network Address options, with a number of bit errors less than or equal to the threshold programmed into NTW_ADR_THR, a read only status bit, NTW_ADR_MCH[x] becomes set, indicating which of the 4 patterns matched. After a match has been detected, the PHY shall hold NTW_ADR_MCH[x] valid (sticky), until the next receiver warmup, or the next RX recycle. (Specifically, the TSM output rx_init clears the NTW_ADR_MCH[x] bits; TSM will assert this signal during either an RX warmup or a recycle). A table listing the registers which apply to Network Address Management, for each of the 4 Network Address options, is shown below.

NTW_ADR PATTERN REGISTER	NTW_ADR ENABLE BIT	NTW_ADR SIZE REGISTER	NTW_ADR THRESHOLD REGISTER	NTW_ADR PATTERN-MATCH STATUS BIT
NTW_ADR_0[31:0]	NTW_ADR_EN[0]	NTW_ADR_SZ[1:0]	NTW_ADR_THR[2:0]	NTW_ADR_MCH[0]
NTW_ADR_1[31:0]	NTW_ADR_EN[1]	NTW_ADR_SZ[1:0]	NTW_ADR_THR[2:0]	NTW_ADR_MCH[1]
NTW_ADR_2[31:0]	NTW_ADR_EN[2]	NTW_ADR_SZ[1:0]	NTW_ADR_THR[2:0]	NTW_ADR_MCH[2]
NTW_ADR_3[31:0]	NTW_ADR_EN[3]	NTW_ADR_SZ[1:0]	NTW_ADR_THR[2:0]	NTW_ADR_MCH[3]

For Network Addresses of size less than 4 octets, the PHY expects the desired pattern to occupy the least-significant byte positions of the NTW_ADR_x register. Unused byte positions may be filled with any value.

NOTE

When programming the NTW_ADR_x, NTW_ADR_EN[x], NTW_ADR_SZ, and NTW_ADR_THR for any RX operation, the following software restrictions apply:

1. All NTW_ADR_x patterns which are enabled by their respective NTW_ADR_EN[x]=1 must be unique (disabled NTW_ADR_x registers need not follow this restriction).
2. Given restriction #1 above, not only must the patterns be unique, but the bit patterns of NTW_ADR_x and NTW_ADR_y must differ by an amount greater than the bit-error threshold, i.e., NTW_ADR_THR.

These restrictions are designed to preclude multiple, simultaneous pattern matches during a pattern search. Results are indeterminate if the restrictions are violated.

When the PHY detects a pattern match on any enabled NTW_ADR_x during reception, and asserts NTW_ADR_MCH[x], the GENERIC_FSK Link Layer Controller shall assert NTW_ADR_IRQ. The NTW_ADR_IRQ is the Link Layer's first indication that

a packet is being received. After any NTW_ADR_IRQ assertion, a packet will be received and stored into the Packet Buffer RX buffer.

HEADER

The packet header is comprised of H0, LENGTH, and H1, in that order. Each of the 3 fields of the header has programmable length, from 0 to 16 bits. A size of zero for any of the header components infers that the component is skipped in the packet structure. Although the size of the individual H0, LENGTH, and H1 components need not be aligned to a byte boundary, the overall header must be byte-aligned. That is, the sum of the sizes (in bits) of H0, LENGTH, and H1, must be a integer multiple of 8 bits. This is a software restriction, a violation of which may result in indeterminate behavior.

For GENERIC_FSK protocol, the purpose of the header is twofold:

1. To frame the LENGTH field, so that LENGTH can be extracted and interpreted by the GENERIC_FSK packet processor, so as to determine how and when perform the required bit-stream processing on the data. In most wireless protocols with regular structures, the LENGTH field is embedded within the packet header, but its positioning, number of LENGTH bits, bit ordering, and interpretation can differ. The GENERIC_FSK configurability thus provides support for a wide range of header formats, yielding a high likelihood that most current and future packet formats and LENGTH fields will be parsable by the GENERIC_FSK Link Layer controller.
2. To provide rudimentary packet filtering on the non-LENGTH header bits, namely H0 and H1. Screening of packets based on H0 and/or H1 bit comparisons can reduce power consumption, by avoiding notification to the host processor on received packets that do not comply with pre-configured H0 and/or H1 bitmasks, enabling longer MCU sleep durations.

The header components, and their configurability options, are further described below.

H0

H0 is an optional header packet element. Its length in bits is determined by register H0_SZ[4:0]. Legal values for H0_SZ are:

0 <= H0_SZ <= 16

If H0 is present (non-zero size), it can be filtered-on by the GENERIC_FSK packet processor; each bit of H0 can be forced to match a programmable bitmask, or ignored. Every bit of H0 that is forced to match the bitmask, must match; otherwise, the packet is rejected. The registers H0_MATCH[15:0] and H0_MASK[15:0] control H0 filtering. For every bit of H0_MASK that is set to one, the received bit in the H0 portion of the header must match the corresponding bit of H0_MATCH. Bits of H0_MASK which are cleared, do not require a match of received data bits to H0_MATCH bits. If any bit of received data which is qualified with H0_MASK[x]=1 fails to match its corresponding H0_MATCH[x] bit, the H0_FAIL register status bit will be set, an RX recycle will occur, and the host processor will not be notified. (The recycle operation will self-clear the H0_FAIL bit). For debug purposes, the control bit REC_BAD_PKT has been provided. If REC_BAD_PKT=1 and a H0-mismatch occurs, the RX recycle is prevented, the full packet is received, the MCU is notified, and the H0_FAIL flag stays asserted. In this case, the H0-fail flag will stay asserted until the next RX warmup (specifically, rx_init clears the bit).

The equation for H0_FAIL is thus:

H0_FAIL = [$H0_{received}$^H0_MATCH[15:0]) & H0_MASK > 0

LENGTH

LENGTH is an optional header packet element. Its length in bits is determined by the register LENGTH_SZ[4:0]. Legal values for LENGTH_SZ are:

0 <= LENGTH_SZ <= 16

If LENGTH is present (non-zero size), its value determines the number of octets remaining in the packet after the header is complete, that is, PAYLOAD octets plus CRC octets, as depicted below.

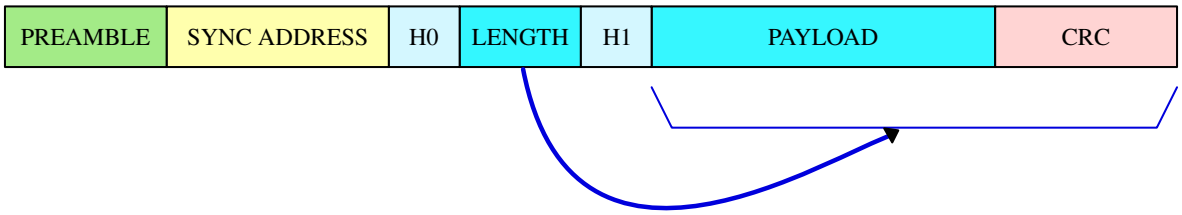


Figure 408. LENGTH Field Interpretation with No LENGTH_ADJ

For example, if the payload consists of 16 octets, and CRC is 4 octets, the LENGTH field of the header should be set to 20. The packet processor will extract the packet length from the LENGTH field of the header, and use this information to determine the timing of the CRC, Whitening control signals, as well as end-of-packet interrupt triggering.

In case a protocol exists for which the LENGTH field is to be interpreted differently, a configurability option has been provided to compensate for this. Hypothetically, this could happen, for example, in a proprietary packet structure, in which LENGTH is to be defined to represent PAYLOAD only, excluding CRC. The register LENGTH_ADJ, allows a fixed offset to be applied to the extracted LENGTH field, such that the sum of LENGTH + LENGTH_ADJ represents the number of PAYLOAD + CRC octets, as depicted below:

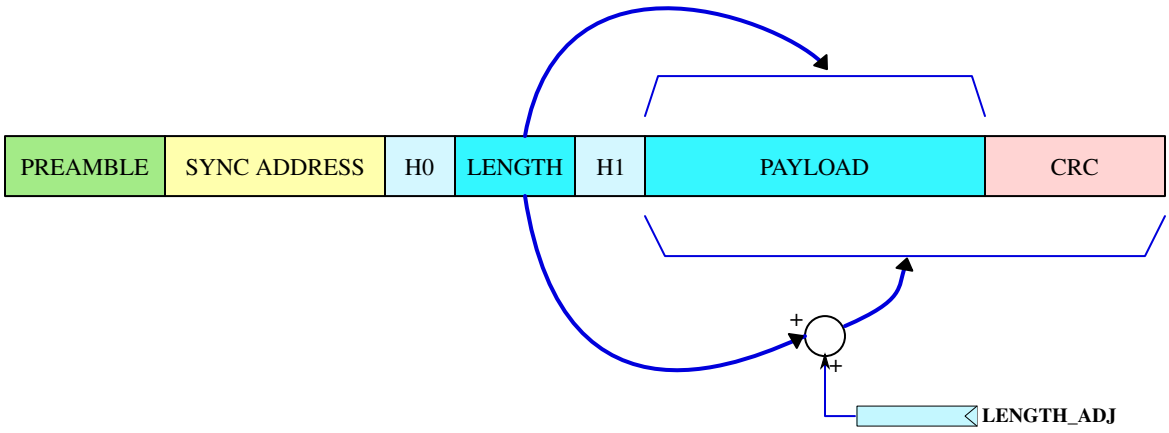


Figure 409. LENGTH Field Interpretation with LENGTH_ADJ

Applying the previous example, if, using a proprietary format, LENGTH were to be defined as payload-length only, LENGTH would be set to 16. In order for the packet processor to correctly time the CRC and Whitening control signals, etc., the LENGTH_ADJ register should be programmed to 4 in this case, so that the packet processor would operate on a 20-octet "PAYLOAD + CRC" PDU. The LENGTH_ADJ register represents a signed value, so that negative offsets can be applied. The legal range for LENGTH_ADJ is:

-1024 <= LENGTH_ADJ <= 1023

The LENGTH field could be ordered MSB first or LSB first. The packet processor must be able to accommodate either case, in order to extract the LENGTH field properly from the header, so a configurability option has been provided. The register bit LENGTH_BIT_ORD specifies the bit order. If LENGTH_BIT_ORD=0, ordering is LSB-first, as shown below.

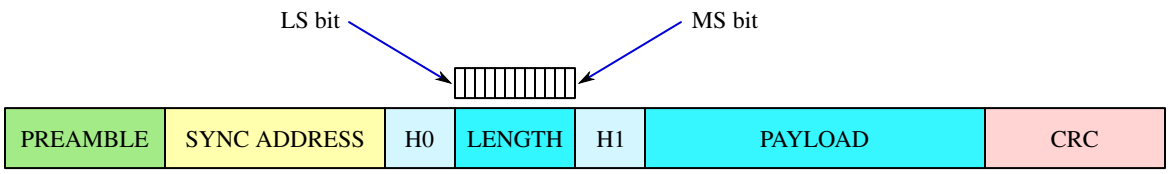


Figure 410. LENGTH Bit Ordering (LSB First)

If LENGTH_BIT_ORD=1, ordering is MSB-first, as shown below.

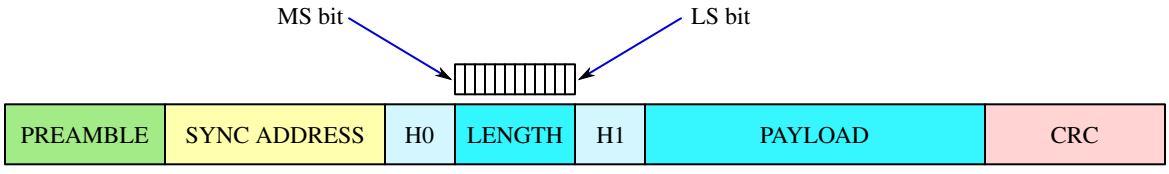


Figure 411. LENGTH Bit Ordering (MSB First)

If LENGTH_SZ=0, then LENGTH is not present in the header. The packet processor interprets this as LENGTH=0. If, in this case, LENGTH_ADJ=0 also, the packet processor interprets this to mean that the packet ends with the last bit of H1, and there are no PAYLOAD or CRC octets (CRC_SZ should be programmed to 0 in this case). There is no CRC verification in this scenario. If, in this case, LENGTH_SZ=0, meaning an empty payload field, but a CRC check is desired, CRC_SZ should be programmed to the desired number of CRC octets, and LENGTH_ADJ should be programmed to the same value: LENGTH_ADJ = CRC_SZ.

For GENERIC_FSK, valid payload lengths are from 0 to 2047 octets, i.e.,

0 <= LENGTH <= 2047

If LENGTH_ADJ is non-zero, then the maximum limit of 2047 applies to the sum of LENGTH and LENGTH_ADJ, i.e.,

0 <= (LENGTH + LENGTH_ADJ) <= 2047

A restriction on LENGTH_ADJ follows:

If the packet ends immediately after the LENGTH field of the header, i.e., H1_SZ=0 and there is neither payload nor CRC, then LENGTH_ADJ must be set to 0.

Although the LENGTH field has a maximum size of 16 bits, since the maximum receivable packet size is 2047 bytes, any received LENGTH bits above bit [10] are ignored by the packet processor, which effectively clamps LENGTH at 2047 bytes. There may be cases where it is desirable to limit the maximum received packet length to values less than 2047, so that received packets whose extracted LENGTH field indicates to be larger than a pre-set limit, are rejected by the packet processor. A configurability option has been provided for this, the register LENGTH_MAX[6:0]. The LENGTH_MAX setting configures a pre-set limit on the extracted LENGTH field, in multiples of 16 octets. A special case exists for LENGTH_MAX=0 (the register default), which implies no hardware-limiting; this setting should be used to allow reception of full-length 2047-octet packets. If LENGTH_MAX=1, packets with LENGTH >= 16 octets are rejected. If LENGTH_MAX=2, packets with LENGTH >= 32 octets are rejected. If LENGTH_MAX=127 (the maximum setting) packets with LENGTH >= 2032 octets are rejected. To determine compliance with LENGTH_MAX, the packet processor compares the LENGTH_MAX register with the extracted LENGTH field directly, not the adjusted length (LENGTH + LENGTH_ADJ). If a packet is received during which the extracted LENGTH field exceeds the LENGTH_MAX setting, the LENGTH_FAIL register status bit will be set, an RX recycle will occur, and the host processor will not be notified. (The recycle will self-clear the LENGTH_FAIL bit). For debug purposes, the control bit REC_BAD_PKT has been provided. If REC_BAD_PKT=1 and a LENGTH_MAX violation occurs, the RX recycle is prevented, the full packet is received, the MCU is notified, and the LENGTH_FAIL flag stays asserted. In this case, the LENGTH_FAIL flag will stay asserted until the next RX warmup (specifically, rx_init clears the bit).

H1

H1 is an optional header packet element. Its length in bits is determined by register H1_SZ[4:0]. Legal values for H1_SZ are 0 ≤ H1_SZ ≤ 16. If H1 is present (non-zero size), it can be filtered-on by the GENERIC_FSK packet processor; each bit of H1 can be forced to match a programmable bitmask, or ignored. Every bit of H1 that is forced to match the bitmask, must match; otherwise, the packet is rejected. The registers H1_MATCH[15:0] and H1_MASK[15:0] control H1 filtering. For every bit of H1_MASK that is set to one, the received bit in the H1 portion of the header must match the corresponding bit of H1_MATCH. Bits of H1_MASK which are cleared, do not require a match of received data bits to H1_MATCH bits. If any bit of received data which is qualified with H1_MASK[x]=1 fails to match its corresponding H1_MATCH[x] bit, the H1_FAIL register status bit will be set, an RX recycle will occur, and the host processor will not be notified. (The recycle will self-clear the H1_FAIL bit). For debug purposes, the control bit REC_BAD_PKT has been provided. If REC_BAD_PKT=1 and a H1-mismatch occurs, the RX recycle is prevented, the full packet is received, the MCU is notified, and the H1_FAIL flag stays asserted. In this case, the H1-fail flag will stay asserted until the next RX warmup (specifically, rx_init clears the bit).

The equation for H1_FAIL is thus:

H1_FAIL = [$H1_{received}$^H1_MATCH[15:0]) & H1_MASK > 0

PAYLOAD

In the GENERIC_FSK packet structure, the payload directly followed the header. For transmission, payload bytes are serialized, and transmitted, LSB first, in order, as they are received from the Packet Buffer interface. During reception, payload bytes are received LSB first, undergo serial-to-parallel conversion to octets, 4 octets are packed into a 32-bit word, and the word is transferred to memory via Packet Buffer. For both TX and RX, since the packet processor interprets the extracted LENGTH to represent the number of Payload + CRC octets, the actual number of payload octets is computed from the following equation:

Number_of_Payload_Octets = ($LENGTH_{extracted}$ + LENGTH_ADJ – CRC_SZ)

For TX, payload bytes are subjected to whitening . They are also shifted through CRC to compute a CRC checksum value. Whitening and CRC computing are all optional. For RX, payload bytes are subjected to de-whitening and CRC verification. De-whitening and CRC verification are all optional. For both TX and RX, (de-)whitening can precede CRC, or follow it. For more details on CRC, whitening, see Section Bitstream Processing.

CRC

In the GENERIC_FSK packet structure, the CRC field directly followed the payload. The size of the CRC field, in octets, is set by the CRC_SZ register. The valid range for CRC_SZ is:

0 <= CRC_SZ <= 4

For both TX and RX, the portion of the packet which are subjected to CRC is programmable, via the register CRC_START_BYTE[3:0]. The register CRC_START_BYTE programs the absolute byte count of the first byte for which CRC shifting is desired. The CRC_START_BYTE value is referenced to the GENERIC_FSK Byte Counter. The Byte Counter starts at 0 for the first octet of Sync Address, and increments octet-for-octet henceforth, until the last octet of CRC is transmitted or received. An example of CRC_START_BYTE programming is shown below.

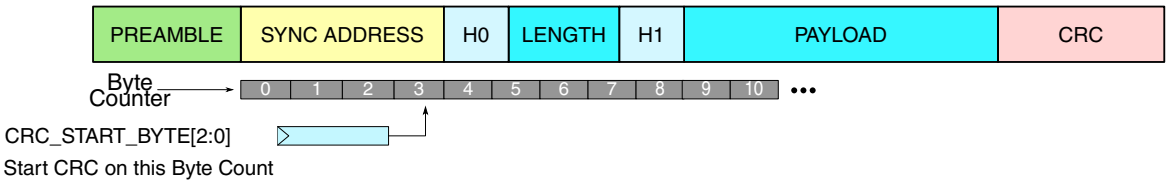


Figure 412. CRC Start Byte

In this example

SYNC_ADDR_SZ = 4

Header Size = 4 i.e. (H0_SZ + LENGTH_SZ +H1_SZ)/8 = 4

To initiate CRC shifting at the start of Sync Address, program CRC_START_BYTE = 0

To initiate CRC shifting at the start of Header, program CRC_START_BYTE = 4

To initiate CRC shifting at the start of Payload, program CRC_START_BYTE = 9

The CRC_START_BYTE should not be programmed so large as to exceed the length of the PDU; that is, larger than extracted LENGTH – CRC_SZ. If CRC_SZ=0, program CRC_START_BYTE=0.

For more details on how the GENERIC_FSK packet processor controls the CRC generation and verification processes, see Section Bitstream Processing.

EXAMPLE

A real-world example follows, describing how the GENERIC_FSK Link Layer packet configuration can be applied to transmit and receive packets which conforms to, e.g., IEEE 802.15.4-2015 SUN FSK PHY packet format.

IEEE 802.15.4-2015 SUN FSK PHY header(PHR) is as follows.

Table 478. Format of the PHR(without mode switching) for MR-FSK

Bit string index	0	1-2	3	4	5-15
Bit mapping	MS	R_1-R_0	FCS	DW	L_{10}-L_0
Field Name	Mode Switch	Reserved	FCS Type	Data Whitening	Frame Length

The GENERIC_FSK Link Layer controller supports such a header format, and can be programmed to transmit and receive packets which conform to it. To correctly size and order the GENERIC_FSK header elements so as to conform with this header format:

1. The LENGTH field ends the header, so there is no H1 (H1_SZ=H1_MASK=0)
2. The 5 bits which precede LENGTH in the header, are grouped into H0 (H0_SZ=5)
3. The size of the LENGTH field is 11 bits (LENGTH_SZ=11)
4. The LENGTH is specified as MSB-first (LENGTH_BIT_ORD=1)

For this example, we would like to leverage the H0-filtering capability built into the GENERIC_FSK Link Layer to screen the 802.15.4-2015 SUN FSK header, to require that the bit fields in the received packet comply with the following conditions:

1. MS (Mode Switch, bit [0] of header) = 0
2. DW (Data Whitening, bit [4]) = 1
3. FCS (FCS Type, bit [3]) = 1 (Four-octet FCS)
4. Bits [1] and [2] are marked as Reserved, to be ignored on reception

Given these parameters, the following diagram shows how we would program the GENERIC_FSK Link Layer controller to accept this packet configuration, and then screen H0 accordingly:

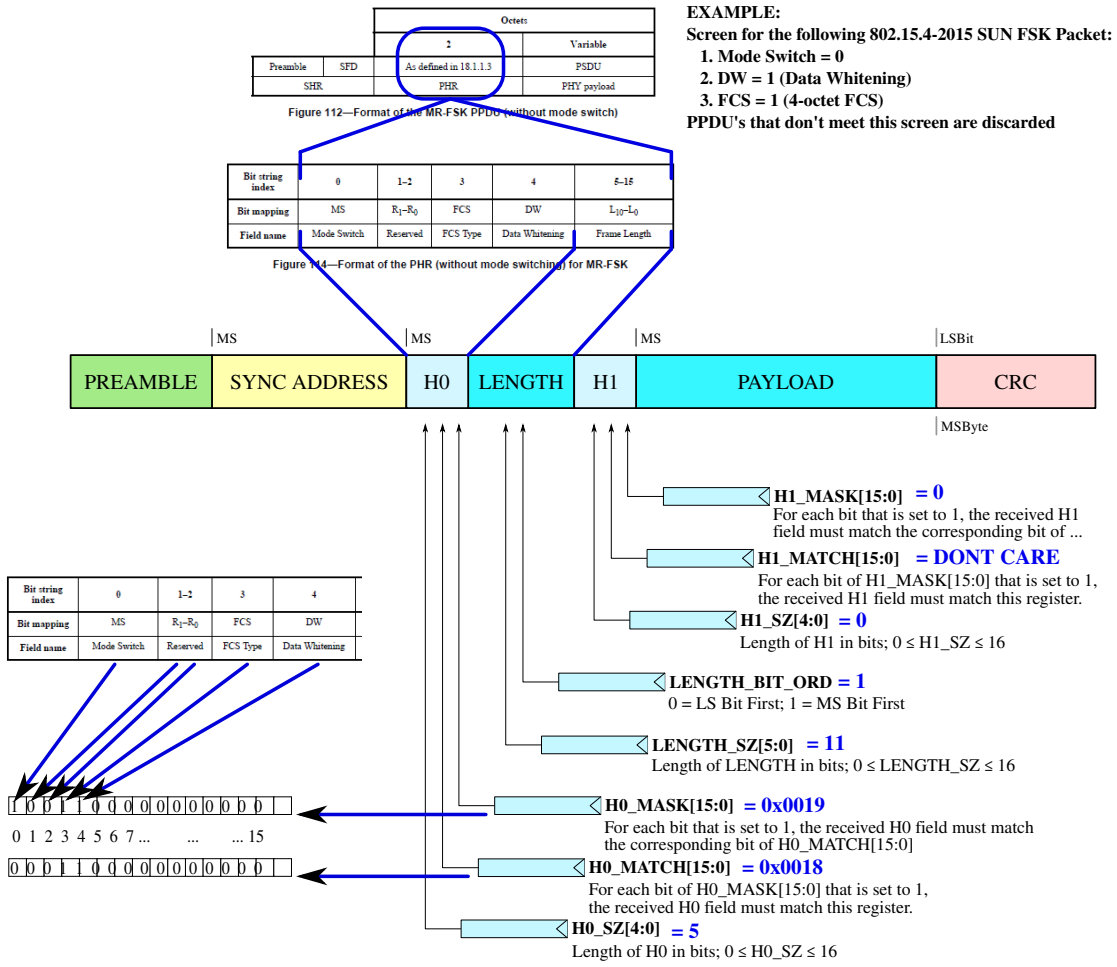


Figure 413. GENERIC FSK PACKET STRUCTURE AS APPLIED TO 802.15.4-2015 SUN FSK PHR

Received packets which meet the H0 filtering screen are fully received, and Data Indication is provided to the host processor. Packets which fail the screen result in an RX recycle, and no notification to the host.

55.4.9.2.3.2 Bitstream Processing

The CRC/Whitener block is inside RBME.

The GENERIC_FSK Link Layer controller provides control for CRC verification and Data De-whitening as the packet is being received.

The RBME provides control for CRC generation and Data Whitening as the packet is being transmitted.

The following diagram provides another perspective of bitstream processing.

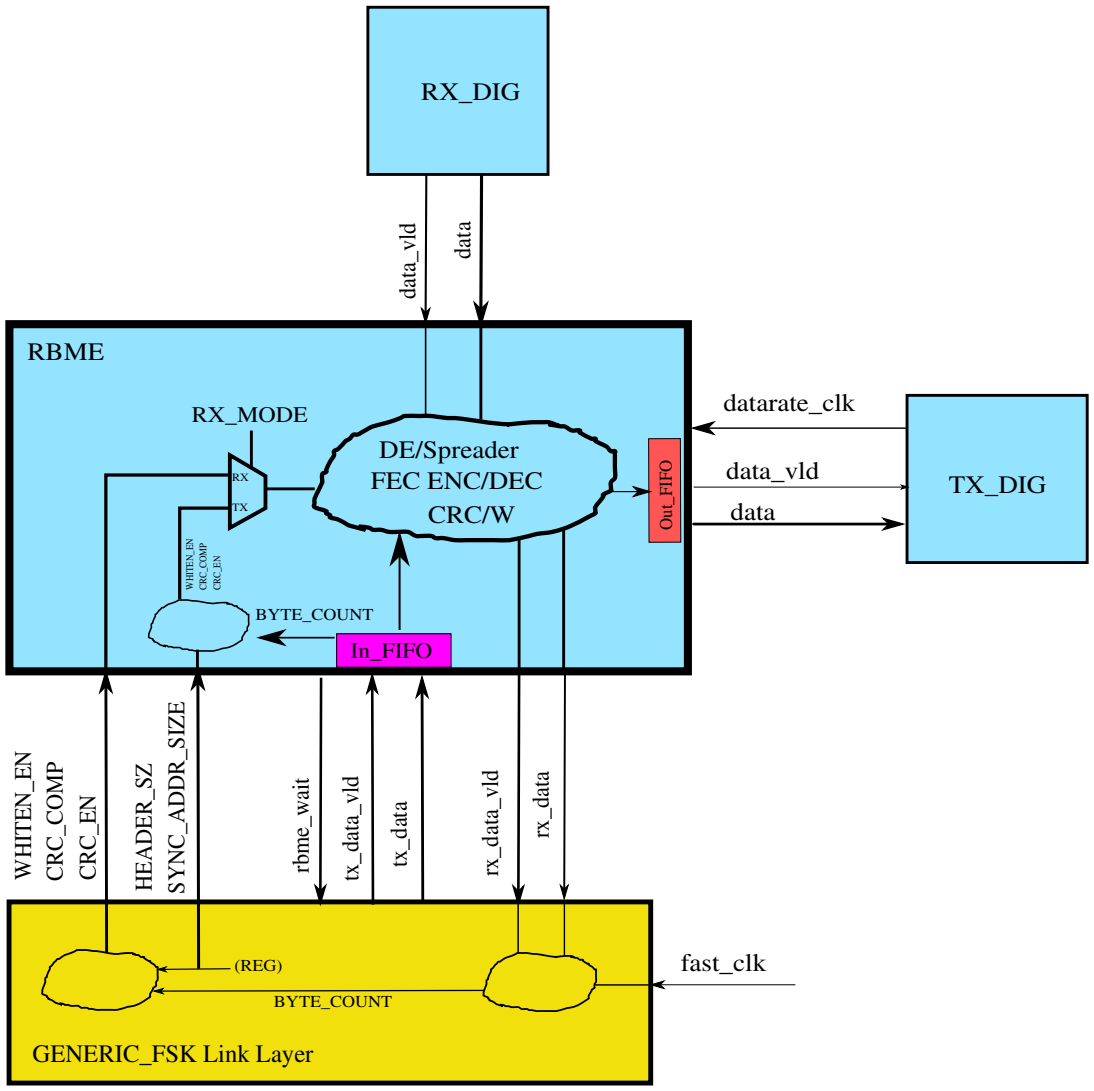


Figure 414. Basic Bitstream Between RBME and Link Layer

The following sections provide more detail on how CRC and whitening, are controlled, and describes the configuration options associated with each.

CRC & WHITENING

The GENERIC_FSK Link Layer controller dynamically manages CRC verification process (for RX), RBME manages the CRC generation process (for TX), by way of 3 timing signals that it asserts to the CRC/Whitener module.

The CRC engine in the CRC/Whitener block shall only shift incoming bits (data_in) through its LFSR when all the following conditions are met:

1. crcw_init=0
2. crc_en=1
3. tx_data_in_vld=1 (signal is data_in_vld at the CRC/Whitener)

The CRC/Whitener shall only shift bits out of its CRC LFSR, to the TX digital block, when all the following conditions are met:

1. crcw_init=0
2. crc_comp=1

3. tx_data_in_vld=1 (signal is data_in_vld at the CRC/Whitener)

The GENERIC_FSK Link Layer and RBME use the extracted LENGTH field of the packet, as well as the registers which control the size of the individual packet elements, in order to determine when to assert the CRC-related control signals, **crc_en** and **crc_comp**. This is true for both transmission and reception.

The calculations used by the GENERIC_FSK Link Layer to determine when to assert the CRC-related control signals are summarized in the Figure 415.

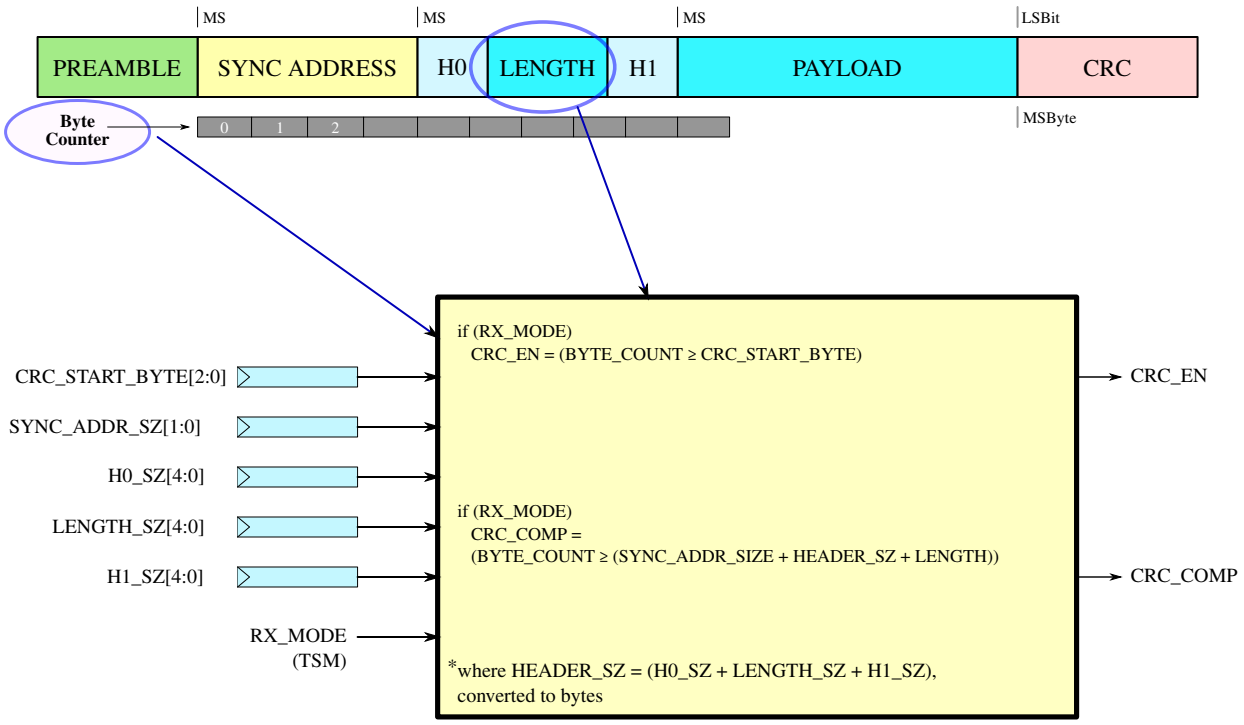


Figure 415. CONTROL OF CRC IN LINK LAYER (RX)

The calculations used by the RBME to determine when to assert the CRC related control signals are summarized in the Figure 416.

Note, the RBME and Link Layer has their own BYTE_COUNT, for TX and RX respectively.

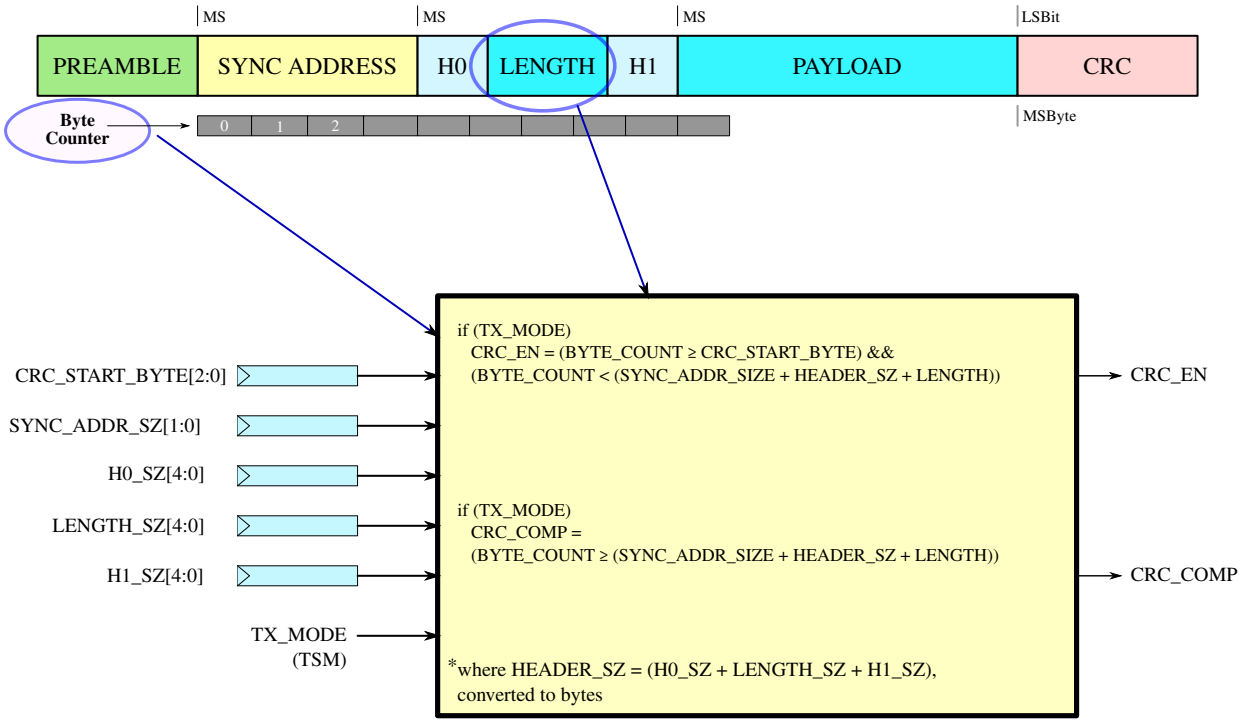


Figure 416. CONTROL OF CRC IN RBME (TX)

Likewise, the Link Layer controller and RBME optionally enables whitening and de-whitening of the packet bitstream, and controls the portions of the packet which are subjected to whitening. For both TX and RX, the portion of the packet which are subjected to whitening/de-whitening is programmable; the following table describes the 3 registers which determine which elements of the packet bitstream are whitened/de-whitened.

REGISTER NAME	FUNCTIONAL DESCRIPTION
WHITEN_START[1:0]	**0**: No whitening/de-whitening **1**: Begin whitening/de-whitening at the start of the header (H0) field, regardless of packet length. **2**: Begin whitening/de-whitening at the start of the header H1 field, but only if the extracted length field meets or exceeds WHITEN_SZ_THR, i.e., ($LENGTH_{extracted}$ >= WHITEN_SZ); otherwise, there is no whitening/de-whitening of the bitstream **3**: Begin whitening/de-whitening at the start of the PAYLOAD, but only if the extracted length field meets or exceeds WHITEN_SZ_THR, i.e., ($LENGTH_{extracted}$ >= WHITEN_SZ); otherwise, there is no whitening/de-whitening of the bitstream
WHITEN_END	**0**: Whitening/de-whitening ends with the last bit of the last PAYLOAD byte

Table continues on the next page...

Table continued from the previous page...

REGISTER NAME	FUNCTIONAL DESCRIPTION
	1: Whitening/de-whitening ends with the last bit of the last CRC byte
WHITEN_SZ_THR[12:0]	If WHITEN_START=2 or WHITEN_START=3, apply whitening/de-whitening at the point determined by WHITEN_START, but only if the extracted LENGTH field of the packet meets or exceeds this WHITEN_SZ_THR threshold; if LENGTHextracted < WHITEN_SZ_THR, there is no whitening/de-whitening of the bitstream. If WHITEN_START < 2, this register has no effect.

The GENERIC_FSK Link Layer controller dynamically manages the data whitening process (for TX), and de-whitening (for RX), via 2 timing signals that it asserts to the CRC/Whitener module.

The Whitening/de-whitening engine in the CRC/Whitener block shall only shift incoming bits (data_in) through its LFSR when all the following conditions are met:

1. crcw_init=0
2. whiten_en=1
3. tx_data_in_vld=1 (signal is data_in_vld at the CRC/Whitener)

The GENERIC_FSK Link Layer and RBME use the extracted LENGTH field of the packet, as well as the registers which control the size of the individual packet elements, in order to determine when to assert the whitening-related control signal, whiten_en. This is true for both transmission and reception.

The calculations used to determine when to assert whiten_en control signals are summarized in the following diagram.

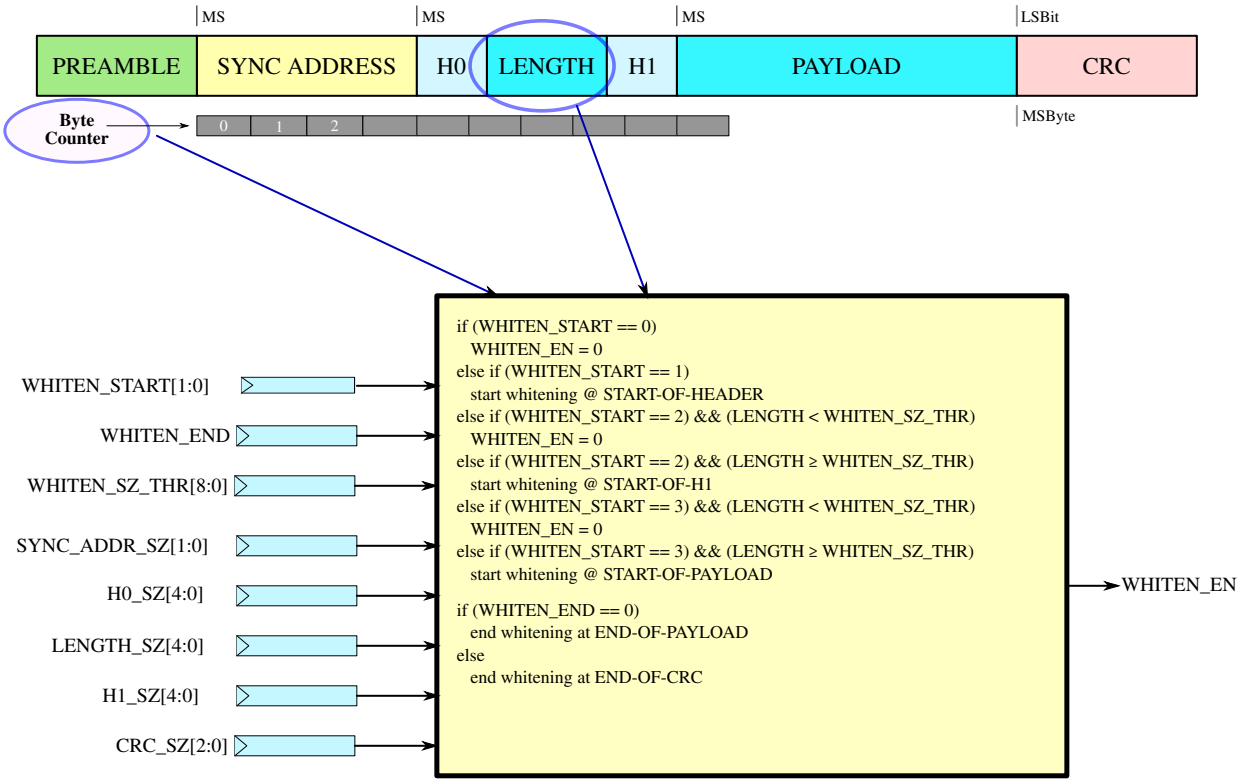


Figure 417. CONTROL OF WHITENING (TX and RX)

The order in which CRC and whitening are performed, is configurable, via the static register bit WHITEN_B4_CRC. This bit is programmed in the RBME, and sent to the CRC/Whitening engine to control the ordering. The following table describes how the bitstream is processed for both settings of WHITEN_B4_CRC, for both TX and RX.

	TX	RX
WHITEN_B4_CRC=1	In the CRC/Whitener, the CRC engine shifts in data bits which have been whitened first through the whitener LSFR. The entire post-whitened bitstream is then transmitted to the TX Digital.	Since the entire bitstream, including CRC, has been pre-whitened, the incoming bitstream from the PHY is shifted directly through the CRC LFSR to verify the checksum. The entire bitstream is simultaneously de-whitened by the whitener LFSR and shifted out to the Link Layer.
WHITEN_B4_CRC=0	In the CRC/Whitener, the CRC engine shifts in data bits which have not been whitened. Whitening is performed on the CRC LFSR output. The entire post-whitened bitstream is then transmitted to the TX Digital.	The entire bitstream is de-whitened first by the whitener LFSR, and then shifted through the CRC LFSR to verify the checksum. The entire de-whitened bitstream is simultaneously shifted out to the Link Layer.

A diagram depicting an example of bitstream processing flow during packet transmission, with WHITEN_B4_CRC=1 (the register default), is shown below.

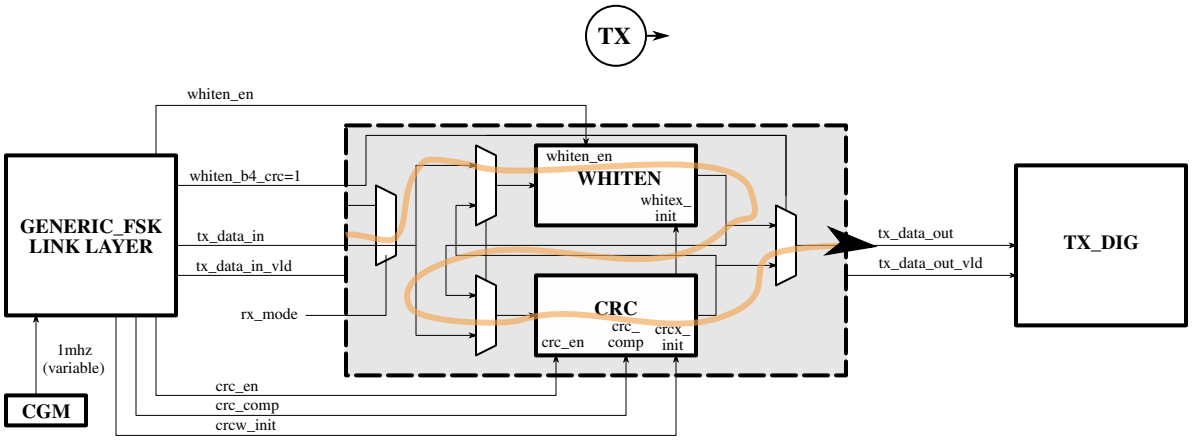


Figure 418. Whitening Before CRC (TX)

A diagram depicting an example of bitstream processing flow during packet reception, with WHITEN_B4_CRC=1, is shown below.

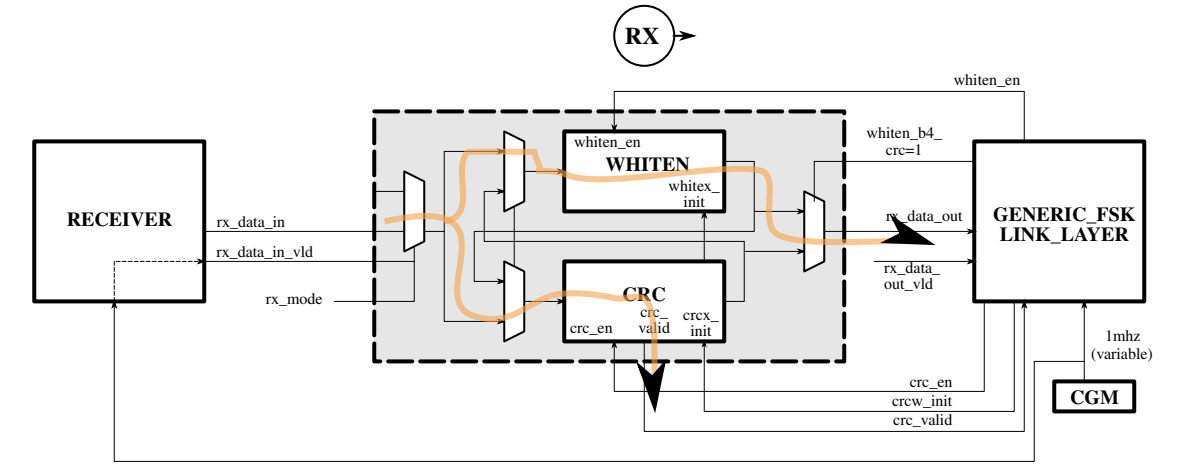


Figure 419. Whitening Before CRC (RX)

A timing diagram depicting a real-world example of TX bitstream processing timing is shown below. In this example, the desired start point for CRC shifting is the start of Network Address, so CRC_START_BYTE=0. The desired start point for whitening is the start of the header (H0), so WHITEN_START=1. It is also desired to stop whitening after the last bit of the last PAYLOAD byte, so WHITEN_END=0.

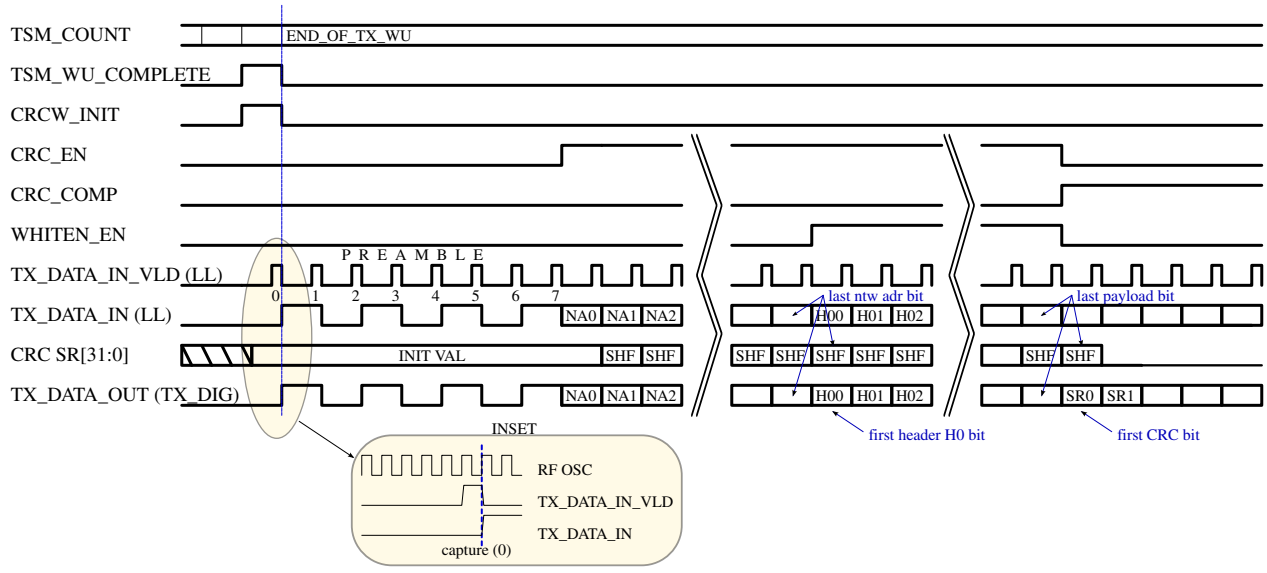


Figure 420. BIT STREAM PROCESSING CRC AND WHITENING TX TIMING

The bitstream depicted in the previous timing diagram, with CRC and whitening applied, is shifted out to the TX digital block, and transmitted over the air. A receiving device, will receive and process this same bitstream, by applying de-whitening, and verifying CRC on the de-whitened bitstream. The CRC and whitening parameters (registers) would be programmed to the receiving device, identically to how they are programmed for the transmitting device, specifically:

CRC_START_BYTE=0

WHITEN_START=1

WHITEN_END=0

A similar timing diagram depicting this RX bitstream processing timing, is shown below.

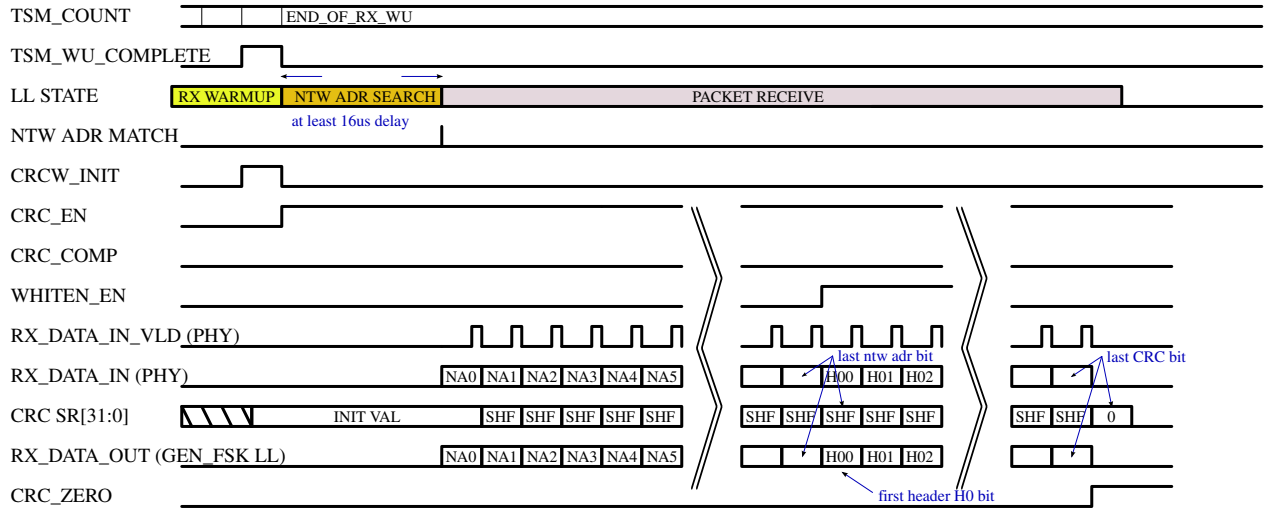


Figure 421. BIT STREAM PROCESSING CRC AND WHITENING RX TIMING

The GENERIC_FSK Link Layer controller provides a special whitening option, that causes the whitening LFSR to be re-initialized at the start of the PAYLOAD. The register bit WHITEN_PAYLOAD_REINIT, when 1, will result in the LFSR being re-initialized to its programmed initial state, determined by WHITEN_SEED[8:0], coincident with the first bit of the first byte of PAYLOAD. This applies to both TX and RX. When WHITEN_PAYLOAD_REINIT=0, there is no such re-initialization. This feature is only relevant when whitening/de-whitening is programmed to start before the PAYLOAD, i.e., WHITEN_START=1 or WHITEN_START=2.

The RBME provides a number of other configurability options, which affect the internal operation of the CRC engine, and not the Link Layer controller. A listing of these registers which affect CRC internal operation, appears in the table below. For more details on how these registers affect CRC operation, see the CRC/Whitener Chapter.

CRC-related Register	Brief Description
CRC_POLY[31:0]	CRC Polynomial 1: XOR exists in the bit's feedback path 0: no XOR in the bit's feedback path
CRC_SEED[31:0]	CRC Initialization Value
CRC_BYTE_ORD	0: LS Byte First 1: MS Byte First
CRC_REF_IN	0: do not manipulate input data stream 1: reflect each byte in the input stream bitwise
CRC_REF_OUT	0: do not manipulate CRC result 1: CRC result is to be reflected bitwise (operated on entire word)
CRC_XOR_OUT[31:0]	XOR mask for CRC result (for no mask, should be 0)

Similarly, there are a number of configurable whitening options provided by the RBME, which affect the internal operation of the whitening engine, and not the Link Layer controller. A listing of these registers which affect whitener internal operation, appears in the table below. For more details on how these registers affect whitener operation, see the CRC/Whitener Chapter.

Whitener-related Register	Brief Description
WHITEN_ POLY[8:0]	Polynomial Value for Whitening/De-whitening
WHITEN_POLY_TYPE	Whiten polynomial type. A Fibonacci type LFSR is used with the whiten polynomial if this bit is asserted. Otherwise, a Galois type LFSR is used.
WHITEN_SEED[31:0]	Initialization Value for Whitening/De-whitening
WHITEN_SIZE[3:0]	Whitener Length
WHITEN_REF_IN	The input data stream is reflected, bit-wise, per byte, if this bit is asserted. Bit 7 becomes bit 0, bit 6 becomes bit 1, etc. This bit will only cause the reflection of the payload data bits as they are used in the whiten calculation and will not cause any change in the output bit order.

55.4.9.2.3.3 Fixed Packet Length

The GENERIC_FSK Link Layer controller expects a stream of consecutive bits specifying the length of the packet, to be transmitted and received over-the-air, as part of the packet bitstream, with the length specification typically encoded into the packet header, as shown below.

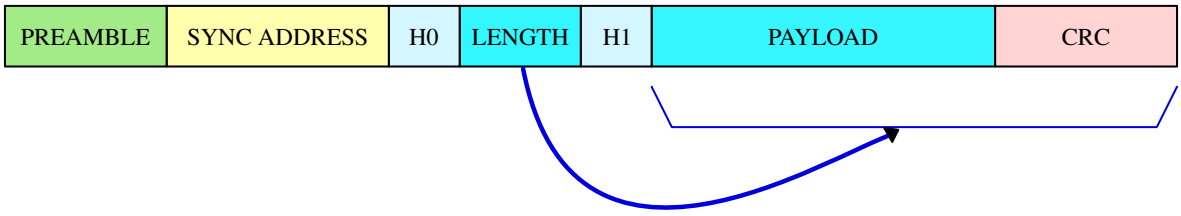


Figure 422. PACKET STRUCTURE WITH LENGTH FIELD IN HEADER

Some protocols don't include any length instructions in the packet header, as shown below.

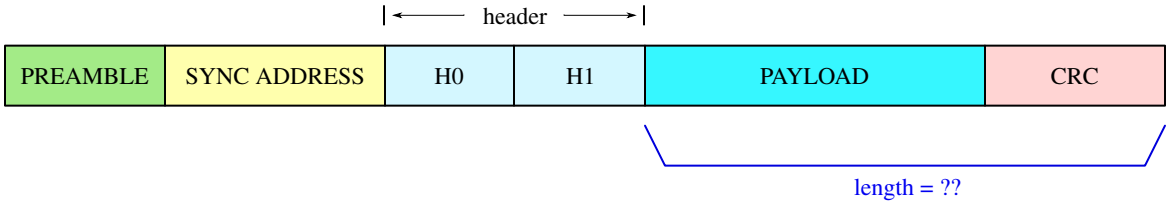


Figure 423. PACKET STRUCTURE WITH NO LENGTH FIELD IN HEADER

And, some protocols may not include a header at all.

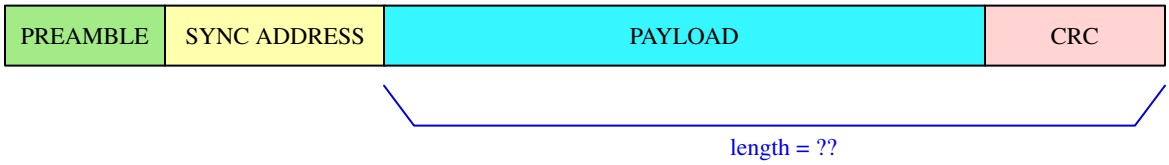


Figure 424. PACKET STRUCTURE WITH NO HEADER

In the aforementioned cases, another method of specifying length to the GENERIC_FSK Link Layer hardware must be employed. In both cases, if the length is not specified as part of the over-the-air bitstream, then GENERIC_FSK software must know the length in advance of the TX or RX operation, or know how to calculate it. So GENERIC_FSK software will provide this information to the hardware, in advance of the TX or RX operation, by programming the following registers:

- PACKET_CFG[LENGTH_SZ] = 0 : instructs the hardware that there is no LENGTH info in the packet header, and thus forces the hardware to use LENGTH_ADJ for packet length
- PACKET_CFG[LENGTH_ADJ] = ??? : program the desired (known) length of PAYLOAD + CRC
- CRC_CFG[CRC_SZ] = ??? : program the desired number of CRC octets

As an example, a packet is to be transmitted or received using the GENERIC_FSK Link Layer hardware. The packet has no header, but the length of the payload is known in advance to be 57 octets, and the number of CRC octets is 3. In this case, LENGTH_ADJ is programmed to 60 (57 + 3) and CRC_SZ is programmed to 3, as shown below.

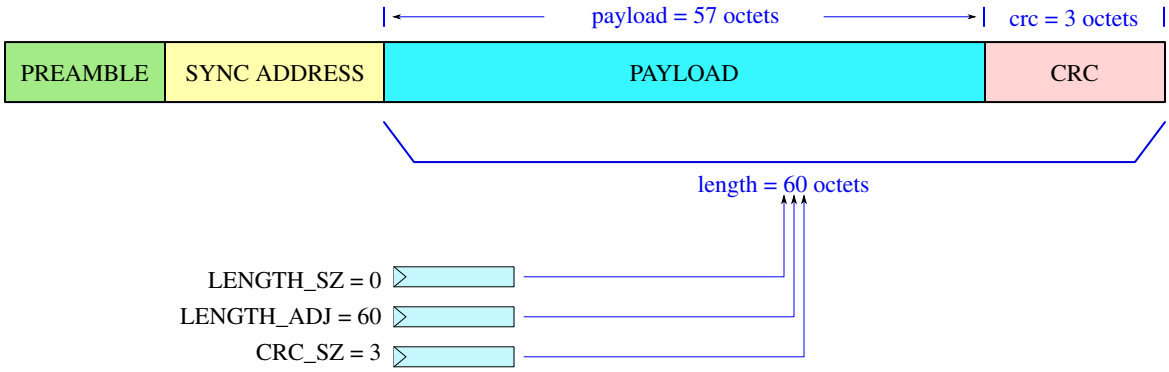


Figure 425. PACKET STRUCTURE WITH NO HEADER AND FIXED PACKET LENGTH

55.4.9.2.3.4 Packet Storage

The GENERIC_FSK Link Layer has access to a Packet Buffer (also known as Packet RAM), to store data to be transmitted, and to receive incoming packet data. The GENERIC_FSK Link Layer software prepares data to be transmitted, by loading the octets, in order, into the Packet Buffer. After reception, the octets received over the air are stored into the Packet Buffer, in the order they are received. The Packet Buffer contains separate spaces for TX and RX data, so incoming receive octets don't overwrite previously-loaded content intended for transmission. The Packet Buffer is large enough to accommodate the longest GENERIC_FSK packet, 2047 octets, plus header and CRC octets, for either transmit or receive.

The GENERIC_FSK Packet Buffer is contained within the multi-protocol Packet RAM. When operating in GENERIC_FSK mode, the entire Packet RAM is allocated to the GENERIC_FSK packet buffer. Separate TX and RX buffers within the consolidated RAM means that incoming RX packets don't overwrite TX packet data. Because the Packet RAM is 32/64bits wide, the GENERIC_FSK Link Layer controller hardware will address the consolidated RAM in an interleaved fashion as it stores octets to, or fetches octets from, the RAM-based packet buffer.

During packet reception, the MCU is allowed to access the RX packet buffer of the consolidated RAM, in order to download the packet while it's being received, if so desired. In cases where both the MCU and the GENERIC_FSK Link Layer controller attempt to simultaneously access the RAM during reception, internal hardware arbitration is provided to delay the MCU access until the GENERIC_FSK Link Layer completes its access to the RAM. Link Layer accesses complete in a single reference oscillator clock period, so one and only 1 wait state shall be inserted on the IPS bus whenever access attempts coincide.

During packet transmission, MCU access to the consolidated RAM is not allowed. The GENERIC_FSK software must program the TX buffer before commanding a TX operation, and must not access the RAM during the transmission.

As mentioned previously, incoming receive data lands in the RX section of the Packet Buffer. Nominally, a newly received packet will overwrite the contents of the previously received packet (the TX section of the Packet Buffer remains intact). Software can inhibit receive-packet overwriting of the RX section of the Packet Buffer contents by setting the PB_PROTECT bit in the PACKET_RAM_CTRL register. When this bit is set, incoming receive data will be blocked from overwriting the existing contents of the RX section of the Packet Buffer. This will block RX packet overwriting, but will not inhibit TX content loading of the TX section of the Packet Buffer.

GENERIC_FSK software must not attempt to access the Packet RAM unless GENERIC_FSK Link Layer is the controlling Link Layer.

When using another Link Layer, if the GENERIC_FSK Link Layer software attempts to access the Packet RAM via the GENERIC_FSK IPS bus, the access attempt will fail, as described below:

1. RAM contents shall not be altered on a GENERIC_FSK write attempt
2. Readback data on a GENERIC_FSK read attempt shall be indeterminate
3. The **ips_xfr_err** shall be asserted on the GENERIC_FSK IPS bus

55.4.9.2.3.5 Send and Receive multiple packets

The feature of "Send and Receive multiple packets" is enabled by setting the register MULT_PKT_EN.

To support this, the Tx and Rx RAM are divided into segments with same size. The segment size is defined by register SEG_SZ.

The first packet received is stored in the Segment 0. The second packet is stored in Segment 1, and so on. Register PKT_INDEX is initially 0 and is automatically increased by 1 after successfully received a packet. So PKT_INDEX has two meanings: the number of packet received; the segment number to store the next coming packet.

Tx is like the Rx process. The first packet to sent is placed in Segment 0. After transmitted the first packet, the PKT_INDEX is automatically increased to 1, and so on. So PKT_INDEX has two meanings in Tx: the number of packet sent; the segment number to send the next packet.

User should monitor the PKT_INDEX to avoid it moves to outside of Tx/Rx RAM.

User should write RESET_PKT_IDX to clear the PKT_INDEX to 0 before the round of packet reception.

Figure 426 shows an example of 8 packets. After received or sent 8 packets, the PKT_INDEX is increased to 8. If user wants to start another around, the PKT_INDEX should be cleared to 0.

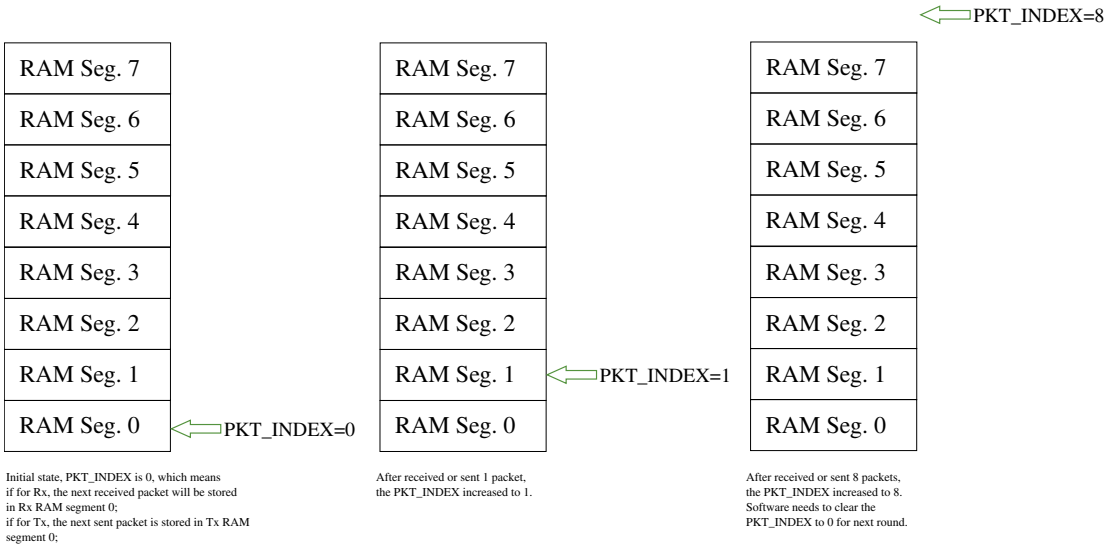


Figure 426. Send or Receive 8 packets

Notes:

1. If a packet is recieved with error, like CRC fail, and this fail causes the link layer to recycle, the PKT_INDEX will not be increased. This means link layer ignores the current received packet
2. For Rx with AutoACK case, the PKT_INDEX is increased after Rx is done. The AutoACK packet is not effected by PKT_INDEX. It is always in the dedicated RAM.
3. For TR case, the PKT_INDEX is increased after the TR sequence end. The T and R sequence use the same PKT_INDEX to detemine the segment to send or receive a packet.

55.4.9.2.3.6 Timebase

The GENERIC_FSK Link Layer includes a dedicated, 32-bit multi-purpose Event Timer (EVENT_TMR), that is updated at a 1 MHz rate. The EVENT_TMR maintains the timebase for all GENERIC_FSK-related activities, except during intervals of Deep Sleep Mode. The clock source for the EVENT_TMR is a high-precision, crystal-referenced RF Oscillator, which can be either 32MHz or 26MHz. The RF Oscillator is divided down to 1MHz for GENERIC_FSK in the Clocks Generation Module (CGM). To reduce power consumption to a minimum, most of the GENERIC_FSK Link Layer Controller runs off this 1MHz clock. The exceptions are the Register IP bus interface, Command Decoder, and Interrupt Control Unit, parts of which must run at the IP bus frequency.

The 32-bit EVENT_TMR running at 1 MHz rolls over (wraps around) approximately every 4295 seconds.

The EVENT_TMR count can be read at any time, via the register of the same name. A 32-bit read is recommended to ensure coherency of the 4 bytes that make up the counter. No synchronization to the IP bus is required.

If the EVENT_TMR needs to be updated (changed), this can be done by writing the new, desired EVENT_TMR value to the register EVENT_TMR_LD. The update to EVENT_TMR will take effect at the next 1MHz clock edge.

If it is desired to increment the EVENT_TMR by a known amount, this can be done by writing the desired increment value to the EVENT_TMR_ADD register. The increment value is 32-bit signed, thus enabling a decrement function as well. The update to EVENT_TMR will take effect immediately. The EVENT_TMR_ADD avoids the necessity to perform an read-modify-write operation to the EVENT_TMR to adjust its value; this facilitates accurate restoration of the EVENT_TMR state after a period of Deep Sleep Mode.

Two 32-bit timer-compare registers are provided, T1_CMP and T2_CMP.

GENERIC_FSK software may load T1_CMP with a future EVENT_TMR count value. When EVENT_TMR reaches the T1_CMP value, either (or both) of the following events can be triggered:

1. T1_IRQ (Interrupt)
2. Sequence Command Trigger (can automatically launch or stop a transceiver operation)

Similarly, GENERIC_FSK software may load T2_CMP with a future EVENT_TMR count value. When EVENT_TMR reaches the T2_CMP value, either (or both) of the following events can be triggered:

1. T2_IRQ(Interrupt)
2. Sequence Command Trigger (can automatically launch or stop a transceiver operation)

For either type of T1-triggered events to be enabled, i.e., interrupt or sequence-launch, set T1_CMP_EN=1 after loading T1_CMP with the desired value. If T1_CMP_EN=1, the T1_IRQ interrupt status bit will always become set on a T1 timer match. (Whether or not T1_IRQ generates an GENERIC_FSK Interrupt to the MCU depends on the settings of T1_IRQ_EN and GENERIC_FSK_IRQ_EN, see section: Interrupts). If T1_CMP_EN=0, neither an interrupt nor a sequence command can be triggered by a T1 match.

For either T2-triggered events to be enabled, i.e., interrupt or sequence launch, set T2_CMP_EN=1 after loading T2_CMP with the desired value. If T2_CMP_EN=1, the T2_IRQ interrupt status bit will always become set on a T2 timer match. (Whether or not T2_IRQ generates a GENERIC_FSK Interrupt to the MCU depends on the settings of T2_IRQ_EN and GENERIC_FSK_IRQ_EN, see Section: Interrupts). If T2_CMP_EN=0, neither an interrupt nor a sequence command can be triggered by a T2 match.

See Section Interrupts for a description of how the Link Layer Controller manages the T1 and T2 interrupts, and how software can clear them.

See Section Sequence Manager for a description on how to launch and stop sequences on T1 and T2 timer matches.

55.4.9.2.3.7 Sequence Manager

55.4.9.2.3.7.1 Introduction

This Section describes the Generic Link Layer Sequence Manager (GSM).

55.4.9.2.3.7.1.1 Overview

The GSM is a hardware state machine which controls the timing of all transmit, receive, and CCA operations for the Link Layer.

Designed into the GSM are a variety of complex sequences, called autosequences, which are sequential concatenations of simpler, building-block sequences. The hardware autosequences provide the following advantages:

- smaller SW memory footprint: HW automates operations previously done by SW
- increases MCU sleep times, reducing current consumption
- handles timing-critical sequences (rx-to-tx), e.g., data-polling, not possible in SW

Sequences can be initiated by software directly, or alternatively, can be initiated automatically at the expiration of a hardware timer. The general parameters of the sequences can be programmed by software prior to initiating the sequences. Such parameters allow software to select, for example:

- whether CCA is required ahead of a transmit sequence
- whether an automatic Ack frame should be transmitted after a receive sequence
- slotted vs. unslotted mode
- whether the device is a PAN Coordinator or not
- whether the device resides on 2 networks simultaneously (Dual PAN mode)

55.4.9.2.3.7.1.2 Features

- Supports many working modes, defined by register GENLL_MODE
- Supports complex sequences
- Unburdens software by automating many transceiver operations
- Data-polling accelerator hardware enables critical-timing sequences
- Includes capability to conditionally perform CCA prior to every transmit operation
- Includes capability to conditionally transmit an Ack frame after every receive operation
- Supports software- or timer-initiated sequences
- Broad support of 2FSK/GFSK PHYs in IEEE 802.15.4-2015
- Supports slotted and unslotted modes
- Supports orderly transceiver shutdown on PLL unlock event
- Increases software visibility by making SEQ_STATE state readable
- Supports Dual PAN Mode, including on-the-fly channel changing

55.4.9.2.3.7.1.3 Modes and operations

The GSM controls, coordinates, and automates all transceiver operations within the 2.4GHz radio. Software selects the desired sequence, sets the general parameters, and (optionally) configures a hardware timer to trigger the sequence at a precise time in the future. Subsequently, the MCU can enter a low-power state, or tend to other activities, while the sequence manager state machine conducts the programmed transceiver operations. When the sequence completes, an interrupt alerts the MCU. At this point the MCU can check the completion status of the sequence, download received data from the packet buffer, and then prepare the next sequence. Sequences can be simple transceiver operations. For example, transmit one frame, or, perform a CCA on a channel. Alternatively, the sequence manager supports complex sequences, which use simple operations as building blocks, strung together in a back-to-back fashion, with precise timing intervals between operations. For example, a "TR" sequence calls for a transmit frame followed by a receive frame. For complex sequences, interrupts can optionally be generated at the completion of each stage of the sequence, providing greater visibility for the MCU into the timeline of the sequence. Software can determine the precise state of the sequence manager state machine at any time. Software can opt to abort a sequence at any time; when this happens, the sequence manager will return to its IDLE state in an orderly fashion.

55.4.9.2.3.7.2 Functional Description

The GSM provides a command set of transceiver commands, which can be used to launch and stop sequences. Software can issue sequence commands by writing to the SEQCMD(XCVSEQ) field of the XCVR_CTRL register. The SEQCMD field always reads back what was previously written by software, enabling software to quickly recall the last command written. The sequence commands are as listed in the table below:

SEQCMD[3:0]	COMMAND	DESCRIPTION
0x0	IDLE	Abort All - Cancels all pending events and abort any sequence-in-progress
0x1	TX_START_NOW	TX Start Now
0x2	TX_START_T1	TX Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP)
0x3	TX_START_T2	TX Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP)
0x4	TX_CANCEL	TX Cancel -- Cancels pending TX events but do not abort TX-in-progress[1]
0x5	RX_START_NOW	RX Start Now
0x6	RX_START_T1	RX Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP)
0x7	RX_START_T2	RX Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP)
0x8	RX_STOP_T1	RX Stop @ T1 Timer Compare Match (EVENT_TMR = T1_CMP)
0x9	RX_STOP_T2	RX Stop @ T2 Timer Compare Match (EVENT_TMR = T2_CMP)
0xA	RX_CANCEL	RX Cancel -- Cancels pending RX events but do not abort RX-in-progress[2]
0xB	ABORT	Abort All - Cancels all pending events and abort any sequence-in-progress
0xC	TR_START_NOW	TR Start Now
0xD	TR_START_T1	TR Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP)
0xE	TR_START_T2	TR Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP)
0xF	TR_CANCEL	TR Cancel -- Cancels pending TR events but do not abort TR-in-progress[3]
0x10	CCA_NOW	CCA Start Now
0x11	CCA_START_T1	CCA Start @ T1 Timer Compare Match (EVENT_TMR = T1_CMP)
0x12	CCA_START_T2	CCA Start @ T2 Timer Compare Match (EVENT_TMR = T2_CMP)
0x13	CCA_CANCEL	CCA Cancel -- Cancels pending CCA events but do not abort CCA-in-progress[4]

1. TX-in-progress implies TX Warmup is complete, and packet transmission is underway (i.e., command decoder state = TX_PKT)
2. RX-in-progress implies RX Warmup is complete, and Network Address search or packet reception is underway (i.e., command decoder state = RX_PKT)
3. TR-in-progress implies TX Warmup is complete, and packet transmission is underway (i.e., command decoder state = TX_PKT). For CCA+TX case, if cancel command is issued at CCA phase, then need to wait CCA complete, the following TX is cancelled.
4. CCA-in-progress implies RX Warmup is complete(i.e., command decoder state = RX_CCA1/RX_CCA2/CCA2_WAIT))

As described above, sequences can be launched immediately (actually on the next 1MHz clock edge) by issuing one of the following commands:

- TX_START_NOW
- RX_START_NOW
- TR_START_NOW

- CCA_START_NOW

Alternatively, sequences can be scheduled to launch (or stop) in the future on a T1 timer compare, by loading T1_CMP with the desired launch time (relative to the EVENT_TMR), setting T1_CMP_EN=1, and then issuing one of the following commands:

- TX_START_T1
- RX_START_T1
- RX_STOP_T1
- TR_START_T1
- CCA_START_T1

Similarly, sequences can be scheduled to launch (or stop) in the future on a T2 timer compare, by loading T2_CMP with the desired launch time (relative to the EVENT_TMR), setting T2_CMP_EN=1, and then issuing one of the following commands:

- TX_START_T2
- RX_START_T2
- RX_STOP_T2
- TR_START_T2
- CCA_START_T2

Any transceiver sequence that has been launched, either immediately (by way of the *_NOW commands), or in the future (by way of the *_T1 or *_T2 commands), can be terminated immediately by issuing the ABORT/IDLE command. This is true whether or not the sequence has actually begun, or is merely pending because the T1- (or T2-) match has not yet occurred. In either case, this is called a software abort. The ABORT/IDLE command will terminate any active sequence and cancel any pending events.

Regarding the issuance of sequence commands, both immediate and timer-triggered, software should be mindful of the following Guidelines and Restrictions:

PROGRAMING GUIDELINES AND RESTRICTIONS	
1	T1 can only be assigned to 1 function at a time {TX_START, RX_START, RX_STOP, TR_START, CCA_START}. Once assigned to a function, attempts to change the assignment will be ignored
2	T2 can only be assigned to 1 function at a time {TX_START, RX_START, RX_STOP, TR_START, CCA_START}. Once assigned to a function, attempts to change the assignment will be ignored
3	To change a T1- or T2- assignment, do one of the following first: **a.** issue an ABORT or IDLE command **b.** issue an TX_CANCEL command if T1 (or T2) is already assigned to TX_START, and the sequence has not yet started **c.** issue an RX_CANCEL command if T1 (or T2) is already assigned to RX_START or RX_STOP, and the sequence has not yet started **d.** issue an TR_CANCEL command if T1 (or T2) is already assigned to TR_START, and the sequence has not yet started **e.** issue an CCA_CANCEL command if T1 (or T2) is already assigned to CCA_START, and the sequence has not yet started

Table continues on the next page...

Table continued from the previous page...

PROGRAMING GUIDELINES AND RESTRICTIONS	
4	T1 and T2 should not be assigned to the same function (results are indeterminate)
5	If T1 and T2 are both assigned to functions, then T1_CMP shall not be equal to T2_CMP (results are indeterminate)
6	If any TX or RX or TR or CCA sequence is underway (e.g., TSM not idle), the ongoing sequence will not be interrupted or perturbed by any of the following: **a.** a T1 timer-match with T1 assigned to a function. The sequence trigger will be ignored, but the associated "pending" status bit will be cleared. **b.** a T2 timer-match with T2 assigned to a function. The sequence trigger will be ignored, but the associated "pending" status bit will be cleared. **c.** a TX_START_NOW or RX_START_NOW or TR_START_NOW or CCA_START_NOW command. Both commands will be ignored

The GSM controls high-level autosequence timing, but offloads low level warmup and warmdown tasks to a dedicated, protocol-neutral Transceiver Sequence Manager, or TSM (see TSM Chapter). The GSM initiates high-level functions, such as a CCA measurement operation or a transmit acknowledge operation, by commanding the TSM to execute an RX warmup, a TX warmup, a RX warmdown, a TX warmdown, or an RX recycle. Based on these commands, TSM generates the precise timing to control all of the digital, analog, and RF components of the TX and RX chains. The TSM is fully programmable, allowing all of the enabling and control signals to the transceiver components to be independently scheduled with 1us precision. TSM also allows all enable and control signals to be optionally overridden at any time, giving software complete control of each transceiver block, and ultimately the entire warmup/warmdown process if desired.

The GSM and TSM sequence managers operate on the same clock domain, and both have 1us resolution. The TSM is fully programmable, while the GSM is largely hardwired, with a few key exceptions. In the GSM/TSM relationship, GSM is the master and TSM is the slave. The key GSM/TSM interface signaling is shown below:

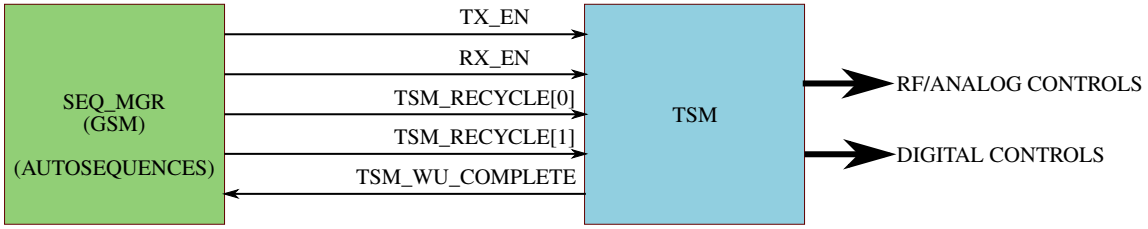


Figure 427. GSM/TSM INTERFACE SIGNALS

A description of the interface signals is provided in the table below.

GSM/TSM INTERFACE SIGNAL	DESCRIPTION
TX_EN	A low-to-high transition initiates a TSM TX warmup, starting 1us after the transition. A high-to-low transition initiates a TSM TX warmdown, starting 1us after the transition.
RX_EN	A low-to-high transition initiates a TSM RX warmup, starting 1us after the transition. A high-to-low transition initiates a TSM RX warmdown, starting 1us after the transition.

Table continues on the next page...

Table continued from the previous page...

GSM/TSM INTERFACE SIGNAL	DESCRIPTION
TSM_RECYCLE[0]	This signal asserts in the GSM RX_CYC state, and causes the TSM to jump from its "ON" phase, back to a point in the TSM RX warmup determined by the TSM's RECYCLE_COUNT0[7:0] register in XCVR space. This process is used to execute an RX recycle.
TSM_RECYCLE[1]	This signal asserts in the GSM RX_PAN1 state, and causes the TSM to jump from its "ON" phase, back to a point in the TSM RX warmup determined by the TSM's RECYCLE_COUNT1[7:0] register in XCVR space. This process is used to re-initialize the PLL digital on a new RF channel, during a Dual PAN mode on-the-fly channel change.
TSM_WU_COMPLETE	For the GSM states which assert TX_EN or RX_EN, this 1us-wide signal from the TSM indicates that the warmup is complete, and the GSM is now free to advance to its next state. The TSM_WU_COMPLETE is asserted 1us before the actual end of sequence, which is determined by the END_OF_TX_WU[7:0] register for TX warmup, and the END_OF_RX_WU[7:0] register for RX warmup.

The Command Decoder also maintains an set of status bits, that make it easy for Link Layer software to track, not only state, but pending events; that is, timer-linked events scheduled for future triggering. These status bits reside in the SEQ_STS register. A description of these bits is provided in the table below.

SEQSTS[12:0]	BIT	DESCRIPTION
TX_START_T1_PEND	[0]	TX Sequence will start @ next T1 Match
TX_START_T2_PEND	[1]	TX Sequence will start @ next T2 Match
TX_IN_WARMUP	[2]	TX Sequence in TSM Warmup
TX_IN_PROGRESS	[3]	TX Packet Transmission Currently Underway
TX_IN_WARMDN	[4]	TX Sequence in TSM Warmdown
RX_START_T1_PEND	[5]	RX Sequence will start @ next T1 Match
RX_START_T2_PEND	[6]	RX Sequence will start @ next T2 Match
RX_STOP_T1_PEND	[7]	RX Sequence will stop @ next T1 Match
RX_STOP_T2_PEND	[8]	RX Sequence will stop @ next T2 Match
RX_IN_WARMUP	[9]	RX Sequence in TSM Warmup
RX_IN_SEARCH	[10]	RX Sequence in Network Address Search
RX_IN_PROGRESS	[11]	RX Packet Reception Currently Underway
RX_IN_WARMDN	[12]	RX Sequence in TSM Warmdown
TR_START_T1_PEND	[13]	TR Sequence will start @ next T1 Match
TR_START_T2_PEND	[14]	TR Sequence will start @ next T2 Match
CCA_START_T1_PEND	[15]	CCA Sequence will start @ next T1 Match
CCA_START_T2_PEND	[16]	CCA Sequence will start @ next T2 Match

The GSM state diagram is shown below.

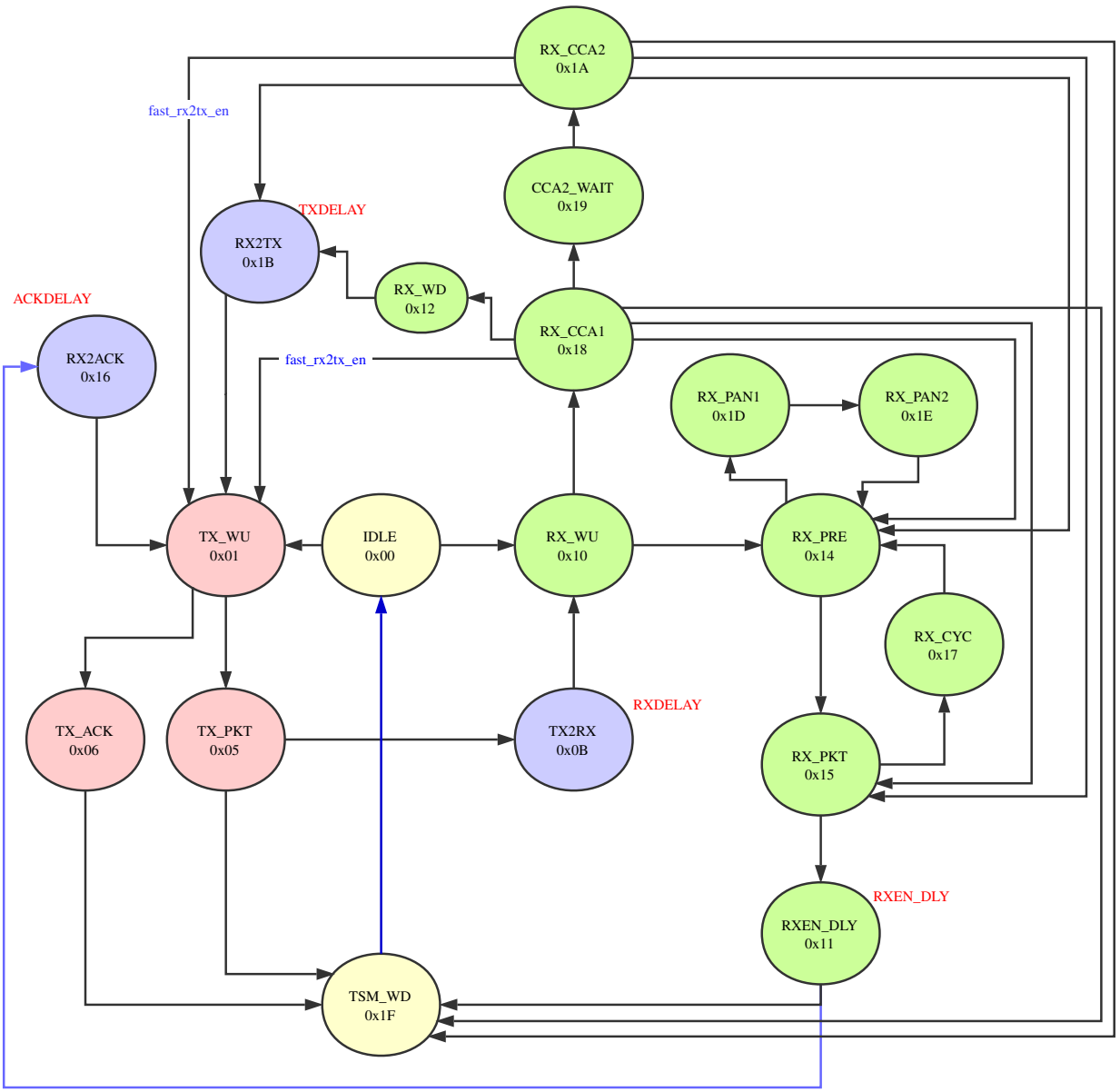


Figure 428. GSM State Diagram

Each GSM state is represented in the hardware by a 5-bit state vector. The GSM state can be monitored at any time by way of the read-only SEQ_STATE[4:0] register. The mapping of GSM state names to state vectors is shown in the table below:

GSM STATE	VECTOR
SEQ_IDLE	0x00
RX_WU	0x10

Table continues on the next page...

Table continued from the previous page...

GSM STATE	VECTOR
RX_PRE	0x14
RX_PKT	0x15
RX2ACK	0x16
RX_CYC	0x17
RX_PAN1	0x1D
RX_PAN2	0x1E
RX_CCA1	0x18
CCA2_WAIT	0x19
RX_CCA2	0x1A
RX_WD	0x12
RX2TX	0x1B
TX_WU	0x01
TX_PKT	0x05
TX_ACK	0x06
TX2RX	0x0B
TSM_WD	0x1F

55.4.9.2.3.7.2.1 Supported Sequences

55.4.9.2.3.7.2.1.1 Sequence I (Idle)

When the IDLE/ABORT value is written to XCVSEQ, the sequence manager goes to its SEQ_IDLE state. If not already in SEQ_IDLE, the sequence manager executes an orderly warmdown to return to SEQ_IDLE. A SEQIRQ interrupt is then issued. Writing IDLE/ABORT value to XCVSEQ is the proper way to abort a sequence.

When a sequence is aborted in this way (called a software abort), the SW_ABORTED bit will become set. This bit will be self-cleared at the start of the next autosequence.

55.4.9.2.3.7.2.1.2 Sequence R (Receive)

Sequence R can be lunched by command RX_START_*. Sequence R is the basic receive sequence.

In GLL mode, sequence R can be used to receive all frames.

In PAN mode, sequence R can be used to receive all types of PHY- and MAC-compliant frames, including reserved frame types. However, reception of Acknowledge frames is not recommended using Sequence R. This is because reception of an Acknowledge frame, usually follows a transmitted frame, with a designated Sequence Number, so that the received Acknowledge frame can be verified for the matching Sequence Number. In a standalone Sequence R, there is no Sequence Number to verify against, so any Acknowledge frame is merely transferred to the Packet RAM. (Note: the appropriate way to receive Acknowledge frames is with Sequence TR, with the RXACKRQD bit asserted). Using Sequence R, *all* frames which pass frame-filtering rules and CRC check, are transferred to the Packet RAM.

In FAN mode, sequence R can be used to receive all types of PHY- and FAN-compliant frames. Same as in PAN mode, it is recommended to use TR to receive Acknowledge frames.

The basic execution of a successful Sequence R in GLL/PAN/FAN mode is as follows:

- SEQ_MGR executes RxWarmup
- SFD Detected and received, transferred to Packet RAM
- Payload and CRC bytes received, transferred to Packet RAM.
- If no H0/H1/LENGTH violation and CRC check pass:
- And if frame-filtering rules(PAN rule) and PHR check pass:(PAN mode)
- And if frame-filtering rules(FAN rule) and PHR check pass:(FAN mode)
- ---> RXIRQ interrupt issued
- ---> SEQ_MGR executes RxWarmdown
- ---> If AUTOACK=1
- ---> And if received Frame Control Field indicates AckRequest=1 (PAN/FAN mode)
- ---- ---> SEQ_MGR executes TxWarmup
- ---- ---> an Ack frame is transmitted (programmed in Packet RAM) (GLL mode)
- ---- ---> an imm-Ack frame is transmitted using the received Sequence Number (PAN mode)
- ---- ---> an Enh-Ack frame is transmitted, with IEs and ASH field programmed in Packet RAM (FAN mode)
- ---- ---> SEQ_MGR executes TxWarmdown
- SEQIRQ interrupt issued

When Sequence R is initiated, the GSM sequence manager warms up the receiver (analog and digital elements), via the TSM (Transceiver Sequence Manager). Once a SFD is detected, the sequence manager's slot timer is activated. This is because a slotted "transmit acknowledge" frame (TxAck) may be required later in this sequence. The slot timer is loaded with the following value (in us):

Slot Timer Preload = SLOT_TIME - (PREAMBLE + SFD) + ACKDELAY

The slot timer will count down and eventually rollover at to 0, then to (SLOT_TIME-1). (There are SLOT_TIME us in a backoff slot).

The receiver receives the SFD octets . Starting with this octet, and continuing through the remainder of the frame, each octet that is received is transferred to Packet RAM. After each octet is transferred to Packet RAM, the Packet RAM address is incremented by 1.

During the reception, link layer will check the healthy of frame:

- CRC (all modes)
- H0/LENGTH_MAX/H1 violation (all modes)
- BCH/Parity fail (PAN/FAN mode)
- PAN packet-filtering rules (PAN mode)
- FAN packet-filtering rules (FAN mode)

Assuming no fail, an interrupt (RXIRQ) is issued to the MCU.

At this point, the sequence manager must determine if a TxAck is required. A TxAck is required if the following conditions are all met:

- AUTOACK=1 (register bit) (all modes)
- PROMISCUOUS=0 (register bit) (FAN and PAN mode)
- The received FrameControlField has AckRequest=1 (FAN and PAN mode)

If these conditions are met, and SLOTTED mode is not in effect, the sequence manager loads its TxAck timer with the following value (in us):

TxAck Timer Preload = TURNAROUND_TIME + ACKDELAY - TXWARMUPTIME

The TURNAROUND_TIME is the non-slotted RX-to-TX turnaround requirement in the 802.15.4 standard. The TXWARMUPTIME takes into account the fact that the TxAck timer must expire early for the Tx warmup to be complete at exactly the TURNAROUND_TIME us point. The ACKDELAY is a signed, fine-tune adjustment to the TxAck transmission time (+/-).

If SLOTTED mode is in effect, the slot timer was previously loaded (at SFD detect), but the sequence manager needs to determine how much time is left in the current backoff slot. If less than TURNAROUND_TIME remain, the sequence manager must wait an *additional* backoff slot before initiating the Ack transmission. So it sets a flag indicating it must allow the slot counter to rollover an additional iteration.

Finally, at TxAck timer expiration (in non-SLOTTED mode), or at the qualified rollover of the slot timer (in SLOTTED mode), a Tx warmup is initiated. The conclusion of the Tx warmup will coincide precisely with a backoff slot boundary if SLOTTED mode is in effect. The sequence manager will then initiate the transmission of the Ack frame.

In GLL mode, entire Ack frame should be programmed by software in Packet RAM.

In FAN mode, software should program ASH and IEs field in the Packet RAM and point out their total length in register LENGTH_ACK. The other part of the Enh-Ack frame are built up by hardware.

In PAN mode, entire Ack frame will be built up by hardware. The hardware will insert the following parameters into the Ack frame's Frame Control Field:

Table 479. Frame Control Field for Hardware-generated Auto-TxAck Frame (PAN Mode)

FCF Field	Value
FrameType	Acknowledge
SecurityEnabled	0
FramePending	Determined by the Packet Processor. (see Packet Processor Chapter, Source Address Management)
AckRequest	0
PanIDCompression	0
Reserved (bits 7-9)	000
DstAddrMode	0
FrameVersion	0
SrcAddrMode	0

The Sequence Number for the Ack packet, will be the Sequence Number obtained from the previously received frame.

Table 480shows the register bits used to control the Auto-Ack sequence in different modes.

Table 480. Auto-Ack Behavior and Register Setting(AUTOACK=1, SLOTTED=0/1)

Mode	GENLL_MODE[3:0]	SW_BUILD_ACK	ACKBUFSEL	LENGTH_ACK[10:0]	ACK Condition
GLL-ACK	0x0	0	ACK frame's SFD/H0/H1 field is from received frame. LENGTH field is from LENGTH_ACK.	Payload length + CRC_SZ - LENGTH_ADJ	1. Pass CRC 2. Pass H0/H1/LENGTH_MAX check (No SAM)

Table continues on the next page...

Table 480. Auto-Ack Behavior and Register Setting(AUTOACK=1, SLOTTED=0/1) (continued)

Mode	GENLL_ MODE[3:0]	SW_BUILD_ ACK	ACKBUFSEL	LENGTH_ACK[10:0]	ACK Condition
			0: Payload is in 64-byte dedicated RAM. 1: Payload is in TX RAM.		
		1	ACK frame's SFD/H0/H1/LENGTH/ Payload field is from RAM 0: ACK frame is in 64-byte dedicated RAM. 1: ACK frame is in TX RAM.	Not used	
Imm-ACK	0x1 or 0x3	0	ACK frame's SFD is from received. HW build the PHR according to PHR type HW build the payload HW does not access RAM	Not used	1. Pass CRC 2. Pass H0/H1/LENGTH_MAX check 3. Pass PHR check 4. Is MAC-compliant 5. Is addressed to the device 6. Is DATA or CMD frame (Do SAM when is a data polling CMD frame)
		1	ACK frame's SFD/H0/H1/LENGTH/ Payload field is from RAM 0: ACK frame is in 64-byte dedicated RAM. 1: ACK frame is in TX RAM.	Not used	
Enh-ACK	0x2 or 0x3	0	ACK frame's SFD is from received HW build the PHR according to PHR type HW build part of the Payload 0: ASH and IEs is in 64-byte dedicated RAM. 1: ASH and IEs is in TX RAM.	ASH/IEs length	1. Pass CRC 2. Pass H0/H1/LENGTH_MAX check 3. Pass PHR check 4. Is FAN-compliant 5. Is addressed to the device (No SAM since FAN use DATA frame only.)
		1	ACK frame's SFD/H0/H1/LENGTH/ Payload field is from RAM 0: ACK frame is in 64-byte dedicated RAM. 1: ACK frame is in TX RAM.	Not used	

After the Ack frame has been transmitted, the sequence manager will issue a TXIRQ interrupt, and then perform a Tx warmdown. Finally, a SEQIRQ interrupt will be issued, and the sequence manager returns to SEQ_IDLE state.

Sequence R Timing Diagrams

The following timing diagram depicts an example of "good" packet reception, highlighting the following events:

1. AUTOACK=0, so no auto-TxAck
2. Software loads T2_CMP with desired RX start time, sets T2_CMP_EN=1
3. Software schedules RX sequence with RX_START_T2 command
4. EVENT_TMR matches T2_CMP, RX sequence starts automatically
5. Network Address search, followed by NTW_ADR_IRQ interrupt, TIMESTAMP capture

6. Packet reception, Generic_FSK LL write octets to Packet RAM
7. CRC verification passes, and no H0, H1, or LENGTH_MAX violations(GLL/PAN/FAN mode)
8. PHR verification passes(PAN/FAN mode)
9. Payload verification passes PAN rules(PAN mode)
10. Payload verification passes FAN rules(FAN mode)
11. RX_IRQ asserted, Packet RX RAM valid, RSSI valid
12. SEQ_END_IRQ asserts after TSM returns to IDLE (RX warmdown complete)

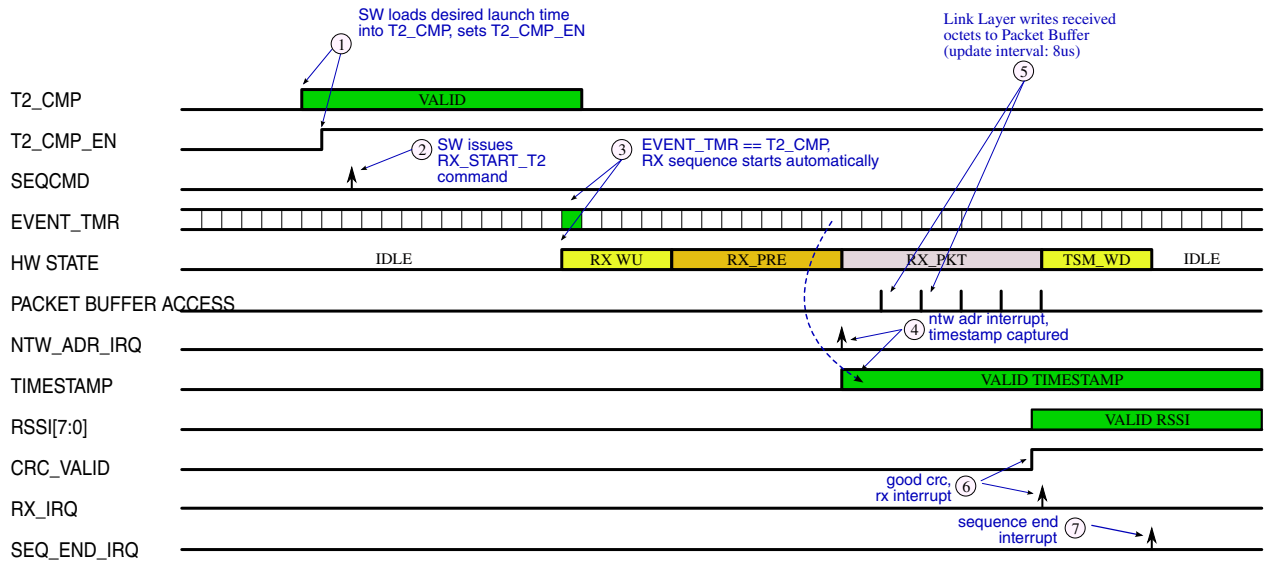


Figure 429. SEQUENCE R: LAUNCH VIA T2 TIMER, GOOD PACKET, AUTOACK=0 (All Modes)

The following timing diagram depicts an example of "good" packet reception, highlighting the following events:

1. Register RXEN_DLY is not zero and RXEN_DLY_OVERRIDE is one. After the last byte(determined by LENGTH field) of packet is received, the RX_EN signal will delay (RXEN_DLY +1) microseconds to de-assert
2. RX_IRQ asserted after the last byte(determined by LENGTH field) of packet is received.(if SEL_RXIRQ=0)
3. After extra packet received, RX_IRQ is asserted(if SEL_RXIRQ=1), GSM jumps to TSM_WD from RXEN_DLY.
4. SEQ_END_IRQ asserts after TSM returns to IDLE (RX warmdown complete)
5. **This RX mode is particularly suitable for protocols that have other fields after the CRC field. For example, the CTE(Constant Tone Extension) field in the Bluetooth LE 5.1 protocol.**

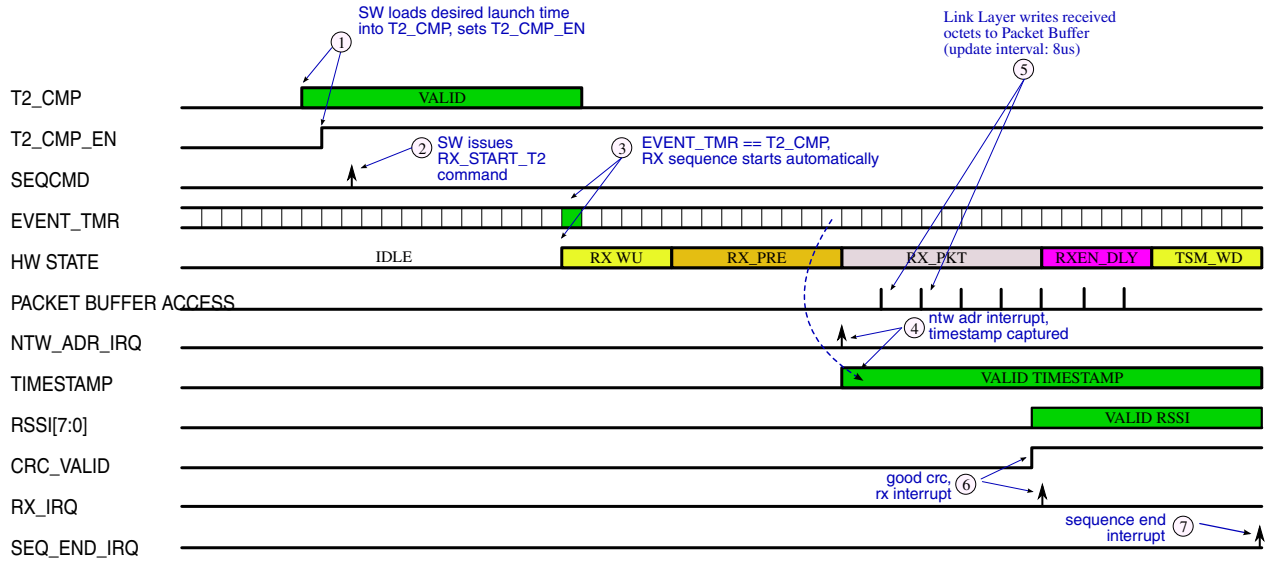


Figure 430. SEQUENCE R: RXEN DELAY (All Modes)

The following timing diagram depicts an example of "good" packet reception, highlighting the following events:

1. Register STAY_IN_RX is 1.
2. After packet received with CRC valid, GSM jumps to RX_CYC after RX_IRQ
3. This avoid the full RX re-warmup to facilitate sniffer SW.

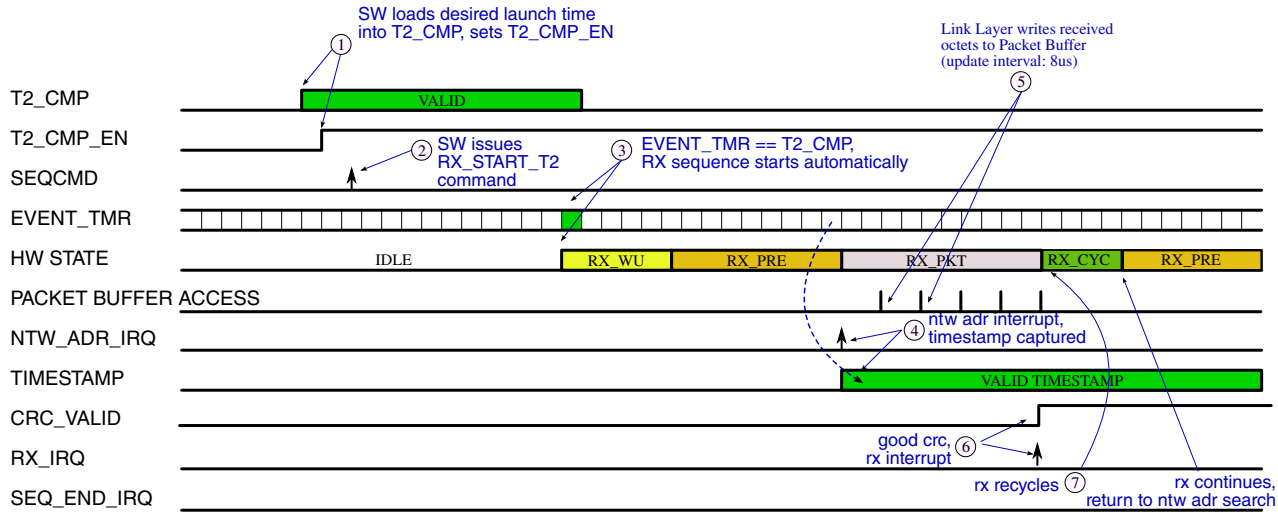


Figure 431. SEQUENCE R: STAY IN RX (All Modes)

The following timing diagram depicts an example of GENERIC_FSK "bad" packet reception, highlighting the following events:

1. Software loads T2_CMP with desired RX start time, sets T2_CMP_EN=1
2. Software schedules RX sequence with RX_START_T2 command

3. EVENT_TMR matches T2_CMP, RX sequence starts automatically
4. Network Address search, followed by NTW_ADR_IRQ interrupt, TIMESTAMP capture
5. Packet reception, Generic_FSK LL write octets to Packet RAM
6. CRC verification fails, no CRC_VALID, no RX_IRQ, RX Recycle
7. No SEQ_END_IRQ, return to Network Address Search

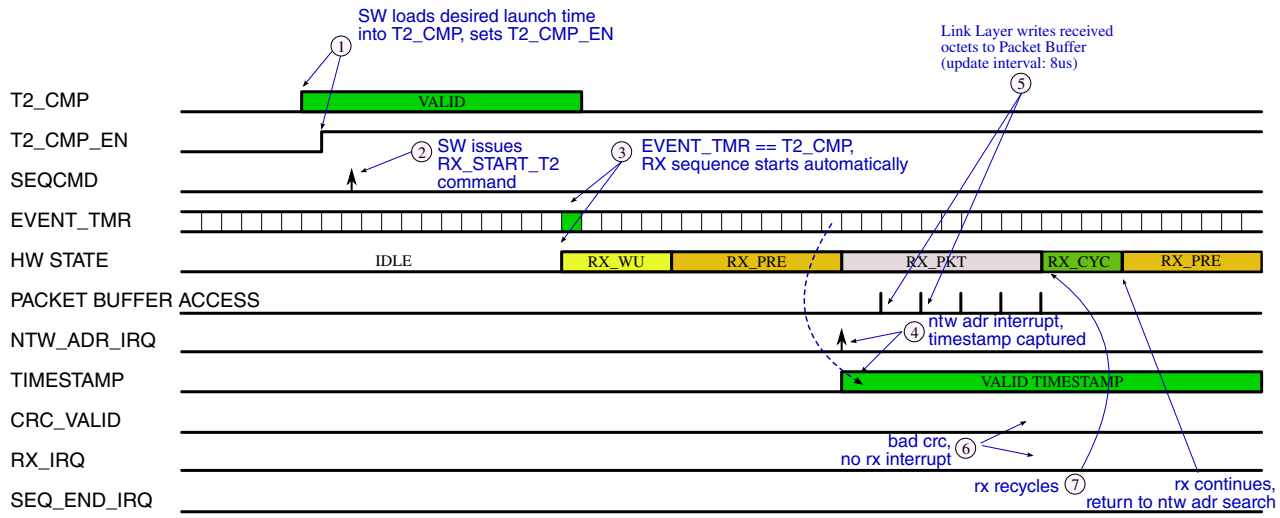


Figure 432. R SEQUENCE: LAUNCH VIA T2 TIMER, BAD PACKET, RX RECYCLE(All Modes)

The following timing diagram depicts a GENERIC_FSK RX search operation, which fails to detect a Network Address before timing out. The following events are highlighted:

1. Software loads T1_CMP with desired RX start time, sets T1_CMP_EN=1
2. Software loads T2_CMP with desired RX stop time, sets T2_CMP_EN=1
3. Software schedules RX sequence start with RX_START_T1 command
4. Software schedules RX sequence end with RX_STOP_T2 command
5. EVENT_TMR matches T1_CMP, RX sequence starts automatically
6. EVENT_TMR matches T2_CMP, RX sequence goes into warmdown automatically
7. SEQ_END_IRQ asserts to indicate TSM has returned to IDLE. No NTW_ADR_IRQ, No RX_IRQ.

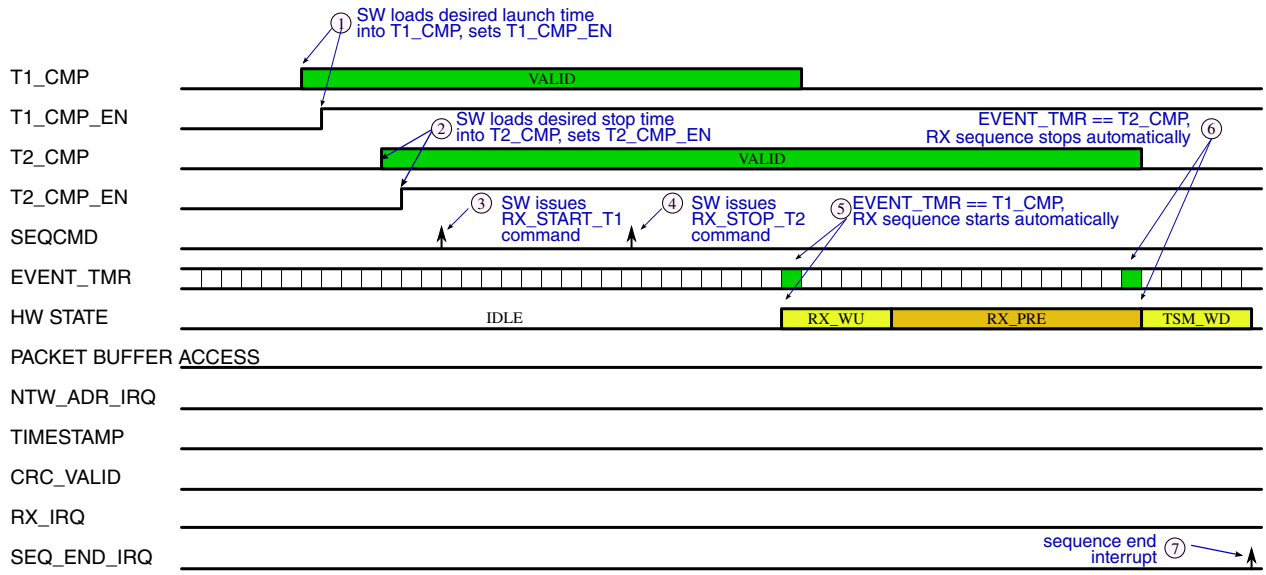


Figure 433. R SEQUENCE: SLAVE CHANNEL SEARCH, LAUNCH VIA T1, STOP VIA T2 (All Modes)

The following timing diagram depicts the GSM state transition of the Auto-Ack in PAN mode, highlighting the following events:

1. The Imm-Ack is built by hardware (SW_BUILD_ACK=0)
2. The Rx to TX time is TURNAROUND_TIME+ACKDELAY(±) (SLOTTED=0)
3. RX_IRQ is asserted when RX ends, TX_IRQ is asserted when TX ends, SEQ_END_IRQ is asserted when sequence finishes.

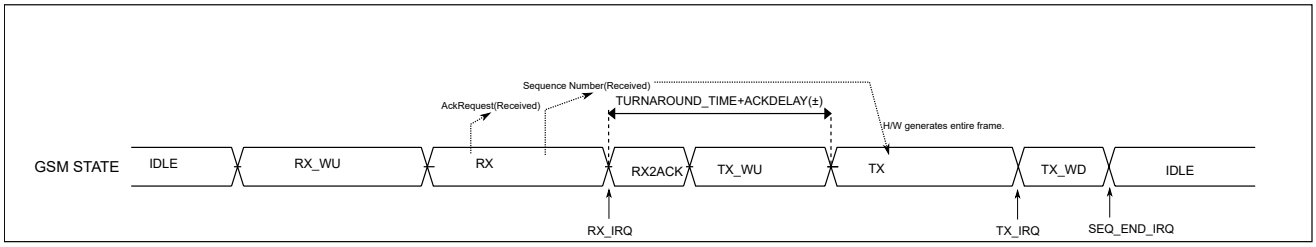


Figure 434. SEQUENCE R: Imm-Ack, SW_BUILD_ACK=0, SLOTTED=0, No SAM (PAN Mode)

The following timing diagram depicts the GSM state transition of the Auto-Ack in PAN mode, highlighting the following events:

1. The Imm-Ack is built by software in Packet RAM (SW_BUILD_ACK=1)
2. The Imm-Ack can be built by software at ACK_IRQ or any time before TX_PKT
3. ACK_IRQ are asserted when RX ends.

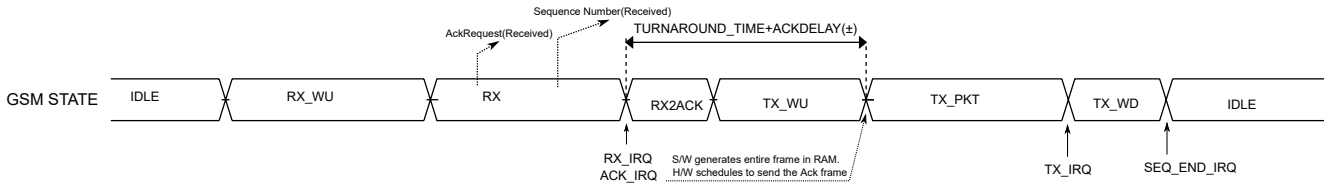


Figure 435. SEQUENCE R: Imm-Ack, SW_BUILD_ACK=1, SLOTTED=0, No SAM (PAN Mode)

The following timing diagram depicts the GSM state transition of the Auto-Ack in PAN mode, highlighting the following events:

1. The received frame is a data polling CMD frame, so Source Address Management is trigged.

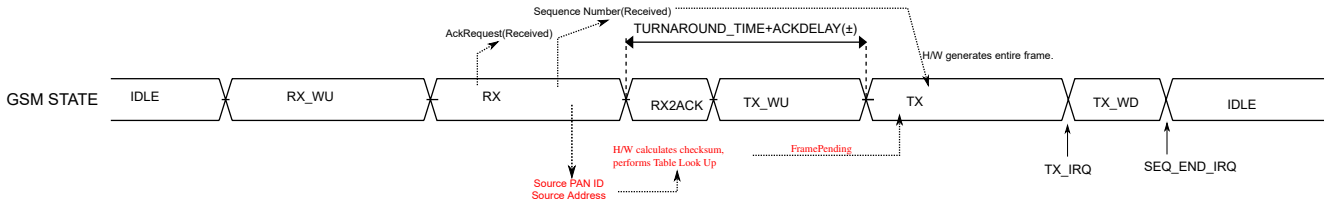


Figure 436. SEQUENCE R: Imm-Ack, SLOTTED=0, With SAM (PAN Mode)

The following timing diagram depicts the GSM state transition of the Auto-Ack in PAN mode, highlighting the following events:

1. The Slot Timer is preloaded at SFD detected with value SLOT_PRELOAD.
2. In Slot Mode, if the time between the RX END and next slot boundary is smaller than TURNAROUND_TIME, Link Layer will Tx-Ack at the next 2 slot boundary.

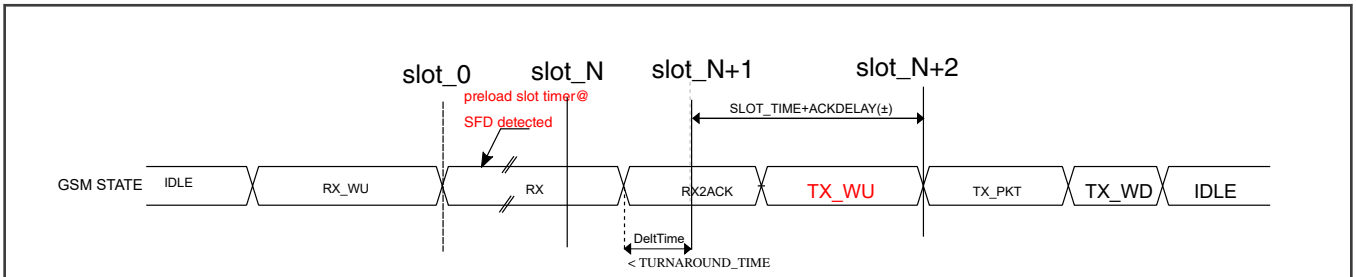


Figure 437. SEQUENCE R: Imm-Ack, SLOTTED=1, CASE 1(PAN Mode)

The following timing diagram depicts the GSM state transition of the Auto-Ack in PAN mode, highlighting the following events:

1. The Slot Timer is preloaded at SFD detected with value SLOT_PRELOAD.
2. In Slot Mode, if the time between the RX END and next slot boundary is larger than or equal to TURNAROUND_TIME, Link Layer will Tx-Ack at the next slot boundary.

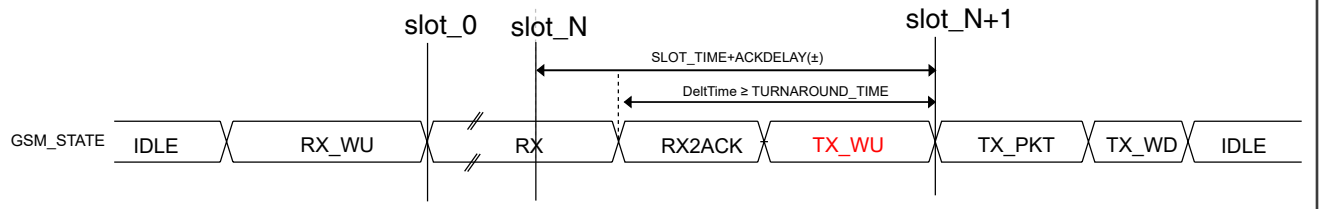


Figure 438. SEQUENCE R: Imm-Ack, SLOTTED=1, CASE 2 (PAN Mode)

55.4.9.2.3.7.2.1.3 Sequence T (Transmit)

Sequence T can be lunched by command TX_START_*. Sequence T is the basic, standalone transmit sequence.

Sequence T can be used to transmit all types of 802.15.4 PHY- and MAC-compliant frames. However, transmission of Acknowledge frames is not recommended using Sequence T. This is because transmission of an Acknowledge frame, usually follows a received frame, with a designated Sequence Number. The GSM sequence manager automates the transmission of an Ack frame which follows a received frame, using Sequence R with the AUTOACK bit asserted. This is the recommended method for transmitting an Ack frame. This is especially beneficial, because the transmitted Ack frame octets are generated by hardware, and so no setup of a transmit packet in Packet RAM, is required. However, sequence manager hardware does not prohibit Ack frames using Sequence T.

Sequence T allows for the insertion of 1 or 2 CCA (clear channel assessment) measurements prior to transmission, to ensure that the selected channel is idle. The CCA-before-TX feature is governed by two register bits, CCABFRTX and SLOTTED. All CCA measurements must indicate channel-idle, in order for the sequence manager to proceed to transmission; if the channel is determined by CCA to be busy, the sequence manager will terminate the sequence without a transmission. The number of CCA measurements attempted by the sequence manager, and the timing of the measurements, is shown in the following table.

Table 481. Timing of CCA Operations for Sequence T

CCABFRTX	SLOTTED	Number of CCA measurements	Timing of Initiation of CCA measurement #1	Timing of Initiation of CCA measurement #2	Timing of First Bit of Transmitted Frame (assumes channel idle)
0	X	0	-	-	Immediately follows TxWarmup
1	0	1	Immediately follows RxWarmup	-	Immediately follows RxWarmdown and TxWarmup
1	1	2	Immediately follows RxWarmup	SLOT_TIME us after Initiation of CCA measurement #1	SLOT_TIME us after Initiation of CCA measurement #2

CCA mode 1 can be used for Sequence T and Sequence TR.

The basic execution of a successful Sequence T is as follows:

- MCU sets CCABFRTX if one or more CCA's are required prior to transmit
- MCU sets SLOTTED if in slotted mode (2 CCA's will be attempted)
- MCU writes XCVSEQ = RX_START_*

- Wait for T1 or T2 match (if RX_START_T*)
- If CCABFRTX=1 :
- ---> SEQ_MGR executes RxWarmup
- ---> SEQ_MGR initiates CCA measurement (takes CCA_TIME us (CCA_TIME = SLOT_TIME - TURNAROUND_TIME))
- ---> if CCA indicates channel busy, sequence terminates, CCAIRQ & SEQIRQ issued
- ---> otherwise, if CCA indicates channel idle:
- ---- ---> if SLOTTED=0, SEQ_MGR executes a RxWarmdown, followed by a TxWarmup
- ---- ---> if SLOTTED=1, SEQ_MGR initiates a 2nd CCA, SLOT_TIME us after initiating 1st CCA.
- ---- ---> if SLOTTED=1, and channel busy, sequence terminates, CCAIRQ & SEQIRQ issued
- ---- ---> if SLOTTED=1, and channel idle, SEQ_MGR executes a RxWarmdown, followed by a TxWarmup
- if SLOTTED=1, SEQ_MGR waits until SLOT_TIME us elapse after 2nd CCA initiation
- SEQ_MGR executes TxWarmup
- Preamble transmitted
- SFD are obtained from Packet RAM and transmitted
- Payload are obtained from Packet RAM and transmitted
- After Payload octet transmitted, FCS is available from CRC engine.
- SEQ_MGR issues TXIRQ
- SEQ_MGR executes TxWarmdown
- SEQ_MGR issues SEQIRQ

When Sequence T is initiated, the sequence manager first must determine if one or more CCA measurements are required.

If CCABFRTX bit is asserted, the sequence manager warms up the receiver (analog and digital elements), via the TSM (Transceiver Sequence Manager). Immediately after the conclusion of the warmup, a CCA measurement is initiated. The CCA measurement takes CCA_TIME us. If CCA indicates the channel is busy, the sequence manager warms down the receiver, issues CCAIRQ and SEQIRQ interrupts, and terminates the sequence. If, however, CCA indicates the channel is idle, the sequence manager inspects the SLOTTED bit to determine what to do next.

If the SLOTTED bit is clear, the sequence manager issues a CCAIRQ interrupt, and then executes a warmdown of the receiver.

If, however, the SLOTTED bit is set, the sequence manager does not issue a CCAIRQ interrupt; instead, the sequence manager initiates a second CCA measurement, precisely SLOT_TIME us after the initiation of the first CCA. This SLOT_TIME us interval is timed by the slot timer, which had been preloaded with SLOT_TIME (microseconds) at the instant that Rx warmup was complete.

At the conclusion of the second CCA, the sequence manager issues a CCAIRQ. If the second CCA indicates the channel is busy, the sequence manager warms down the receiver, issues a SEQIRQ interrupt, and terminates the sequence. If, however, CCA indicates the channel is idle, the sequence manager executes a warmdown of the receiver. Prior to executing the warmdown, the sequence manager reloads the slot timer, in order to precisely schedule the ensuing transmission. This time, the slot timer is loaded with a smaller value, in order to account for the fact that the timer must expire early in order to trigger the Tx warmup, which must complete at precisely the next backoff slot boundary. Thus, this time the slot timer is preloaded with:

Slot Timer Preload = TURNAROUND_TIME - TXWARMUPTIME + TXDELAY.

The SLOT_TIME us is the duration of the backoff slot. The TXWARMUPTIME takes into account the fact that the TxAck timer must expire early for the Tx warmup to be complete at exactly the SLOT_TIME us point. The TXDELAY is a signed, fine-tune adjustment to the TxAck transmission time.

If the SLOTTED bit is asserted, once the slot timer expires, the sequence manager executes a Tx warmup. The conclusion of this warmup coincides with the backoff slot boundary. If the SLOTTED bit is deasserted, the sequence manager does not wait for the slot timer, and instead executes a Tx warmup immediately after the Rx warmdown.

At this point, the sequence manager begins the transmit operation. After the transmit operation is complete. The sequence manager issues a TXIRQ interrupt. The sequence manager then executes a complete Tx warmdown, and follows up with a SEQIRQ interrupt. The sequence manager returns to SEQ_IDLE state.

Sequence T Timing Diagrams

The following timing diagram depicts an example of packet transmission, highlighting the following events:

1. No CCA before TX
2. Software stores packet octets into Packet RAM, to be transmitted
3. Software loads T1_CMP with desired TX start time, sets T1_CMP_EN=1
4. Software schedules TX sequence with TX_START_T1 command
5. EVENT_TMR matches T1_CMP, TX sequence starts automatically
6. Link Layer reads octets from Packet RAM **and sends to RBME when requested**
7. Packet is transmitted at the end of TX_WU
8. TX_IRQ asserts after last bit of packet transmitted
9. SEQ_END_IRQ asserts after TSM returns to IDLE (TX warmdown complete)

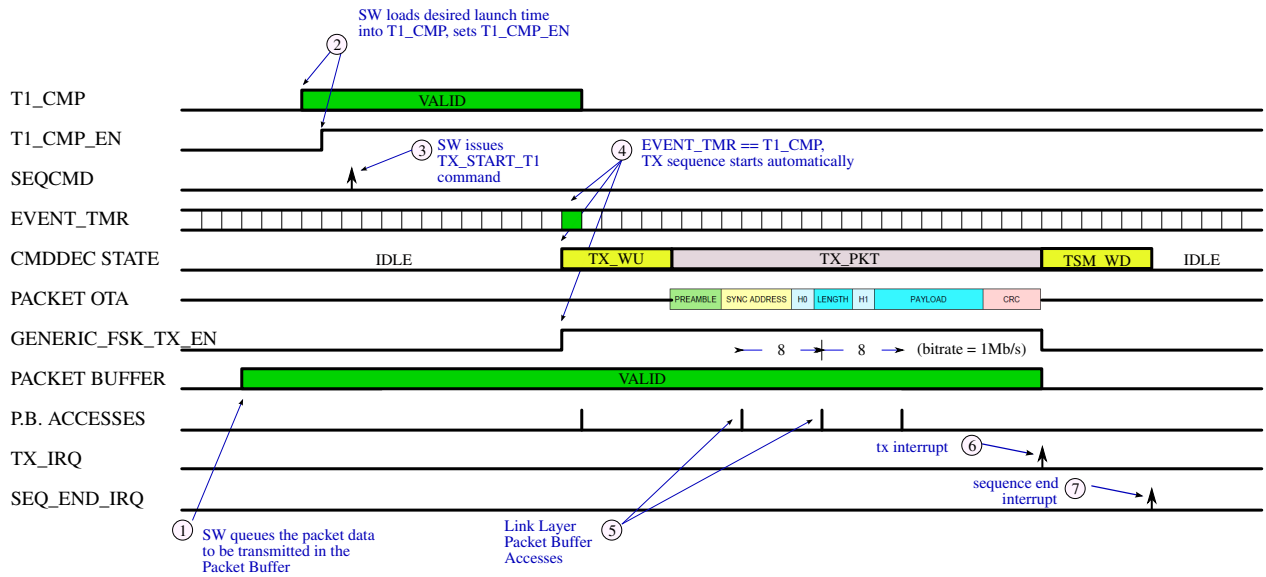


Figure 439. SEQUENCE T: LAUNCH VIA T1 TIMER

The following timing diagram depicts the GSM state transition of Tx, highlighting the following events:

1. Un-slotted mode with CCA before TX
2. The CCA measurement takes CCA_TIME us, the result shows IDLE channel
3. After CCA Rx warmdown, Link Layer inserts TXDELAY time (positive or zero) before Tx warmup
4. CCA_IRQ is asserted when CCA ends, TX_IRQ is asserted when Tx ends, SEQ_END_IRQ is asserted when sequence finishes

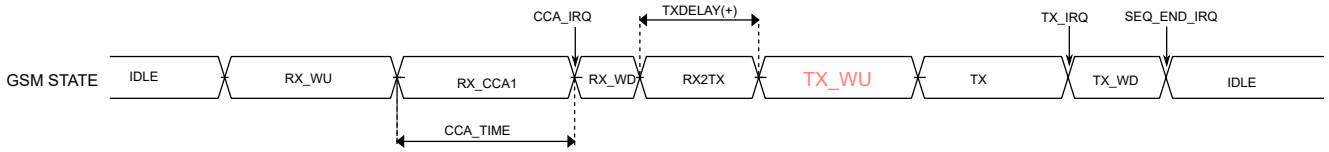


Figure 440. SEQUENCE T: Unslotted CCABFRTX, Channel is IDLE

The following timing diagram depicts the GSM state transition of Tx, highlighting the following events:

1. FAST_RX2TX_EN=1, enables TSM Fast RX-to-TX
2. Link Layer assert fast_rx2tx_wu to TSM, requests a Fast RX-to-TX warmup
3. The TSM will respond to fast_rx2tx_wu by jumping from the "ON" phase of the RX operation to the "WARMUP" phase of the TX operation at the programmed point.
4. Simultaneously, GSM jumps to its TX_WU state to wait for the TX warmup to complete.

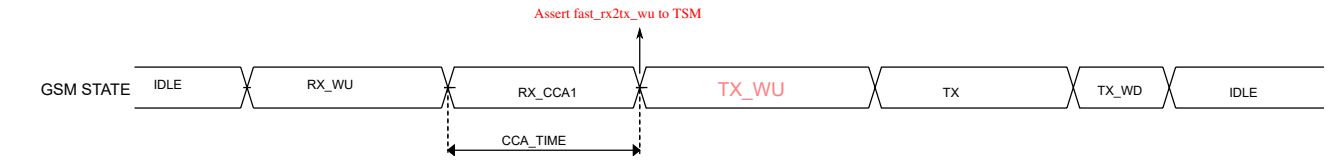


Figure 441. SEQUENCE T: Unslotted CCABFRTX, Channel is IDLE, TSM FAST RX-to-TX

The following timing diagram depicts the GSM state transition of Tx, highlighting the following events:

1. Slotted mode with CCA before TX
2. CCA_IRQ is asserted after 2nd CCA is done
3. Link Layer sends out the packet at next slot boundary(TXDELAY(±))

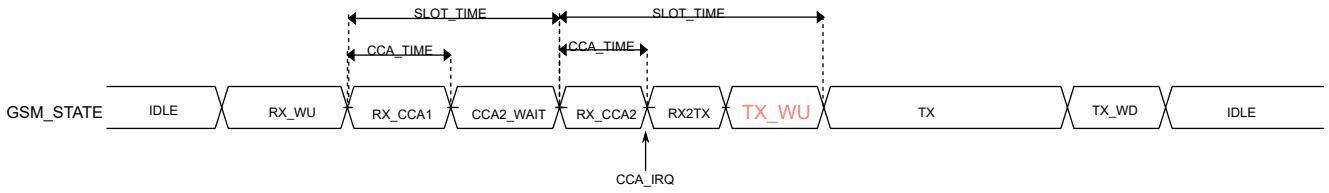


Figure 442. SEQUENCE T: Slotted CCABFRTX, Channel is IDLE

55.4.9.2.3.7.2.1.4 Sequence C (CCA)

CCA/ED/LQI

The CCA/ED/LQI block resides inside RX DIG. The purpose of this block is to perform the following operations:

1. Clear Channel Assessment: Modes 1
2. Energy Detection
3. Link Quality Indicator

This block takes RSSI value, as an input, for computations related to CCA modes 1, Energy Detection and LQI.

At the output, it provides a signal called (cca) to indicate if the channel of interest is busy (cca=1) or idle (cca=0), the final computed energy value (cca1_ed_fnl) and Link Quality Indicator (LQI) value.

The cca signal is further used to generate cca interrupt, according to the set interrupt mask. The cca1_ed_fnl is a signed 8 bit value that can be read back as a register. LQI is an unsigned 8 bit value that is written at the end of each valid packet that is received and is available to be read in the packet buffer.

Sequence C

Sequence C can be lunched by command CCA_START_*. Sequence C is the standalone CCA sequence.

During Sequence C, the sequence manager executes a Clear Channel Assessment (CCA). The result of the CCA measurement is reported back to software in the CCA bit. The CCA bit indicates either a busy channel (1), or an idle channel (0).

The basic execution of a Sequence C is as follows:

- MCU writes XCVSEQ=CCA_START_*
- Wait for T1 or T2 match (if CCA_START_T*)
- SEQ_MGR asserts one of **cca1_en** to CCA_DIG
- SEQ_MGR executes RxWarmup, via Transceiver Sequence Manager
- SEQ_MGR initiates CCA measurement by asserting **rx_cca_en** to CCA_DIG
- SEQ_MGR waits for CCA to complete (**CCA_TIME=SLOT_TIME - TURNAROUND_TIME**)
- CCA bit is set to the CCA status (1=busy 0=idle)
- CCAIRQ interrupt issued
- SEQ_MGR executes RxWarmdown
- SEQIRQ interrupt issued

The asserted control signal **cca1_en** will remain asserted for the duration of the Sequence C. The sequence manager warms up the receiver (analog and digital elements), via the Transceiver Sequence Manager (TSM). At the completion of the warmup, the sequence manager initiates the CCA by asserting the **rx_cca_en** signal to CCA_DIG. While CCA is enabled, the CCA_DIG measures and averages the energy or signal on the channel. At the completion of the CCA process, the CCA bit is set to the status of the channel (idle or busy). The CCAIRQ is issued, the receiver is warmed down, the SEQIRQ is issued, and the sequence manager returns to SEQ_IDLE state.

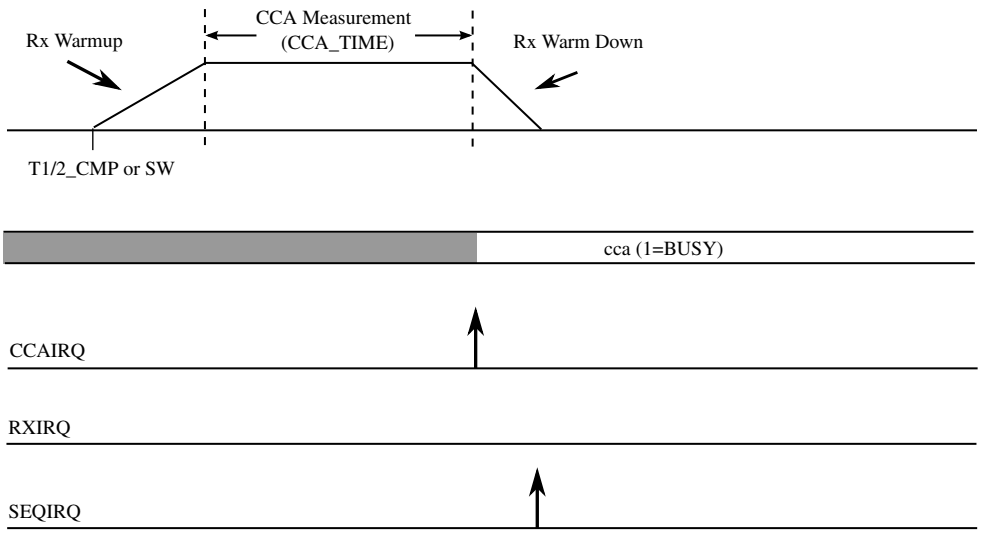


Figure 443. Sequence C Timing Diagrams

55.4.9.2.3.7.2.1.5 Sequence TR (Transmit/Receive)

Sequence TR can be lunched by command TR_START_*. Sequence TR is a combination Transmit/Receive sequence.

The sequence is executed as a concatenation of 1 transmit operation followed by 1 receive operation, with a minimum TX-to-RX turnaround time in between.

In PAN and FAN mode, there are 2 permutations of Sequence TR, depending on the RXACKRQD bit. For both permutations, the Sequence T which constitutes the first half of a Sequence TR, is identical to the standalone Sequence T (XCVSEQ=TX_START_*). This means that the transmit operation can be slotted or unslotted, and can be preceded by 1 or 2 CCA measurements, depending on the state of the CCABFRTX and SLOTTED bits.

If RXACKRQD=0, then Sequence TR is executed as a Sequence T followed by a Sequence R, with a minimum TX-to-RX turnaround time in between. The Sequence R, which constitutes the second half of the Sequence TR, is identical to the standalone Sequence R (XCVSEQ=RX_START_*). This means that the receive operation can be followed by an automatic, hardware-generated transmit Acknowledge frame, if all the necessary conditions are met. As with basic Sequence R, data for a successful receive operation is always transferred to Packet Buffer.

If RXACKRQD=1, then Sequence TR is executed as a Sequence T followed by a Receive-Acknowledge-Only frame. This type of receive operation is special, and not the same as a standalone Sequence R. The Ack-only receive operation filters all incoming frames, looking only for an Acknowledge frame whose Sequence Number matches the Sequence Number which was transmitted in the Sequence T portion of the sequence. All non-matching frames are discarded, and after each non-matching frame, the sequence manager will return the receiver to preamble-detect mode. This receive operation will continue until the matching Ack frame is received. Since this is a Receive-Acknowledge-Only operation, and not a Sequence R, there will be no receive octets transferred to Packet Buffer. The sequence will end with a RXIRQ interrupt, which indicates that the matching Ack frame was successfully received.

Needless to say, if during the transmit operation of a Sequence TR, an Acknowledge frame is being requested of the receiving end device (Frame Control Field: AckRequest=1), then Sequence TR with RXACKRQD=1 is the appropriate procedure. (Using Sequence TR with *RXACKREQ=0* is *not* recommended for this scenario, since there is no Sequence Number matching.)

Conversely, if during the transmit operation of a Sequence TR, an Acknowledge frame is *not* being requested of the receiving end device (Frame Control Field: AckRequest=0), then Sequence TR with RXACKRQD=1 should *never* be used, since a receive acknowledge frame will not be forthcoming. Instead, use Sequence TR with RXACKRQD=0, or simply Sequence T, if no followup receive frame is expected.

In GLL mode, there is nothing to check against to the received Acknowledge frame, so RXACKRQD is not used.

Sequence TR Timing Diagrams

The following timing diagram depicts the GSM state transition of TR, highlighting the following events:

1. Unslotted mode
2. The TX-to-RX time is TURNAROUND_TIME+RXDELAY(±)
3. TX_IRQ is asserted when TX ends, RX_IRQ is asserted when RX ends, SEQ_END_IRQ is asserted when sequence finishes.

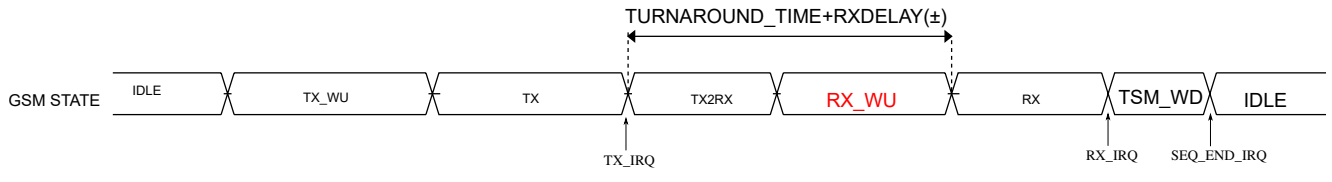


Figure 444. SEQUENCE TR: UNSLOTTED

The following timing diagram depicts the GSM state transition of TR, highlighting the following events:

1. Slotted mode

2. The time between the TX END and next slot boundary is smaller than TURNAROUND_TIME, Link Layer starts to receive at next 2 slot boundary (RXDELAY(±))

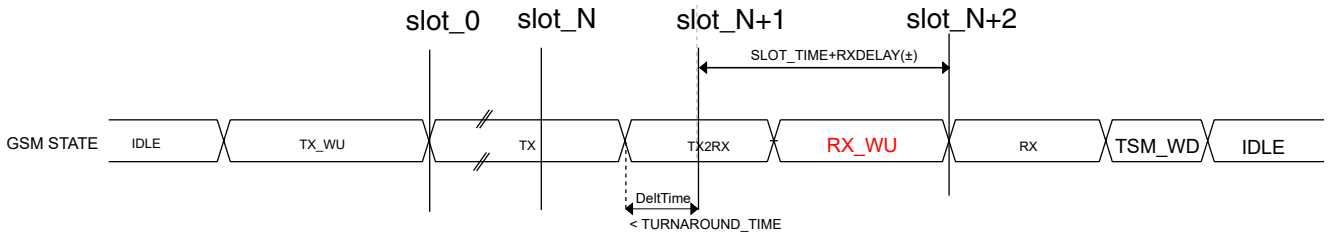


Figure 445. SEQUENCE TR: SLOTTED, CASE 1

The following timing diagram depicts the GSM state transition of TR, highlighting the following events:

1. Slotted mode
2. The time between the TX END and next slot boundary is larger than or equal to TURNAROUND_TIME, Link Layer starts to receive at next slot boundary (RXDELAY(±))

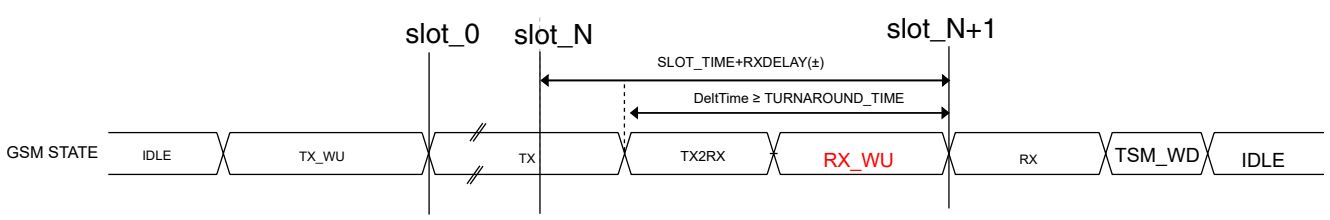


Figure 446. SEQUENCE TR: SLOTTED, CASE 2

55.4.9.2.3.7.2.2 Dual PAN Mode

Dual PAN On-the-fly Channel Change

The Sequence Manager (GSM) supports Dual PAN mode, which allows the device to reside on 2 networks simultaneously, switching back and forth between the 2 networks under software, or hardware control. (See Dual PAN Modefor more details on Dual PAN mode).

In Automatic Dual PAN Mode, the device must switch from one network to another under hardware control, at a software-programmed rate. Since the 2 networks often occupy different channels (frequencies), the sequence manager supports an "on-the-fly channel change", whereby, during a Sequence R, in preamble-search state (RX_PRE), the sequence manager will request the TSM to toggle **pll_dig_en** and switch to a new RF channel, wait out a PLL settling time, and then resume preamble search on the new channel. The entire on-the-fly channel change consumes 93us (based on default TSM programming), and consumes 2 GSM states. The first state (RX_PAN1) issues **tsm_recycle[1]** to the TSM, which causes TSM to jump from its "ON" phase, back to the tsm_count value which is programmed into the RECYCLE_COUNT1[7:0] register in XCVR space. Based on the default programming for RECYCLE1_COUNT, along with the default programming for the TSM timing registers, this TSM recycle will cause the TSM to jump back to the point 1us before **pll_dig_en** is asserted, resulting in a 1us-long deassertion of **pll_dig_en**. The TSM will resume counting from that point, which will re-assert **pll_dig_en**, and the GSM will simultaneously select the alternate Dual Pan channel. Meanwhile, the GSM will advance to RX_PAN2 state and wait there pending **tsm_wu_complete** from the TSM. The TSM will resume counting until it once again until it reaches END_OF_RX_WU[7:0], and then will return to its ON phase. At the end of the TSM RX warmup, TSM will issue **tsm_wu_complete** to GSM, which will return GSM to RX_PRE (preamble search) state.

Normally, during RX sequences, the PLL is monitored for unlock conditions, and if such an unlock were to occur, the GSM would abort the RX sequence by deasserting RX_EN to TSM. During RX_PAN1 and RX_PAN2 states, GSM asserts **in_pan_switch** to the transceiver, which indicates that PLL unlock detection should be temporarily suspended while the channel change and subsequent PLL re-lock takes place. Once the GSM transitions from RX_PAN2 state back to RX_PRE, GSM will deassert **in_pan_switch** to resume PLL unlock monitoring.

As mentioned, the entire on-the-fly channel change consumes 93us, based on default TSM programming. This 93us represents a short "blind spot" during which preamble detection can't occur, while the device switches from one RF frequency to another. See diagram below.

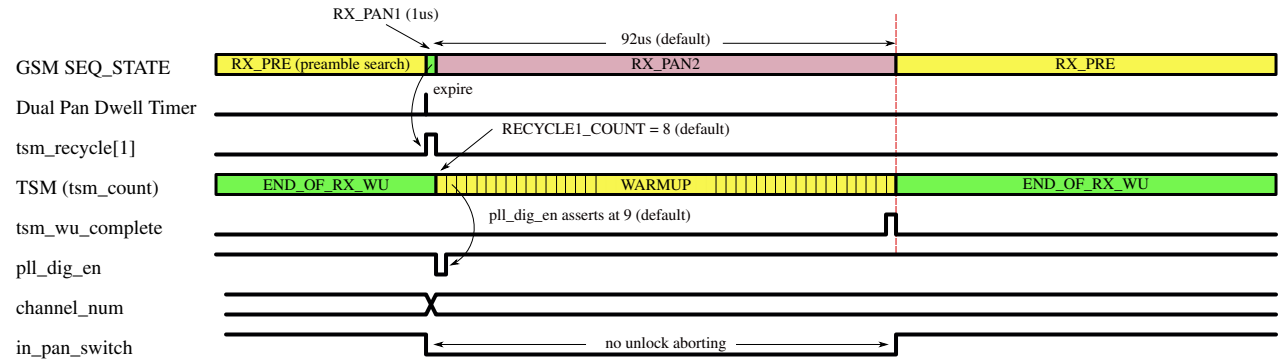


Figure 447. GSM Dual PAN on-the-fly Channel Change

Dual PAN Channel Override

When using Dual PAN mode, the channels are assigned to the 2 PAN's by programming the CHANNEL_NUM0[6:0] and CHANNEL_NUM1[6:0] registers. The channel number is processed downstream into the necessary Integer, Numerator, and Denominator components, required to generate the correct RF frequencies (see PLL Digital Chapter).

In Dual PAN mode, in case there is a need to generate a frequency which may be offset from the 16 prescribed 5MHz-spaced channels, to, for example, avoid interference on one of the Dual PAN channels, a method has been provided to do that, by designating one of the two PAN channels to use the transceiver's set of direct frequency-programming registers, instead of CHANNEL_NUMx. Programming the direct frequency-programming registers – integer, numerator, and denominator, allows an RF frequency to be selected with much more precision than the 5MHz granularity of the 802.15.4 mapped-channel registers, CHANNEL_NUM0 and CHANNEL_NUM1.

Two bits have been provided to realize this feature: DP_CHAN_OVRD_SEL and DP_CHAN_OVRD_EN. When DP_CHAN_OVRD_EN=1, this enables one of the Dual PAN channels to use the direct frequency programming. The DP_CHAN_OVRD_SEL bit determines *which* channel uses the direct programming, according to the following table:

Table 482. Dual PAN Channel Override

DP_CHAN_OVRD_EN	DP_CHAN_OVRD_SEL	PAN0 Frequency Determined by …	PAN1 Frequency Determined by …
0	X	CHANNEL_NUM0	CHANNEL_NUM1
1	0	DIRECT FREQUENCY PROGRAMMING	CHANNEL_NUM1
1	1	CHANNEL_NUM0	DIRECT FREQUENCY PROGRAMMING

Direct Frequency Programming is accomplished by setting the PLL's Integer, Numerator, and Denominator registers to the appropriate values for the desired RF frequency. These registers are in XCVR address space, and are shown in the following table:

Table 483. Direct Frequency Programming

DIRECT FREQUENCY PROGRAMMING PARAMETER	REGISTER NAME (XCVR SPACE)	REGISTER MNEMONIC
Integer	LPM_INTG[6:0]	PLL_LP_SDM_CTRL1
Numerator	LPM_NUM[27:0]	PLL_LP_SDM_CTRL2
Denominator	LPM_DENOM[27:0]	PLL_LP_SDM_CTRL3

See the PLL Digital Chapter for RF frequency programming details.

In Dual PAN automatic mode, if DP_CHAN_OVRD_EN=1, when the hardware switches to the PAN selected by DP_CHAN_OVRD_SEL as a direct-programming channel, the CHANNEL_NUMx register is ignored and the directly programmed frequency is used instead; when the hardware switches back to the other channel, the CHANNEL_NUMx register is used once again.

Hybrid Dual PAN Mode

Dual PAN Mode can be in Hybrid Mode by configuring the register GENLL_MODE. In Hybrid Mode, the two PAN can be one in PAN mode and another in FAN mode, as shown in the following table:

Table 484. Hybrid Dual PAN Mode

MODE_PAN0	MODE_PAN1	PAN0 Mode	PAN1 Mode
0	0	PAN mode	PAN mode
0	1	PAN mode	FAN mode
1	0	FAN mode	PAN mode
1	1	FAN mode	FAN mode

55.4.9.2.3.7.2.3 Simultaneous CCA/RX

The Link Layer controller features an autosequence that precedes a packet transmission with a CCA (Clear Channel Assessment), to ensure the channel is idle before transmitting. If the CCA indicates a busy channel, the autosequence ends after the CCA, without transmitting. In slotted mode, 2 CCA measurements take place, with the start points for the measurements SLOT_TIME apart. Both must indicate an idle channel for transmission to proceed.

During intervals of severe network congestion, high node density, and packets which are heavily fragmented, the situation can arise where Device A is preparing to transmit a packet, or fragment, and performing pre-transmit CCA, and during the CCA, Device B attempts to transmit another packet or fragment to Device A. The capability to receive-during-CCA is referred to as "Simultaneous CCA/RX".

Using Simultaneous CCA/RX, the Link Layer controller needs to be able to anticipate preamble, and even SFD detection, during the states in which it is taking CCA measurement. Those states are RX_CCA1, for non-slotted, and RX_CCA2, for slotted operation. If preamble and/or SFD is detected in those states, the sequence manager needs to transition to a packet reception state, and then receive the packet in its entirety as if the called sequence was Sequence R. To address this, these new state transitions have been added to the Sequence Manager state machine:

- RX_CCA1 -> RX_PRE (preamble detected during 1st CCA but no SFD)
- RX_CCA1 -> RX_PKT (preamble + SFD detected during 1st CCA)
- RX_CCA2 -> RX_PRE (preamble detected during 2nd CCA but no SFD)
- RX_CCA2 -> RX_PKT (preamble + SFD detected during 1st CCA)

False preambles may occur during the CCA states as well. False preambles occur when the demodulator detects a sufficient number of consecutive symbols to meet the preamble threshold, but the symbol which follows is not right SFD. False preambles

result in the demodulator signal sfd_rst asserting either during, or after, the CCA state. If a false preamble is declared within the CCA time window, then the fact that there was a false preamble has no bearing on the channel status, the CCA measurement determines channel status, as usual. If the false preamble is declared after the CCA window, the channel is assessed to be busy and the autosequence terminates without transmission, because the scheduled transmission time has been missed

In defining the Simultaneous CCA/RX operation, 9 "scenarios", or cases, were developed to help ensure correct hardware response to various permutations of preamble and SFD arrival times, preamble falsing, channel status, and RX packet goodness. These scenarios were used to guide the hardware design, for each of the 3 CCA modes, and for mode 3, both boolean combinations. The 9 scenarios are listed in the table below(**take the CCA_TIME 128us as example**), with the correct hardware response indicated. The hardware response consists of a sequence of state transitions, as well as assertion (or non-assertion) of the various 802.15.4 interrupts. The final combination of asserted interrupts assists software in determining what transpired during the autosequence, and therefore what must be done next.

Table 485. SIMUL CCA/RX SCENARIOS(*CCA_TIME is 128us in table*)

#	Scenario	Auto Sequence Ends After	Mode Switch to SFD Search Occurs At	Mode Switch to PKT Reception Occurs At	CCAIRQ?	RXIRQ?	TXIRQ?
1	Channel IDLE @ 128us	TX	--	--	Yes(CCA=0)	No	Yes
2	Channel Busy @ 128us	CCA	--	--	Yes(CCA=1)	No	No
3	Preamble Asserted During CCA SFD_RST Occurs During CCA Channel Otherwise IDLE @128us	TX	--	--	Yes(CCA=0)	No	Yes
4	Preamble Asserted During CCA SFD_RST Occurs During CCA Channel Otherwise Busy @128us	CCA	--	--	Yes(CCA=1)	No	No
5	Preamble Asserted @128us SFD Not Asserted @128us SFD_RST Asserts Later (False PRE)	SFD_RST	128us	--	Yes(CCA=1)	No	No
6	Preamble Asserted @128us SFD Not Asserted @128us SFD Asserts Later (Bad PKT)	RX	128us	--	Yes(CCA=1)	No	No
7	Preamble Asserted @128us SFD_RST Asserted @ 128us (Bad PKT)	RX	--	SFD DET	Yes(CCA=1)	No	No
8	Preamble Asserted @128us SFD Not Asserted @128us SFD Asserts Later (Good PKT)	RX	128us	--	Yes(CCA=1)	Yes	No
9	Preamble Asserted @128us SFD Asserted @128us(Good PKT)	RX	--	SFD DET	Yes(CCA=1)	Yes	No

A single register bit enables Simultaneous CCA/RX mode: SIMUL_CCA_RX. Simultaneous CCA/RX works with both nonslotted (single CCA before TX) and slotted (2 CCA before TX) modes.

Note that to use the Simultaneous CCA/RX feature, software (not just hardware) needs to anticipate packet reception during a Sequence T (not just a Sequeunce TR)

55.4.9.2.3.8 Work Modes

The Generic Link Layer work modes are controlled by register bits GENLL_MODE[3:0] . With proper setting, Generic Link Layer and RBME can support to receive and send different kinds of packet, like 802.15.4, FCP, Bluetooth LE packets.

- The default mode is called Generic Link Layer mode(**GLL mode**). GLL mode is the most basic mode, all commands and sequences can be used in GLL mode.
- The PAN, FAN modes are to support GFSK, MSK, SUN FSK, and LECIM FSK in the 802.15.4 protocol. In these modes, the PHRs are parsed and checked according to the protocol. Both modes also check whether the payload complies with the 802.15.4 MAC frame format. The difference is that in PAN mode, link layer only supports frames with frame version less than 2, and supports Imm-Ack frame sending and receiving. While in FAN mode, link layer supports FAN protocol-specific frame, and supports Enh-Ack frame only.
- The link layer can work in Dual PAN mode when in PAN or FAN mode. In this case, both PANs are in PAN mode or both PANs are in FAN mode. When configured to Hybrid Dual PAN Mode, one PAN can be in PAN mode while the other can be in FAN mode.
- Link layer can also work in FCP/Bluetooth LE mode.
- The test mode (GTM mode) is also supported.

55.4.9.2.3.8.1 802.15.4 Protocol Support

The link layer supports to parse the IEEE 802.15.4-2015 FSK/GFSK mode packet header when configured in PAN or FAN mode. The objective is to be able to parse the PHR field and identify when to mode switch, extract the PSDU configuration and verify the health of the header. It is intended to support the FSK/GFSK modes as specified for GFSK, MSK, SUN-FSK and LECIM-FSK PHYs.

The 4 PHR structures are shown as Figure 448

<table>
<tr><td>PPDU field</td><td colspan="2">SHR</td><td colspan="17">PHR[15:0]</td><td colspan="2">PSDU</td></tr>
<tr><td>Gen LL field</td><td>Preamble</td><td>SYNC ADDRESS (SFD)</td><td colspan="16">H0 (bit)</td><td>LENGTH (bit)</td><td>H1 (bit)</td><td>PAYLOAD</td><td>CRC (byte)</td></tr>
<tr><td></td><td></td><td></td><td>bit 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td></td><td></td><td></td><td></td></tr>
<tr><td>GFSK</td><td>0x55 x 4</td><td>0xA7</td><td colspan="16">0</td><td>7</td><td>1</td><td>N bytes</td><td>2/4</td></tr>
<tr><td>MSK</td><td>0xAA x 4</td><td>0x0B67</td><td colspan="16">0</td><td>7</td><td>1</td><td>N bytes</td><td>2/4</td></tr>
<tr><td rowspan="2">SUN FSK (w/o mode switch PHR)</td><td rowspan="2">0x55 x n, where n is 4–64</td><td rowspan="2">0x72F6(coded) 0x7209 0xB4C6(coded) 0x705E</td><td colspan="16">5</td><td>11</td><td></td><td></td><td></td></tr>
<tr><td>Mode Switch(0)</td><td colspan="2">Reserved</td><td>FCS Type</td><td>Data Whitening</td><td colspan="11"></td><td>Frame Length[10:0]</td><td>0</td><td>N bytes</td><td>by FCS Type</td></tr>
<tr><td rowspan="2">SUN FSK (w/ mode switch PHR)</td><td colspan="16">16</td><td rowspan="2">0</td><td rowspan="2">0</td><td rowspan="2">0</td><td rowspan="2">0</td></tr>
<tr><td>Mode Switch(1)</td><td colspan="2">Mode Switch Parameter</td><td>New Mode FEC</td><td colspan="3">New Mode</td><td colspan="2">Checksum</td><td>Parity</td></tr>
<tr><td></td><td></td><td></td><td colspan="4"></td><td>Page</td><td colspan="2">Modulation Scheme</td><td>Mode</td><td colspan="3"></td></tr>
<tr><td rowspan="2">LECIM FSK</td><td rowspan="2">0x55 x n, where n is 4–64</td><td rowspan="2">0x4B770E</td><td colspan="16">5</td><td>11</td><td></td><td></td><td></td></tr>
<tr><td colspan="2">Reserved</td><td>Parity</td><td>FCS Type</td><td>Data Whitening</td><td colspan="11"></td><td>Frame Length[10:0]</td><td>0</td><td>N bytes</td><td>by FCS Type</td></tr>
</table>

Figure 448. PPDU Mapping to Link Layer Packet Structure

Software should configure the register PHR_TYPE[2:0] to select the proper PHR type before Tx or Rx.

For SUN FSK, user should put the **uncoded network address in NTW_ADR_0 or NTW_ADR_2** and put the **coded network address in NTW_ADR_1 or NTW_ADR_3**.

And also software should configure the related registers to correctly work with different PHRs, as shown in Table 486.

Table 486. Link Layer Register Configuration for 802.15.4 Emulation

Register	GFSK/MSK	SUN FSK	LECIM FSK
GENLL_MODE[3:0]	• 0x1 - PAN Mode • 0x2 - FAN Mode • 0x3 - Hybrid Dual PAN Mode		

Table continues on the next page...

Table 486. Link Layer Register Configuration for 802.15.4 Emulation (continued)

Register	GFSK/MSK	SUN FSK	LECIM FSK
PREAMBLE_SZ[8:0]	As needed		
PREAMBLE_SEL[2:0]	• 000b - The controller hardware selects the preamble pattern based on the first transmitted bit of Network Address, such that the last bit of preamble is the opposite polarity from the first bit of Network Address, forcing a bit transition at this boundary. • 001b - Preamble is programmed by register GEN_PREAMBLE[7:0]		
GEN_PREAMBLE[7:0]	As needed		
NTW_ADR_SZ[1:0]	According to SFD length		
SYNC_ADDR_SZ[1:0]	According to SFD length		
H0_SZ[4:0]	0	• 5 (MS=0) • 16 (MS=1)	5
LENGTH_SZ[4:0]	7	• 11(MS=0) • 0 (MS=1)	11
H1_SZ[4:0]	1	0	
LENGTH_BIT_ORD	1 - MS Bit First 0 - LS Bit First		
LENGTH_ADJ[10:0]	0		
AA_PLAYBACK_CNT	0 - Access Address will not play back to Link Layer		
RXEN_DLY[9:0]	0		
PHR_TYPE[2:0]	• 000b - The packet type is GFSK • 001b - The packet type is MSK	010b - The packet type is SUN FSK	011b - The packet type is LECIM FSK

For SUN FSK PHR, there will be two cases: one is mode switch PPDU and the other is PPDU without mode switch. LL distinguishes between the two modes by the "Mode Switch (MS)" bit received. If MS=1, then link layer hardware will override the H0_SZ to 16, override LENGTH_SZ to 0. And if MS=0, then link layer hardware will override the H0_SZ to 5 and override the LENGTH_SZ to 11.

For SUN FSK PHR (MS=0) and LECIM FSK PHR type, when Rx, link layer hardware will override the CRC_SZ and WHITEN_START register according to the received FCS Type and Data Whitening bits. That is when FCS Type = 0, CRC_SZ is 4; when FCS Type = 1, CRC_SZ is 2. And when Data Whitening = 0, WHITEN_START=0; when Data Whitening = 1, WHITEN_START=3. When Tx, the PHR is programmed in Packet RAM by software. The FCS Type and Data Whitening bits in PHR should align with the CRC_SZ and WHITEN_START registers. When link layer is performing an auto-Ack sequence, the FCS Type and Data Whitening bits are automatically calculated by hardware according to the CRC_SZ and WHITEN_START registers.

When Rx, for SUN FSK(MS=1), RBME will do the BCH correction and Parity check for the PHR and it will send out the corrected result and fail/pass flag to link layer. If the flag shows pass, an interrupt flag MS_IRQ will be set in IRQ_CTRL register. If MS_IRQ_EN=1, then the interrupt will occur. The PHR[10:0] will be saved into PHR_MISC register for software read. For LECIM FSK, link layer will check the Parity bit. When either fail happens, BCH fail or Parity fail, the PHRFAIL_IRQ will be set and RX_IRQ

will not set. if PHRFAIL_IRQ_EN=1, then the fail interrupt will occur. If PHR_FAIL_IGNORE =1, RX_IRQ will be set even for a received packet which fails PHR verification.

When Tx, for SUN FSK(MS=1), RBME will do the BCH encode and Parity calculation, and fill the Checksum and Parity Check bits in PHR. For LECIM FSK, the Parity bit is calculated by software and programmed into the Packet RAM.

When auto-Ack, the Parity bit for LECIM FSK are calculated by link layer hardware.

55.4.9.2.3.8.1.1 Packet Processor

55.4.9.2.3.8.1.1.1 Introduction

The Packet Processor works in PAN or FAN mode.

In PAN mode, the Packet Processor performs sophisticated hardware filtering of the incoming received packet, to determine whether the packet is both PHY- and MAC-compliant, whether the packet is addressed to this device, and if the device is a PAN Coordinator, whether a message is pending for the sending device.

In FAN mode, the Packet Processor checks whether the received packet passes the rules in FAN standard.

The packet processor greatly reduces the packet filtering burden on software, allowing software to tend to higher-layer tasks with a lower latency and smaller software footprint.

Table 487. References

Revision	Title
IEEE Std 802.15.4 -2003	Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)
IEEE Std 802.15.4 -2006	802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)
IEEE Std 802.15.4 -2011	Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)
IEEE Std 802.15.4 -2015	Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) (Amendment to IEEE Std 802.15.4-2011)
Version 1v10	Technical Profile Specification Field Area Network (FAN)

55.4.9.2.3.8.1.1.1.1 Overview

The Packet Processor performs aggressive hardware filtering of the incoming packet as it is being received, to increase battery life by not awakening the MCU to process packets which are non-compliant or not addressed to the device, or that violate the 802.15.4 MAC Frame structure.

The advanced frame parsing enables the sequence manager to handle complex sequences, by, for example, determining whether a receive frame requires an acknowledge, and generating the Sequence Number for that acknowledge, so that the transmit/receive combination sequence can be executed without any MCU intervention.

Finally, the packet processor includes a hardware accelerator that allows the device to handle a critical timing requirement that had been problematic for 8-bit MCU's in the past: the quick-turnaround requirement for an indirect queue-lookup to determine the correct response to an end device sending a data polling request.

The packet processor contains a register-based table of end devices which have messages pending, allowing a single-cycle hardware table lookup to be performed, and a minimum-turnaround-time acknowledge packet to be sent to the end device without MCU intervention.

The packet processor also includes support for Dual PAN mode, which allows the device to simultaneously reside on 2 networks, with 2 separate IEEE addresses.

55.4.9.2.3.8.1.1.1.2 Features

- Aggressive packet filtering to enable long, uninterrupted MCU sleep periods

- Fully compliant with both 2003 and 2006 versions of the 802.15.4 wireless standard
- Supports all Frame Types, including reserved types
- Enables auto-Tx Acknowledge frames (no MCU intervention) by parsing of Frame Control Field and Sequence Number
- Supports all Source and Destination Address modes, and also PAN ID Compression
- Supports Broadcast address for PAN ID and short address mode
- Supports "Promiscuous" mode, to receive all packets regardless of address- and rules-checking
- Allows Frame Type-specific filtering (e.g., reject all but Beacon frames)
- Supports SLOTTED and non-SLOTTED modes
- Includes special filtering rules for PAN Coordinator devices
- Enables minimum-turnaround TX-Acknowledge frames for data-polling requests by automatically determining message-pending status
- Assists MCU in locating pending messages in its indirect queue for data-polling end devices
- Assists MCU in determining if a poll request originated from a valid child end device
- Makes available to MCU detailed status of frames which fail address- or rules-checking
- Supports Dual PAN mode, allowing the device to exist on 2 PAN's simultaneously
- Supports 2 IEEE addresses for the device
- Includes provisions to enable software support of 802.15.4-2015 frame types and versions
- **Supports FAN standard DATA and ACK frame filter.**

55.4.9.2.3.8.1.1.2 Functional Description

The packet processor is responsible for receiving symbols from the demodulator, combining them into octets, assembling the octets into packets, filtering the packets in compliance with the 802.15.4 wireless MAC standard, and storing the packets into the Packet Buffer (Packet RAM).

The packet processor parses the incoming packets on-the-fly to determine:

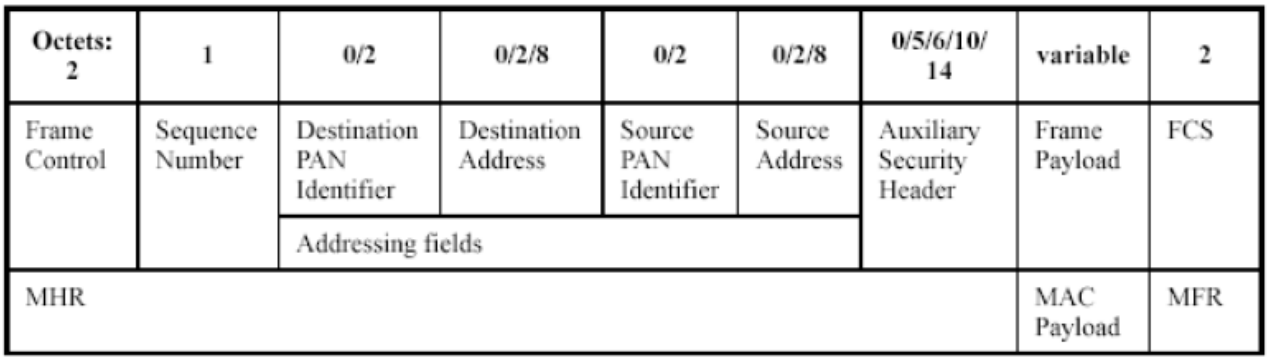
1. whether the packet is MAC-compliant, and if it is,
2. whether the packet is addressed to the end device.

The packet processor performs aggressive filtering on incoming packets, in order to reduce unnecessary MCU wake-ups, to allow longer MCU sleep durations, and to provide sufficient hardware automation so as to enable complex sequences to be executed by the sequence manager. The packet processor also includes hardware support for Dual PAN operation (allowing the device to reside on two networks simultaneously).

The packet processor also includes functionality to allow a critical-timing requirement to be met, which in the past has been problematic for low-MIPS receiving devices: the ability of a coordinator device to respond to a MAC Command data request from an end device, with an Acknowledge frame containing a FramePending bit accurately reflecting the presence of a message for the end device in the coordinator's indirect queue, or the absence of the end device in the coordinator's neighbor table, within the prescribed turnaround time window. This functionality is described in more detail in the Source Address Management

55.4.9.2.3.8.1.1.2.1 Packet Filtering Background

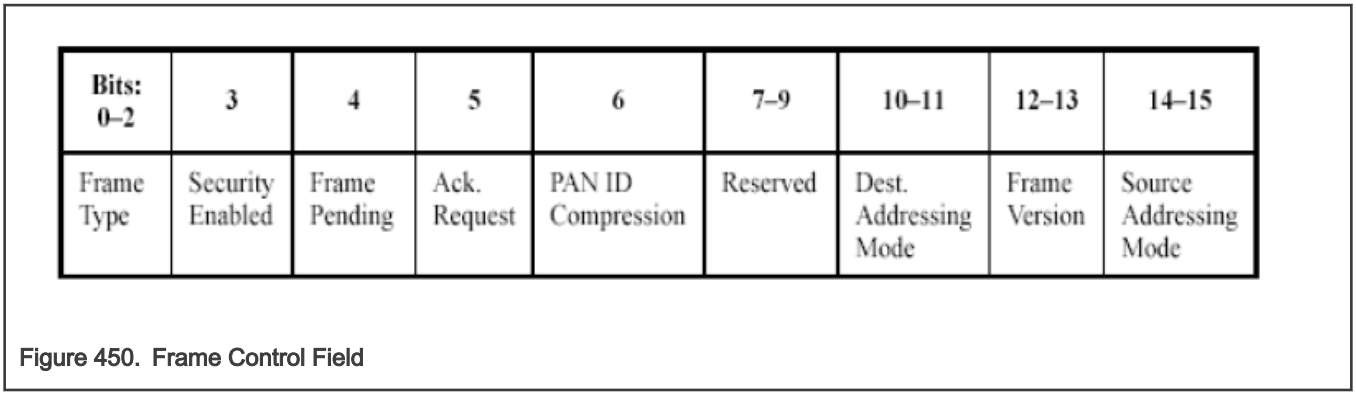
The packet processor parses packets to verify compliance with the 802.15.4 MAC frame format. The basic MAC frame format is shown in the figure below.



Octets: 2	1	0/2	0/2/8	0/2	0/2/8	0/5/6/10/14	variable	2
Frame Control	Sequence Number	Destination PAN Identifier	Destination Address	Source PAN Identifier	Source Address	Auxiliary Security Header	Frame Payload	FCS
		Addressing fields						
MHR							MAC Payload	MFR

Figure 449. MAC Frame Format

The packet processor parses the incoming octets on the fly, as they are received. The Frame Control Field, two octets in length, contains subfields which encode "instructions" on how to parse the remainder of the MHR (MAC Header). The structure of the Frame Control Field is shown in the figure below.



Bits: 0–2	3	4	5	6	7–9	10–11	12–13	14–15
Frame Type	Security Enabled	Frame Pending	Ack. Request	PAN ID Compression	Reserved	Dest. Addressing Mode	Frame Version	Source Addressing Mode

Figure 450. Frame Control Field

The packet processor utilizes the Frame Control Fields subfields in the following manner:

FCF Subfield	Utilization by Packet Processor
FrameType	Interprets the remaining MHR as specific to Beacon, Ack, Data, Command, or Reserved frame types. Each frame type has a unique MHR structure, and so different parsing rules apply.
SecurityEnabled	If SecurityEnabled=1 and FrameVersion>0 (frame versions 2006 and later), an Auxiliary Security Header field will be present in the MHR and will need to be further parsed by the packet processor if this is a MAC Command frame (see Section Source Address Management). There is no other use of SecurityEnabled by the packet processor.
FramePending	ignored by packet processor
Ack. Request	will be copied into the **pp_ack_request** signal to the sequence manager. An auto-TxAck frame will follow the incoming receive frame if necessary conditions are met
PAN ID Compression	used for addressing mode rules-checking and addressing field parsing.

Table continues on the next page...

Table continued from the previous page...

FCF Subfield	Utilization by Packet Processor
Reserved	ignored by packet processor
Destination Addressing Mode	used for addressing mode rules-checking and addressing field parsing
FrameVersion	For Beacon, Data, and MAC Command frames, FrameVersion is checked against the allowed frame versions according to FRM_VER_FILTER For Acknowledge frames, FrameVersion is ignored. If SecurityEnabled=1 and FrameVersion>0 (frame versions 2006 and later), an Auxiliary Security Header field will be present in the MHR and the MHR will need to be further parsed by the packet processor if this is a MAC Command frame (see SectionSource Address Management).
Source Addressing Mode	used for addressing mode rules-checking and addressing field parsing

The Packet Processor parses the FrameType field of the Frame Control Field per the following table:

Frame type value b_2 b_1 b_0	Description
000	Beacon
001	Data
010	Acknowledgment
011	MAC command
100–111	Reserved

Figure 451. Frame Type Field

Directly following the Frame Control Field is the Sequence Number field. The Sequence Number field of the MHR is captured by the packet processor. If an auto-TxAck frame follows the incoming receive frame, the captured Sequence Number will be copied to the transmitted Acknowledge packet, and the Packet Processor will insert FrameVersion=00 into the Frame Control Field of the transmitted frame, regardless of the received FrameVersion subfield of the original frame.

The Addressing Fields follow the Sequence Number. The format of the Addressing Fields depends upon the Source and Destination Addressing Mode subfields of the Frame Control Field. The Addressing Modes are defined in the table below:

Addressing mode value b_1 b_0	Description
00	PAN identifier and address fields are not present.
01	Reserved.
10	Address field contains a 16-bit short address.
11	Address field contains a 64-bit extended address.

Figure 452. Addressing Mode Fields

The packet processor uses these Source and Destination Address Modes shown above, to extract the Source PAN ID and Address (if present), and the Destination PAN ID and Address (if present), from the MHR, according to the table below:

FrameType	Destination Addressing Mode	Source Addressing Mode	PanID-Compression	Addressing Fields
Acknowledge only (reject all other frame types)	0	0	0	None
Data or Command (reject Ack and Beacon frames)	2 or 3	0	0	DstPanID + DstAddr
Beacon (all devices), or Data or Command (PAN Coord Only) (reject Ack frames)	0	2 or 3	0	SrcPanID + SrcAddr
Data or Command (reject Ack and Beacon frames)	2 or 3	2 or 3	0	DstPanID + DstAddr + SrcPanId + SrcAddr
Reject all frames (Illegal as per Section 7.2.2.1.5)	0	0	1	-
Reject all frames (Illegal as per Section 7.2.2.1.5)	2 or 3	0	1	-
Reject all frames	0	2 or 3	1	-

Table continues on the next page...

Table continued from the previous page...

FrameType	Destination Addressing Mode	Source Addressing Mode	PanID-Compression	Addressing Fields
(Illegal as per Section 7.2.2.1.5)				
Data or Command (reject Ack or Beacon frames)	2 or 3	2 or 3	1	DstPanID + DstAddr + SrcAddr
Reject all frames (Illegal as per Section 7.2.2.1.6)	1	-	-	-
Reject all frames (Illegal as per Section 7.2.2.1.8)	-	1	-	-

The Source PAN ID and Address, and Destination PAN ID and Address, extracted by the packet processor, are then checked against the MACPANID, MACSHORTADDRS, and MACLONGADDRS register settings, depending on the FrameType of the incoming packet, to determine:

1. Is the addressing mode combination valid for this frame type?
2. Do the address fields indicate that the packet is indeed addressed to the end device?

The details on how the packet processor answers those 2 questions, are described in the following sections.

55.4.9.2.3.8.1.1.2.2 Packet Filtering Detail

The figure below is a high-level representation of the packet filtering performed by the 802.15.4 packet processor. The sections which follow provide more detail on the packet filter implementation.

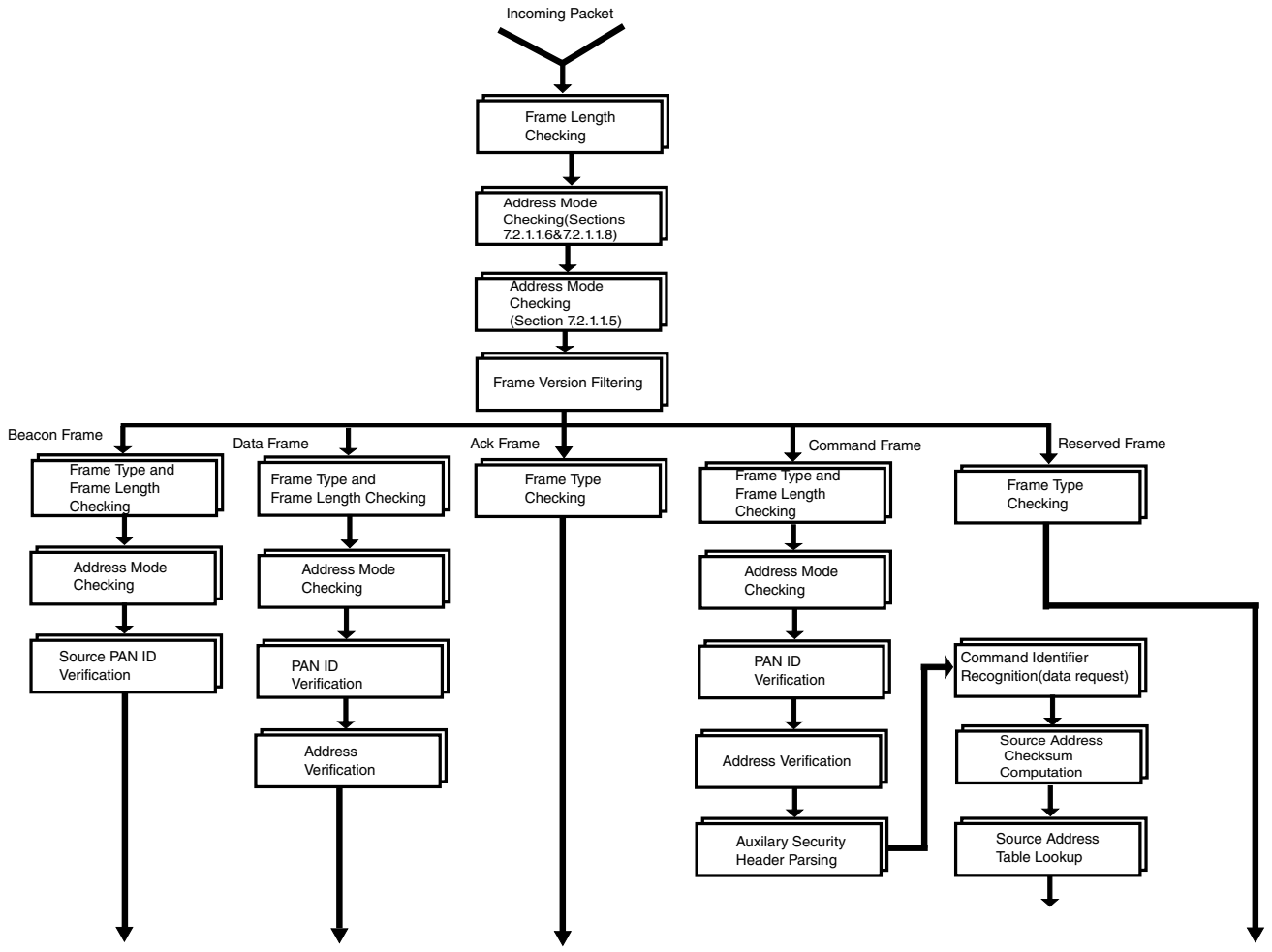


Figure 453. Packet Filtering Flow Diagram

First-Stage Packet Filtering

The first stage of packet filtering examines the incoming packet, and implements basic rule-checking, based on the 2006 version of the 802.15.4 standard, and is applied to *all* frame types. First stage rule-checking is described in the table below:

	Filtering Step	Description
1	Frame Length Checking	Check that all frames are at least 3 octets payload (not include FCS field).
2	Address Mode Checking (Sections 7.2.1.1.6 and 7.2.1.1.8)	Check that neither Source nor Destination Address Mode set to 1 (illegal addressing mode)
3	Address Mode Checking (Section 7.2.1.1.5)	Check for PAN ID Compression enabled, with either Source or Destination Address Mode set to 0 (illegal addressing mode)
4	FrameVersion Checking	Check that incoming packet's Frame Version is allowed based on FRM_VER_FILTER setting. (FRM_VER_FILTER is a programmable register, see Register Descriptions Section).

Packets which pass first stage filtering, proceed to the second stage of packet filtering, which is *frame type-specific* filtering. Packets which fail initial rule checking are marked as rejected.

Second-Stage Packet Filtering

The second stage of packet filtering is *frame type-specific*. Different variations of the MAC Header (MHR) apply to different frame types, so different parsing mechanisms apply to each.

Note: In the following section, the following references are to register bits; see Register Description Section for register descriptions: BEACON_FT, ACK_FT, DATA_FT, CMD_FT, NS_FT, PANCORDNTR, MACPANID, MACSHORTADDRS, and MACLONGADDRS.

1) For Beacon frames, these are the basic filtering steps:

- Check that Beacon frames are enabled by software (BEACON_FT)
- Check that Payload Length >= 7
- Check for illegal combinations of PAN ID Compression and Source/Destination Address Modes
- Verify the Source PAN ID matches MACPANID, or that MACPANID = 0xffff.

2) For Acknowledge frames, these are the basic filtering steps:

- Check that Ack frames are enabled by software (ACK_FT)
- If Sequence TR is active and RXACKRQD = 1, check the received Sequence Number matches the transmitted Sequence Number; FrameVersion bits are ignored, as per section 7.2.2.3.1 of the 802.15.4 standard.

Note: It is recommended to use Sequence TR to receive Acknowledge frames, not Sequence R. See Chapter Sequence Manager, Section.Sequence TR (Transmit/Receive)

3) For Data frames, these are the basic filtering steps:

- Check that Data frames are enabled by software (DATA_FT)
- Check that Payload Length >= 7
- Check for illegal combinations of Source and Destination Address Modes
- If PANCORDNTR=0, check for missing Destination Address field or present Source Address field
- If no Destination Address field (PANCORDNTR=1 only), check Source PAN ID matches MACPANID
- If Destination Address present, check Destination PAN ID matches MACPANID or broadcast (0xffff)
- If Short Destination Address field, check address matches MACSHORTADDRS or broadcast (0xffff)
- If Long Destination Address field, check address matches MACLONGADDRS.

4) For MAC Command frames, these are the basic filtering steps:

- Check that Command frames are enabled by software (CMD_FT)
- Check that Payload Length >= 7
- Check for illegal combinations of Source and Destination Address Modes
- If PANCORDNTR=0, check for missing Destination Address field or present Source Address field
- If no Destination Address field (PANCORDNTR=1 only), check Source PAN ID matches MACPANID
- If no Destination Address field (PANCORDNTR=1 only), capture Source PAN ID and Source Address for later use
- If Destination Address present, check Destination PAN ID matches MACPANID or broadcast (0xffff)
- If Short Destination Address field, check address matches MACSHORTADDRS or broadcast (0xffff)
- If Long Destination Address field, check address matches MACLONGADDRS.

- At end of the addressing field, if incoming packet's FrameVersion=0 or SecurityEnabled=0, capture the next octet (Command Identifier)
- At end of the addressing field, if incoming packet's FrameVersion!=0 and SecurityEnabled=1, parse the Auxiliary Security Header, then capture the next octet (Command Identifier)
- If Command Identifier indicates the MAC Command is a data request, compute a "Source Address Checksum" to uniquely identify the sending device.

55.4.9.2.3.8.1.1.2.3 Source Address Management

The 802.15.4 wireless MAC standard envisions a scenario whereby an end device may interrogate a coordinator as to whether the coordinator is storing data (i.e., a pending message) for the end device. The situation arises in a beacon-enabled network, when a coordinator includes in its transmitted beacon frame the MAC address of the end device in its "Pending Address Fields". The "Pending Address Fields" are part of the required MAC payload of the beacon frame. The "Pending Address Fields" contain a list of end device addresses for which messages are pending. In this scenario, an end device which finds its address included in the "Pending Address Fields" of the received beacon frame, must respond to the beacon (coordinator) with a MAC Command of type "data request". Alternatively, in non-beacon-enabled networks, an end device may periodically wake up and "poll" a coordinator, to determine if a message is pending for the end device. In either case, the coordinator stores messages for its end devices in its "indirect queue". The coordinator must respond to an incoming MAC Command data request (which *must* have AckRequest=1 in its Frame Control Field), with an Acknowledge frame containing a FramePending subfield indicating the presence (or absence) of a message for the requesting end device, in the coordinator's indirect queue. For a coordinator built around a low-MIPS MCU, a significant amount of MCU time is typically required, simply to ascertain whether a given end device has a message pending or not; the indirect queue can be quite large, and for 8-bit MCU's, the queue must be searched byte-by-byte. This makes it difficult, or impossible, for an underpowered MCU to establish frame-pending status within the RX-to-TX turnaround time required by the standard.

Also, coordinators may include, in software, a "Neighbor Table", the contents of which consist of all the end devices with which the coordinator is in communication. The neighbor table requires periodic maintenance by coordinator (parent) software, as end devices (children) come and go within the network, join different parents, or simply never return. Child tables have limited capacity and parents need to create space for new devices that may show up. A timeout mechanism is the only way that can guarantee that child tables will eventually be cleaned up. Coordinator software must "age out" end devices with which there has not been recent communication. In order for an end device to refresh its status within its parent's Neighbor Table, the end device may periodically send a "keepalive" message in the form of a MAC Data Poll Request. Upon receipt of such a poll request, the parent must search its neighbor table to determine if the source address of the end device is a valid table entry. If the end device exists in the table, an Acknowledgment packet must be transmitted to the end device, with the FramePending subfield set to 0, and coordinator software resets that end device's timeout counter; if the child is not present in the table, an Acknowledgment packet must be transmitted to the end device, with the FramePending subfield set to 1, and the parent device will follow up by transmitting a "Leave Packet" to the end device, instructing the end device to sever its child status with the parent. The end device may opt to rejoin at a later time. Due to the size of the parent's Neighbor table, the search a child's source address can be time consuming, and may not always be possible within the RX-to-TX turnaround time required by the standard.

To alleviate both of these problems, the 802.15.4 packet processor includes hardware acceleration to expedite the source address table search, greatly relieving the software bottleneck during the critical turnaround time. The table search acceleration consists of 2 components: 1) a search for the *presence* of a specific end device's source address, which expedites the process of determining whether a message is pending for the device; and 2) a search for the *absence* of a specific end device's source address, which expedites the process of determining whether the end device is a valid child of the parent.

The packet processor maintains a register table (128 deep, 16-bit wide). The table is divided into multiple partitions, one to perform the "Source Address Present" (SAP) search, and another to perform the "Source Address Absent" (SAA) search. In the SAP partition of the table, coordinator software will store the address of end devices for which it has a message pending; in the SAA partition of the table, coordinator software will store the address of end devices for which it is the parent. To reduce area, the address information is stored in a "compressed" format. Software compresses the address information, and stores it into the table partitions, during a non-critical time. Then, when the coordinator's packet processor receives a MAC Command data request, it extracts the source address information from the incoming packet, and compresses it using the *same* compression algorithm used by software to populate the source address table. The packet processor performs 2 steps simultaneously:

1. it compares the just-received, compressed source address information, or "checksum", against all of the enabled entries in the SAP partition of the table. If a match is found in the SAP partition, the packet processor *asserts* the **rx_frame_pending** signal to the transmit block (TX_PACKET), so that an auto-TxAck packet can be sent to the requesting end device, with the FramePending subfield set to 1, indicating that the end device has a message pending in the coordinator's indirect queue. A message packet from the coordinator to the end device will follow.
2. it compares the just-received, compressed source address information, or "checksum", against all of the enabled entries in the SAA partition of the table. If *no* match is found in the SAA partition, the packet processor *asserts* the **rx_frame_pending** signal to the transmit block (TX_PACKET), so that an auto-TxAck packet can be sent to the requesting end device, with the FramePending subfield set to 1, indicating that the end device is not a valid child of this parent, and a "Leave Packet" from the coordinator to the end device, will follow.

If neither condition is met, i.e., there is no address match in the SAP partition, and a positive match in the SAA partition, the packet processor *deasserts* the **rx_frame_pending** signal, so that an auto-TxAck packet can be sent indicating to the end device that it has no data pending, and that it remains a valid child of this parent; the end device may then deactivate its receiver immediately. For the packet processor, the entire table lookup occurs in 128 clock cycles, once the last source address octet has been received. The SAP and SAA lookups are performed simultaneously. The checksum is also computed on-the-fly by hardware as the source address octets are received, taking no more than 1 clock cycle per octet. Finally, table-search status is reported to the MCU:

1. after the packet processor completes its SAP table search, the status bit SAP0_ADDR_PRESENT will be set to 1 if a match is found in the SAP partition, and the table index which matched will be indicated in the SAP0_MATCH[6:0] register, to further assist software in locating the relevant message within its indirect queue; if no match is found in the SAP partition, the status bit SAP0_ADDR_PRESENT will be set to 0.
2. after the packet processor completes its SAA table search, the status bit SAA0_ADDR_ABSENT will be set to 1 if *no* match is found in the SAA partition; if a positive match is found in the SAA partition, the status bit SAA0_ADDR_ABSENT will be set to 0, and the table index which matched will be indicated in the SAA0_MATCH[6:0] register, to further assist software in locating the child device within its neighbor table.

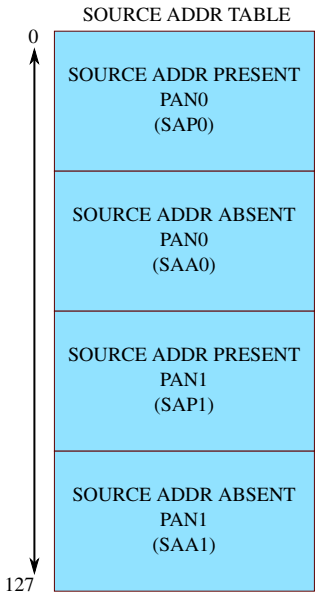
Both features, SAP and SAA acceleration, are optional, and functionally independent. Coordinator software can elect to enable both, either, or neither feature. The control bit SAP0_EN enables SAP acceleration. The control bit SAA0_EN enables SAA acceleration. If either feature is disabled, that table partition (SAP or SAA) is not searched, and so is not a factor in determining FramePending status for the transmitted Acknowledge frame. If *both* features are disabled, FramePending status defaults to software control, and merely tracks the state of the ACK_FRM_PND register bit; in this case software must manage FramePending status for all auto-TxAck frames.

Software can also take direct control of FramePending status for the next auto-TxAck frame at any time (regardless of whether SAP or SAA acceleration is enabled), by setting the ACK_FRM_PND_CTRL bit; in this case, hardware FramePending status will track the state of ACK_FRM_PND.

In order to support Dual PAN mode, a device could theoretically be a PAN Coordinator on both networks. Thus, a coordinator device may need to maintain distinct message pending lists and/or neighbor tables for the 2 PANs. Accordingly, the packet processor further subdivides the Source Address Management table into 4, instead of 2, partitions:

1. SAP0: Source Address Present Partition for PAN0
2. SAA0: Source Address Absent Partition for PAN0
3. SAP1: Source Address Present Partition for PAN1
4. SAA1: Source Address Absent Partition for PAN1

SOURCE ADDRESS TABLE PARTITIONS



Four register control bits enable each partition individually. If a partition is enabled, and a MAC poll request is received on the corresponding PAN (e.g. SAP0 or SAA0 partitions on PAN0), then the partition is searched by the packet processor, looking for a source address match (or absence), and the result of the search affects the FramePending status; if a partition is disabled, the partition is not searched, and will not influence FramePending status. The register bits are described in the table below.

REGISTER NAME	DESCRIPTION
SAP0_EN	Enables SAP0 Partition
SAA0_EN	Enables SAA0 Partition
SAP1_EN	Enables SAP1 Partition
SAA1_EN	Enables SAA1 Partition

For maximum flexibility in supporting all combinations of message-pending status, neighbor-table status, and dual PAN mode, the dividing-line between all the partitions is fully programmable. The SAP0 partition always starts at table index 0. Three registers control the starting point for the other 3 partitions. The starting point is defined the first table index (0-127) of that partition. The registers are described below, along with some software restrictions to ensure table consistency.

REGISTER NAME	DESCRIPTION	RESTRICTIONS
SAA0_START[6:0]	First Index of SAA0 partition.	If SAA0_START=0, then there is no SAP0 partition (SAP0_EN should be set to 0)
SAP1_START[6:0]	First Index of SAP1 partition.	Software must ensure that: **SAP1_START ≥ SAA0_START** If SAP1_START = SAA0_START, then there is no SAA0 partition (SAA0_EN should be set to 0)
SAA1_START[6:0]	First Index of SAA1 partition.	Software must ensure that:

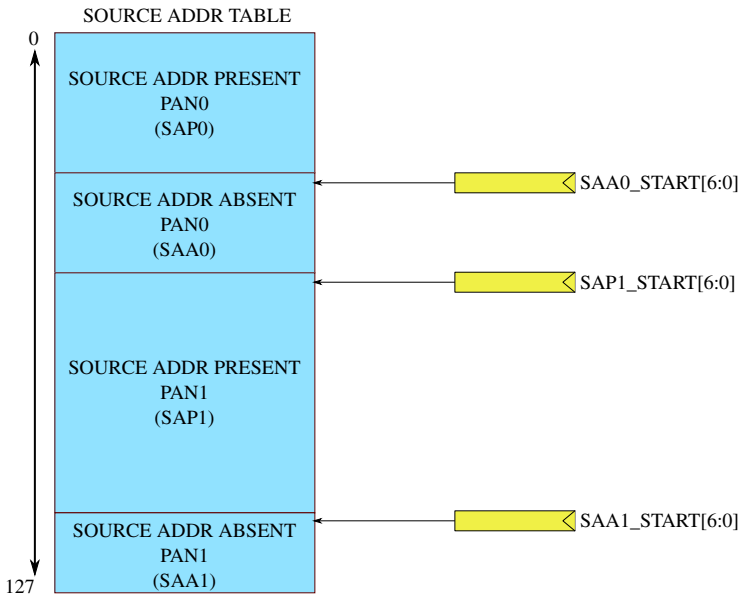
Table continues on the next page...

Table continued from the previous page...

REGISTER NAME	DESCRIPTION	RESTRICTIONS
		SAA1_START ≥ SAP1_START If SAA1_START = SAP1_START, then there is no SAP1 partition (SAP1_EN should be set to 0)

A diagram which depicts one example of how the dividing-line registers affect partitioning of the table, is shown below.

SOURCE ADDRESS TABLE PARTITIONS -- NON-EQUAL SIZES (EXAMPLE)



Each table partition consists of one or more indices, which can be used to store source address checksums. The first index for any partition, is defined by the aforementioned *PARTITION*_START[6:0] register, where *PARTITION*_ is one of {SAP0, SAP1, SAA0, SAA1}. The last index for any partition, is defined by the next higher partition's *PARTITION*_START[6:0] register, minus 1. When populating the table with source address information, software must be sure to select an index that is within the defined range for the partition. For example, when installing a checksum in the SAP1 partition:

SAP1_START ≤ Selected_Index ≤ (SAA1_START – 1)

To populate the table partitions, a 7-bit register SAM_INDEX[6:0] is provided, as well as a "index enable bit", and a "index invalidate" bit. A 16-bit SAM_CHECKSUM[15:0] is provided, into which software will write the computed checksum. Installing a checksum into any partition is referred to as "Enabling an Index", and removing a checksum from any partition is referred to as "Invalidating an Index". A description of the registers is provided in the following table:

REGISTER NAME	DESCRIPTION
SAM_CHECKSUM[15:0]	Software-computed source address checksum, to be installed into a table index
SAM_INDEX[6:0]	Contains the table index to be enabled or invalidated. Software must ensure that the index is within the range of the desired partition.

Table continues on the next page...

Table continued from the previous page...

REGISTER NAME	DESCRIPTION
SAM_INDEX_WR	For 32-bit writes, this must be set to indicate that the table entry specified by SAM_INDEX[6:0] is to be written; if SAM_INDEX_WR=0, the table entry is not written, but the SAM_INDEX[6:0] register is updated. For 8-bit writes, this bit is ignored.
SAM_INDEX_EN	Enable the index selected by SAM_INDEX[6:0] for searching.
SAM_INDEX_INV	Invalidate the index selected by SAM_INDEX[6:0]
FIND_FREE_IDX	After modifying Valid bits (enabling or invalidating), write this bit to 1 to force HW to update the "First Free Index" registers to account for the changed Valid bits. This HW update process takes 4us. SW can poll SAM_BUSY to determine when the table update is complete. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.
SAM_BUSY	HW is in the process of updating the Source Address table, either in response to a poll indication from the packet processor, or due to SW setting FIND_FREE_IDX=1. In the latter case, SW should poll SAM_BUSY until low before accessing the "First Free Index" registers. Read-only bit.
INVALIDATE_ALL	Writing a 1 to this bit clears all 128 Valid bits. Invalidates the entire table. Write-only bit. Writing 0 to this bit has no effect. Readback value is indeterminate.

All the index-control registers reside within a single 32-bit word, so that a single 32-bit write can store a new checksum in the table, and simultaneously enable, or invalidate, the index. For 32-bit writes, SAM_INDEX_WR must be set to 1 to cause the SAM_CHECKSUM[15:0] to be loaded into the table at the address specified by SAM_INDEX[6:0]. For 32-bit reads, it is necessary first to update SAM_INDEX[6:0] to the desired table index, if it is not already set to the desired index; this requires a write to SAM_INDEX[6:0] with SAM_INDEX_WR=0, so as to not update the table contents. Then, a subsequent 32-bit read will return the table contents at the desired index.

For systems that don't support 32-bit writes, the register fields should be written in the following order:

1. SAM_INDEX[6:0]
2. SAM_CHECKSUM (the 2 bytes can be written in either order)
3. SAM_INDEX_EN (to enable), or SAM_INDEX_INV (to invalidate)

During operation, when a packet is received and the packet processor searches the table for a source address match, only enabled indices within each partition are searched. Invalidated indices are ignored.

To assist software in finding an available index to store the next source address checksum, the packet processor makes available the following 4 read-only registers:

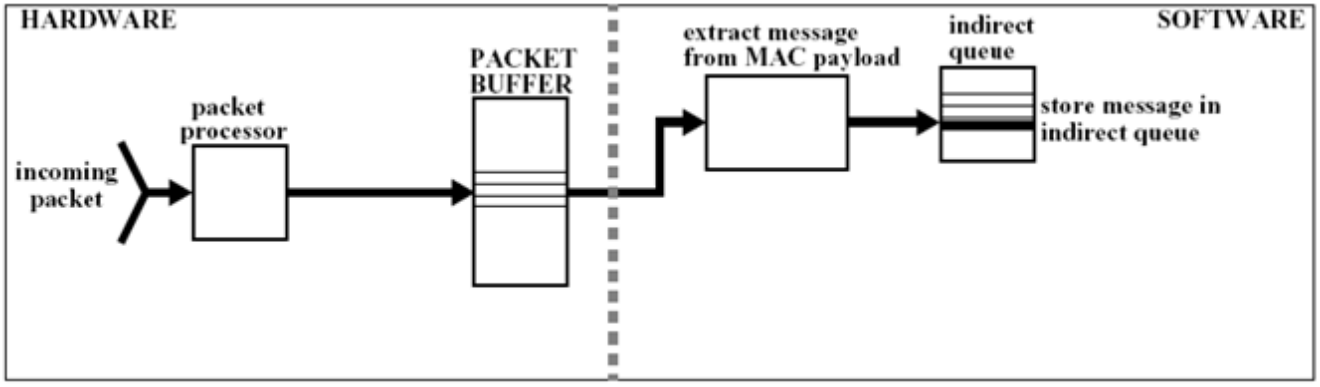
REGISTER NAME	DESCRIPTION
SAP0_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAP0 partition
SAA0_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAA0 partition
SAP1_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAP1 partition
SAA1_1ST_FREE_IDX[6:0]	First non-enabled (invalid) index in the SAA1 partition

Operational Use

The following flow description illustrates how hardware and software interact to utilize the Source Addressing Management feature to facilitate pending-message handling and neighbor table management. This flow assumes both SAP and SAA acceleration are enabled, and Dual PAN mode is in effect. There are 11 sequential steps.

Step 1: Message Packet Reception, Extracting the Message, Indirect Queue Management

A message (data frame) is received by a coordinator. The message is received from one end device, and is intended for another end device. The coordinator's packet processor receives the packet, and stores it in the packet buffer. Coordinator software downloads the packet data from the packet buffer, extracts the message from the MAC payload, and stores the message in its indirect queue.



Step 2: Software computes address checksum

Coordinator software also extracts the destination information (Destination PANID and Destination Address), from the data packet stored in the packet buffer. Software uses this information to compute a 16-bit checksum. This checksum uniquely identifies the sending device, with a high degree of reliability. The checksum is computed according to the following rules:

Destination Addressing Mode	Rule
Mode 2 (short address)	Checksum = (Destination PAN ID + DstAddr[15:0]) % 65536
Mode 3 (long address)	Checksum = (Destination PAN ID + DstAddr[15:0]) % 65536 Checksum = (Checksum + DstAddr[31:16]) % 65536 Checksum = (Checksum + DstAddr[47:32]) % 65536 Checksum = (Checksum + DstAddr[63:48]) % 65536

Step 3: Software populates the address table

Coordinator software chooses an index in the Source Address Table in which it will store the just-computed checksum. If the data packet was received on PAN0, an index in the SAP0 partition should be chosen. Software may consult the register SAP0_1ST_FREE_IDX[6:0] to find the first available index in this partition. If the data packet was received on PAN1, an index in the SAP1 partition should be chosen. Software may consult the register SAP1_1ST_FREE_IDX[6:0] to find the first available index in this partition.

If an index is chosen which is already valid, software should first invalidate the index with the following procedure:

1. Set SAM_INDEX[6:0] to the selected index
2. Set SAM_INDEX_EN=0 and SAM_INDEX_INV=1

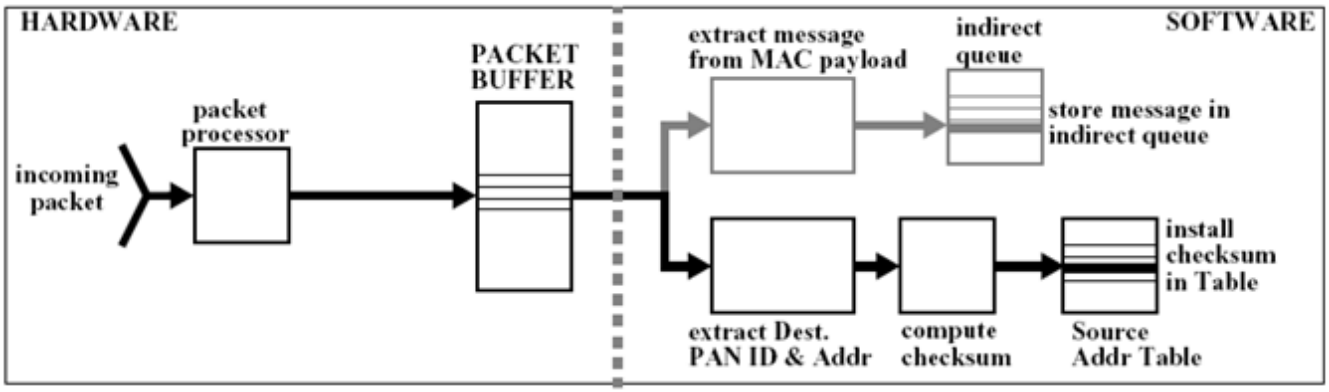
These steps can be performed in a single, 32-bit register write.

To install the source address checksum into the index and enable the index, the following procedure should be used:

1. Program SAM_CHECKSUM[15:0] to the computed checksum
2. Set SAM_INDEX[6:0] to the selected index
3. Set SAM_INDEX_EN=1 and SAM_INDEX_INV=0

These steps can be performed in a single, 32-bit register write.

The index is now enabled, and the packet processor will include it in its search when the next MAC Poll Request is received.



Step 4: Incoming Poll request, Packet Processor parses the Auxiliary Security Header

An incoming packet is received. The packet processor performs routine packet filtering. After determining that the packet is type 'MAC Command', the packet processor must perform additional packet parsing in order to locate the packet's Command Frame Identifier. This is the octet that distinguishes this commandas being of type "data request", from other MAC Command types. For the packet processor, this extra parsing is more complex, because the Command Frame Identifier is part of the MAC payload; this is theonly scenario where the packet processor is required to parse the payload. This is further complicated by the fact that the revision 2006 (and subsequent) of the 802.15.4 wireless MAC standard, includes a variable-length "Auxiliary Security Header", prior to the MAC payload. This header is only present for FrameVersion > 0, and SecurityEnabled = 1. The packet processor must conditionally parse this header in order to locate the start of the MAC payload, where the Command Frame Identifier is located. The packet processor uses 2 subfields of the Frame Control Field to parse the Auxiliary Security Header (A.S.H.): FrameVersion, and SecurityEnabled. The packet processor parses the Auxiliary Security Header (A.S.H.) according to the following table:

Length of Auxiliary Security Header (Octets)	SecurityEnabled (Frame Control Field)	FrameVersion (Frame Control Field)	KeyIdentifierMode (Security Control Field)	Remarks
0	0	00 or 01	-	If !SecurityEnabled, no ASH regardless of FrameVersion
0	1	00	0	If packet is 2003-compliant, no ASH
5 Security Control (1 octet) + Frame Counter (4 octet)	1	01	0x00	ASH present. Length of ASH computed by HW.
6 Security Control (1 octet) +	1	01	0x01	ASH present. Length of ASH computed by HW.

Table continues on the next page...

Table continued from the previous page...

Length of Auxiliary Security Header (Octets)	SecurityEnabled (Frame Control Field)	FrameVersion (Frame Control Field)	KeyIdentifierMode (Security Control Field)	Remarks
Frame Counter (4 octet) + Key Identifier (1 octet)				
10 Security Control (1 octet) + Frame Counter (4 octet) + Key Identifier (5 octet)	1	01	0x2	ASH present. Length of ASH computed by HW.
14 Security Control (1 octet) + Frame Counter (4 octet) + Key Identifier (9 octet)	1	01	0x3	ASH present. Length of ASH computed by HW.
-	-	10 or 11	-	HW will handle these FrameVersion's indentically to FrameVersion 01

The contents of the A.S.H. are irrelevant to the packet processor; only its length is important. This is because the octet which immediately follows the A.S.H. is the Command Frame Identifier. This is all the packet processor needs to know.

Step 5: Hardware extracts the Command Frame Identifier

After parsing the A.S.H. (if present), the next octet in the MAC payload, in accordance with the 802.15.4 standard, is the Command Frame Identifier.

Command frame identifier	Command name	RFD		Subclause
		Tx	Rx	
0x01	Association request	X		5.3.1
0x02	Association response		X	5.3.2
0x03	Disassociation notification	X	X	5.3.3
0x04	Data request	X		5.3.4
0x05	PAN ID conflict notification	X		5.3.5
0x06	Orphan notification	X		5.3.6
0x07	Beacon request			5.3.7
0x08	Coordinator realignment		X	5.3.8
0x09	GTS request			5.3.9
0x0a–0xff	Reserved			—

The packet processor checks to see if the Command Frame Identifier = 0x4. If it is not, packet processing for this frame is complete, the remainder of the packet is received, without inspection by the packet processor, and is stored the packet buffer. If the Command Frame Identifier = 0x4, then this is a data request, and assuming the prior rules-checking and address-checking on this packet have already passed, then the Source Address Matching function will be applied to this packet, continuing in STEP 6.

Step 6: Hardware Computes the Source Address Checksum

The packet processor uses the received Source PAN ID and Source Address for this packet to compute a 16-bit Source Address checksum. This checksum uses precisely the same algorithm that software uses to compute its Address checksum (see STEP 2), except that the *Source* PAN ID and address are used, rather than the Destination PAN ID and address. This checksum uniquely identifies the sending device, with a high degree of reliability. Note if the incoming packet's Frame Control Field has the PANIDCompression subfield asserted, the packet processor will substitute "Destination PAN ID" for "Source PAN ID" in the computation, since Source PAN ID will not be present under these circumstances.

Step 7: Hardware performs "Source Address Present" Table Search

If the MAC poll request was received on PAN0, and SAP0_EN=1 to enable searches of the PAN0 "Source Address Present" partition, the packet processor uses its Source Address checksum to look for a match in the SAP0 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If a single table match is found (i.e., 1 software-computed checksum in SAP0 partition matches the hardware-computed checksum for the incoming packet), this means that there is a message for this requestor in the coordinator's message queue. For a SAP0 match, the packet processor takes these 2 steps:

1. Set SAP0_ADDR_PRESENT=1
2. Set SAP0_MATCH[6:0] to the index which matched

If multiple table matches are found (i.e., more than 1 software-computed checksum in the SAP0 partition matches the hardware-computed checksum for the incoming packet), this means that there are multiple messages for this requestor in the coordinator's message queue. In this case, the packet processor takes the same steps as for a single match, with the SAP0_MATCH[6:0] register being set to the lowest-value matching index (i.e., the smallest number) within the SAP0 partition.

If no source address match is found in the SAP0 table search, the packet processor will set SAP0_ADDR_PRESENT=0.

If, on the other hand, the MAC poll request was received on PAN1, and SAP1_EN=1 to enable searches of the PAN1 "Source Address Present" partition, the packet processor uses its Source Address checksum to look for a match in the SAP1 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If a single table match is found (i.e., 1 software-computed checksum in SAP1 partition matches the hardware-computed checksum for the incoming packet), this means that there is a message for this requestor in the coordinator's message queue. For a SAP1 match, the packet processor takes these 2 steps:

1. Set SAP1_ADDR_PRESENT=1
2. Set SAP1_MATCH [6:0] to the index which matched

If multiple table matches are found (i.e., more than 1 software-computed checksum in the SAP1 partition matches the hardware-computed checksum for the incoming packet), this means that there are multiple messages for this requestor in the coordinator's message queue. In this case, the packet processor takes the same steps as for a single match, with the SAP1_MATCH[6:0] register being set to the lowest-value matching index (i.e., the smallest number) within the SAP1 partition.

If no source address match is found in the SAP1 table search, the packet processor will set SAP1_ADDR_PRESENT=0.

Step 8: Hardware performs "Source Address Absent" Table Search

If the MAC poll request was received on PAN0, and SAA0_EN=1 to enable searches of the PAN0 "Source Address Absent" partition, the packet processor uses its Source Address checksum to look for a match in the SAA0 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If one (or more) table match is found (i.e., a software-computed checksum in SAA0 partition matches the hardware-computed checksum for the incoming packet), this means that the end device which sent the poll request is a valid child of this parent. For a SAA0 match, the packet processor takes these 2 steps:

1. Set SAA0_ADDR_ABSENT=0
2. Set SAA0_MATCH[6:0] to the first index which matched

If *no* source address match is found in the SAA0 table search, the packet processor will set SAA0_ADDR_ABSENT=1.

If, on the other hand, the MAC poll request was received on PAN1, and SAA1_EN=1 to enable searches of the PAN1 "Source Address Absent" partition, the packet processor uses its Source Address checksum to look for a match in the SAA1 partition of the table. Only table entries that are enabled, are compared against (see STEP 3); table indices which are not enabled, are ignored in the search.

If one (or more) table match is found (i.e., a software-computed checksum in SAA1 partition matches the hardware-computed checksum for the incoming packet), this means that the end device which sent the poll request is a valid child of this parent. For a SAA1 match, the packet processor takes these 2 steps:

1. Set SAA1_ADDR_ABSENT=0
2. Set SAA1_MATCH[6:0] to the first index which matched

If *no* source address match is found in the SAA1 table search, the packet processor will set SAA0_ADDR_ABSENT=1.

Step 9: Hardware determines FramePending status

The packet processor maintains an internal flag called **rx_frame_pending**, which is sent to the TX_PACKET block after every received packet, to directly control the state of the FramePending subfield of the Frame Control Field in the upcoming auto-TxAck frame (if enabled with AUTOACK=1).

If the Source Address Management feature is completely disabled, i.e.,

SAP0_EN = SAA0_EN = SAP1_EN = SAA1_EN = 0

then the packet processor will set the **rx_frame_pending flag** to the value of the ACK_FRM_PND register, putting the state of FramePending under direct software control.

Otherwise, if *any* of the following conditions are met, the packet processor will set its **rx_frame_pending** flag to 1:

1. The MAC Poll Request was received on PAN0, SAP0_EN=1, and SAP0_ADDR_PRESENT=1
2. The MAC Poll Request was received on PAN1, SAP1_EN=1, and SAP1_ADDR_PRESENT=1
3. The MAC Poll Request was received on PAN0, SAA0_EN=1, and SAA0_ADDR_ABSENT=1
4. The MAC Poll Request was received on PAN1, SAA1_EN=1, and SAA1_ADDR_ABSENT=1

If none of the conditions are met, **rx_frame_pending** flag will be set to 0.

Step 10: Receive interrupt RXIRQ

The packet processor receives the remainder of the packet, and verifies the FCS. Assuming the CRC check passes, the RXIRQ interrupt is issued. In the coordinator software's RXIRQ Interrupt Service Routine, software can determine if the received packet was a MAC Poll Request by reading the PI bit of the IRQSTS register. If PI=0, the received packet was not a poll request, and no action relating to Source Address Matching need be taken.

Otherwise, if *either* SAP0_ADDR_PRESENT=1 or SAP1_ADDR_PRESENT=1, an auto-TxAck packet is being sent with the FramePending subfield set to 1, so software should follow up by constructing a data packet (or packets) for the requesting end device, with a payload consisting of the message for this device which had been stored in the coordinator's indirect queue.

Also, if *either* SAA0_ADDR_ABSENT=1 or SAA1_ADDR_ABSENT=1, an auto-TxAck packet is being sent with the FramePending subfield set to 1, so software should follow up by constructing a "Leave Message" for the requesting end device.

Step 11: Software disables the spent index

In the final step, coordinator software must disable (invalidate) the index which was just used to locate and send the stored message to the requesting end device. To disable the index:

1. Set SAM_INDEX[6:0] to the selected index
2. Set SAM_INDEX_EN=0 and SAM_INDEX_INV=1

At this point, the disabled index is now an available table entry, which can be used to install a new Address checksum for another message in the indirect queue.

55.4.9.2.3.8.1.1.2.4 Dual PAN Mode

In the past, radio transceivers designed for 802.15.4 applications allow a device to associate to one and only one PAN (Personal Area Network), at any given time. The Packet Processor includes hardware support for a device to reside on two networks simultaneously. In optional Dual PAN mode, the device will alternate between the 2 PAN's, under hardware or software control. Hardware support for Dual PAN operation consists of 2 sets of PAN and IEEE addresses for the device, 2 different channels (one for each PAN), a programmable timer to automatically switch PAN's (including on-the-fly channel-changing) without software intervention, control bits to configure and enable Dual PAN mode, and read-only bits to monitor status in Dual PAN mode. A device can be configured to be a PAN Coordinator on either network, both networks, or neither.

For the purpose of defining a "PAN" in the context of Dual PAN mode, two sets of network parameters are maintained. In this document, "PAN0" and "PAN1" will be used to refer to the 2 PAN's. Each parameter set uniquely identifies a PAN for Dual PAN mode. These parameters are:

PAN0	PAN1
CHANNEL_NUM0	CHANNEL_NUM1
MACPANID0 (16-bit register)	MACPANID1 (16-bit register)

Table continues on the next page...

Table continued from the previous page...

PAN0	PAN1
MACSHORTADDRS0 (16-bit register)	MACSHORTADDRS1 (16-bit register)
MACLONGADDRS0 (64-bit registers)	MACLONGADDRS1 (64-bit registers)
PANCORDNTR0 (1-bit register)	PANCORDNTR1 (1-bit register)

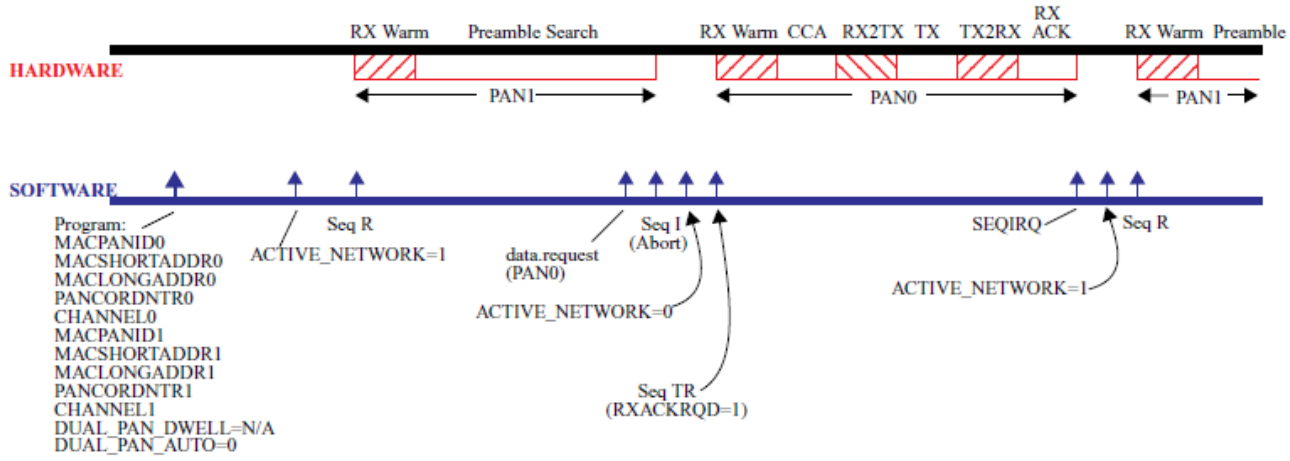
During device initialization, if Dual PAN mode is to be utilized, software will program both parameters sets, to configure the hardware for operation on two networks.

Two modes are available for Dual PAN operation: manual and automatic.

Manual Dual PAN

In manual Dual PAN mode, software controls which PAN is selected for transmission and reception, at all times. Software populates both PAN network parameter-set registers listed above. Software then selects a PAN to begin transmission or reception. A single control bit **ACTIVE_NETWORK**, selects the PAN. To select PAN0, **ACTIVE_NETWORK** should be set to 0; to select PAN1, **ACTIVE_NETWORK** should be set to 1. Software then initiates an autosequence by writing the appropriate value to XCVSEQ (PHY_CTRL1 Register). At any point, software can elect to switch to the alternate PAN. To do so, software aborts the sequence (if one is active) by writing XCVSEQ=IDLE. Software then toggles **ACTIVE_NETWORK**, and restarts a new sequence with XCVSEQ. Software is responsible for aborting and initiating sequences. Hardware is responsible for warming up the transceiver on the selected channel (CHANNEL_NUM0 or CHANNEL_NUM1), and performing address-filtering on the selected network's address parameters (PANCORDNTR, MacPanID, Short Address, and/or IEEE address). Software overhead is significantly reduced in this mode, since there is no need to re-program all of the PAN-select registers every time the PAN is toggled in Dual PAN mode; only a single bit (**ACTIVE_NETWORK**) need be written. For manual Dual PAN operation, **DUAL_PAN_AUTO** should be set to 0.

The following diagram depicts Dual PAN operation in manual mode. In this example, software populates both PAN parameters sets during initialization. In this example, software elects to initiate packet reception on PAN1, so **ACTIVE_NETWORK** is set to 1. Software initiates the Sequence R by writing to XCVSEQ. Hardware warms up the receiver on CHANNEL_NUM1 and enters preamble-search. In this example, no packet is received, but a data request from higher-level software requires a switch to PAN0. Software responds by aborting the reception by writing Sequence I to XCVSEQ. Software then sets **ACTIVE_NETWORK** to 0 to select PAN0. In the example, software initiates a Sequence TR. Hardware warms up the receiver for CCA on CHANNEL_NUM0, performs CCA (channel is idle), warms down the receiver, and warms up the transmitter on CHANNEL_NUM0, transmits the requested packet, warms down the transmitter and warms up the receiver again (as per the autosequence) on CHANNEL_NUM0, to receive the acknowledge packet. The acknowledge packet is received and the autosequence completes successfully with RXIRQ and SEQIRQ. Software elects to resume packet reception on PAN1 (which was in progress before the data request). Software toggles **ACTIVE_NETWORK** back to 1, and re-initiates Sequence R.



Auto Dual PAN

In automatic Dual PAN mode, hardware is responsible for alternating the selected PAN, at a pre-programmed interval. Software programs a "PAN dwell time", which determines the amount of time during which the hardware will receive on a given PAN. This is done by programming the **DUAL_PAN_DWELL** register. Software sets the **DUAL_PAN_AUTO** bit to 1 for automatic Dual PAN operation, and sets **ACTIVE_NETWORK** to correspond to the PAN on which it desires packet reception to begin. Software then initiates a Sequence R. Hardware warms up the receiver on the selected PAN, and begins preamble search. When the dwell time expires, assuming preamble and SFD have not been detected, hardware will switch to the alternate PAN. This includes an on-the-fly channel change. Once switched to the alternate PAN, hardware will use the address filtering parameters (PANCORDNTR, MacPanID, Short Address, and/or IEEE address) programmed for the alternate PAN. Once the dwell time expires on the alternate PAN, assuming preamble and SFD have not been detected, hardware will switch back to the original PAN. This process will repeat indefinitely, until one of the following occurs:

1. A packet is successfully received (address match and CRC valid).
2. Software aborts the Sequence R
3. A PLL unlock automatically aborts the Sequence R (if unlock aborting is enabled)

If the hardware PAN dwell timer expires *during* a packet reception (i.e., preamble and SFD have been detected during a Sequence R), the PAN-switch will be delayed until the packet is completely received. If the packet is received successfully (address match and CRC valid), the receive sequence will terminate with RXIRQ and SEQIRQ (if an auto-Acknowledge was requested and enabled, the sequence will complete after the Acknowledge packet was transmitted, and TXIRQ will also be set, in addition to RXIRQ and SEQIRQ); the PAN switch will occur at this point, after the Sequence has returned to IDLE.

If the received packet fails FCS (CRC check), or, if the packet was not intended for the device (address mismatch), the hardware will toggle the PAN, and return to preamble-search on the new PAN.

If the hardware PAN dwell timer expires during a state *other* than RX preamble search during Sequence R (i.e., if expiration occurs during TX, CCA, ED, Sequence TR, or any warmup or transitional state), PAN-switch will be delayed until either:

1. Preamble Search state is reached during a Sequence R (not Sequence TR)
2. Sequence returns to IDLE.

At any point, software can elect to interrupt the automatic Dual PAN operation, by aborting the sequence with a XCVSEQ=IDLE, and the setting the **DUAL_PAN_AUTO** bit to 0. In response, hardware will freeze the current state of the dwell timer, and capture the currently-selected PAN, so that hardware-controlled PAN alternation can resume where it left off, later. Software can then tend to that tasks that caused the Dual PAN interruption. When software wants to resume Dual PAN operation, it then sets **DUAL_PAN_AUTO**=1, and re-initiates Sequence R. Hardware will resume the dwell timer from the point it was frozen earlier, on the same PAN, and will warmup the receiver to receive on the PAN which was being monitored before

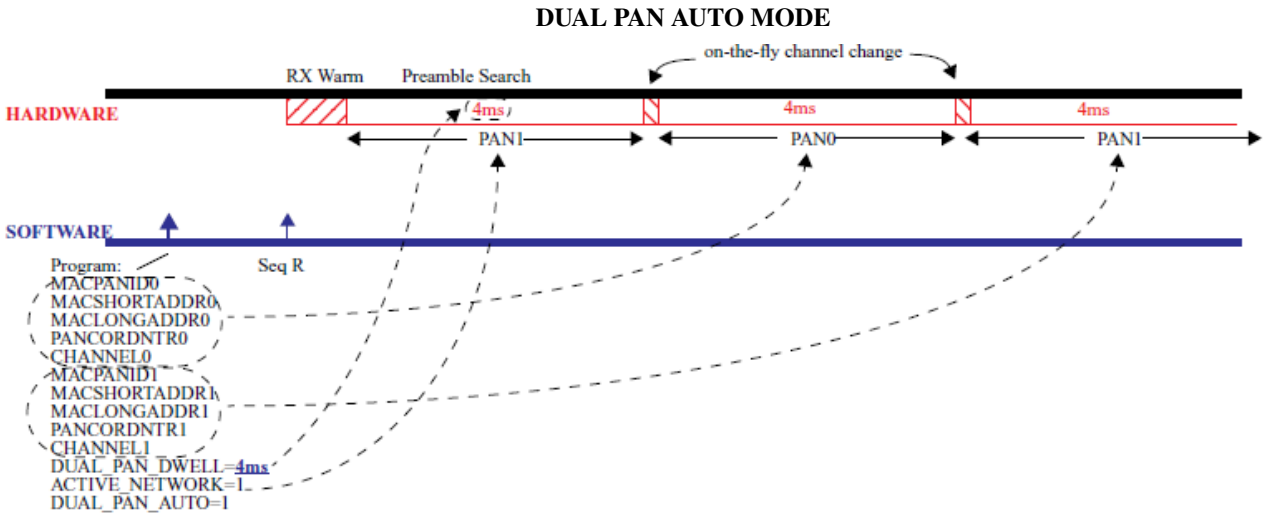
the interruption. Hardware will dwell on that PAN until the timer expires, and then switch PAN's (assuming no preamble and SFD detected, subject to the conditions listed above). Software can always determine the selected PAN at any time, in any mode, by reading **CURRENT_NETWORK** bit. If **DUAL_PAN_AUTO**=0, then **CURRENT_NETWORK** = **ACTIVE_NETWORK**. If **DUAL_PAN_AUTO**=1, then **CURRENT_NETWORK** is controlled by hardware.

To initiate auto Dual PAN operation for the first time, or to reset the dwell timer to its programmed value and allow software to select the initial PAN for auto Dual PAN operation, software should program the desired initial PAN to **ACTIVE_NETWORK** (0 for PAN0, 1 for PAN1), and then program the desired dwell time into the **DUAL_PAN_DWELL** register. Hardware always responds to a write to **DUAL_PAN_DWELL**, by resetting the dwell timer to its programmed value, and selecting the initial PAN for auto Dual PAN operation, based on the state of **ACTIVE_NETWORK**. A write to **DUAL_PAN_DWELL**, always re-initilizes the DWELL TIMER to the programmed value. If a write to **DUAL_PAN_DWELL** occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to **DUAL_PAN_DWELL** occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the autosequence warmup. Under all circumstances, **DUAL_PAN_AUTO** must be asserted to allow the DWELL TIMER to count. If **DUAL_PAN_AUTO**=0, the DWELL TIMER will freeze and hold its state, until **DUAL_PAN_AUTO**=1. However, a write to **DUAL_PAN_DWELL** will still initialize the DWELL TIMER to its programmed value, regardless of the state of **DUAL_PAN_AUTO**.

At any point during auto Dual PAN mode, software can ascertain the PAN currently being serviced by hardware, by reading the **CURRENT_NETWORK** bit. Software can also determine the time-remaining in the current dwell, by reading the **DUAL_PAN_REMAIN** Register.

In auto Dual PAN mode, software retains responsibility for initiating and aborting sequences. There is no provision for hardware-initiated sequences.

The following diagram depicts Dual PAN operation in automatic mode. In this example, software populates both PAN parameters sets during initialization. In this example, software wants to initiate packet reception in auto Dual PAN mode, on PAN1, so **ACTIVE_NETWORK** is set to 1. For auto Dual PAN operation, software programs the desired PAN-switching interval into **DUAL_PAN_DWELL** register, in this case, 4ms. Software initiates the Sequence R by writing to XCVSEQ. The dwell timer begins counting as soon as the receiver warmup begins. Hardware warms up the receiver on CHANNEL_NUM1 (since PAN1 was selected as the initial PAN in this example), and begins preamble search. If a packet is received during this interval, hardware applies PAN1 address filtering. In this example, no preamble is detected at the point at which the dwell timer expires. So, at this point, hardware executes a PAN-switch to PAN0. which includes an on-the-fly channel change. The on-the-fly channel change requires approximately 93us and no preamble detection is possible during this "blind spot". (For more details on how the hardware executes the Dual PAN on-the-fly channel change, see Chapter GSM Sequence Manager, Section Dual PAN Mode.) After the PAN switch, hardware resumes preamble search on CHANNEL_NUM0, and applies PAN0 address filtering if a packet is received. In this example, no preamble is detected at the point at which the dwell timer expires. So, at this point, hardware performs a PAN-switch back to PAN1. And so on and so forth.



Two-PAN-one-channel

A special case occurs if a device is operating in Dual PAN mode (either Auto or Manual mode), and both PAN's occupy the same channel; in other words, CHANNEL_NUM0=CHANNEL_NUM1. In this case, hardware will detect this condition automatically, and apply both sets of address-filtering parameters simultaneously. Hardware will provide Data Indication if the received packet matches either the PAN0 address parameter set {**MACPANID0, MACSHORTADDRS0, MACLONGADDRS0, PANCORDNTR0**}, or the PAN1 address parameter set {**MACPANID1, MACSHORTADDRS1, MACLONGADDRS1, PANCORDNTR1**}. At Data Indication, two status bits can be queried by software to quickly determine on which PAN the packet was received, **RECD_ON_PAN0** or **RECD_ON_PAN1**. In the "Two-PAN-one-channel" scenario, it is possible for a packet to satisfy both sets of addressing parameters (PAN0 and PAN1). When this happens, both **RECD_ON_PAN0** and **RECD_ON_PAN1** will be set. In the "Two-PAN-one-channel" scenario, the **CURRENT_NETWORK** bit should be ignored, since both networks are simultaneously active.

If Dual PAN mode is not to be utilized, **ACTIVE_NETWORK** should be maintained at 0 (default) to select the PAN0 parameters for transceiving. In single PAN operation, the PAN1 parameter registers need not be programmed, and can be used as scratchpad. CHANNEL_NUM1 should not be programmed, and should instead be left in its default state, if Dual PAN mode is not in use.

Dual PAN Channel Override

In Dual PAN mode, in case there is a need to generate a frequency which may be offset from the 16 prescribed 5MHz-spaced channels, to, for example, avoid interference on one of the Dual PAN channels, a method has been provided to do that, by designating one of the two PAN channels to use the transceiver's set of direct frequency-programming registers, instead of CHANNEL_NUMx. Programming the direct frequency-programming registers – integer, numerator, and denominator, allows an RF frequency to be selected with much more precision than the 5MHz granularity of the 802.15.4 mapped-channel registers, CHANNEL_NUM0 and CHANNEL_NUM1. This is referred to as "Dual PAN Channel Override". For more details on this feature, see Chapter GSM Sequence Manager, Section Dual PAN Channel Override.

55.4.9.2.3.8.1.1.2.5 Promiscuous Mode

Functional description

Promiscuous Mode are defined in PAN mode and FAN mode.

The 802.15.4 Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs) defines the following behavior for Promiscuous Mode operation:

> "In promiscuous mode, the MAC sublayer shall pass all frames received after the first filter directly to the upper layers without applying any more filtering or processing."

Where first level filtering is described as:

> "For the first level of filtering, the MAC sublayer shall discard all received frames that do not contain a correct value in their FCS field in the MFR (see 7.2.1.9)"

Promiscuous mode allows a device to receive and process all 802.15.4 frames which pass FCS check (CRC), regardless of whether or not they are addressed to the device. This allows a device to receive and process all valid traffic on a network. In Promiscuous mode, the Packet Processor will provide a Data Indication on all such packets. This capability allows a simple network sniffer to be constructed, which can display comprehensive packet data for all packets transmitted on a network within receiving range of the device.

In PAN mode, promiscuous mode will bypass the filtering rules described in Packet Filtering Detail.

In FAN mode, promiscuous mode will bypass the filtering rules described in FAN Mode Filter.

Since a device operating in promiscuous mode is designed to do so discreetly, automatic acknowledgement of all received packets is inhibited, regardless of the state of the Ack Request field in the Frame Control Field of the received packet; otherwise, the sniffing device would be acknowledging received frames for which it was not the addressee.

Special Handling for Auto Receive Acknowledge Frame

The Packet Processor features an autosequence to automatically receive, and verify, Acknowledge Frames after a Transmit frame has been sent by the device. This is accomplished by programming a Sequence TR with RXACKRQD=1. During normal operation, these receive Acknowledge Frames are not stored in the Packet Buffer. They are merely checked for matching Sequence Number and Frame Version, with a Data Indication issued for a valid match on both. Promiscuous Mode should therefore **NOT** be used with Sequence TR with RXACKRQD=1.

55.4.9.2.3.8.1.1.2.6 Provisions to Enable Software Support 802.15.4-2015

Legacy frame types as specified in 802.15.4, such as LLDN Frame Type, Multipurpose Frame Type, , as well as the Extended Frame Type, Fragment Frame Type, modify the format and definition of the Frame Control Field in such a way as to render parsing of the MHR by the packet processor impossible, without major upgrades to the hardware. For example, the packet processor hardware does not support 8-bit (simple) addressing, nor Sequence Number Suppression.

As in interim solution, provisions have been added to the packet processor to allow these new packet variations to be recognized and conditionally admitted (Data Indication to software). Three new Frame Type control bits, and a modification of the Frame Version Filtering bits, are included in the Packet Processor, and affect hardware packet parsing as indicated in the following table:

REGISTER	DEFINITION
FRM_VER_FILTER[3:0]	Frame Version selector. The incoming packet's Frame Control Field is parsed to obtain the FrameVersion subfield, and that value is compared against this register, in accordance with the following: xxx1: Accept received packets with FrameVersion=00 xx1x: Accept received packets with FrameVersion=01 x1xx: Accept received packets with FrameVersion=10 1xxx: Accept received packets with FrameVersion=11 These filtering rules apply to Beacon, Acknowledge, Data, and MAC Command Frame Types only, since these frame types require a 2-octet Frame Control Field which embeds a 2-bit FrameVersion subfield.
LLDN_FT	1: LLDN frame type enabled (Frame Type 4) 0: reject all LLDN frames
MULTIPURPOSE_FT	1: Multipurpose frame type enabled (Frame Type 5) 0: reject all Multipurpose frames
EXTENDED_FT	1: Extended frame type enabled (Frame Type 7) 0: reject all Extended frames
FRAGMENT_FT	1: Fragment/Frak frame type enabled (Frame Type 6) 0: reject all Fragment/Frak frames

The Extended Frame Type does not comply with the General MAC Frame Format, which is shown in the following diagram:

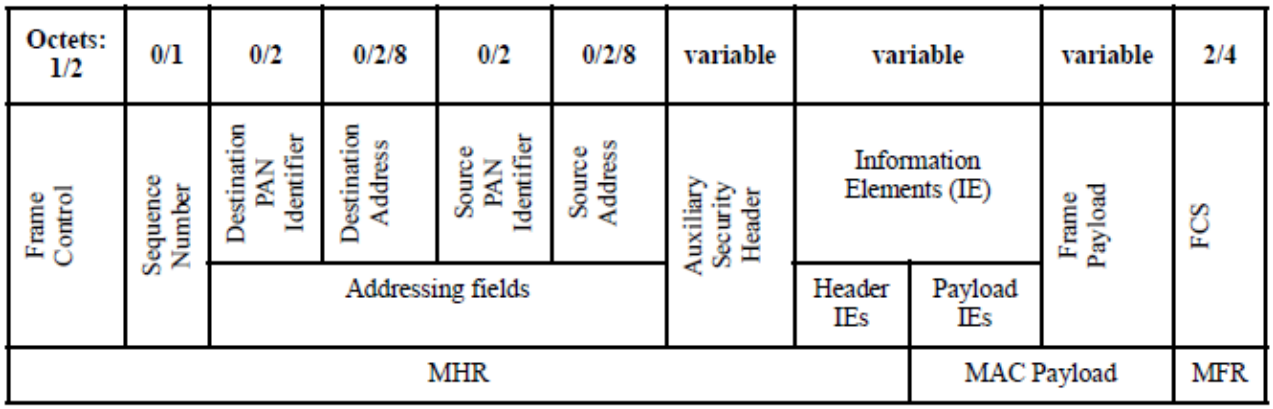


Figure 86—General MAC frame format

Figure 454. General MAC Frame Format

Instead, the Extended Frame has its own unique format, which is shown in the following diagram:

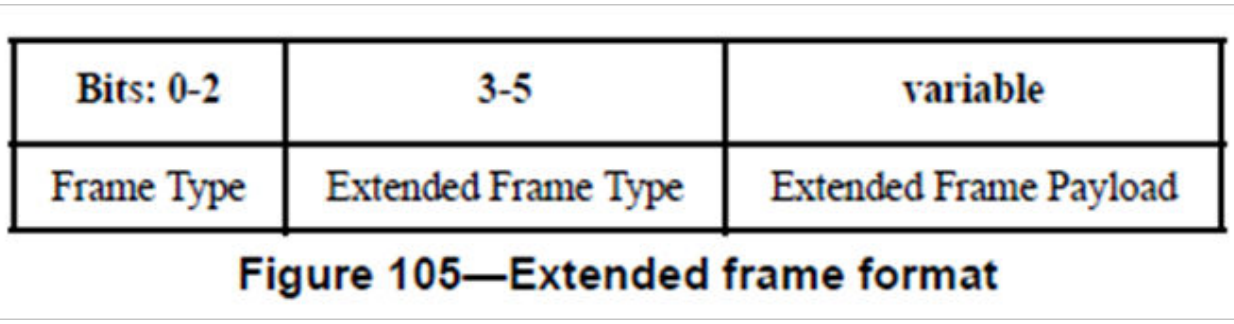


Figure 105—Extended frame format

Figure 455. Extended Frame Format

Because the Extended Frame Payload, which is not defined in the 802.15.4-2015 revision to the standard, would be intended to contain and fully specify the FCS, the Packet Processor will not be able to verify FCS on received packets with Frame Type Extended. FCS checking will need to be implemented in software.

Note: If the EXTENDED_FT bit is set to enable Data Indication on packets of Frame Type Extended, software should expect a substantial increase in the frequency of notifications, as a consequence of the lack of hardware verification of FCS, which will have implications for software workload as well as power consumption in the SoC.

When the packet processor receives any of the legacy frames, if the corresponding control bit is set, the hardware will skip Stage 2 filtering, and merely verify FCS. If FCS passes, the packet processor will notify software of the packet, without inspection of the addressing fields or payload. For such packets, software will be responsible for parsing the MHR and payload sections of the frame. If the corresponding control bit is not set, the packet processor will reject the received packet and there will be no Data Indication.

When the packet processor receives a packet of Frame Type Extended, if the EXTENDED_FT bit is set, the hardware will skip Stage 2 filtering, and after the entire packet is received, the packet processor will merely notify software of the packet, without inspection of the Extended Frame Type bits or the Extended Frame Payload. For such packets, software will be responsible for parsing the Extended Frame Type and the Extended Frame Payload sections of the frame. If the corresponding control bit is not set, the packet processor will reject the received packet and there will be no Data Indication.

Like all existing frames prior to 802.15.4-2015, a received frame may require an acknowledgement. In the case of the 802.15.4-2015 frames, the required timing for the acknowledge frame transmission, has been modified. The acknowledgement no longer immediately follows the packet reception. For this reason, upon reception of any of these 2015-compliant packets, the packet processor will cancel any auto-ACK transmission, regardless of the state of the PHY_CTRL[AUTOACK] bit and AR

field of the received packet's Frame Control Field. Software will be responsible for constructing the acknowledgment packet and transmitting it at the appropriate time using a Sequence T.

To help expedite software processsing of the 2015-compliant Frame Types and Versions, additional status bits have been added to the 802.15.4 register set, which are described in the following table:

STATUS BIT	RESIDES IN REGISTER	DEFINITION
ENH_PKT_STATUS	RX_FRAME_FILTER	1: The last packet received was 2015-compliant (SW should query the RX_FRAME_FILTER register for additional status bits) 0: The last packet received was not 2015-compliant
BEACON_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Beacon with FrameVersion 2 0: The last packet received was not a FrameType Beacon with FrameVersion 2
DATA_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Data with FrameVersion 2 0: The last packet received was not a FrameType Data with FrameVersion 2
ACK_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Acknowledge with FrameVersion 2 0: The last packet received was not a FrameType Acknowledge with FrameVersion 2
CMD_FV2_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType MAC Command with FrameVersion 2 0: The last packet received was not a FrameType MAC Command with FrameVersion 2
LLDN_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType LLDN 0: The last packet received was not a FrameType LLDN
MULTIPURPOSE_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Multipurpose with FrameVersion 2 0: The last packet received was not a FrameType Multipurpose
EXTENDED_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Extended 0: The last packet received was not a FrameType Extended
FRAGMENT_RECD	RX_FRAME_FILTER	1: The last packet received was a FrameType Fragment 0: The last packet received was not a FrameType Fragment

55.4.9.2.3.8.1.1.2.7 FAN Mode Filter

The Packet Processor supports to filter the DATA and ACK frame in FAN standard which is based on 802.15.4-2015. Only Data and Enhanced Acknowledge frames are used in FAN mode.

The following table summarizes the frame format used in FAN.

	Frame Control, Octets: 2											1	8	2	8	6	Variable	4
	Bit:0-2	3	4	5	6	7	8	9	10-11	12-13	14-15							
	Frame Type	Security	Frame Pending	AR	PAN ID Compression	Reserved	Sequence Number Suppression	IEs Present	Destination Addressing Mode	Version	Source Addressing Mode	Sequence Number	Destination Address	Source PAN ID	Source Address	ASH	IE	FCS
PA	1	0	0	0	0	0	1	1	0	2	3	N	N	Y	Y	N	Y	Y
PAS	1	0	0	0	1	0	1	1	0	2	3	N	N	N	Y	N	Y	Y
PC	1	1	0	0	0	0	1	1	0	2	3	N	N	Y	Y	Y	Y	Y
PCS(same as PA)	1	0	0	0	0	0	1	1	0	2	3	N	N	Y	Y	N	Y	Y
unicast ULAD	1	1	0	0/1	1	0	0/1	1	3	2	0/3	Y/N	Y	N	Y/N	Y	Y	Y
broadcast ULAD	1	1	0	0	0	0	0	1	0	2	3	Y	N	Y	Y	Y	Y	Y
Acknowledgement	2	1	0	0	1	0	1	1	3	2	3	N	Y	N	Y	Y	Y	Y
EAPOL	1	0	0	0	1	0	0	1	3	2	3	Y	Y	N	Y	N	Y	Y

Figure 456. Frame Type in FAN

The Packet Processor checks the following rules:

1. Auto-RxAck Checking (in Sequence TR)
    a. Received Destination Address == Device's Address
    b. Received Source Address = Transmitted Destination Address
    c. Received Frame Control filed is same as WiSUN spec
2. Data or ACK frame format check
    a. Received Destination Address == Device's Address
    b. Frame Version field should be b10
    c. IEs Present field should be 1
    d. Frame Type should be DATA or ACK

55.4.9.2.3.8.1.1.2.8 Appendix A

FILTERFAIL CODES

A description of the FILTERFAIL_PAN codes can be found in the following table.

FILTERFAIL CODE_PAN BIT	REASON FOR FILTERFAIL
[0]	Fails Stage 1 Frame Length Checking (FL < 5) **Note:** FL < 3 will not generate an SFD, so this bit will not be set
[1]	Fails Stage 1 Section 7.2.1.1.6 or Section 7.2.1.1.8 Checking (DST_ADDR_MODE or SRC_ADDR_MODE = 1)
[2]	Fails Stage 1 Section 7.2.1.1.5 Checking (Illegal PAN_ID_COMPRESSION Usage)
[3]	Fails Stage 1 Frame Version Checking
[4]	Fails Stage 2 Auto-RxAck Checking (Illegal Ack Frame Format in Sequence TR)
[5]	Fails Stage 2 Frame Type Checking (Incorrect Frame Filter Bit setting)
[6]	Fails Stage 2 Frame Length Checking (Illegal Beacon, Data, or Cmd FL)
[7]	Fails Stage 2 Addressing Mode Checking (Illegal Addressing Mode for Beacon, Data, OR Cmd)
[8]	Fails Stage 2 Sequence Number Matching (Sequence TR Only)
[9]	Fails Stage 2 PAN ID or Address Checking (Beacon, Data, or Cmd)

A description of the FILTERFAIL_FAN codes can be found in the following table.

FILTERFAIL CODE_FAN BIT	REASON FOR FILTERFAIL
[0]	Fails Stage 2 Auto-RxAck Checking (Illegal Ack Frame Format in Sequence TR)
[1]	Fails Stage 2 Received frame has Destination Address != Device's Address or Frame Version field != b10 or IEs Present field != 1 or Frame Type != DATA or ACK

55.4.9.2.3.8.1.1.2.9 Appendix B

LENIENCY BITS

The Packet Processor performs filtering on all received packets, in order to determine whether the packet is intended for the device. The packet filtering is based on rules. In case any of the packet filtering rules need to be overriden, a 45-bit "leniency register" has been provided. When the leniency register is programmed to its default value (0), all hardware packet filtering rules are in effect, and if an incoming packet violates any rule, a "Filter Fail" will occur (packet will be rejected). When a given leniency register bit is asserted, the packet filtering rule assigned to that bit will not be in effect, and if any incoming packet violates that rule (but no other rules), then a "Filter Fail" will not occur, the packet will not be rejected, the packet will be treated as "intended for the device", and software will be notified of the incoming packet. The table below shows the assignment of leniency bits to packet filtering rules.

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[0]	Override Stage 1 Frame Length Checking (FL < 5)
leniency[1]	Override Stage 1 Section 7.2.1.1.6 Checking (DST_ADDR_MODE = 1)
leniency[2]	Override Stage 1 Section 7.2.1.1.8 Checking (SRC_ADDR_MODE = 1)
leniency[3]	Override Stage 1 Section 7.2.1.1.5 Checking (Illegal PAN_ID_COMPRESSION Usage)
leniency[4]	Override Stage 2 Auto-RxAck Checking (Illegal Ack Frame Format in Sequence TR)
leniency[5]	Override Stage 2 Frame Length Checking (Illegal Beacon, Data, or Cmd FL)
leniency[6]	Override Stage 2 Beacon Frame Address Mode Violations
leniency[7]	Override Stage 2 Data Frame Address Mode Violations
leniency[8]	Override Stage 2 MAC Command Frame Address Mode Violations
leniency[9]	Override Stage 2 Sequence Number Matching
leniency[10]	Override Stage 2 Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_SHORT (Beacon Only)
leniency[11]	Override Stage 2 Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_SHORT (Data and MAC Command Only)
leniency[12]	Override Stage 2 Dst PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_NONE (Data and MAC Command Only)

Table continues on the next page...

Table continued from the previous page...

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[13]	Override Stage 2 Dst Short Addr Filter for DST_ADDR_MODE_SHORT/ SRC_ADDR_MODE_NONE (Data and MAC Command Only)
leniency[14]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Beacon Only)
leniency[15]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[16]	Override Stage 2 Dst Short Addr Filter for DST_ADDR_MODE_SHORT/ SRC_ADDR_MODE_SHORT/PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[17]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[18]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESS ION (Data and MAC Command Only)
leniency[19]	Override Stage 2 Dst Addr Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[20]	Override Stage 2 Src PAN ID Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[21]	Override Stage 2 Src PAN ID Filter for DST_ADDR_MODE_NONE/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[22]	Override Stage 2 Dst PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_NONE (Data and MAC Command Only)
leniency[23]	Override Stage 2 Dst Long Addr Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_NONE (Data and MAC Command Only)
leniency[24]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ PAN_ID_COMPRESSION (Beacon Only)
leniency[25]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[26]	Override Stage 2 Dst Long Addr Filter for DST_ADDR_MODE_SHORT/ SRC_ADDR_MODE_LONG/PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[27]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Beacon Only)
leniency[28]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ PAN_ID_COMPRESSION (Data and Mac Command Only)
leniency[29]	Override Stage 2 Dst Long Addr Filter for DST_ADDR_MODE_LONG/ SRC_ADDR_MODE_SHORT/PAN_ID_COMPRESSION (Data and Mac Command Only)
leniency[30]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[31]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)

Table continues on the next page...

Table continued from the previous page...

LENIENCY BIT	PACKET FILTERING RULE OVERRIDDEN
leniency[32]	Override Stage 2 Short Addr Filter for DST_ADDR_MODE_SHORT/SRC_ADDR_MODE_LONG/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[33]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Beacon Only)
leniency[34]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSI ON (Data and MAC Command Only)
leniency[35]	Override Stage 2 Long Addr Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_SHORT/ NO_PAN_ID_COMPRESSION (Data and MAC Command Only)
leniency[36]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_LONG (Beacon Only)
leniency[37]	Override Stage 2 PAN ID Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_LONG (Data and MAC Command Only)
leniency[38]	Override Stage 2 Long Addr Filter for DST_ADDR_MODE_LONG/SRC_ADDR_MODE_LONG (Data and MAC Command Only)
leniency[39]	Allow an auto-TxAck frame to be sent, after a receive frame which has all of the following parameters: 1. Destination PAN ID = Broadcast (0xFFFF) 2. Destination Address = !Broadcast (not 0xFFFF) 3. Destination Address Mode = Short Nominally, the SEQ_MGR inhibits auto-TxAck on such frames.
leniency[40]	Override FAN mode Ack frame filter for Received Destination Address == Device's Address
leniency[41]	Override FAN mode Ack frame filter for Received Source Address = Transmitted Destination Address
leniency[42]	Override FAN mode Ack frame filter for Received Frame Control filed is same as WiSUN spec
leniency[43]	Override FAN mode filter for Data frame Received Destination Address = Device's Address
leniency[44]	Override FAN mode filter for Frame Version field should be b10, IEs Present field should be 1 and Frame Type should be DATA or ACK

55.4.9.2.3.8.2 FCP(Frame delimiter before CRC Protocol) Protocol Support

The FCP package structure and its mapping relationship with the GENLL package structure is shown in Figure 457.

The Device Address, Control and Data Payload are mapping to PAYLOAD. User should set H0_SZ=0,H1_SZ=0, LENGTH_SZ=0, and use LENGTH_ADJ to inform the length of (Device Address + Control + Data Payload + CRC-CCITT). Standard fixed sized packets have a total length of 18bytes. Advanced Burst packets can be 19, 27 or 35 bytes. The sizes for all packets are pre-determined and will be specified by the FCP stack prior to each Tx/Rx.

Bit ordering when defining packets conforms to a Big Endian approach. First bit transmitted over the air is the MSB. So register LENGTH_BIT_ORD should set to 1.

Pre-amble is either 0xAA or 0x55 depending on the MSB of the Network Address. If MSB of Network Address is 1, preamble is 0xAA, otherwise the preamble is 0x55.

Network Address is programmable by FCP stack and transmitted in big-endian byte-order. This is the sync word of the RF packet in which corresponding FCP receivers are configured to sync on when trying to receive FCP packets.

Device Address specified and by FCP stack. Address configurable by host interface.

Control byte specified and controlled by FCP stack. Packet sequencing control.

Data payload specified and controlled by FCP stack. Data contents altered by host interface.

CRC-CCITT of packet calculated from Network Address (inclusive) to Data Payload (inclusive). Initial value is 0xFFFF with polynomial (X^16+X^12+X^5+1). CRC is to be transmitted in big-endian byteorder. **CRC_START_BYTE[2:0] in RBME should set to 0.**

Whitening is done to avoid long sequences of 0s or 1s and can be performed by the FCP stack SW or the HW (if it is capable). Whitening is perfomed by XORing against a pseudo-random number sequence generated from a 9-bit pseudo-random number generator with the following polynomial X^9+X^5+X^0.

- Whitening is only applied to the Device Address, Control and Data Payload. The other parts of the packet are not whitened. **So WHITEN_START[1:0] in RBME should set to 0x3.**
- Whitening is performed for any packet that is larger than the standard 18 byte packet (any packet where the Data Payload is larger than 8 bytes).
- Whitening is performed prior to CRC Generation and dewhitening is performed after the CRC is checked.

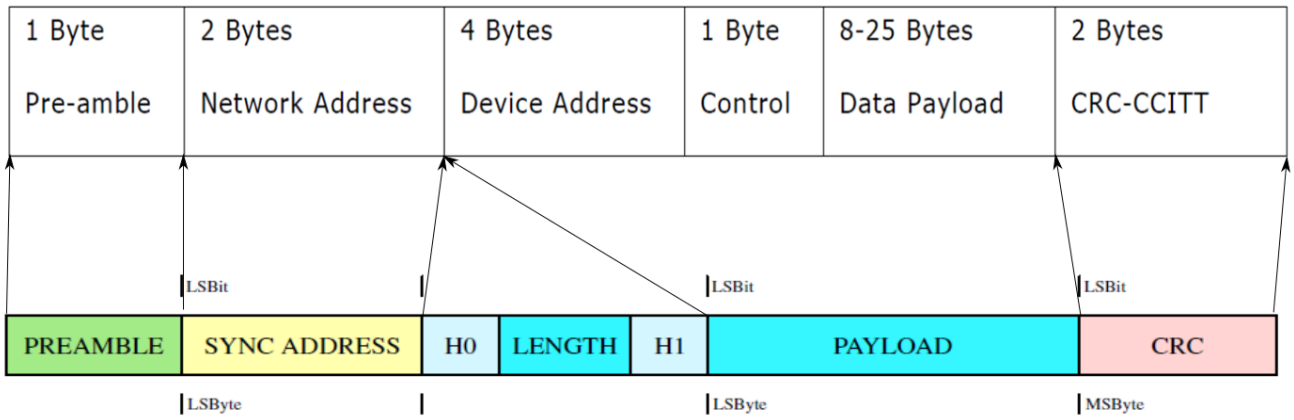


Figure 457. FCP Packet Format and Mapping to Generic Link Layer Format

Table 488 shows the register configuration for the FCP emulation.

Table 488. Link Layer Register Configuration for FCP Emulation

Register	Value
GENLL_MODE[3:0]	0x6 - FCP Mode
PREAMBLE_SZ[8:0]	0x000 - 1 octet preamble
PREAMBLE_SEL[2:0]	000b - The controller hardware selects the preamble pattern based on the first transmitted bit of Network Address, such that the last bit of preamble is the opposite polarity from the first bit of Network Address, forcing a bit transition at this boundary.
GEN_PREAMBLE[7:0]	Not used
NTW_ADR_SZ[1:0]	01b - Network Address 0/1/2/3 requires a 16-bit correlation
SYNC_ADDR_SZ[1:0]	01b - Network Address is 16-bit

Table continues on the next page...

Table 488. Link Layer Register Configuration for FCP Emulation (continued)

Register	Value
H0_SZ[4:0]	0
LENGTH_SZ[4:0]	0
H1_SZ[4:0]	0
LENGTH_BIT_ORD	0 - MS Bit First
LENGTH_ADJ[10:0]	Device Address + Control + Data Payload + CRC-CCITT
AA_PLAYBACK_CNT	1 - Access Address will play back to Link Layer
RXEN_DLY[9:0]	0

55.4.9.2.3.8.3 Bluetooth LE Protocol Support

- Bluetooth LE 5.0 packet format for the LE uncoded PHY

  The following packet format is defined for the LE Uncoded PHYs (LE 1M and LE 2M) and is used for both advertising channel packets and data channel packets. This packet format is shown in Figure 458. Each packet consists of four mandatory fields. The mandatory fields are Preamble, Access Address, PDU, and CRC.

  When a packet is transmitted on either the primary or secondary advertising channel, the PDU shall be the Advertising Channel PDU as defined in Figure 460. When a packet is transmitted on the data physical channel, the PDU shall be the Data Channel PDU as defined in Figure 462.

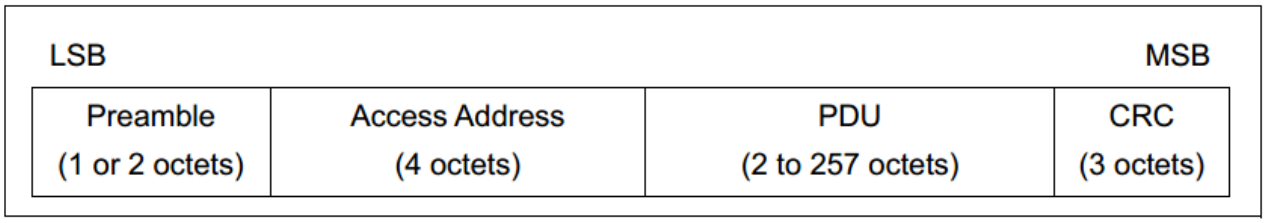


Figure 458. Link Layer packet format for the LE Uncoded PHYs

- Bluetooth LE 5.0 packet format for the LE coded PHY

  The following packet format is defined for the LE Coded PHY and is used for both advertising channel packets and data channel packets. This packet format is shown in Figure 459 .

  Each packet consists of the Preamble, FEC block 1, and FEC block 2. The Preamble is not coded. The FEC block 1 consists of three fields: Access Address, Coding Indicator (CI), and TERM1. The CI field determines which coding scheme is used for FEC block 2. The FEC block 2 consists of three fields: PDU, CRC, and TERM2.

  When a packet is transmitted on either the primary or secondary advertising channel, the PDU shall be the Advertising Channel PDU as defined in Figure 460. When a packet is transmitted on the data physical channel, the PDU shall be the Data Channel PDU as defined in Figure 462.

  When TX, RBME will pad CI, TERM1 and TERM2 into the packet.

  When RX, Link Layer will only get PDU, CRC from RBME.

  Users can also set RXEN_DLY_OVERRIDE to 1 and use RXEN_DLY to override the hardware calculation result. The waveform is show in Figure 430.

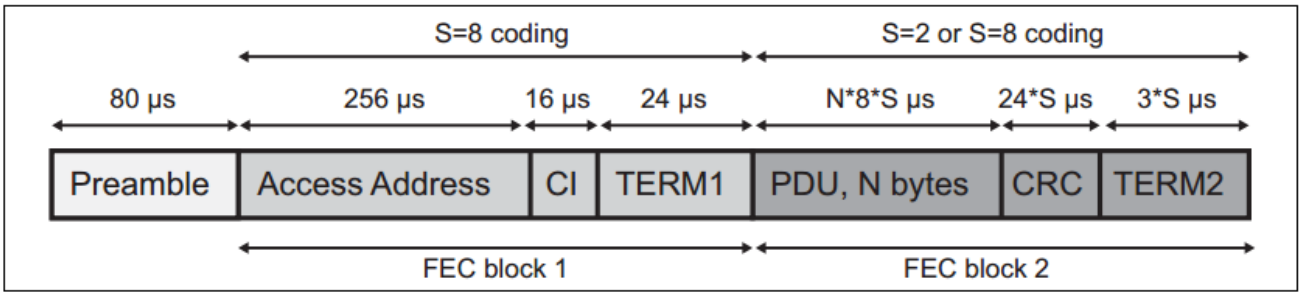


Figure 459. Link Layer packet format for the LE Coded PHY

- Advertising channel PDU

  The advertising channel PDU has a 16-bit header and a variable size payload. Its format is as shown in Figure 460. The 16-bit Header field of the advertising channel PDU is as shown in Figure 461.

  The Length field of the advertising channel PDU header indicates the payload field length in octets. The valid range of the Length field shall be 1 to 255 octets.

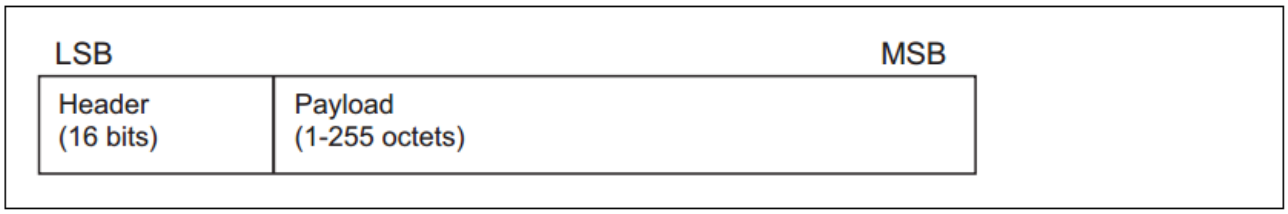


Figure 460. Advertising channel PDU

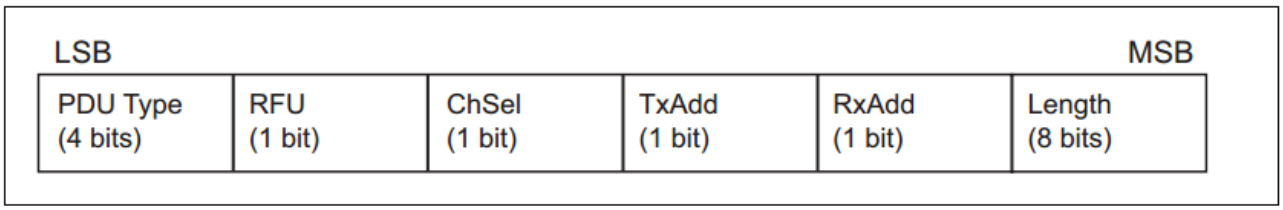


Figure 461. Advertising channel PDU Header

- Data channel PDU

  The Data Channel PDU has a 16 bit header, a variable size payload, and may include a Message Integrity Check (MIC) field. The Data Channel PDU is as shown in Figure 462. The Header field of the Data Channel PDU is as shown in Figure 463.

  The Length field of the Header indicates the length of the Payload and MIC if included. The length field has the range of 0 to 255 octets. The Payload field shall be less than or equal to 251 octets in length. The MIC is 4 octets in length.

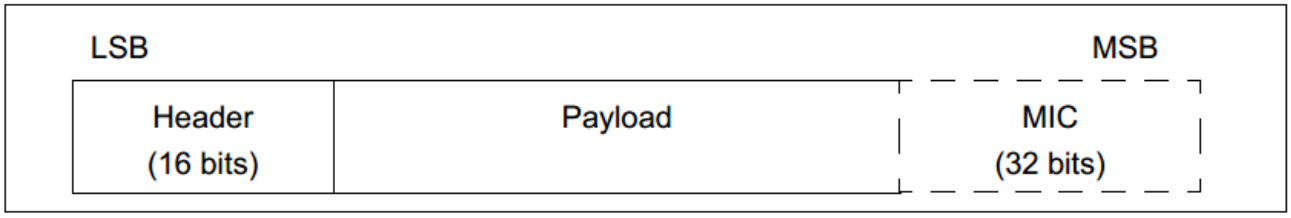


Figure 462. Data Channel PDU

Header					
LLID (2 bits)	NESN (1 bit)	SN (1 bit)	MD (1 bit)	RFU (3 bits)	Length (8 bits)

Figure 463. Data channel PDU header

- Bluetooth LE 5.1 packet format for the LE uncoded PHYs

  Be different with Bluetooth LE 5.0, the CRC is followed by an optional Constant Tone Extension field, which consists of a constantly modulated series of unwhitened 1s. The constant Tone Extension field is not included in CRC or MIC calculations.

  When TX, TX_DIG adds the CTE field.

  When RX, the CTE field can be received by Link Layer, using the RX_EN delay feature. LL hardware will recognize the uncoded packet and automatically calculate the duration of CTE according to the PDU type and CTETime field parsed, and expand the RX_EN signal accordingly. Users can also set RXEN_DLY_OVERRIDE to 1 and use RXEN_DLY to override the hardware calculation result. The waveform is shown in Figure 430.

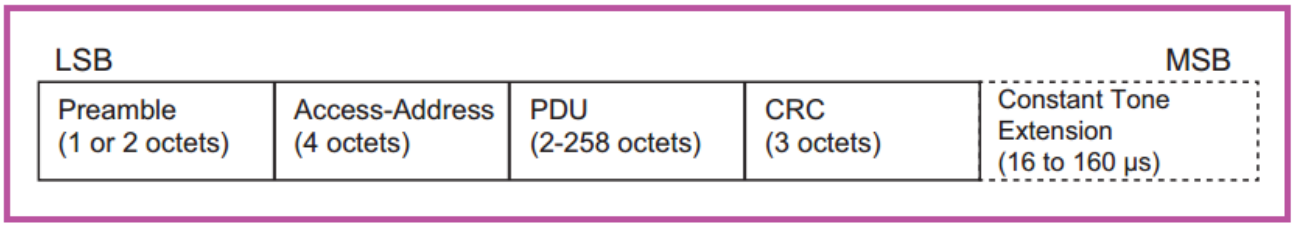


Figure 464. Link Layer packet format for the LE Uncoded PHY (Bluetooth LE 5.1)

To map the Bluetooth LE packet format to the generic packet format, just map the Length to the LENGTH field and the other 8 bits in the Header to the H0 field. Table 489 shows the register configuration for the Bluetooth LE emulation.

Table 489. Link Layer Register Configuration for Bluetooth LE Emulation

Register	Value
GENLL_MODE[3:0]	• 0x9 - Bluetooth LE Uncoded Mode • 0xA - Bluetooth LE LR Mode • 0xB - Bluetooth LE Concurrent Mode (RX configuration only; TX uses either Bluetooth LE Uncoded or Bluetooth LE LR configuration)

Table continues on the next page...

Table 489. Link Layer Register Configuration for Bluetooth LE Emulation (continued)

Register	Value
PREAMBLE_SZ[8:0]	• 0x000 - 1 octet preamble • 0x001 - 2 octets preamble
PREAMBLE_SEL[2:0]	001b - Preamble is programmed by register GEN_PREAMBLE[7:0]
GEN_PREAMBLE[7:0]	As needed
NTW_ADR_SZ[1:0]	11b - Network Address 0/1/2/3 requires a 32-bit correlation
SYNC_ADDR_SZ[1:0]	11b - Network Address is 32-bit
H0_SZ[4:0]	8
LENGTH_SZ[4:0]	8
H1_SZ[4:0]	0 or 8(TX of Bluetooth LE 5.1 Data Channel PDU with CTE)
LENGTH_BIT_ORD	0 - LS Bit First
LENGTH_ADJ[10:0]	3 - Same as CRC_SZ[2:0] (RBME register)
AA_PLAYBACK_CNT	0 - Access Address will not play back to Link Layer
BLE_V5P1_CTE_EN	When set, hardware will parse the CTE field as defined in Bluetooth LE V5.1 specification
RXEN_DLY_OVERRIDE	Override the hardware calculation result for the duration of TERM2 or CTE
RXEN_DLY[9:0]	When not zero and RXEN_DLY_OVERRIDE is set, the RX_EN signal will delay (RXEN_DLY +1) microseconds to de-assert after payload is received. The delay time range is from 2~1024 microseconds.
SEL_RXIRQ	• 0: RX_IRQ is asserted at the end of RX_PKT state • 1: RX_IRQ is asserted at the end of RXEN_DLY state(only if RX_EN is delayed).

CTE duration and parsing and signal cte_present resolve process:

- When RXEN_DLY_OVERRIDE=1, it means for all the RX process, there will be a delay defined by RXEN_DLY[9:0];
- — If BLE_V5P1_CTE_EN==1, cte_present=1 in the whole RX process
  — Otherwise, cte_present=0
- When RXEN_DLY_OVERRIDE=0, hardware will decide the RX_EN delay time.
- — If Bluetooth LE coded packet is received(i.e., internal signal phy_mode[2:0]=1), RX_EN is not delayed.
  — If Bluetooth LE uncoded packet is received(i.e., phy_mode[2:0]=2),
  — ◦ If it is Bluetooth LE 5.0 use case(BLE_V5P1_CTE_EN=0), no RX_EN delay
    ◦ If it is Bluetooth LE 5.1 use case(BLE_V5P1_CTE_EN=1),
    ◦ ▪ If received data channel PDU (ADV_CHANNEL_EN=0) , as shown in Figure 465
      ▪ • If [CP] =0, then on-the-fly change H1_SIZE to 0, no RX_EN delay
        • If [CP] =1, then on-the-fly change H1_SIZE to 8, and get the [CTETime] as the RX_EN Delay; cte_present=1 when parsed
      ▪ If received advertising channel PDU(ADV_CHANNEL_EN=1) , as shown in Figure 466
      ▪ • If the [Extended Header Length]=0, no RX_EN delay

- If the [Extended Header Length]>0, get the [CTEInfo] (if exists), and get the [CTETime] as the RX_EN Delay; cte_present=1 when parsed

NOTE

1. The [CTETime] field defines the length of the Constant Tone Extension in 8 us units. The value of the [CTETime] field shall be between 2 and 20 inclusive; all other values are reserved for future use. But LL will not check the length range.
2. Once asserted, cte_present will not be de-asserted even CRC fails. Register bit CRC_IGNORE can be used in debug cases to bypass the CRC pass condition.
3. If CRC fail and CRC_IGNORE=0, LL will recycle according to its inherent behavior.

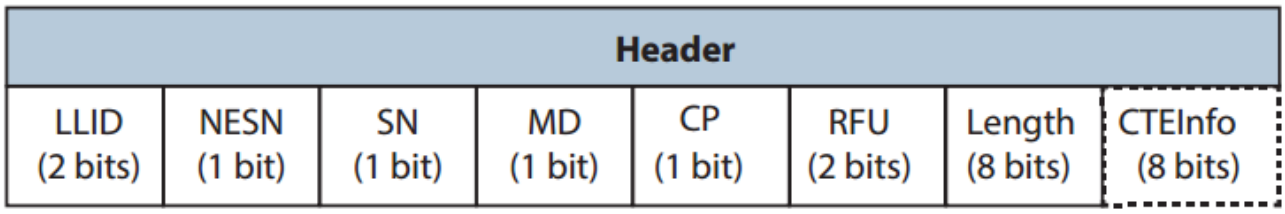


Figure 465. Bluetooth LE 5.1 Data Channel PDU Header

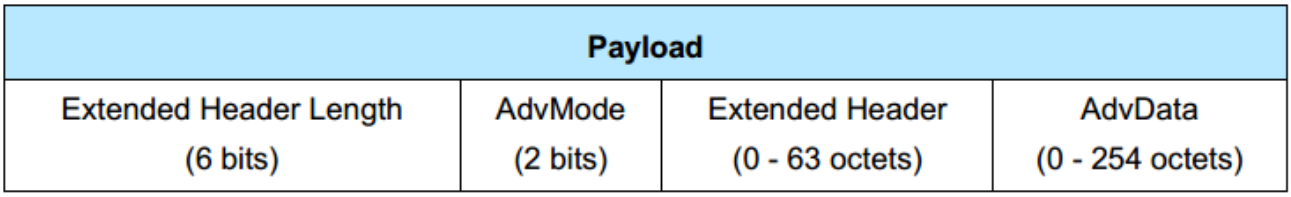


Figure 466. Bluetooth LE 5.1 Common Extended Advertising Payload

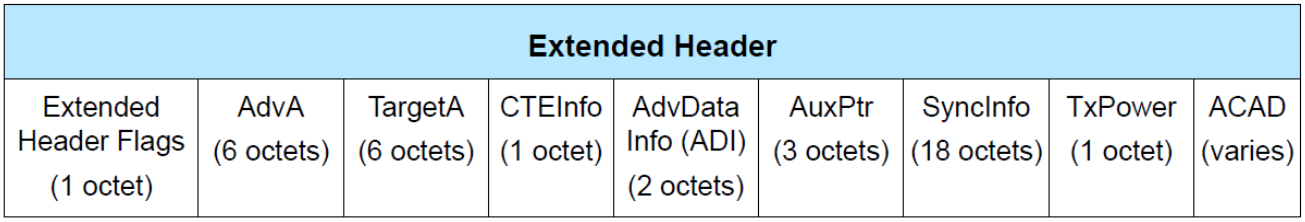


Figure 467. Bluetooth LE 5.1 Extended Header

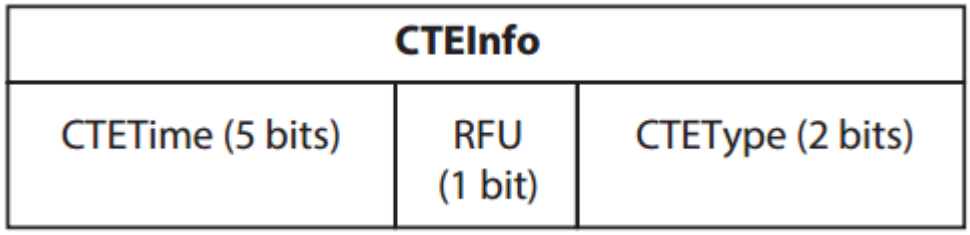


Figure 468. Bluetooth LE 5.1 CTEInfo Field

55.4.9.2.3.8.3.1 RPA and White List

The format of the resolvable private address is shown in Figure 469.

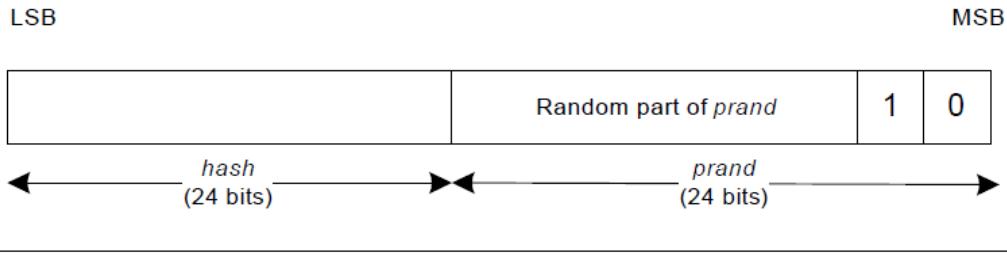


Figure 469. Format of resolvable private address

Private device address generation

The resolvable private address is generated with the Identity Resolving Key (IRK) and a randomly generated 24-bit number(prand).

The hash is generated using the function *ah* as *hash = ah(IRK, prand)*.

The prand and hash are concatenated to generate the resolvable private address in the following manner:

resolvable private address = hash || prand

The least significant octet of hash becomes the least significant octet of RPA and the most significant octet of prand becomes the most significant octet of RPA.

Private device address resolution

A resolvable private address may be resolved if the corresponding device's IRK is available using this procedure. If a resolvable private address is resolved, the device can associate this address with the peer device.

A localHash value is generated as *localHash = ah(IRK, prand)*

The localHash value is then compared with the hash value extracted from RPA. If the localHash value matches the extracted hash value, then the identity of the peer device has been resolved.

If a device has more than one stored IRK, the device repeats the above procedure for each stored IRK to determine if the received resolvable private address is associated with a stored IRK, until either address resolution is successful for one of the IRKs or all have been tried.

The RPA resolution and White List check are related with advertising channel PDUs. Register ADV_CHANNEL_EN is to indicate that it is to receive an advertising channel PDU.

Linklayer supports resolvable private address(RPA) resolution during Rx process when register RPA_EN is enabled.

There are 8 entries for the RPA resolution in RAM. Each entry includes the fields: Peer IRK[127:0], Local IRK[127:0], Peer_Identity_Address[47:0], Peer_Identity_Address_Type, Peer_RPA[47:0], Local_RPA[47:0]. The organization of these fields are shown in Figure 470.

Register RPA_VALID_ENTRY[7:0] defines that if the entry is valid or not. If the entry is not valid, it is bypassed during RPA search and resolution.

When RPA_EN is asserted, and the received Peer Address is resolvable, the address is first be searched in the PRPA list. If the address is found in the PRPA list, the address is resolved and register PEER_RESOLVED_INDEX[3:0] returns matched index of the entry. If the address is not in the PRPA list, link layer resolves the address with Peer IRK, one by one. If the address is resolved by one of the Peer IRK, register PEER_RESOLVED_INDEX[3:0] returns matched index of the entry. Link layer will also update the received peer address to the PRPA list.

If the peer address is not resolved, PEER_RESOLVED_INDEX[3:0] returns 0xF and interrupt flag PEER_RPA_FAIL_IRQ is asserted. In this case, if register PEER_RPA_FAIL_IRQ_EN is set, the interrupt is generated to the host. If register IGNORE_RPA_FAIL is set, the link layer continues to receive the whole packet; otherwise, the link layer aborts.

When RPA_EN is asserted, and the received Local Address is resolvable, the address is first be searched in the LRPA list. If the address is found in the LRPA list, the address is resolved and register LOCAL_RESOLVED_INDEX[3:0] returns matched index of the entry. If the address is not in the LRPA list, link layer resolves the address with Local IRK, one by one. If the address is resolved by one of the Local IRK, register LOCAL_RESOLVED_INDEX[3:0] returns matched index of the entry. Link layer will also update the received peer address to the LRPA list.

If the local address is not resolved, LOCAL_RESOLVED_INDEX[3:0] returns 0xF and interrupt flag LOCAL_RPA_FAIL_IRQ is asserted. In this case, if register LOCAL_RPA_FAIL_IRQ_EN is set, the interrupt is generated to the host. If register IGNORE_RPA_FAIL is set, the link layer continues to receive the whole packet; otherwise, the link layer aborts.

Entry #	Byte Address	Peer IRK	Byte Address	Local IRK	Byte Address	PEER Identity Address	Byte Address	Peer RPA	Byte Address	Local RPA
Entry[0]	PEER_IRK_BASE+0	peer_irk[0][31:0]	LOCAL_IRK_BASE+0	local_irk[0][31:0]	PEER_ID_BASE+0	peer_id_addr[0][31:0]	PRPA_BASE+0	peer_rpa[0][31:0]	LRPA_BASE+0	local_rpa[0][31:0]
	PEER_IRK_BASE+4	peer_irk[0][63:32]	LOCAL_IRK_BASE+4	local_irk[0][63:32]	PEER_ID_BASE+4	peer_id_addr_type[0], 15'h0, peer_id_addr[0][47:32]	PRPA_BASE+4	16'h0, peer_rpa[0][47:32]	LRPA_BASE+4	16'h0, local_rpa[0][47:32]
	PEER_IRK_BASE+8	peer_irk[0][95:64]	LOCAL_IRK_BASE+8	local_irk[0][95:64]						
	PEER_IRK_BASE+12	peer_irk[0][127:96]	LOCAL_IRK_BASE+12	local_irk[0][127:96]						
Entry[1]	PEER_IRK_BASE+16	peer_irk[1][31:0]	LOCAL_IRK_BASE+16	local_irk[1][31:0]	PEER_ID_BASE+8	peer_id_addr[1][31:0]	PRPA_BASE+8	peer_rpa[1][31:0]	LRPA_BASE+8	local_rpa[1][31:0]
	PEER_IRK_BASE+20	peer_irk[1][63:32]	LOCAL_IRK_BASE+20	local_irk[1][63:32]	PEER_ID_BASE+12	peer_id_addr_type[1], 15'h0, peer_id_addr[1][47:32]	PRPA_BASE+12	16'h0, peer_rpa[1][47:32]	LRPA_BASE+12	16'h0, local_rpa[1][47:32]
	PEER_IRK_BASE+24	peer_irk[1][95:64]	LOCAL_IRK_BASE+24	local_irk[1][95:64]						
	PEER_IRK_BASE+28	peer_irk[1][127:96]	LOCAL_IRK_BASE+28	local_irk[1][127:96]						
Entry[2]	PEER_IRK_BASE+32	...	LOCAL_IRK_BASE+32	...	PEER_ID_BASE+16	...	PRPA_BASE+16	...	LRPA_BASE+16	...
	PEER_IRK_BASE+36	...	LOCAL_IRK_BASE+36	...	PEER_ID_BASE+20	...	PRPA_BASE+20	...	LRPA_BASE+20	...
	PEER_IRK_BASE+40	...	LOCAL_IRK_BASE+40	...						
	PEER_IRK_BASE+44	...	LOCAL_IRK_BASE+44	...						
Entry[3]	PEER_IRK_BASE+48	...	LOCAL_IRK_BASE+48	...	PEER_ID_BASE+24	...	PRPA_BASE+24	...	LRPA_BASE+24	...
	PEER_IRK_BASE+52	...	LOCAL_IRK_BASE+52	...	PEER_ID_BASE+28	...	PRPA_BASE+28	...	LRPA_BASE+28	...
	PEER_IRK_BASE+56	...	LOCAL_IRK_BASE+56	...						
	PEER_IRK_BASE+60	...	LOCAL_IRK_BASE+60	...						
Entry[4]	PEER_IRK_BASE+64	...	LOCAL_IRK_BASE+64	...	PEER_ID_BASE+32	...	PRPA_BASE+32	...	LRPA_BASE+32	...
	PEER_IRK_BASE+68	...	LOCAL_IRK_BASE+68	...	PEER_ID_BASE+36	...	PRPA_BASE+36	...	LRPA_BASE+36	...
	PEER_IRK_BASE+72	...	LOCAL_IRK_BASE+72	...						
	PEER_IRK_BASE+76	...	LOCAL_IRK_BASE+76	...						
Entry[5]	PEER_IRK_BASE+80	...	LOCAL_IRK_BASE+80	...	PEER_ID_BASE+40	...	PRPA_BASE+40	...	LRPA_BASE+40	...
	PEER_IRK_BASE+84	...	LOCAL_IRK_BASE+84	...	PEER_ID_BASE+44	...	PRPA_BASE+44	...	LRPA_BASE+44	...
	PEER_IRK_BASE+88		LOCAL_IRK_BASE+88	...						
	PEER_IRK_BASE+92		LOCAL_IRK_BASE+92	...						
Entry[6]	PEER_IRK_BASE+96	...	LOCAL_IRK_BASE+96	...	PEER_ID_BASE+48	...	PRPA_BASE+48	...	LRPA_BASE+48	...
	PEER_IRK_BASE+100	...	LOCAL_IRK_BASE+100	...	PEER_ID_BASE+52	...	PRPA_BASE+52	...	LRPA_BASE+52	...
	PEER_IRK_BASE+104	...	LOCAL_IRK_BASE+104	...						
	PEER_IRK_BASE+108	...	LOCAL_IRK_BASE+108	...						
Entry[7]	PEER_IRK_BASE+112	...	LOCAL_IRK_BASE+112	...	PEER_ID_BASE+56	...	PRPA_BASE+56	...	LRPA_BASE+56	...
	PEER_IRK_BASE+116	...	LOCAL_IRK_BASE+116	...	PEER_ID_BASE+60	...	PRPA_BASE+60	...	LRPA_BASE+60	...
	PEER_IRK_BASE+120	...	LOCAL_IRK_BASE+120	...						
	PEER_IRK_BASE+124	...	LOCAL_IRK_BASE+124	...						

Figure 470. RPA entry organization in RAM

Link layer supports White List search and filter during Rx process when register WL_EN is enabled.

There are 2 White Lists in RAM, WL0 and WL1. Each White List has 32 entries. Register WL_SEL is used to select which White List is used for the search.

Each entry includes fields WL_Device_Address[47:0] and WL_Device_Address_Type, as shown in Figure 471.

For each White List, there is a register WL_VALID_ENTRY[31:0] to enable the 32 entries. Once disabled, the entry is skipped during the search.

If RPA is resolved, the Peer_Identity_Address[47:0] and Peer_Identity_Address_Type are searched in the White List. If RPA is not enabled or no RPA entry is valid, the received Peer Address and its type are directly searched in the White List.

After search, if there is an entry matches the address to be searched, the register WL_MATCH_INDEX[5:0] returns the matched index of the entry.

If there is no entry matched or all entries are invalid, WL_MATCH_INDEX[5:0] returns the 0x3F value, and interrupt flag WL_FAIL_IRQ is asserted. In this case, if register WL_FAIL_IRQ_EN is set, the interrupt is generated to the host. If register IGNORE_WL_FAIL is set, the link layer continues to receive the whole packet; otherwise, the link layer aborts.

Byte Addr	Entry #	White List 0
WL_BASE0+0	Entry[0]	wl_device_id_addr[0][31:0]
WL_BASE0+4		wl_addr_typ[0], 15'h0, wl_device_id_addr[0][47:32]
WL_BASE0+8	Entry[1]	wl_device_id_addr[1][31:0]
WL_BASE0+12		wl_addr_typ[1], 15'h0, wl_device_id_addr[1][47:32]
…	…	…
…		…
WL_BASE0+248	Entry[31]	wl_device_id_addr[31][31:0]
WL_BASE0+252		wl_addr_typ[31], 15'h0, wl_device_id_addr[31][47:32]

Byte Addr	Entry #	White List 1
WL_BASE1+0	Entry[0]	wl_device_id_addr[0][31:0]
WL_BASE1+4		wl_addr_typ[0], 15'h0, wl_device_id_addr[0][47:32]
WL_BASE1+8	Entry[1]	wl_device_id_addr[1][31:0]
WL_BASE1+12		wl_addr_typ[1], 15'h0, wl_device_id_addr[1][47:32]
…	…	…
…		…
WL_BASE1+248	Entry[31]	wl_device_id_addr[31][31:0]
WL_BASE1+252		wl_addr_typ[31], 15'h0, wl_device_id_addr[31][47:32]

Figure 471. White List 0 and White List 1 in RAM

For each type of PDUs that needs to do RPA check and WL check, the hardware tasks are summarized in Figure 472.

If the received PDU Type is "0011b" or "0101b" regarding to an ADV_DIRECT_IND PDU, the resolved Peer identity address and address type are not searched in WL, but are compared with register DIRECT_PEER_ADDR[47:0] and DIRECT_PEER_ADDR_TYPE. This case is indicated by set the register ADV_DIRECT_IND_SENT.

If the comparison result is a mismatch, interrupt flag DIRECT_ID_FAIL_IRQ is asserted. In this case, if register DIRECT_ID_FAIL_IRQ_EN is set, the interrupt is generated to the host. If register IGNORE_DIRECT_FAIL is set, the link layer continues to receive the whole packet; otherwise, the link layer aborts.

<table>
<tr><th>PDU Name</th><th>PDU Type</th><th colspan="7">Packet Fields</th><th>Hardware Task</th></tr>
<tr><td>ADV_DIRECT_IND</td><td>0001</td><td>AdvA</td><td>TargetA</td><td></td><td></td><td></td><td></td><td>1. Resolve AdvA
2. Check WL
3. Resolve TargetA</td></tr>
<tr><td>ADV_IND</td><td>0000</td><td>AdvA</td><td>AdvData</td><td></td><td></td><td></td><td></td><td rowspan="4">1.Resolve AdvA
2.Check WL</td></tr>
<tr><td>ADV_NONCONN_IND</td><td>0010</td><td>AdvA</td><td>AdvData</td><td></td><td></td><td></td><td></td></tr>
<tr><td>SCAN_RSP</td><td>0100</td><td>AdvA</td><td>ScanRspData</td><td></td><td></td><td></td><td></td></tr>
<tr><td>ADV_SCAN_IND</td><td>0110</td><td>AdvA</td><td>AdvData</td><td></td><td></td><td></td><td></td></tr>
<tr><td>ADV_EXT_IND</td><td rowspan="6">111</td><td rowspan="7">Extended Header Length (6 bits)+ AdvMode (2 bits)</td><td colspan="5">Extended Header (0 - 63 octets)</td><td rowspan="7">AdvData</td><td rowspan="7">1. Resolve AdvA (if exists)
2. Check WL
3. Resolve TargetA (if exists)</td></tr>
<tr><td>AUX_ADV_IND</td><td rowspan="6">Extended Header Flags (1 octet)</td><td rowspan="6">AdvA</td><td rowspan="6">TargetA</td><td rowspan="6">…</td></tr>
<tr><td>AUX_SCAN_RSP</td></tr>
<tr><td>AUX_SYNC_IND</td></tr>
<tr><td>AUX_CHAIN_IND</td></tr>
<tr><td>AUX_CONNECT_RSP</td><td>1000</td></tr>
<tr><td>SCAN_REQ
AUX_SCAN_REQ</td><td>0011</td><td>ScanA</td><td>AdvA</td><td></td><td></td><td></td><td></td><td rowspan="2">1. Resolve ScanA/InitA
2.1 If it is for ADV_DIRECT_IND, check with DIRECT_PEER_ADDR/DIRECT_PEER_ADDR_TYPE
2.2 Other cases, check WL</td></tr>
<tr><td>CONNECT_IND
AUX_CONNECT_REQ</td><td>0101</td><td>InitA</td><td>AdvA</td><td>LLData</td><td></td><td></td><td></td></tr>
</table>

Figure 472. Hardware tasks for RPA resolution and White List(WL) check

55.4.9.2.3.8.4 Generic Link Layer Test Mode (GTM)

Objective

Generic Test Mode (GTM) is a special test mode with the objective to facilitate time-efficient testing of the Packet Error Rate (PER) for the PHY modes available to the Generic Link Layer. The goal of this mode is to test the radio's TX and RX physical layer and front-end quality assisted by the link layer without necessarily involving complicated link layer processing.

The key objective of this test mode are as follows:

- To make PER testing on NXP radios using Generic Link Layer easy for both internal and external (customer) usage, and
- To minimize the complications in a PER test setup by reducing the need of external triggering and provision of accounting for good (passing CRC), bad (failing CRC) and missing packets on the receiving end.

Generic Test Mode (GTM) Requirements

The GTM HW will facilitate the following simplified testing of RX and TX PHY performances using either a signal analyzer (for TX), a vector signal generator (for RX) and using two NXP radios (where one is configured as TX, while the other is configured as RX). Note that it is conceived that the GTM implementation would leverage the link layer infrastructure as much as possible minimizing the need of additional register space to support GTM programming.

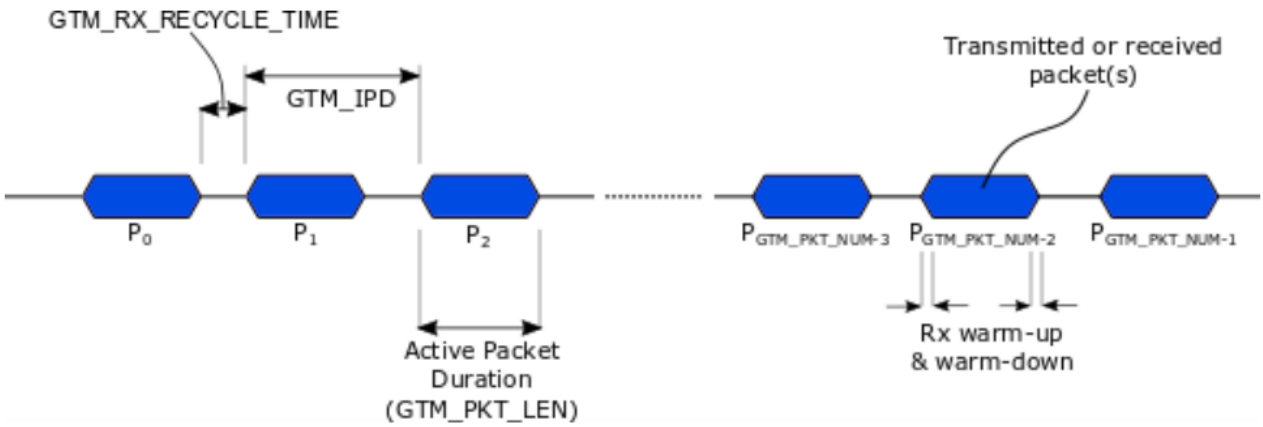


Figure 473. PACKET INTERVALS IN GTM mode

GTM Packet Structure

The packet structure in GTM mode is shown as below, where H0[7:0] = {4'h0, GTM_PDU_TYP}. So in GTM mode, register H0_SZ should be set to 8.

Figure 474. GTM PACKET STRUCTURE

Where

Table 490. GTM PACKET FIELDS

Field	Definition
PREAMBLE	Packet Preamble as supported by Gen-LL. Default size is 1 octet with preamble pattern chosen based on 1st bit of SYNC ADDRESS
SYNC ADDRESS	From registers NTW_ADR_*X*, when NTW_ADR_EN[*X*]==1. When more than one NTW_ADR_EN[*X*] is enabled, NTW_ADR_EN[0] has the highest priority. NTW_ADR*X*_SZ and SYNC_ADDR_SZ should be same value. Default size is 4 octets. It is recommended to use the same default SYNC ADDRESS value as is used by Bluetooth-LE DTM, with default transmission order of '10010100 10000010 01101110 10001110'.
GTM_PDU_TYP	From register GTM_PDU_TYPE[3:0] (reused from MACLONGADDRS1_LSB[27:24]) 0000b PRBS9 Sequence 0001b Programmable 8-bit Pattern (from register GTM_PDU[7:0], reused from MACSHORTADDRS1[7:0]) 0010b PRBS-13 Sequence 0011b PRBS-15 Sequence 0100b Programmable 32-bit Pattern (from register GTM_PDU[31:0], reused from {MACSHORTADDRS1,MACPANID1}) 0101b Programmable packet from Packet RAM (in this case, PKT_LEN is ignored) Others reserved
RES	It is reserved. (0000b)

Table continues on the next page...

Table 490. GTM PACKET FIELDS (continued)

Field	Definition
PKT_LEN	The PKT_LEN is the content of the LENGTH field. When LENGTH_ADJ=0, PKT_LEN means the length of (PAYLOAD + CRC); When LENGTH_ADJ = CRC_SZ, PKT_LEN means the length of PAYLOAD. The PKT_LEN size is controlled by register LENGTH_SZ, the value can be 8 or 16-bit. The PKT_LEN value is controlled by register GTM_PKT_LEN[10:0] (reused from LENGTH_ACK[10:0]). The GTM_PKT_LEN[10:0] value overflows the LENGTH_SZ is ignored. The default packet allocation will be 8 bits and the default value will be 39 octets (same as Bluetooth LE DTM)
GTM_PDU	A string of transmission bits as defined above by GTM_PDU_TYP, which are sent out by default in the same order as Bluetooth-LE.
CRC	All CRC computational combinations defined by Gen-LL are possible. Default size will be 3 octets and the computation will be configured to be the same as Bluetooth-LE, i.e.,

Generic Test Mode Transmit (GTM-TX)

In GTM-TX mode, Generic LL would allow for the following configurability for the radio driver SW:

- Program GENLL_MODE[3:0] to 0xF to enable the GTM mode
- Program PHY modulation parameters
- Program the type and size of the preamble to be used for the packet (PREAMBLE_SZ and PREAMBLE_SEL)
- Program a test mode Access Address (or SFD). All size of the AA/SFD supported by the Gen-LL shall be supported. (NTW_ADR_EN[*X*] , NTW_ADR_*X*, NTW_ADR_SZ and SYNC_ADDR_SZ)
- Program a fixed length for the Packet size in octets (PKT_LEN).
- Program the type of payload (GTM_PDU).
- Program the Bit manipulation engine (BME) configuration to be used, e.g., CRC, whitening, FEC, encoding, repetition or spreading, etc. blocks and processing order be configured as per the desired test configuration.
- Program the interpacket duration (register GTM_IPD[19:0], reused from MACLONGADDRS1_MSB[19:0]).
- Program the number of packets to be sent (register GTM_PKT_NUM[11:0]), reused from MACLONGADDRS1_LSB[11:0]).
- Program the delay time between 1st Tx warmup time and GTM_IN_TX asserted (register GTM_RX_RECYCLE_TIME[19:0], reused from MACLONGADDRS0_MSB[19:0]; This register has different meaning in GTM RX mode)

Once configured, GTM_TX mode will operate as follows:

- Write a "1" to register GTM_IN_TX to trigger the start of the GTM-TX mode.
  - Clear GTM_PKT_COUNT at initial
  - Gen-LL transmits the 1st sequence after GTM_RX_RECYCLE_TIME
  - Gen-LL transmits sequence after every GTM_IPD duration for GTM_PKT_NUM times.
  - Note that for each packet transmission, the TX would warm-up, transmit the packet and warm-down
- After requisite number of TX packets are sent, i.e., GTM_PKT_COUNT == GTM_PKT_NUM; the GTM_IN_TX bit would be de-asserted and GTM_TX would stop.
- The following figure shows the GTM TX timing. The GTM_PKT_PUR represents the transmitted packet duration, where *GTM_PKT_DUR = sizeof(PREAMBLE+AA+H0+LEN+GTM_PDU+CRC)*symbol_time*

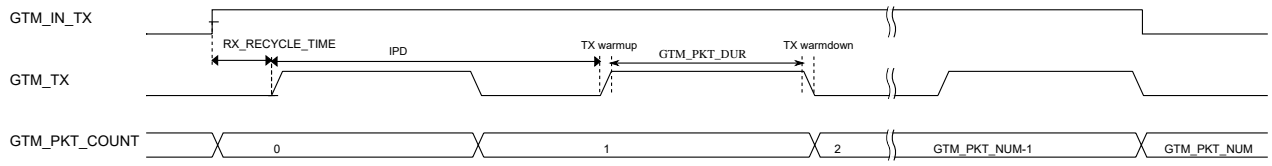


Figure 475. GTM TX Processing

Generic Test Mode Receive (GTM-RX)

For GTM-RX, it is conceived that Gen-LL and BME for the receiver are correctly configured as per the test configuration. This includes

- Program GENLL_MODE[3:0] to 0xF to enable the GTM mode
- Program the register FIRST_SFD2WD, which means the time between the force Rx warmdown and first SFD found. It is possibly the longest packet duration plus some margin.
- Program the register RX_RECYCLE_TIME, which means the time between the force Rx warmdown and Rx warmup.
- Program the register GTM_IPD, same as the GTM Tx side.
- Write a "1" to register GTM_IN_RX to trigger the start of the GTM-RX scan mode. This bit should also clear all GTM packet counters (GTM_GOOD_PKT_COUNT, GTM_BAD_PKT_COUNT and GTM_PKT_COUNT).
- The receiver will wait for the first correctly AA/SFD matched packet. This initiates the GTM-RX state machine.
- Every packet for which AA/SFD match happens should increment the GTM_PKT_COUNT.
- Once GTM_PKT_COUNT == GTM_PKT_NUM, GTM_IN_RX bit should be de-asserted and the periodic RX recycling by the GTM state machine should stop.
- If register ***GTM_PKT_COUNT_CHECK_DIS*** is programmed, then
  - The check GTM_PKT_COUNT == GTM_PKT_NUM is disabled
  - In this case the GTM test would be disabled by SW based time-out
  - As now there is no limit on the upper bound on GTM_PKT_COUNT due to a pre-programmed GTM_PKT_NUM, the counter can possibly overflow GTM_PKT_NUM. The counter will have saturation but not roll over in this case.
- If the packet is successfully received, processed by BME and passes the payload CRC. It is considered a good packet and the GTM_GOOD_PKT_COUNT is incremented.
- If the packet is not successfully received due to corrupted packet length or an unrecoverable bit error causing the packet to fail the payload CRC, or there is a header(H0/Length/H1) check fail or there is no AA matched, it is considered a bad packet and the GTM_BAD_PKT_COUNT is incremented.
- GTM_GOOD_PKT_COUNT and GTM_BAD_PKT_COUNT should be incremented based on CRC pass/fail only and setting a Gen-LL configuration such as CRC_IGNORE should not effect it.
- After receiving either a good or a bad packet, Gen-LL should recycle the receiver after GTM_RX_RECYCLE_TIME, where GTM_RX_RECYCLE_TIME is the time required by the receiver to warm-down and warm-up with some additional timing margins to receive the next packet. As an example, for a radio having a warm-up time of 110us and a warm-down time of 10us, GTM_RX_RECYCLE_TIME ≥ 150us.
- If for a packet, AA/SFD was not detected for a duration of GTM_IPD - GTM_RX_RECYCLE_TIME, it is considered that the packet was missed, GTM_PKT_COUNT is incremented and the receiver is recycled.
- In case the receiver decodes the incoming packets length incorrectly to be too long, although AA/SFD will be matched (causing the GTM_PKT_COUNT to be incremented), RX may stay in the demodulation state during the entire GTM_IPD - GTM_RX_RECYCLE_TIME duration. In this scenario, the receiver is recycled at the scheduled time and the GTM_BAD_PKT_COUNT is incremented as well.

- If register ***GTM_IPD_CHECK_DIS*** is programmed, then,
  - GTM_IPD duration enforcement is disabled
  - After each packet received, Link layer will recycle after "GTM_RX_RECYCLE_TIME"
- The packet statistics are collected as follows:
  - Number of Good CRC Packets = GTM_GOOD_PKT_COUNT
  - Number of Bad CRC Packets = GTM_BAD_PKT_COUNT
  - Number of Missed Packets = GTM_PKT_NUM - (GTM_GOOD_PKT_COUNT + GTM_BAD_PKT_COUNT)
- The following figure shows the normal GTM RX timing.

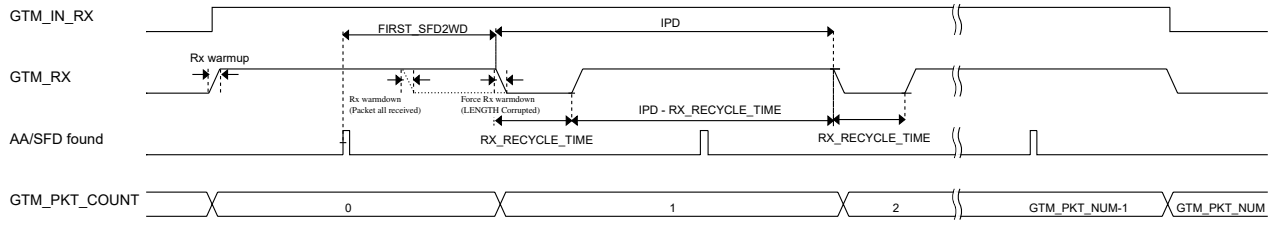


Figure 476. GTM RX Processing

The following figure shows the GTM RX timing with GTM_IPD_CHECK_DIS=1.

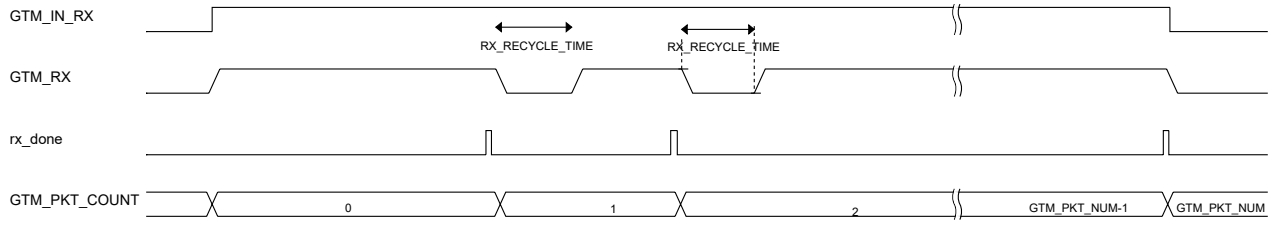


Figure 477. GTM RX Processing with GTM_IPD_CHECK_DIS=1

The following table summarizes the registers used in GTM.

Table 491. Link Layer Registers Used in GTM

Register	RX Side ?	TX Side ?	Description	Reused From
GENLL_MODE[3:0]	X	X	Link layer Mode Select, 1111b - GTM Mode	/
GTM_PDU[31:0]		X	8bits(GTM_PDU_TYP is 0001b) or 32 bits (GTM_PDU_TYP is 0100b) GTM PDU Pattern	{MACSHORTADDRS1, MACPANID1}
GTM_PDU_TYPE[3:0]		X	GTM PDU Type	MACLONGADDRS1[27:24]
GTM_IPD_CHECK_DIS	X		• GTM_IPD duration enforcement is disabled	MACLONGADDRS1[30]

Table continues on the next page...

Table 491. Link Layer Registers Used in GTM (continued)

Register	RX Side ?	TX Side ?	Description	Reused From
			• After each packet received, Link layer will recycle after "GTM_RX_RECYCLE_TIME"	
GTM_PKT_COUNT_CHECK_DIS	X		• The check GTM_PKT_COUNT == GTM_PKT_NUM is disabled; In this case the GTM test would be disabled by SW based time-out; • As now there is no limit on the upper bound on GTM_PKT_COUNT due to a pre-programmed GTM_PKT_NUM, the counter can possibly overflow GTM_PKT_NUM. The counter will have saturation but not roll over in this case.	MACLONGADDRS1[31]
GTM_PKT_NUM[11:0]	X	X	The total number of packets to be sent or received	MACLONGADDRS1[11:0]
GTM_IPD[19:0]	X	X	• In microseconds • Time slot that one packet is sent	MACLONGADDRS1[51:32]
GTM_1ST_SFD2WD[19:0]	X		• In microseconds • The time between the "force Rx warm-down" and "first SFD found", in microseconds • Possibly the longest packet duration plus some margin.	MACLONGADDRS0[19:0]
GTM_RX_RECYCLE_TIME[19:0]	X	X	• In microseconds • For TX, means the delay between "first TX warm up" and "GTM_IN_TX is asserted" • For RX, means the time between "force Rx warm-down" and "next Rx warmup"	MACLONGADDRS0[51:32]
GTM_PKT_LEN[10:0]		X	The LENGTH field	LENGTH_ACK[10:0]
LENGTH_ADJ[10:0]	X	X	• When LENGTH_ADJ=0, PKT_LEN means the length of (PAYLOAD + CRC); • When LENGTH_ADJ=CRC_SZ, PKT_LEN means the length of PAYLOAD.	/
GTM_IN_RX	X		GTM receive mode	/
GTM_IN_TX		X	GTM transmit mode	/
GTM_PKT_CNT[12:0]	X	X	• When RX, count the number of IPD slots after first AA match • When TX, count the number of packets sent.	/
GTM_BAD_PKT_CNT[12:0]	X		GTM Bad Packet Counter (CRC fail or header(H0/Length/H1) fail or no AA matched)	/

Table continues on the next page...

Table 491. Link Layer Registers Used in GTM (continued)

Register	RX Side ?	TX Side ?	Description	Reused From
GTM_GOOD_PKT_CNT[12:0]	X		GTM Good Packet Counter (CRC pass and header(H0/ Length/H1) pass)	/

55.4.9.2.3.9 Interrupts

Interrupt Status Bit Structure

The Generic FSK Link Layer Controller has 15 interrupt sources, each represented in the register map by an interrupt status bit. All interrupt status bits share a common, generic structure. If a particular interrupt source is enabled, a "TRIGGERING EVENT" for that interrupt source, will always set the status bit. A write-1-to-clear input from the IPS bus, will clear the status bit, but only if there is not a simultaneous triggering event. The triggering event prevails over a software clear attempt, if both events coincide. An integrated clock gate guarantees that the status bit only sees a clock when either a triggering event occurs, or a write-1-to-clear pulse arrives from the IPS bus. This reduces interrupt status bit power consumption to an absolute minimum. (Note: the TSM_IRQ does not have an associated status bit in the Link Layer Controller; its status bit resides in the Transceiver Sequence Manager, or TSM).

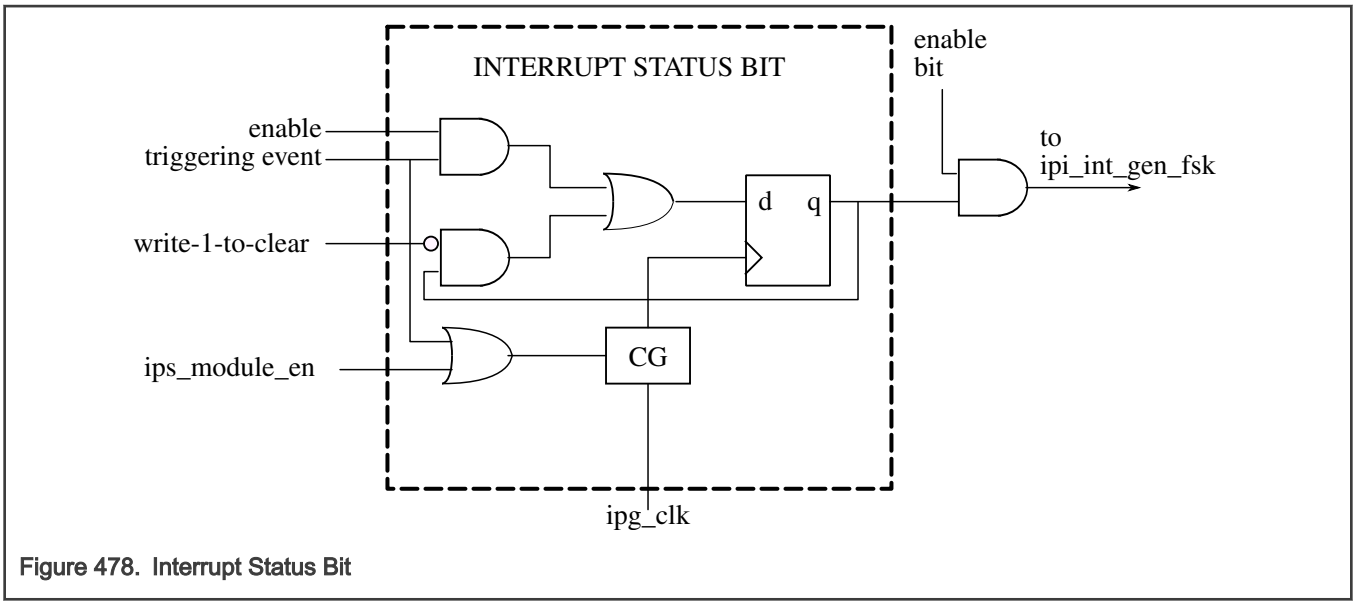


Figure 478. Interrupt Status Bit

Each GENERIC_FSK interrupt status bit, has an ENABLE bit associated with it. The name of the enable bit tracks the name of the interrupt source, with the string "_EN" appended at the end. Note that for any GENERIC_FSK interrupt source, if the triggering event occurs, the Interrupt Status Bit will be set regardless of the state of the corresponding ENABLE bit. If any of the 15 interrupts is to be ignored, software should clear the corresponding ENABLE bit, and apply the appropriate bit mask when reading the IRQ_CTRL register, to mask out the unwanted status bit.

GENERIC_FSK Interrupt Architecture

The 10 interrupt sources (status bits) are combined with their individual ENABLE bits, and then logically OR'ed together, in sum-of-products fashion, to generate single interrupt line to the MCU: **ipi_int_gen_fsk**. A global mask bit, GENERIC_FSK_IRQ_EN, can enable or disable **ipi_int_gen_fsk** altogether. There is no prioritization of interrupt sources; they have equal weight. The **ipi_int_gen_fsk** output is a level-sensitive indicator and will remain asserted until all interrupt status bits are cleared (or the corresponding ENABLE bits cleared). The following diagram depicts the GENERIC_FSK Interrupt Architecture.

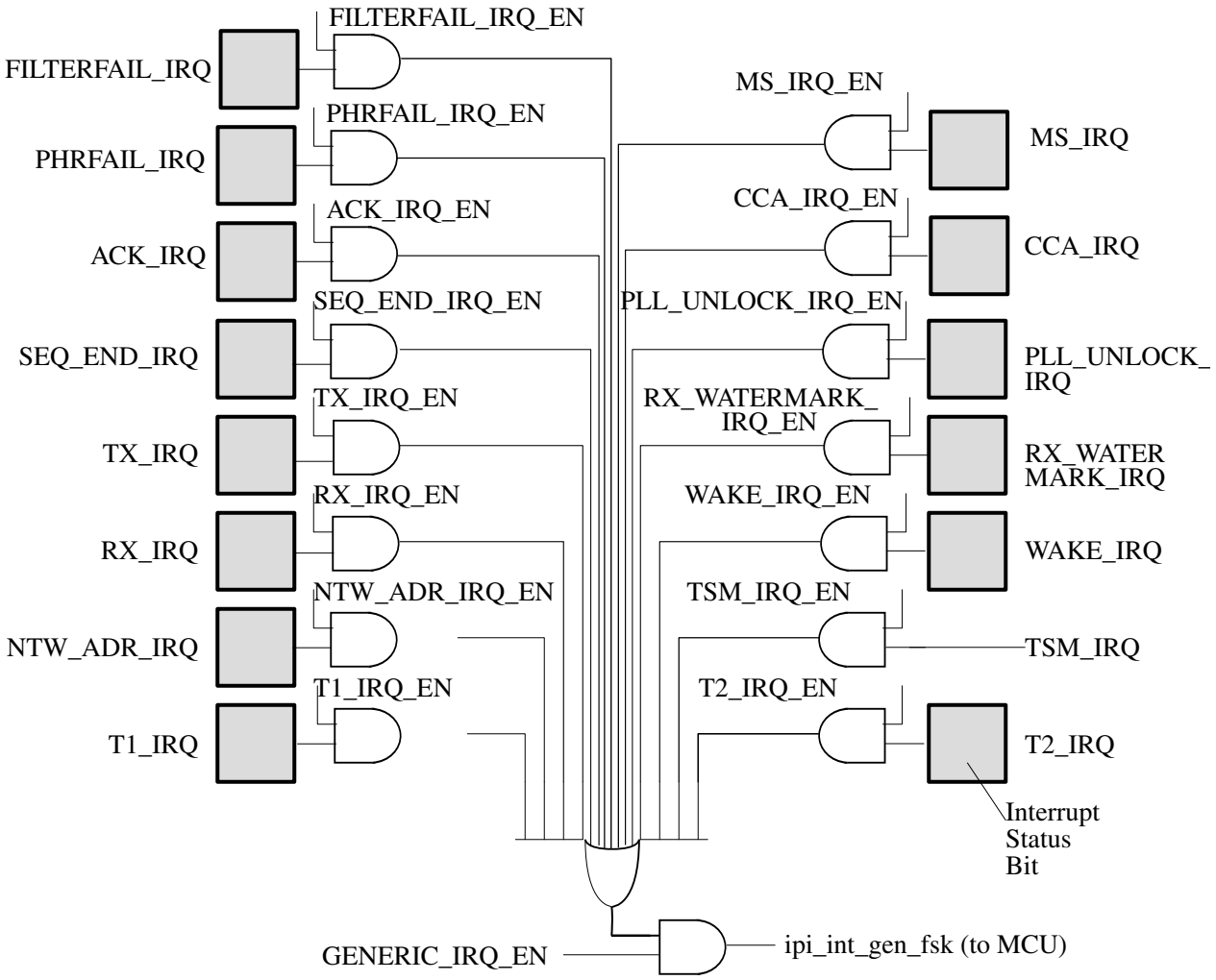


Figure 479. GENERIC_FSK Interrupt Architecture

Clearing Interrupts

All interrupt status bit use a write-1-to-clear protocol. Writing a '1' to the interrupt status bit, in the IRQ_CTRL register, clears the offending interrupt. Writing a '0' to an interrupt status bit has no effect on the bit. Interrupt status bits are not affected by reads.

GENERIC FSK Interrupt Sources

The following table describes the GENERIC_FSK Interrupt Sources

Interrupt	Description
T1_IRQ, T2_IRQ	The GENERIC_FSK Link Layer features a 24-bit Event Timer, which runs at a rate of 1MHz. The Link Layer has 2 Timer interrupts (T1_IRQ, T2_IRQ), each with its own 24-bit compare register (T1_CMP, T2_CMP), and each with its own compare enable (T1_CMP_EN, T2_CMP_EN). For each timer compare enable, if the bit is set, a match on the respective 24-bit compare value to the Event Timer will cause the corresponding interrupt status bit to become set. If the compare enable is

Table continues on the next page...

Table continued from the previous page...

Interrupt	Description
	not set, Event Timer matches won't cause the corresponding interrupt status bit to become set.
TX_IRQ	TX interrupt occurs at the end of a TX operation.
RX_IRQ	RX interrupt occurs at the end of a successful RX packet reception. A successful packet reception implies CRC was verified to be good, H0 matching passed, H1 matching passed, and received packet length did not violate pre-set limits. Nominally, RX interrupts are not generated on packets which fail CRC check, or which failure the H0, H1 or LENGTH limits. A received packet with a failed CRC check or a H0, H1, or LENGTH violation, results in an RX recycle back to the Network Address Search state. To receive a Data Indication (RX_IRQ) on packets which fail CRC, set the bit CRC_IGNORE=1. H0 and H1 matching, and LENGTH limiting, are under SW control.
NTW_ADR_IRQ	A Network Address Match has occurred on an enabled Network Address. At NTW_ADR_IRQ, a timestamp is captured by transferring the current contents of the EVENT_TMR into the TIMESTAMP register. Software can determined which Network Address matched by querying the NTW_ADR_MCH[3:0] field of the NTW_ADR_CTRL register.
SEQ_END_IRQ	The Sequence End Interrupt (SEQ_END_IRQ), indicates that a transceiver operation (TX or RX) has completed, and the Command Decoder State Machine, and the Transceiver Sequence Manager (TSM), have returned to their respective IDLE states. A SEQ_END_IRQ will always occur at the end of a sequence, even if the sequence terminated abnormally (such as a Software Abort, or a PLL Unlock Abort). A SEQ_END_IRQ always occurs whenever the Command Decoder FSM transitions from any non-idle to IDLE state. When SEQ_END_IRQ occurs, software can be sure that the Command Decoder and TSM are in their idle state, and a new sequence can be programmed.
RX_WATERMARK_IRQ	RX Watermark interrupt will occur during packet reception, when the number of received bytes, as indicated by the read-only BYTE_COUNTER register, matches the contents of the RX_WATERMARK register. For the purpose of defining RX_WTR_MARK, the first byte received (BYTE_COUNTER = 0) is the first byte of Network Address; the second byte received (BYTE_COUNTER = 1) is the second byte of Network Address, etc. To cause an RX Watermark Interrupt to occur after a 2-byte Network Address has been received, set RX_WTR_MARK=1. By default, RX_WATERMARK is set to a value larger than the longest supported GENERIC_FSK packet length, so an RX_WATERMARK_IRQ won't be triggered under these conditions.

Table continues on the next page...

Table continued from the previous page...

Interrupt	Description
PLL_UNLOCK_IRQ	When an PLL unlock event occurs during an transceiver operation, and the transceiver has been configured to automatically abort sequences on PLL unlock events, the PLL_UNLOCK_IRQ status bit will become set. The Transceiver Sequence Manager (TSM) will begin monitoring for PLL unlock only after the PLL has been given sufficient time to achieve lock; A PLL unlock which occurs after the warmup period, can be enabled to cause a sequence abort. Individual enables are provided for TX and RX sequences. (See the TSM Chapter for more details).
WAKE_IRQ	**For Manual DSM:** This is WAKE_IRQ. A WAKE_IRQ will be triggered when the GENERIC_FSK Link Layer Controller has awoken from a Manual DSM (Deep Sleep Mode) cycle. WAKE_IRQ indicates that the RF Oscillator has been restarted, and the GENERIC_FSK EVENT_TMR has resumed counting. **For Wake-On-Radio:**: This is WOR_IRQ. See Section Wake-On-Radio
TSM_IRQ	TSM_IRQ is a debug feature, enabling the Transceiver Sequence Manager (TSM) to generate an interrupt at any point in a TX or RX Warmup. TSM is a multipurpose hardware resource shared by all of the protocol engines in the SoC. Thus, TSM does not have its own interrupts; its interrupts are assigned to whichever link layer controller is currently executing an RF operation. TSM Interrupt Status bits reside in Transceiver (XCVR) Address Space, and can be cleared there. There is no intended mission-mode use for TSM_IRQ. See the TSM Chapter for more details.
ACK_IRQ	ACK_IRQ is asserted when RX finishes, and an auto ACK frame is scheduled to TX. At this time, software may need to write packet RAM to prepare the ACK frame or part of the ACK frame.
PHRFFAIL_IRQ	PHRFFAIL_IRQ is asserted when the received frame fails at BCH check or Parity check, for LECIM FSK and SUN FSK Mode Switch frame only.
FILTERFAIL_IRQ	FILTERFAIL_IRQ indicates that the most-recently received packet has been rejected due to elements within the packet. In Dual PAN mode, FILTERFAIL_IRQ applies to either or both networks, as follows: A: If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_IRQ applies to PAN0.

Table continues on the next page...

Table continued from the previous page...

Interrupt	Description
	B: If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_IRQ applies to PAN1. C: If PAN0 and PAN1 occupy the same channel, FILTERFAIL_IRQ is the logical 'AND' of the individual PANs' Filter Fail status.
CCA_IRQ	CCA_IRQ indicates completion of CCA operation
MS_IRQ	MS_IRQ indicates a mode switch frame is received with no BCH or Parity error.

55.4.9.2.3.10 Memory Mapping

In the SoC memory map, space has been allocated to GENERIC_FSK. Within the SoC, peripherals are allocated memory space in 4KB blocks, called IPS slots. GENERIC_FSK occupies one such slot. GENERIC_FSK address space consists of the GENERIC_FSK registers. The partitioning of address space for GENERIC_FSK is shown in the diagram below.

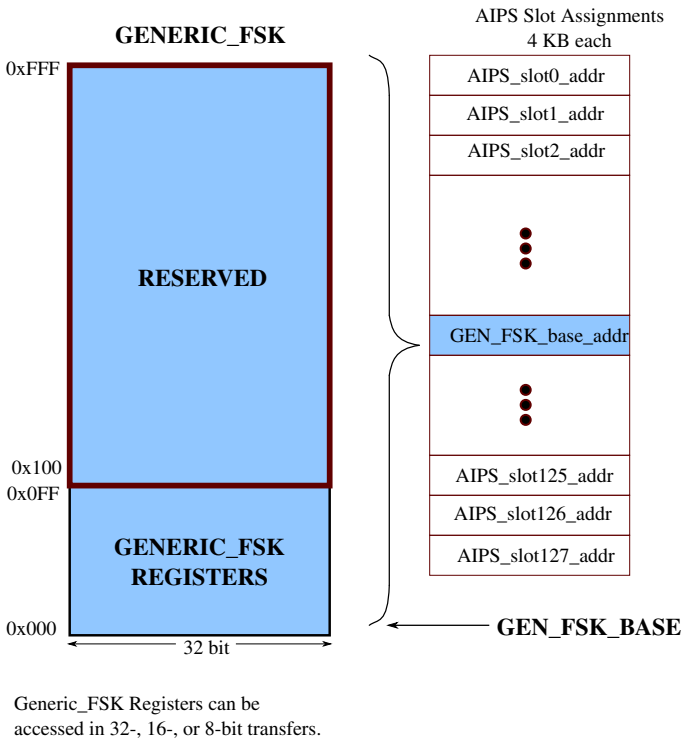


Figure 480. GENERIC_FSK MEMORY MAPPING

NOTE

See AIPS Memory Map for specific memory assignments details.

All registers within GENERIC_FSK memory space, are accessible by the MCU in 8-, 16-, or 32-bit accesses. For efficiency, 32-bit accesses are recommended.

55.4.9.2.3.11 Manual DSM

When the GENERIC_FSK Link Layer anticipates long periods of inactivity, the Link Layer controller can be put into a Deep Sleep Mode (DSM), where all of its clocks are turned off. In addition, the GENERIC_FSK Link Layer can be configured to optionally turn off the RF Oscillator, if it is not otherwise needed by the SoC. Deep Sleep Mode results in dramatic power savings. The procedure described below, is called Manual DSM.

To make Manual DSM possible, the transceiver incorporates a low-frequency, high-precision Deep Sleep Timer, TIMER. This timer is clocked by a crystal-referenced 32.768KHz oscillator. The timer is 24-bit wide, yielding a 8.5 minute rollover (the maximum length of a deep sleep period).

The hardware to implement DSM within the SoC, is partitioned across several modules, and several power domains. This is to allow the maximum achievable power savings, by retaining full voltage for only those modules which need to be active in DSM, and placing the remaining modules in "state retention" mode. The diagram below depicts the partitioning of DSM hardware across the various modules, including the GENERIC_FSK Link Layer Controller.

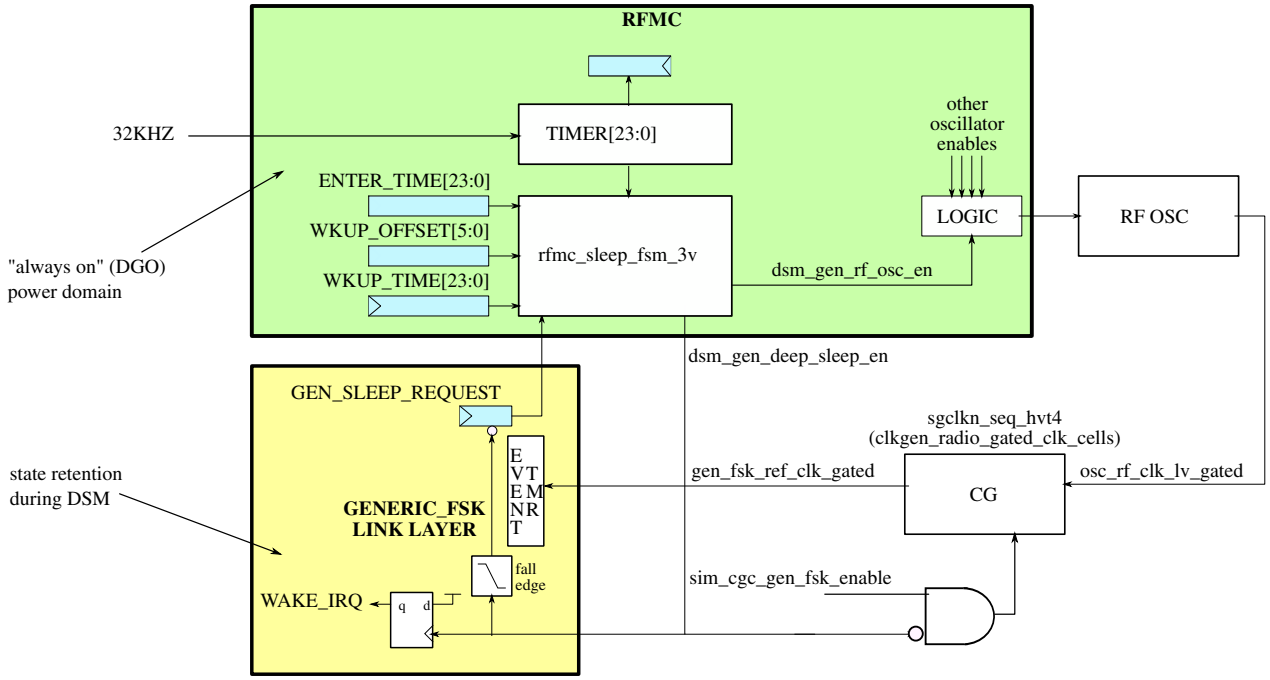


Figure 481. MANUAL DSM BLOCK DIAGRAM

The concept behind Manual DSM, as it is implemented for GENERIC_FSK, is to allow the GENERIC_FSK timebase to be maintained for a period of time, while the RF Oscillator is turned off and much of the SoC is placed in state-retention. During DSM, the timebase management is temporarily transferred from the GENERIC_FSK EVENT_TMR to the low-power TIMER, and after DSM exit, timebase management is transferred back to the EVENT_TMR, under software control. During DSM, the EVENT_TMR is frozen. The total time spent in DSM is known to software, so that upon exiting DSM, the EVENT_TMR can be updated to correct for the precise amount of time it was frozen.

DSM entry and exit, and RF Oscillator re-start during Manual DSM, are governed by 5 registers, as described in the table below.

Field	R/W	Description
ENTER_TIME[23:0]	rw	If GEN_SLEEP_REQUEST=1, enter DSM & freeze EVENT_TMR when ENTER_TIME matches TIMER.

Table continues on the next page...

Table continued from the previous page...

Field	R/W	Description
		NOTE This register resides in RFMC Address Space
WKUP_OFFSET[5:0]	rw	Determines the power and RF oscillator re-start times relative to WKUP_TIME **NOTE** This register resides in RFMC Address Space
WKUP_TIME[23:0]	rw	Exit Deep Sleep Mode, and resume EVENT_TMR, when WKUP_TIME matches TIMER **NOTE** This register resides in RFMC Address Space
GEN_SLEEP_REQUEST	w	Enable a match on ENTER_TIME[23:0] to TIMER[23:0], to enter Deep Sleep Mode, by writing a 1 to this bit. Cancel a pending sleep request by writing a 0 to it, if already set, at least 2 32.768KHz clock cycles prior to ENTER_TIME[23:0]
TIMER[23:0]	r	Current State of the 32KHz Sleep Timer **NOTE** This register resides in RFMC Address Space

Procedure

The procedure to schedule a Manual DSM cycle (entry/exit) as well as to program the desired RF Oscillator re-start time, and the calculation to restore the GENERIC_FSK EVENT_TMR after a DSM cycle, is described below.

1. Software determines the future time at which it would like to enter DSM, by reading the TIMER, computing the number of 32KHz clock cycles remaining until the desired DSM start-time, and writing this value into ENTER_TIME register. The value programmed into ENTER_TIME should be no fewer than 4 clocks greater (in the future) than the current time as read from TIMER.
2. Software determines the future time at which it would like to exit DSM, by computing the number of 32KHz clock cycles remaining until the desired DSM exit-time, and writing this value into WKUP_TIME register.
3. Software writes a 1 to DSM_CTRL[GEN_SLEEP_REQUEST]. This bit resides in GENERIC_FSK address space. Writing this bit to 1 enables a DSM cycle to commence using the values programmed into ENTER_TIME and WKUP_TIME.
4. MCU can go into a low power state (e.g., STOP, VLPS, or VLLSx)
5. When TIMER = ENTER_TIME, the EVENT_TMR will be clock-gated so that it will remain frozen at its current count. The RF Oscillator will be turned off (unless overridden due to a non-Generic-related SoC requirement)
6. All hardware not in the low-voltage domain will be placed into state-retention, or other low-power state.

7. When TIMER = WKUP_TIME, clocking of the GENERIC_FSK Link Layer Controller will resume and the EVENT_TMR will be ungated (allowed to resume counting).
8. Software computes the time that the SoC was in DSM (and hence the time the EVENT_TMR was frozen), in microseconds, with the equation:

   (WKUP_TIME - ENTER_TIME) / 32768 * 1000000
9. Software increments the GENERIC_FSK EVENT_TMR by this amount, by writing this value to EVENT_TMR[23:0] register with EVENT_TMR_ADD=1.
10. On the next 1MHz clock edge, the EVENT_TMR has been restored to where it would have been, had no DSM occurred, with 1.0-microsecond accuracy.

Note that the 1.0-microsecond accuracy quoted above, assumes an ideal 32.768KHz source. Any ppm error in the 32.768KHz source will degrade the microsecond accuracy proportionally to the magnitude of the error.

Once GEN_SLEEP_REQUEST has been set to 1 to request Manual DSM, the request can be cancelled by writing a 0 to the bit. The cancellation must be performed at least 2 32.768KHz before the ENTER_TIME[23:0] match to TIMER[23:0] to avoid a race condition to enter DSM.

The following timing diagram depicts an GENERIC_FSK-driven DSM entry/exit cycle.

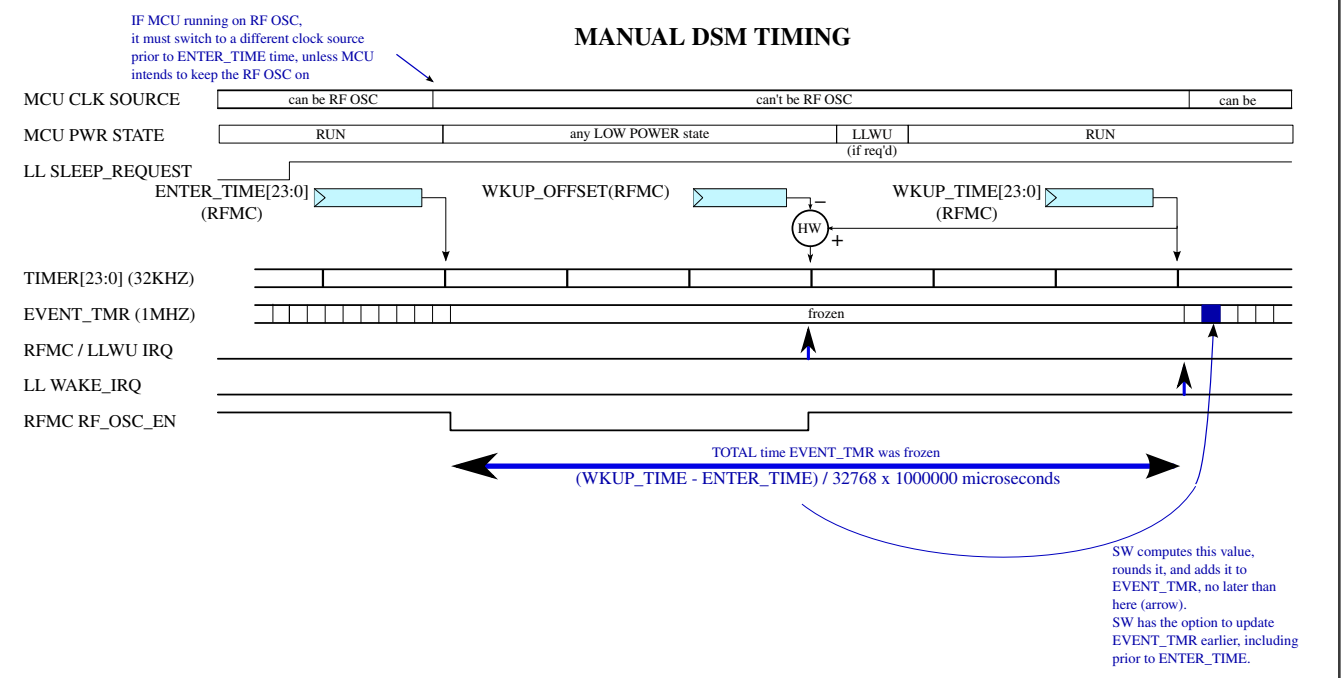


Figure 482. GENERIC_FSK DEEP SLEEP TIMING

DSM Early Exit

Circumstances may arise where the Manual DSM cycle may occasionally need to be cut short, due to circumstances beyond the control of the radio. When this happens, the SoC, starting with the MCU, is awakened out of its low power state, long before the RFMC TIMER reaches WKUP_TIME. Therefore, software handling the event which triggered the wakeup, will determine if the radio needs to be awakened as well. If this is the case, software will advance the WKUP_TIME setting to a point in the near future, but no fewer than 4 32KHz clock cycles beyond the current time ascertained by reading TIMER. So, to execute a Manual DSM Early Exit, software shall read TIMER, add +4 to the value, and write the sum to the RFMC WKUP_TIME register. Software shall then wait for the GENERIC_FSK WAKE_IRQ interrupt, which will assert once the full radio wakeup from the low power state has completed. Software should then take whatever steps are necessary to configure the radio to respond to the conditions which triggered the early exit. Servicing of a DSM early exit interrupt is beyond the scope of this document.

55.4.9.2.3.12 Wake-On-Radio

The GENERIC_FSK Link Layer can take advantage of the automated sequence- and DSM-scheduling provided by Wake-On-Radio. Wake-On-Radio is one of two DSM control schemes that can be selected for GENERIC_FSK, as an alternative to Manual DSM. See Chapter *Wake-On-Radio*.

Several changes to the GENERIC_FSK Link Layer Controller have been made to support Wake-On-Radio.

Interrupt WAKE_IRQ has been made dual-purpose. If Wake-On-Radio is disabled, i.e., WOR_CTRL[WOR_EN]=0, or Wake-On-Radio not assigned to GENERIC_FSK, WAKE_IRQ performs its legacy function for Manual DSM. (See Section Manual DSM). If Wake-On-Radio is enabled, i.e., WOR_EN=1, and assigned to GENERIC_FSK, this interrupt becomes WOR_IRQ, and has new functionality specific to Wake-On-Radio. See Chapter *Wake-On-Radio*, Section *Interrupt Management*, for more details on the meaning, and handling of WOR_IRQ. The Interruptssection of this chapter describes the dual-purpose nature of WAKE_IRQ / WOR_IRQ.

In addition to WOR_IRQ, numerous interface changes have been made to support Wake-On-Radio. The changes allow, for example, WOR to manage GENERIC_FSK's RF interrupts, inhibit timestamp updates, etc. The added port signals are listed in the table below, along with a description or each:

WAKE-ON-RADIO INTERFACE SIGNAL	DIRECTION	DESCRIPTION
wor_event_tmr_add[23:0]	WOR -> GENERIC_FSK	The Event Timer "Addback" value computed by WOR to compensate for the amount of time Event Timer was frozen during DSM
wor_addback	WOR -> GENERIC_FSK	A 1µs-long pulse commanding the GENERIC_FSK Event Timer to auto-increment by the the amount appearing on wor_event_tmr_add[23:0]
wor_force_wake_irq	WOR -> GENERIC_FSK	Force WOR_IRQ to assert, due to an occurrence of one of the 3 WOR_IRQ interrupt triggers
wor_block_wake_irq	WOR -> GENERIC_FSK	Inhibit the legacy WAKE_IRQ assertion when WOR_EN=1 (this interrupt becomes WOR_IRQ)
tx_start_ext	WOR -> GENERIC_FSK	A 1µs-long pulse to the GENERIC_FSK command decoder to initiate a WOR-driven TX sequence
rx_start_ext	WOR -> GENERIC_FSK	A 1µs-long pulse to the GENERIC_FSK command decoder to initiate a WOR-driven RX sequence
seq_stop_ext	WOR -> GENERIC_FSK	A 1µs-long pulse to the GENERIC_FSK command decoder to stop (abort) a WOR-driven RX sequence
wor_allow_irq	WOR -> GENERIC_FSK	Inhibits RF interrupts, so that interrupt assertions can be managed based on WOR descriptor settings. Controlled dynamically by the WOR state machine.
timestamp_inhibit	WOR -> GENERIC_FSK	In Wake-On-Radio, the legacy GENERIC_FSK TIMESTAMP register

Table continues on the next page...

Table continued from the previous page...

WAKE-ON-RADIO INTERFACE SIGNAL	DIRECTION	DESCRIPTION
		becomes WOR TIMESTAMP0 (timestamp for SLOT0). This signal blocks this register from updating on SLOTS 1-3
wake_irq_trig	WOR <- GENERIC_FSK	WOR uses this signal to cancel GEN_SLEEP_REQUEST and transition from IN_DSM -> COMP_ADDBACK state
event_tmr[23:0]	WOR <- GENERIC_FSK	GENERIC_FSK Event Timer, used by WOR for timestamp capture
event_tmr_plus_1[23:0]	WOR <- GENERIC_FSK	GENERIC_FSK Event Timer + 1, used by WOR to schedule RF operation starts and stops
tx_irq_trig	WOR <- GENERIC_FSK	TXIRQ trigger, indicates the completion of a packet transmission, used by WOR to transition out of WAKE_SEQ_END state
rx_irq_trig	WOR <- GENERIC_FSK	RXIRQ trigger, indicates the completion of a good packet reception, used by WOR to transition out of WAKE_SEQ_END state
rx_status_for_wor[2:0]	WOR <- GENERIC_FSK	Internal RX status signals used by WOR to formulate TIMESTAMPx_STS[2:0]. **NOTE** See Section Timestamp Status for Wake-On-Radio following this table
timestamp_clk_en	WOR <- CGM <- GENERIC_FSK	Timestamp clock gate enable, the resultant clock is used by WOR to update the TIMESTAMP1,2, and 3 registers

Timestamp Status for Wake-On-Radio

The GENERIC_FSK Link Layer has 5 methods for filtering and screening packets as they are being received:

H0 filtering

H1 filtering

LENGTH_MAX filtering

PHR filtering(PAN or FAN mode)

MAC rule filtering(PAN or FAN mode)

In Normal operation, when one or more of the filtering rules are violated, an RX recycle will occur immediately after the header, and the cause of the filtering violation (status bit) will not be preserved. In addition, the final CRC status can't be known at the point of recycle. In Wake-On-Radio operation, The TIMESTAMPx_STS[1] bit reflects CRC status, and TIMESTAMPx_STS[2] bit reflects Packet Filter status. For an RX recycle at the end of the packet header, the GENERIC_FSK Link Layer won't be able to provide the necessary signalling to Wake-On-Radio in order to properly set those 2 bits of each TIMESTAMPx_STS register, so these bits should be considered invalid. To render them valid, set the GENERIC_FSK PACKET_CFG[REC_BAD_PKT]=1, which will force reception of the entire packet, including CRC, before recycling. Set RX_FRAME_FILTER[FILTER_FAIL_IGNORE] to ignore MAC rule filtering fail and set PHR_MISC[PHR_FAIL_IGNORE] to ignore PHR filtering rule fail. This will allow the Wake-On-Radio TIMESTAMPx_STS[2:1] bits to yield valid information.

55.4.9.2.3.13 Multi-protocol Arbitration

The GENERIC_FSK Link Layer controller may be included in an SoC that includes similar controllers for other protocols (e.g., Bluetooth Low Energy). GENERIC_FSK may be required to operate simultaneously with other protocols on the SoC, and there are scenarios where both protocol engines require access to the RF channel at the same time.

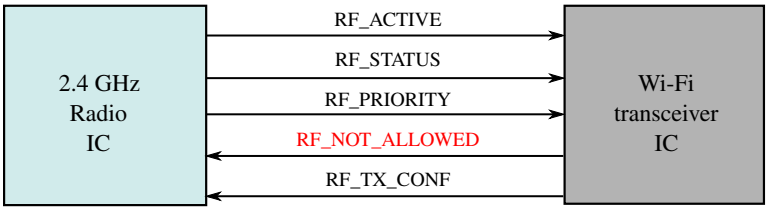
To assist in software arbitration between GENERIC_FSK and other protocols, the read-only status but XCVR_BUSY has been included in the XCVR_STS register. When this bit is set, this indicates that the RF channel is currently in use (TSM is busy), by one of the Radio's four link layer controllers. When this bit it clear, the RF channel is available to GENERIC_FSK (TSM is idle).

55.4.9.2.3.14 Wifi Coexistence

Provisions have been made to allow for the 2.4GHz transceiver to coexist in the same space as a WiFi transceiver IC, which shares the same frequency band. The coexistence scheme designates the WiFi transceiver as the master, and the 2.4GHz transceiver as the slave. The objective of the coexistence strategy is to prevent both the WiFi and 2.4GHz transceivers transmitting simultaneously; a configuration option exists to also prevent the 2.4GHz transceiver from receiving when the WiFi transceiver owns the RF channel.

The 2.4GHz Radio IC support the coexistence interface signals as shown in the figure below, though in a typical application only a subset of these are used.

The WiFi IC generates a signal, RF_NOT_ALLOWED. If this signal is asserted, then 2.4GHz radio does not perform any communication. When this signal is de-asserted, then 2.4GHz radio is free to perform communications.



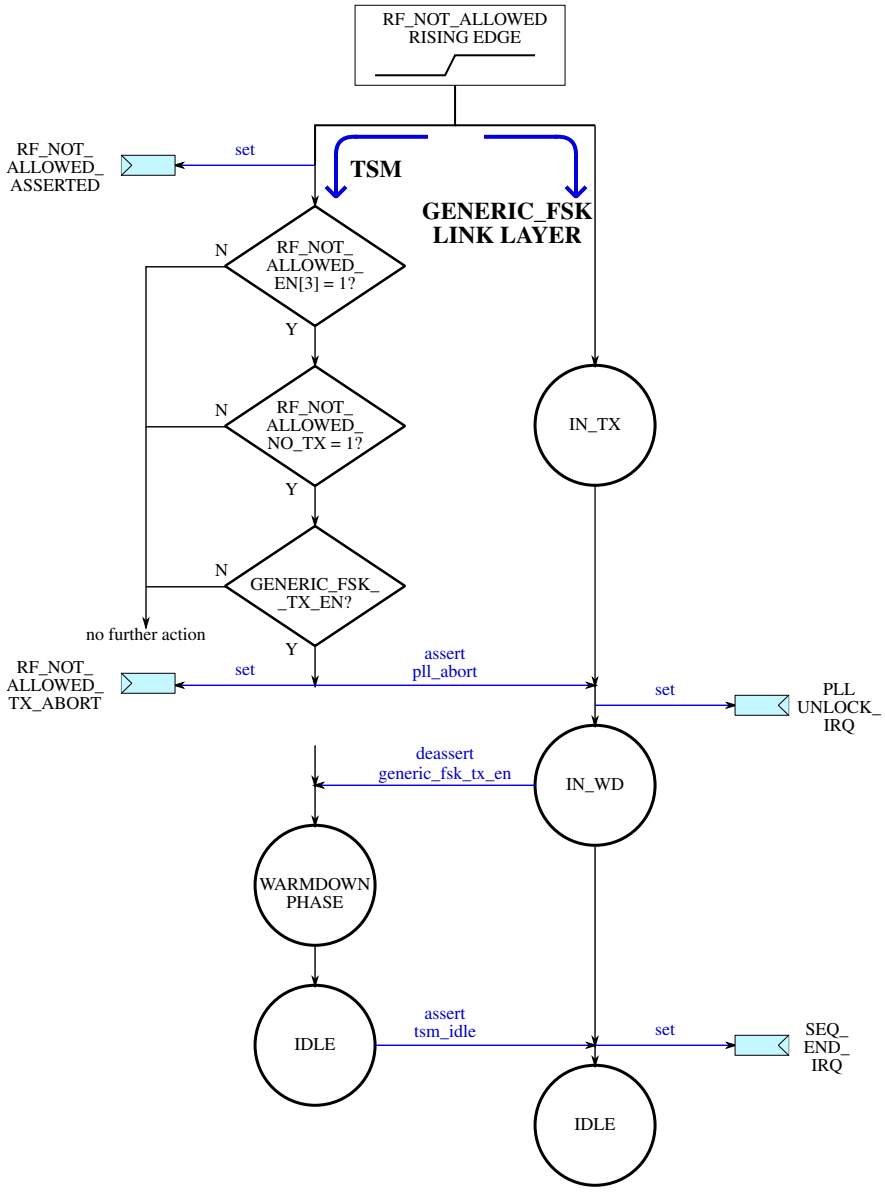
When RF_NOT_ALLOWED aborting is enabled, the GENERIC_FSK Link Layer hardware must respond to RF_NOT_ALLOWED assertions, by aborting any TX or RX sequence which is currently underway, and Link Layer software must not initiate any new sequences until RF_NOT_ALLOWED has been deasserted. The hardware response must be handled autonomously by the Link Layer, with no MCU intervention required. This is because the RF_NOT_ALLOWED assertions can occur during a Wake-On-Radio timeslot which has been designated as "no-MCU-wakeup" in the slot's descriptor. In this scenario, the aborting of the sequence occurs without any assistance from, or notification of, the MCU.

In the multi-protocol 2.4GHz radio, RF_NOT_ALLOWED aborting can be individually enabled/disabled for each protocol engine. For GENERIC_FSK, RF_NOT_ALLOWED_EN[3] is the associated enable control bit. This bit resides in the COEX_CTRL register in XCVR address space. When this bit is 0, transitions on RF_NOT_ALLOWED are ignored by the GENERIC_FSK Link Layer hardware; when this bit is 1, the GENERIC_FSK Link Layer hardware will monitor RF_NOT_ALLOWED at all times, and abort any active sequence which is underway when an assertion on the pin occurs. The complete hardware response to RF_NOT_ALLOWED assertions is described below.

Additional control over RF_NOT_ALLOWED aborting is provided by the RF_NOT_ALLOWED_NO_TX and RF_NOT_ALLOWED_NO_RX control bits. If RF_NOT_ALLOWED_NO_TX=1, then an RF_NOT_ALLOWED abort will occur only if a GENERIC_FSK TX sequence is underway (generic_fsk_tx_en=1). GENERIC_FSK TX sequences will not be aborted if RF_NOT_ALLOWED_NO_TX=0. If RF_NOT_ALLOWED_NO_RX=1, then an RF_NOT_ALLOWED abort will occur only if GENERIC_FSK RX sequence is underway (generic_fsk_rx_en=1). GENERIC_FSK RX sequences will not be aborted if RF_NOT_ALLOWED_NO_RX=0. The RF_NOT_ALLOWED_NO_TX and RF_NOT_ALLOWED_NO_RX control bits reside in the COEX_CTRL register.

For the purposes of triggering a hardware abort, the *pll_abort* input to the GENERIC_FSK Link Layer hardware is used. This is because the hardware response to the RF_NOT_ALLOWED assertion is identical to that of a PLL unlock event. This also means that, when RF_NOT_ALLOWED aborting is enabled for GENERIC_FSK the PLL_UNLOCK_IRQ interrupt status bit, will be dual purpose: it will not only indicate a PLL unlock condition, but also a RF_NOT_ALLOWED abort. PLL aborting and RF_NOT_ALLOWED aborting are enabled separately, with the COEX_CTRL register maintaining the control bits required for the latter. The status bits for RF_NOT_ALLOWED aborting are also available in COEX_CTRL, so that software will be able to distinguish the source of the PLL_UNLOCK_IRQ, in case both PLL and RF_NOT_ALLOWED aborting are enabled.

The sequence of events which results in a hardware abort of an GENERIC_FSK TX operation triggered by an assertion on RF_NOT_ALLOWED, is a collaboration between the TSM (Transceiver Sequence Manager) and the GENERIC_FSK Link Layer hardware, as shown in the following diagram.



Upon assertion on RF_NOT_ALLOWED, the TSM sets the RF_NOT_ALLOWED_ASSERTED status bit in COEX_CTRL, then checks that the conditions for a hardware abort are all met:

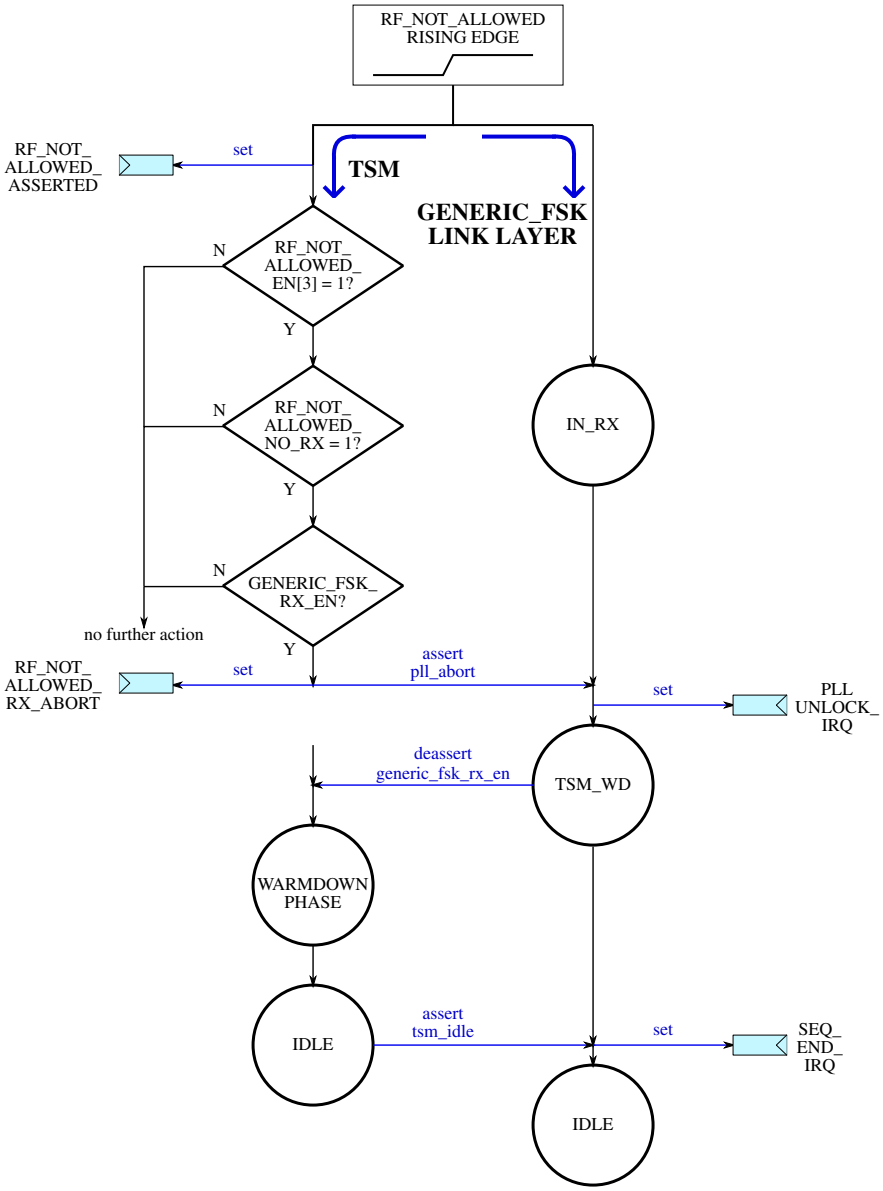
- RF_NOT_ALLOWED_EN[3] = 1, which enables GENERIC_FSK to respond to RF_NOT_ALLOWED events
- RF_NOT_ALLOWED_NO_TX = 1, which enables TX operations to be aborted
- generic_fsk_tx_en = 1, TX request to TSM from GENERIC_FSK, indicating TX operation in progress

If all conditions are not met, no further action is taken. Otherwise, the TSM sets COEX_CTRL[RF_NOT_ALLOWED_TX_ABORT], and also asserts *pll_unlock* to the GENERIC_FSK Link Layer hardware. GENERIC_FSK responds by asserting IRQ_CTRL[PLL_UNLOCK_IRQ] in GENERIC_FSK address space. GENERIC_FSK enters IN_WD state, which deasserts generic_fsk_tx_en to the TSM. This initiates the TSM TX warmdown. GENERIC_FSK holds in IN_WD to wait for TSM to return to idle. Once this occurs, GENERIC_FSK asserts IRQ_CTRL[SEQ_END_IRQ] and returns to its IDLE state. Three status bits are now set to indicate to software that the source of the abort was an RF_NOT_ALLOWED assertion:

- IRQ_CTRL[PLL_UNLOCK_IRQ]
- COEX_CTRL[RF_NOT_ALLOWED_ASSERTED]

- COEX_CTRL[RF_NOT_ALLOWED_TX_ABORT]

The sequence of events which results in a hardware abort of an GENERIC_FSK RX sequence triggered by an assertion on RF_NOT_ALLOWED, is shown in the following diagram.



Upon assertion on RF_NOT_ALLOWED, the TSM sets the RF_NOT_ALLOWED_ASSERTED status bit in COEX_CTRL, then checks that the conditions for a hardware abort are all met:

- RF_NOT_ALLOWED_EN[3] = 1, which enables GENERIC_FSK to respond to RF_NOT_ALLOWED events
- RF_NOT_ALLOWED_NO_RX = 1, which enables RX operations to be aborted
- generic_fsk_rx_en = 1, RX request to TSM from GENERIC_FSK, indicating RX operation in progress

If all conditions are not met, no further action is taken. Otherwise, the TSM sets COEX_CTRL[RF_NOT_ALLOWED_RX_ABORT], and also asserts *pll_unlock* to the GENERIC_FSK Link Layer hardware. GENERIC_FSK responds by asserting IRQ_CTRL[PLL_UNLOCK_IRQ] in GENERIC_FSK address space. GENERIC_FSK enters IN_WD state, which deasserts generic_fsk_rx_en to the TSM. This initiates the TSM TX warmdown. GENERIC_FSK holds in IN_WD to wait for TSM to return to idle. Once this occurs, GENERIC_FSK asserts IRQ_CTRL[SEQ_END_IRQ] and returns to its IDLE state. Three status bits are now set to indicate to software that the source of the abort was an RF_NOT_ALLOWED assertion:

- IRQSTS1[PLL_UNLOCK_IRQ]
- COEX_CTRL[RF_NOT_ALLOWED_ASSERTED]
- COEX_CTRL[RF_NOT_ALLOWED_RX_ABORT]

Chapter 56
Data Stream Buffer (DSB)

56.1 Introduction

The Data Stream Buffer (DSB) module captures streaming data from the radio and transfers the data to system memory with minimal intervention from a host processor. The hardware microarchitecture includes:

- An internal FIFO for temporary storage of the data stream
- A small Direct Memory Access (DMA) controller that performs:
  - Destination address calculations
  - Data-movement operations

The block diagram of DSB is shown in the figure below.

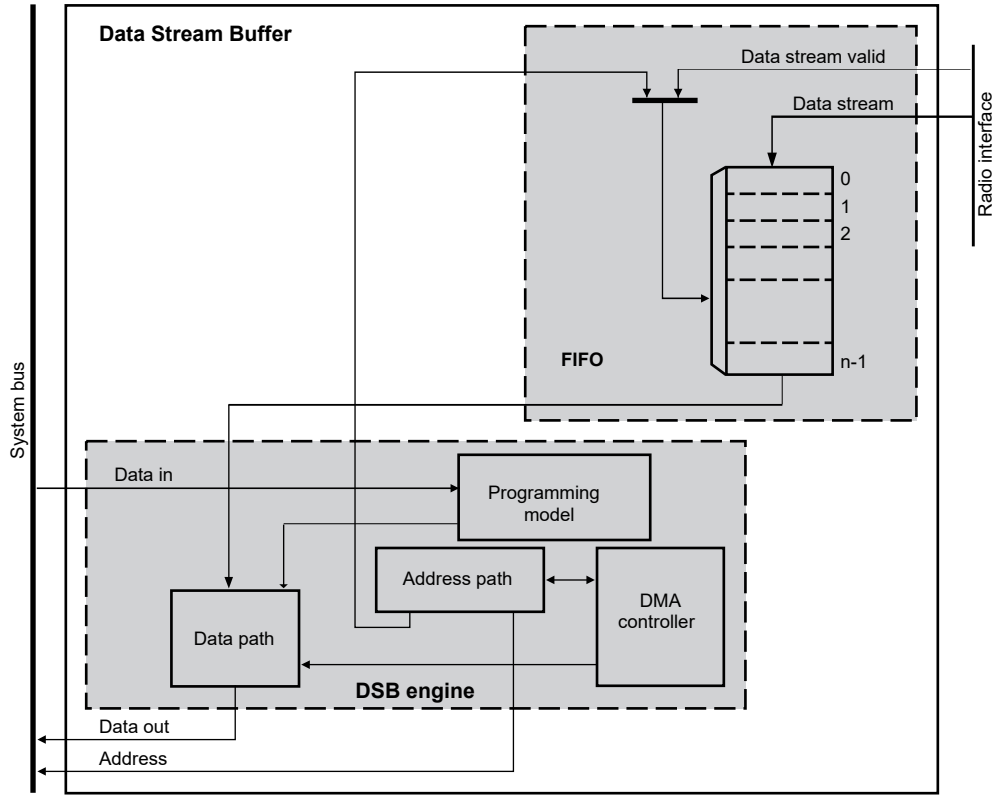


Figure 483. DSB block diagram

56.2 Memory map/register definition

The programming model of DSB is partitioned into four parts:

- The first part, Control/Status, defines a number of registers providing overall control functions.
- The second part corresponds to the interrupt status.

- The third part corresponds to the FIFO attributes.
- The fourth part corresponds to the FIFO read data.

DMA initialization

Before enabling the DMA logic, you must initialize the control fields of DMA with the appropriate transfer profile.

Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading a reserved memory location returns undefined.
- Writes to a reserved memory location is ignored.

56.2.1 DSB register descriptions

56.2.1.1 DSB memory map

DSB0 base address: 4004_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register (CSR)	32	RW	0000_0000h
4h	Interrupt Request Status Register (INT)	32	RW	0000_0000h
8h	Watermark Configuration Register (WMC)	32	RW	1000_0000h
Ch	FIFO Read Data Register (RDATA)	32	R	0000_0000h
10h	DMA Destination Address Register (DADDR)	32	RW	0000_0000h
14h	DMA Transfer Count Register (XCR)	32	RW	0000_0000h

56.2.1.2 Control Register (CSR)

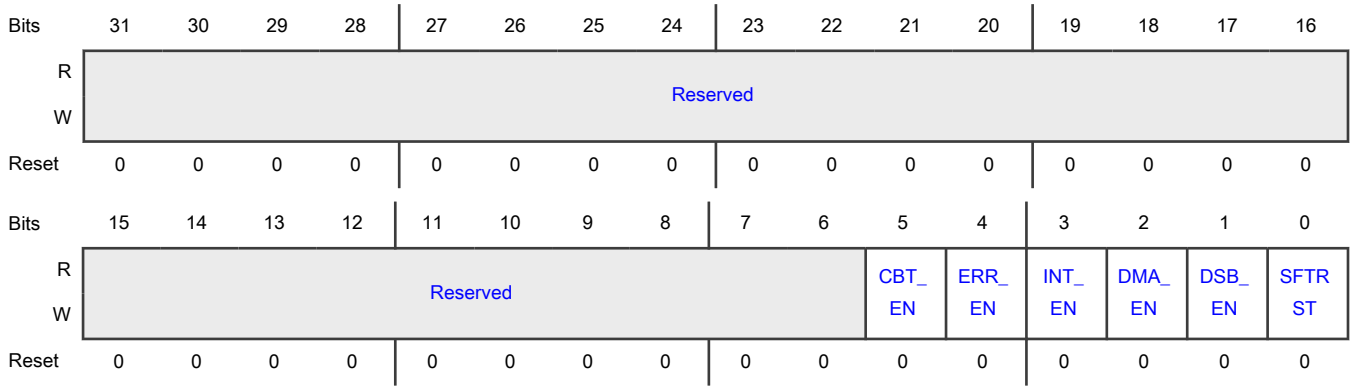
Offset

Register	Offset
CSR	0h

Function

The CSR register defines the basic operating configuration of DSB.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 CBT_EN	Continuous Burst Transfer Enable When enabled, continuous burst transfer streams data from the FIFO to the system bus until either the FIFO is empty or the transfer count, XCR[TCNT], has been achieved. The continous data transfer is executed as a single, undefined length burst transfer. When disabled, the burst transfer length is equal to the watermark value WMC[WMRK]. 0b - Continuous burst transfer mode is disabled. 1b - Continuous burst transfer mode is enabled.
4 ERR_EN	Error Interrupt Request Enable 0b - Error interrupt requests on overflow, underrun, or bus error are disabled. 1b - Error interrupt requests on overflow, underrun, or bus error are enabled.
3 INT_EN	Interrupt Request Enable 0b - Interrupt requests on data ready or DMA done are disabled. 1b - Interrupt requests on data ready or DMA done are enabled. Interrupt requests on data ready or DMA done are enabled. When the DMA is enabled (CSR[DMA_EN]=1), INT_EN enables interrupt requests when the packet transfer is complete (INT[DONE]=1). Otherwise, INT_EN enables interrupt requests when data is ready (INT[DRDY]=1).
2 DMA_EN	DMA Transfer Enable 0b - DMA transfers are disabled. 1b - DMA transfers are enabled.
1 DSB_EN	Data Stream Buffer Enable 0b - Buffer is disabled. 1b - Buffer is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 SFTRST	Soft Reset 0b - No operation. 1b - Reset the data stream buffer.

56.2.1.3 Interrupt Request Status Register (INT)

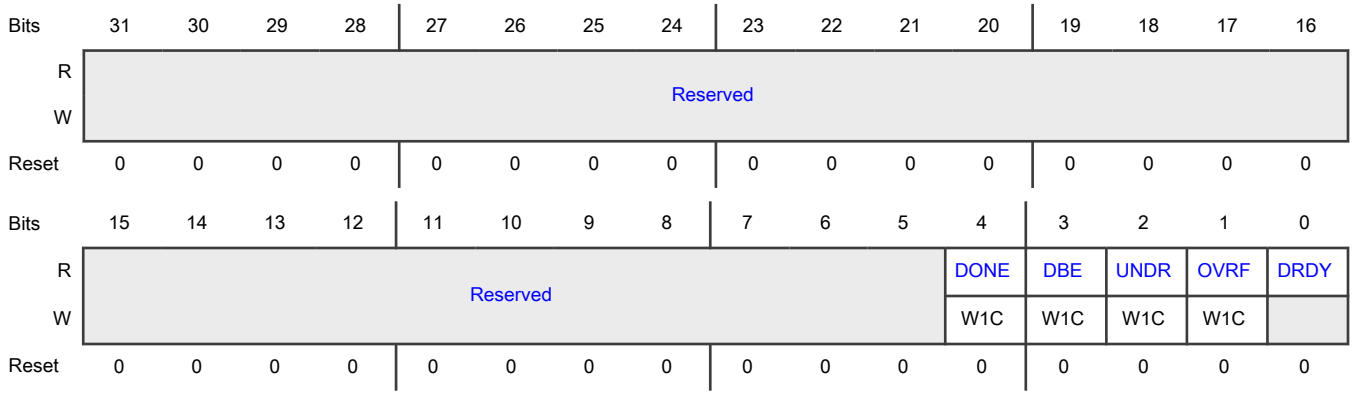
Offset

Register	Offset
INT	4h

Function

The INT register shows the current state of the interrupt service request sources for DSB.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 DONE	DMA Packet Transfer Complete 0b - Packet transfer not done; CCNT less than TCNT 1b - Packet transfer is done; TCNT 32-bit words transferred
3 DBE	Destination Bus Error 0b - No destination bus error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The last recorded error is bus error on a write
2 UNDR	Underrun Error 0b - No underrun error 1b - The last recorded error is an underrun on a read
1 OVRF	Overflow Error 0b - No overflow error 1b - The last recorded error is a buffer overflow
0 DRDY	Data Ready 0b - No data to read (watermark has not been reached) 1b - Data is ready to read (watermark has been reached)

56.2.1.4 Watermark Configuration Register (WMC)

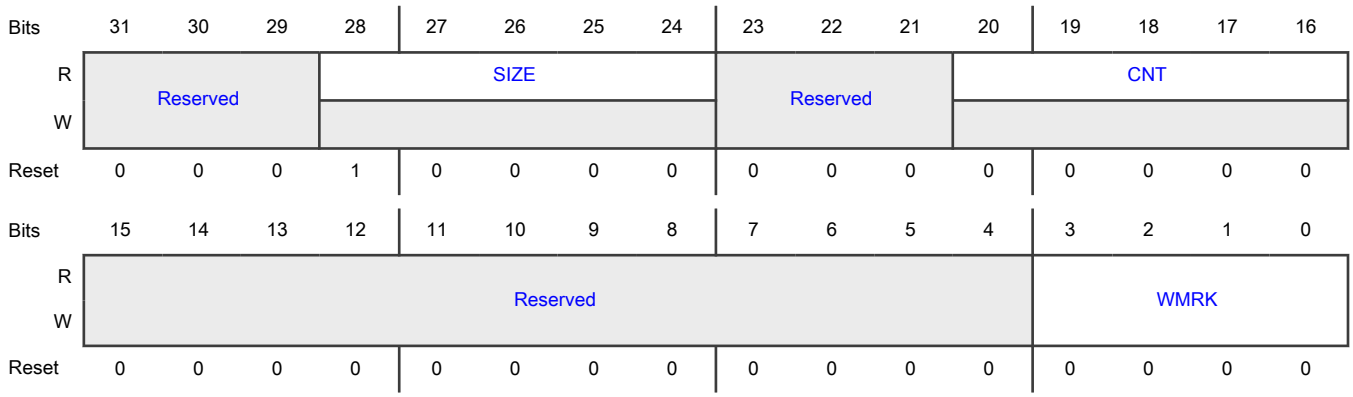
Offset

Register	Offset
WMC	8h

Function

The WMC register controls the DMA operations of DSB and indicates data is available via the INT[DRDY] bit. The WMC register shows the depth of the data stream FIFO and the number of elements written into the FIFO.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 SIZE	FIFO size The FIFO size indicates the number of elements the FIFO can hold.
23-21 —	Reserved
20-16 CNT	FIFO Count The FIFO Count indicates the number of elements currently in the FIFO.
15-4 —	Reserved
3-0 WMRK	Watermark DSB starts DMA transfers, if enabled, and sets DRDY when the number of elements in the FIFO is equal to or greater than the watermark value.

56.2.1.5 FIFO Read Data Register (RDATA)

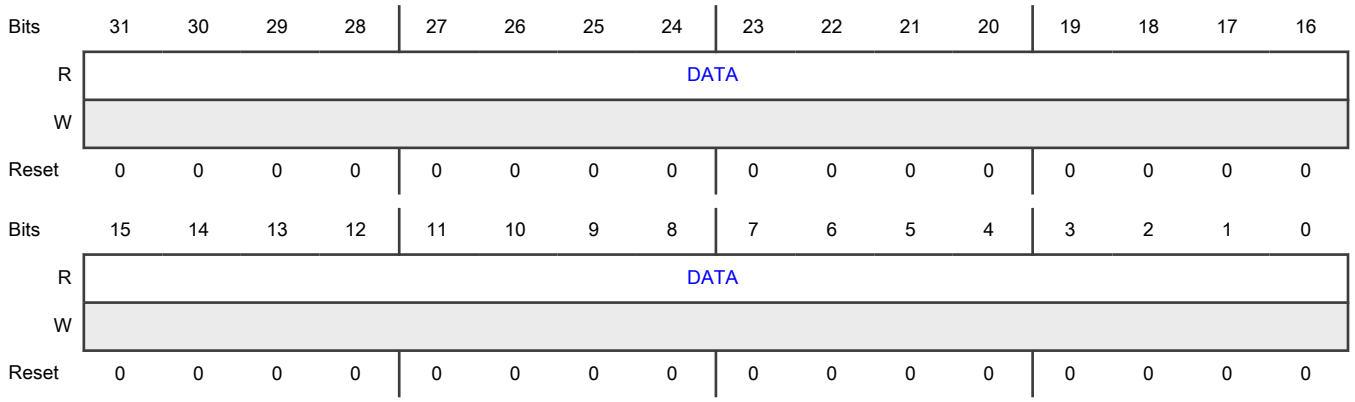
Offset

Register	Offset
RDATA	Ch

Function

The RDATA register is a read-only register that presents the next element in the FIFO. Reading the RDATA register pops the element off the FIFO stack.

Diagram



Fields

Field	Function
31-0 DATA	FIFO Data Read data from FIFO.

56.2.1.6 DMA Destination Address Register (DADDR)

Offset

Register	Offset
DADDR	10h

Function

The DADDR register is the target address in system memory where the FIFO data is transferred to upon reaching the watermark value. The transfer profile to the destination address is undefined length INCR burst, 32-bits per beat. The number of beats is equal to the watermark value until TCNT is reached. This byte address register automatically increments by four after each transfer beat.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DADDR	Destination Address Destination address for the DMA transfer.

56.2.1.7 DMA Transfer Count Register (XCR)

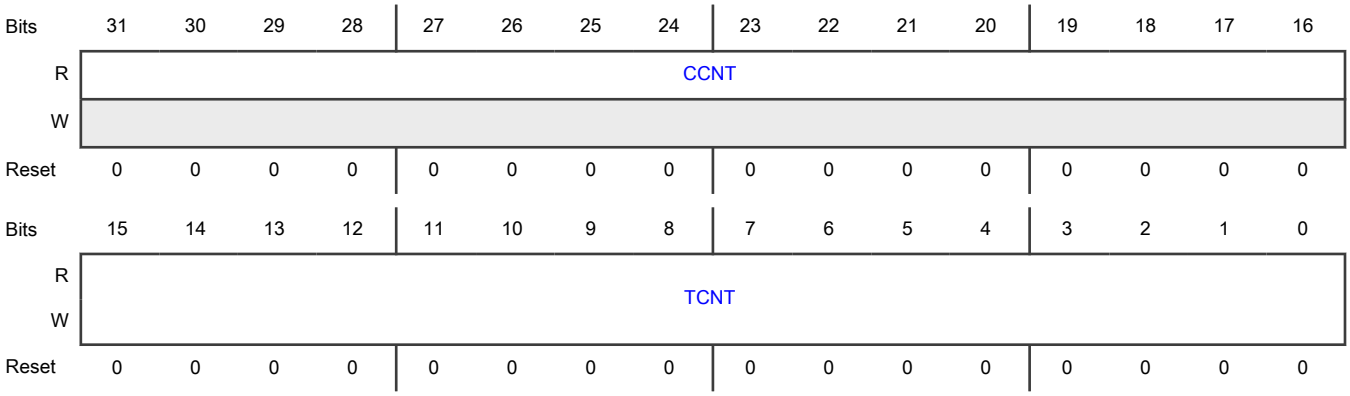
Offset

Register	Offset
XCR	14h

Function

The XCR register indicates the required and actual number of 32-bit words that the internal DMA controller transfers.

Diagram



Fields

Field	Function
31-16 CCNT	Current Transfer Count The current number of elements transferred to system memory.
15-0 TCNT	Total Transfer Count The required total number of elements to transfer to system memory. When the CCNT equals the TCNT: • DSB disables, • DMA operations stop, • any remaining data discards in the FIFO, • the CCNT resets to zero, • an optional DONE interrupt request generates.

56.3 Initialization/application information

This section discusses initialization of DSB and its programming considerations.

To use the internal DMA engine to transfer streaming radio data to system memory, initialize the transfer profile before enabling DSB. Follow the below procedure:

1. Initialize the DADDR with the destination buffer location in system memory.
2. Set the required total transfer count, TCNT, with the total packet length (in 32-bit words).
3. Set the watermark threshold value, WMRK, in the WMC register.
4. Set the DMA_EN bit.
5. Set the INT_EN bit and ERR_EN bit for interrupts on error or packet transfer completion.
6. Set the DSB_EN bit when ready to receive the data stream.

For example:

1. DADDR = 0x80004000 (system memory location)

2. TCNT = 0x1000 (16 KB of data)
3. WMRK = 0x8 (1/2 of the 16 element depth)
4. DMA_EN = 1, INT_EN = 1, ERR_EN = 1
5. DSB_EN = 1

NOTE

Real-time deadline analysis should be performed to ensure that DSB has sufficient system resources to meet all deadlines with respect to transferring the radio data stream to system memory.

Chapter 57
RF Core Mode Controller (RF_CMC)

57.1 Introduction

The Radio Core Mode Controller (RF_CMC) provides the following functions:

- Provides the enable to radio FRO module, which is sequenced off at low power entry, and on at low power exit
- Controls the low power sequencing for the radio flash via Q-channel interface.
- Conditions the low power entry requests from the NBU and WOR (for WOR and MAN) to wait until the NBU/CM3 is ready and the radio flash has entered low power
- On exit from Power Down or Deep Power Down low power modes, holds the radio LV in reset until the radio flash has finished its initialization.
- Creates XTAL ready interrupt for CM3
- Provides programmable control for TPM2 (clock control and input channel selection)
- Generates ifr_ld_en associated with the radio flash IFR outputs
- Implements SOC Core status registers
- Implements RAM related control registers (power controls)

The RF_CMC's logic uses the IPS/APB clock provided by the radio FRO module. It does not use the XTAL clock or the 32.768KHz clock.

57.2 Block Diagram

The following diagram gives a high-level overview of the RF_CMC.

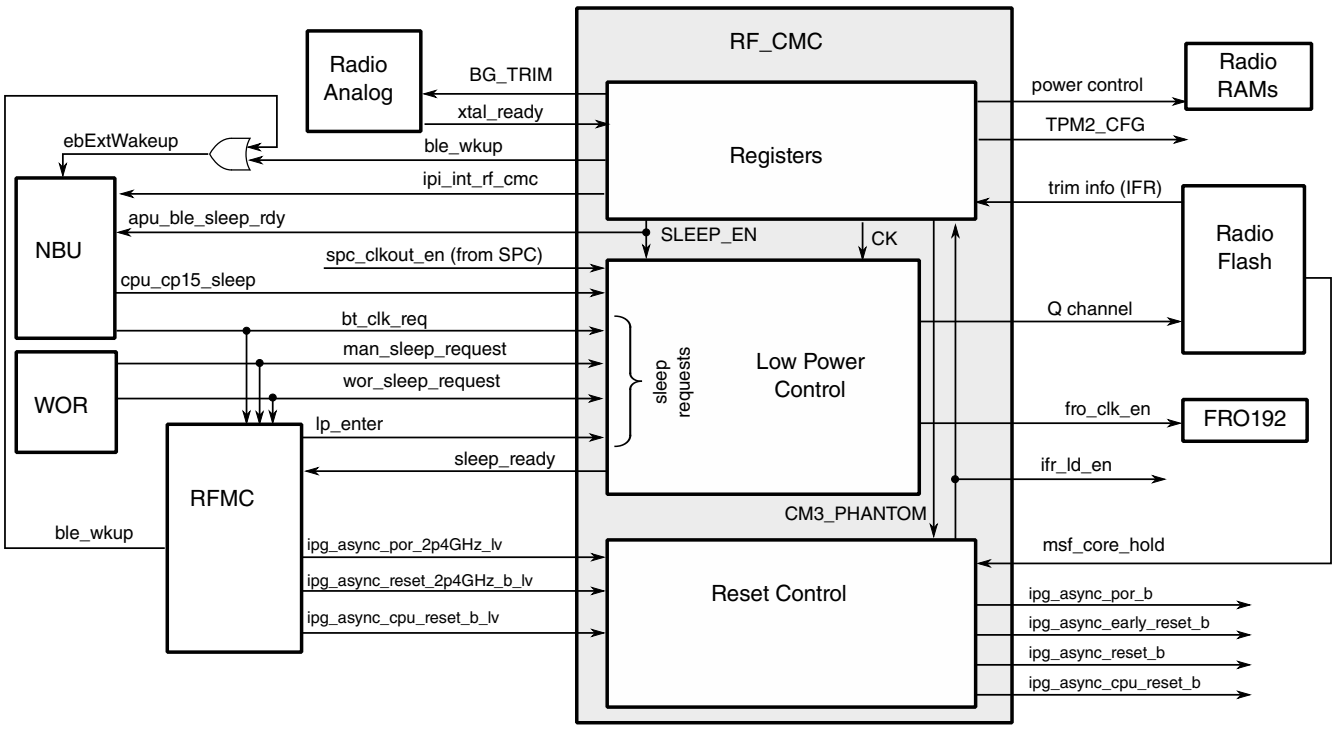


Figure 484. RF_CMC Block Diagram

57.3 Memory Map and Registers

57.3.1 RF_CMC register descriptions

57.3.1.1 RF_CMC memory map

RF_CMC1 base address: 4898_3000h

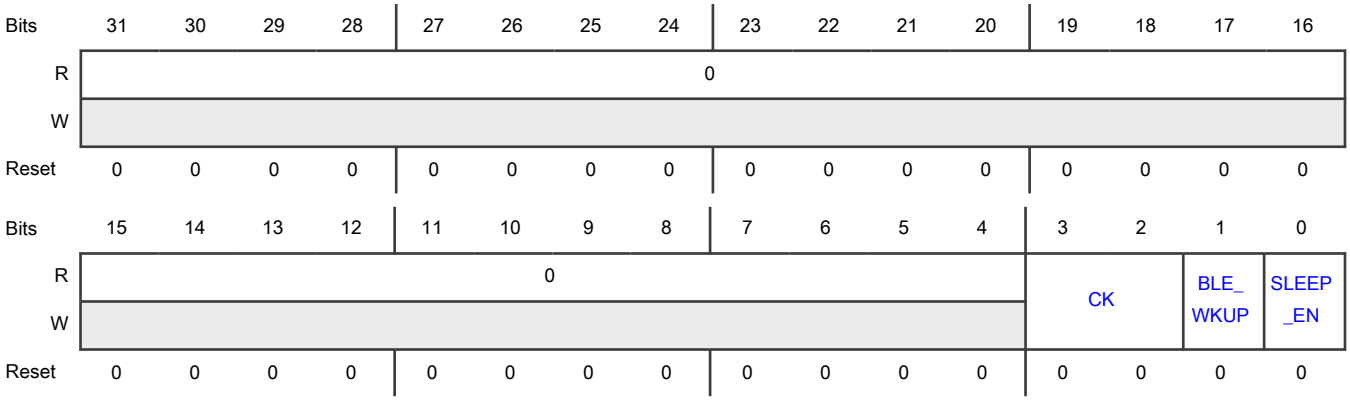
Offset	Register	Width (In bits)	Access	Reset value
0h	Radio Low Power Control Register (RADIO_LP)	32	RW	0000_0000h
4h	SOC Low Power Control and Status Register (SOC_LP)	32	RW	0000_0000h
8h	Interrupt Control Register (IRQ_CTRL)	32	RW	0000_0000h
Ch	TPM2 Configuration Register (TPM2_CFG)	32	RW	0000_0800h
10h	Radio Trim Register (RADIO_TRIM)	32	RW	0000_0073h
14h	RAM Power Control register (RAM_PWR)	32	RW	0000_0000h

57.3.1.2 Radio Low Power Control Register (RADIO_LP)

Offset

Register	Offset
RADIO_LP	0h

Diagram



Fields

Field	Function
31-4 —	Reserved
3-2 CK	Clock Control These bits provide additional configuration for the RF_CMC 00b - Normal configuration. When NBU CPU executes WFI and SLEEP_EN=1 (or if NBU CPU reset is asserted), and a sleep request from RFMC (LP_ENTER) NBU, MAN or WOR is asserted, the flash is put in low power, the sleep_rdy to RFMC asserts and the FRO will be disabled. 01b - Configuration where NBU, FRO and flash are not used. When NBU CPU reset is asserted, or NBU CPU executes WFI and SLEEP_EN=1, the flash will be placed in low power, the FRO disabled, the sleep_rdy to RFMC will assert and the NBU CM3 and AHB clocks will be gated off. The RF_CMC and NBU CPU will be without a clock until the next reset, but low power requests (RFMC LP_ENTER, MAN or WOR) will by accepted by RFMC since RF_CMC's sleep_rdy output will remain asserted. 10b - Configuration where NBU CPU is not used but FRO and flash can still be used. When NBU CPU reset is asserted, or NBU CPU executes WFI and SLEEP_EN=1, the clock to the NBU CPU will be gated. When RFMC (LP_ENTER), MAN or WOR request sleep, the flash is put in low power, the sleep_rdy to RFMC asserts and the FRO will be disabled as in configuration 00.
1 BLE_WKUP	Bluetooth Wakeup

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software control bit which can be used to force the NBU to assert its bt_clk_req output. Note that this bit is OR'ed with RFMC's BLE_WKUP bit; the output of that OR is connected to the NBU's ebExtWakeup input.
0 SLEEP_EN	Sleep Enable Software control bit which gates all sleep entry requests. Software should clear this bit if it is not ready to enter sleep, and set it when it is ready. This bit does not automatically clear on low power exit. This bit is also input to NBU as "apu_ble_slp_rdy".

57.3.1.3 SOC Low Power Control and Status Register (SOC_LP)

Offset

Register	Offset
SOC_LP	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											BUS_AWA...	0			BUS_REQ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved
4 BUS_AWAKE	Bus Awake This status bit indicates that the SOC bus is awake, and the CM3 software can perform a bus access.
3-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 BUS_REQ	Bus Access Request This bit can be set by software to request the SOC to wake the logic so that the CM3 can perform a bus access to the SOC XBAR. This will be connected to SOC WUU_MxDR input.

57.3.1.4 Interrupt Control Register (IRQ_CTRL)

Offset

Register	Offset
IRQ_CTRL	8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							XTAL_RDY	0			RDY_IE	0			RDY_FLAG
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8 XTAL_RDY	XTAL Ready Reflects the instantaneous value of the XTAL rady signal from the XO analog.
7-5 —	Reserved
4 RDY_IE	XTAL Ready Interrupt Enable If set, an interrupt is enabled when RDY_FLAG asserts.
3-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 RDY_FLAG	XTAL Ready Flag Flash which sets after the XTAL ready output from XO analog asserts (write 1 to clear).

57.3.1.5 TPM2 Configuration Register (TPM2_CFG)

Offset

Register	Offset
TPM2_CFG	Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CLK_MUX_SEL		0	CGC	CH1_MUX_SEL				0			CH0_MUX...
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11-10 CLK_MUX_SEL	Clock Mux Select Selects which clock is input to TPM2 00b - No clock 01b - Core Clock 10b - Radio Oscillator 11b - Reserved
9	Reserved

Table continues on the next page...

Table continued from the previous page...

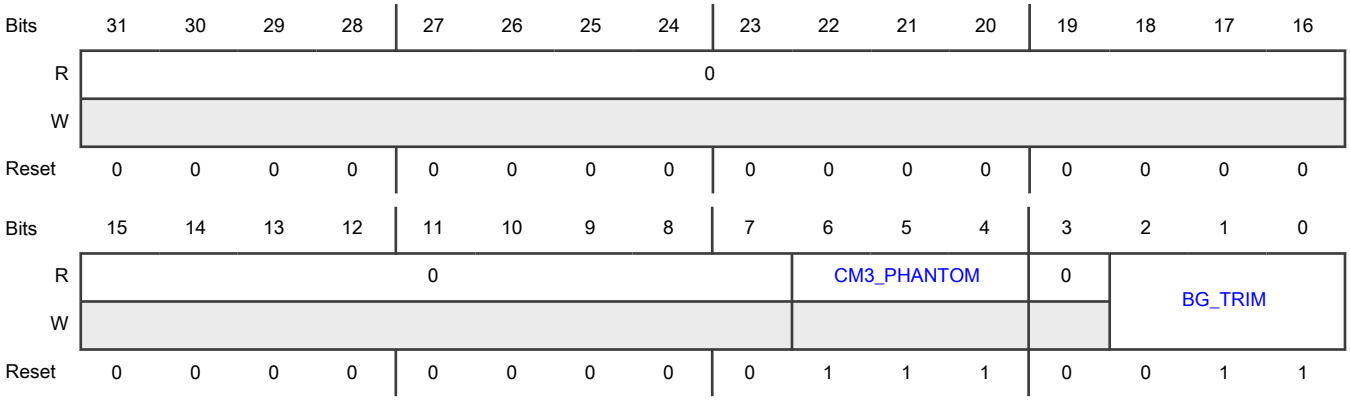
Field	Function
—	
8 CGC	Clock Gate Control 0b - TPM2 clock disabled 1b - TPM2 clock enabled
7-4 CH1_MUX_SEL	Channel1 Input Mux Select Selects which signal is input to TPM2 Channel 1 0000b - TPM2_CH1 pin 0001b - dtest[0] signal from radio 0010b - dtest[1] signal from radio 0011b - dtest[2] signal from radio 0100b - dtest[3] signal from radio 0101b - dtest[4] signal from radio 0110b - dtest[5] signal from radio 0111b - dtest[6] signal from radio 1000b - dtest[7] signal from radio 1001b - dtest[8] signal from radio 1010b - dtest[9] signal from radio 1011b - dtest[10] signal from radio 1100b - dtest[11] signal from radio 1101b - dtest[12] signal from radio 1110b - dtest[13] signal from radio 1111b - Reserved
3-1 —	Reserved
0 CH0_MUX_SEL	Channel0 Input Mux Select Selects which signal is input to TPM2 Channel 0 0b - TPM2_CH0 pin 1b - tof_timestamp_trig signal from radio

57.3.1.6 Radio Trim Register (RADIO_TRIM)

Offset

Register	Offset
RADIO_TRIM	10h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-4 CM3_PHANTOM	CM3 Phantom This bitfield is used by boot ROM. **NOTE** This bitfield resets (to 3'b111) on a power-on-reset, but the value is overwritten after that by IFR load events. 010b - CM3 disabled. The RF_CMC will hold the CM3 in reset 111b - CM3 enabled.
3 —	Reserved
2-0 BG_TRIM	Bandgap Trim Controls the bandgap trim in the analog. **NOTE** This bitfield only resets (to 3'b011) on a power-on-reset, but the value is overwritten after that by IFR load events. Normally, this bitfield should not need to be changed by software except for debug or characterization purposes.

Table continues on the next page...

Table continued from the previous page...

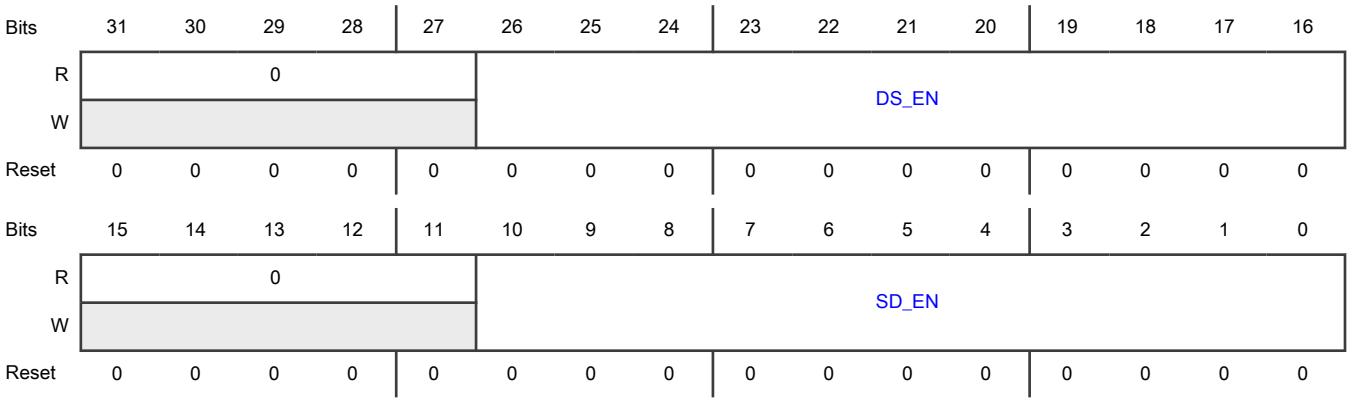
Field	Function
	000b - 787mV 001b - 794mV 010b - 800mV 011b - 806mV 100b - 812mV 101b - 819mV 110b - 825mV 111b - 831mV

57.3.1.7 RAM Power Control register (RAM_PWR)

Offset

Register	Offset
RAM_PWR	14h

Diagram



Fields

Field	Function
31-27 —	Reserved
26-16 DS_EN	Deep Sleep Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	These bits are used to control whether the RAM blocks are retained or power-gated in Power Down and Deep Sleep low power modes. Setting a bit causes the associated RAM block to be powered on in Power Down or Deep Sleep low power modes. Mapping is as shown below Bit [16] TXRAM Bit [17] RXRAM Bit [18] IRAM Bit [19] DRAM block 0, 4KB Bit [20] DRAM block 1, 4KB Bit [21] DRAM block 2, 8KB Bit [22] DRAM block 3, 32KB Bit [23] SMU2RAM block 0, 4KB Bit [24] SMU2RAM block 1, 8KB Bit [25] SMU2RAM block 2, 28KB Bit [26] EBRAM
15-11 —	Reserved
10-0 SD_EN	Shut Down Enable These bits can be used to shut down radio RAM blocks. Setting a bit causes the associated RAM block to be powered off. Mapping is as shown below Bit [0] TXRAM Bit [1] RXRAM Bit [2] IRAM Bit [3] DRAM block 0, 4KB Bit [4] DRAM block 1, 4KB Bit [5] DRAM block 2, 8KB Bit [6] DRAM block 3, 32KB Bit [7] SMU2RAM block 0, 4KB Bit [8] SMU2RAM block 1, 8KB Bit [9] SMU2RAM block 2, 28KB Bit [10] EBRAM

57.4 Functional description

57.4.1 Reset Control

The Reset control module performs the following functions

- On exit from Power Down or Deep Power Down low power modes, holds the radio LV in reset until the radio flash has finished its initialization.
- Supports CM3_PHANTOM feature which can hold CM3 in reset
- Generates the flash ifr_ld_en used to capture the radio flash IFR outputs

57.4.2 Low Power Control

The Low Power control module performs the following functions

- Provides the enable to radio FRO module, which is sequenced off at low power entry, and on at low power exit using the spc_clkout_en signal. The FRO can also be gated off in the CK=01 configuration
- Controls the low power sequencing for the radio flash via Q-channel interface.
- Creates a sleep_rdy signal to the RFMC. This signal must be asserted before the RFMC will accept a request to enter low power mode.

The RF_CMC supports the following low power configurations, selected by the programmable CK bits.

- CK=00. This is the normal/default configuration. When NBU CPU executes WFI and SLEEP_EN=1 (or if NBU CPU reset is asserted), and a sleep request from RFMC (LP_ENTER) NBU, MAN or WOR is asserted, the flash is put in low power, then the sleep_rdy to RFMC asserts and the FRO will be disabled.
- CK=01. This configuration supports use cases where NBU, FRO and flash are not used. When NBU CPU reset is asserted, or NBU CPU executes WFI and SLEEP_EN=1, the flash will be placed in low power, then the FRO disabled, the sleep_rdy to RFMC will assert and the NBU CM3 and AHB clocks will be gated off. Note the following behavior in the configuration after the FRO is disabled:
  - Read/write access to RF_CMC's registers will not be possible
  - RF_CMC's sleep_rdy output will remain asserted, so low power requests from RFMC (LP_ENTER), MAN or WOR will be immediately accepted by RFMC
  - If will not be possible to reprogram the RAM_RM registers, but the RAM control logic will still respond correctly to voltage select changes
- CK=10. Configuration where NBU CPU is not used but FRO and flash can still be used. When NBU CPU reset is asserted, or NBU CPU executes WFI and SLEEP_EN=1, the clock to the NBU CPU will be gated. When RFMC (LP_ENTER), MAN or WOR request sleep, the flash is put in low power, then the sleep_rdy to RFMC asserts and the FRO will be disabled as in configuration 00.</brief_description>

Chapter 58
Radio Flash

58.1 Introduction

The Radio Flash is the wrapper that contains 256 KB flash and FMC. It manages accesses performed by the bus masters of the system to the radio flash memory.

58.2 Memory Map and register definition

This section includes the Radio Flash module memory map and detailed descriptions of all registers.

58.2.1 RadioFlash register descriptions

58.2.1.1 RadioFlash memory map

RF_FMCCFG base address: 4898_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Radio Flash Memory Controller Configuration Register (RFMCCFG)	32	RW	0000_0000h

58.2.1.2 Radio Flash Memory Controller Configuration Register (RFMCCFG)

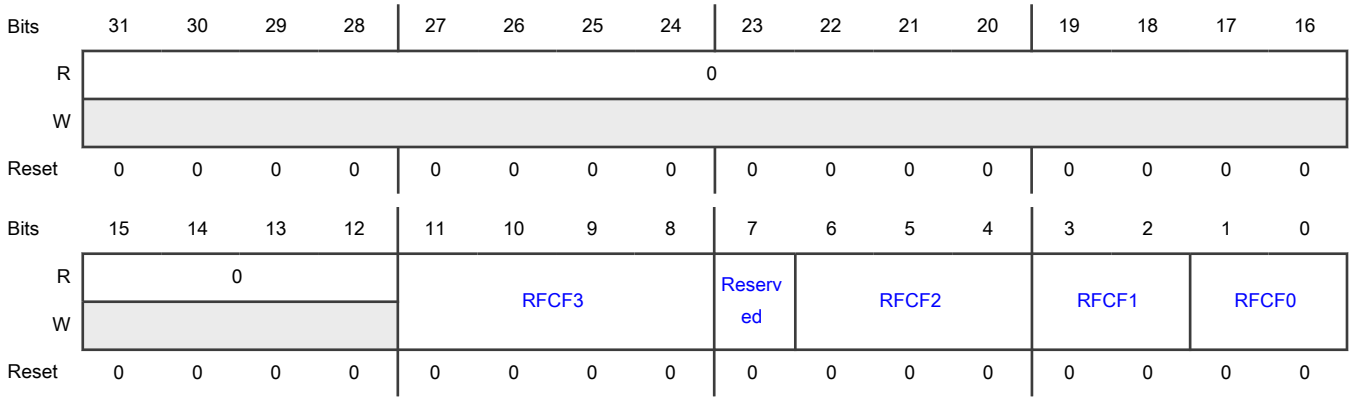
Offset

Register	Offset
RFMCCFG	0h

Function

The flash controller needs to be idle when writing to the RFMCCFG register associated with Flash memory. This means no read/erase/execute/etc operations from the Flash memory should be made while writing to the RFMCCFG register. Changing controller configuration while active can cause undesired results.

Diagram



Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-12
—</td><td>Reserved</td></tr>
<tr><td>11-8
RFCF3</td><td>Radio Flash Control Field 3
CF3 = Control Field 3 - for flash cache
• CF3[3] - disable flash cache
• CF3[2] - disable instruction caching
• CF3[1] - disable flash data caching
• CF3[0] - clear flash cache</td></tr>
<tr><td>7
—</td><td>Reserved</td></tr>
<tr><td>6-4
RFCF2</td><td>Radio Flash Control Field 2
CF2[2] FSE - Flash Stall Enable
• 0b = Flash Stall is disabled on flash busy
• 1b = Flash Stall is enabled on flash busy
CF2[1] DFS - Disable Flash Speculate
CF2[0] DDP - Disable Data Prefetch
CF2[1:0] bits controls whether prefetches (or speculative accesses) are initiated in response to instruction fetches or data references, see section FMC speculative reads. Value 0 means enable and value 1 means disable.
<table>
<tr><th>CF2[1]
Flash Speculate</th><th>CF2[0]
Data Prefetch</th><th>Result</th></tr>
<tr><td>Disabled</td><td>Disabled</td><td rowspan="2">All speculation disabled and speculation buffer is cleared</td></tr>
<tr><td>Disabled</td><td>Enabled</td></tr>
<tr><td>Enabled</td><td>Disabled</td><td>Speculation for Instruction enabled and Speculation for Data disabled</td></tr>
<tr><td>Enabled</td><td>Enabled</td><td>Speculation for both Instruction and Data enabled</td></tr>
</table></td></tr>
<tr><td>3-2
RFCF1</td><td>Radio Flash Control Field 1
CF1 = Control Field 1 - for ECC control functions
CF1[1] DNCBED - Disable non-correctable bus errors on flash data fetches
• 0b = Enable bus error response for non-correctable error on data fetch from flash
• 1b = Disable bus error response for non-correctable error on data fetch from flash</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CF1[0] DNCBEI - Disable non-correctable bus errors on flash instruction fetches • 0b = Enable bus error response for non-correctable error on instruction fetch from flash • 1b = Disable bus error response for non-correctable error on instruction fetch from flash
1-0 RFCF0	Radio Flash Control Field 0 CF0 = Control Field 0 CF0[1] - disable the use of flash, page buffer and speculation buffer CF0[0] - clear of flash cache and page buffer

Chapter 59
FRO Clock Generator (FRO192M)

59.1 Overview

The FRO192 module provides the system clocks for the Radio Mode Controller.

59.1.1 Block Diagram

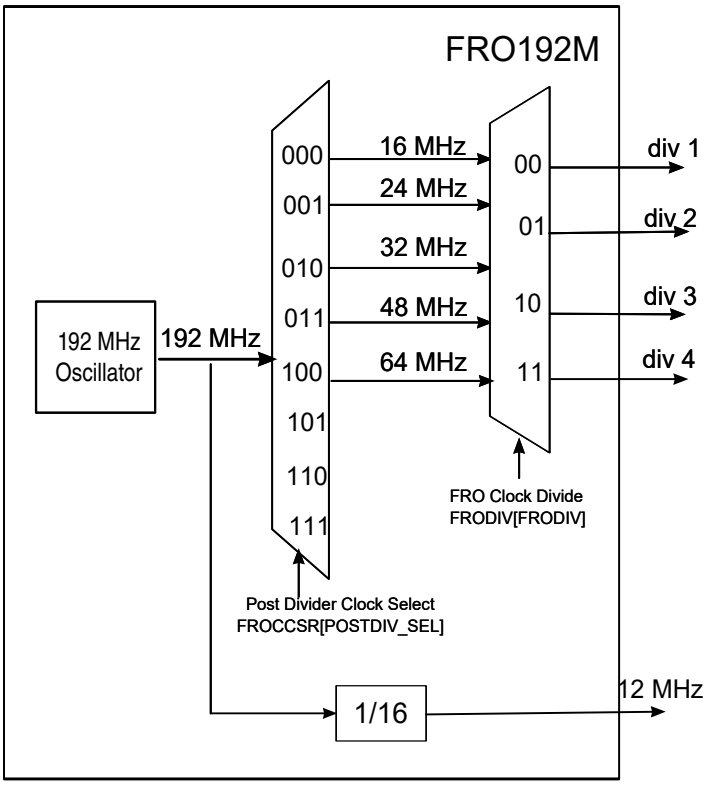


Figure 485. Clock Generator block diagram

59.1.2 Features

Key features of the FRO192 module are:

- Programmable post-divider for up to 5 different frequency ranges

59.2 Functional description

59.2.1 Modes of operation

The following table describes the modes available for FRO192.

Table 492. Operating modes

Mode	Entry	Exit	Description
Reset	POR Warm Reset Disabled	Reset Deassertion	FRO192 is reset.
Run	Enabled	POR Warm Reset Disabled	FRO192 is allowed to run and generate an output clock.
Disabled	Disabled	Run	FRO192 is held in a reset state and waiting to be enabled to switch over to Run mode.

59.2.2 Interrupts

FRO192 does not support any interrupts.

59.2.3 Clocks

FRO192 registers and logic are clocked by the slow bus clock.

The following table describes the clocks of FRO192.

Table 493. Clocks and their description

Clock	Type	Description
Slow Bus Clock	Input	FRO192 register read/write clock source.
FRO Bus Clock	Output	FRO192 Div Clock/FRODIV output divided system clock.
Fixed_clk	Output	FRO192 fixed 12 MHz output clock source.

59.3 External signals

This module has no external signals.

59.4 Initialization

By default, FRO192 is disabled and enabled by external block. It requires no initialization.

59.5 Memory Map/Register Definition

This section includes the memory map and register definition.

59.5.1 FRO192 clock generator register descriptions

- For any writeable registers, only 32-bit writes are allowed. 8-bit or 16-bit writes will result in transfer errors.

59.5.1.1 FRO192 memory map

FRO192M0 base address: 4898_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	FRO192 Clock Control Status Register (FROCCSR)	32	RW	0000_2001h
4h	FRO192 Divide Register (FRODIV)	32	RW	0000_0001h

59.5.1.2 FRO192 Clock Control Status Register (FROCCSR)

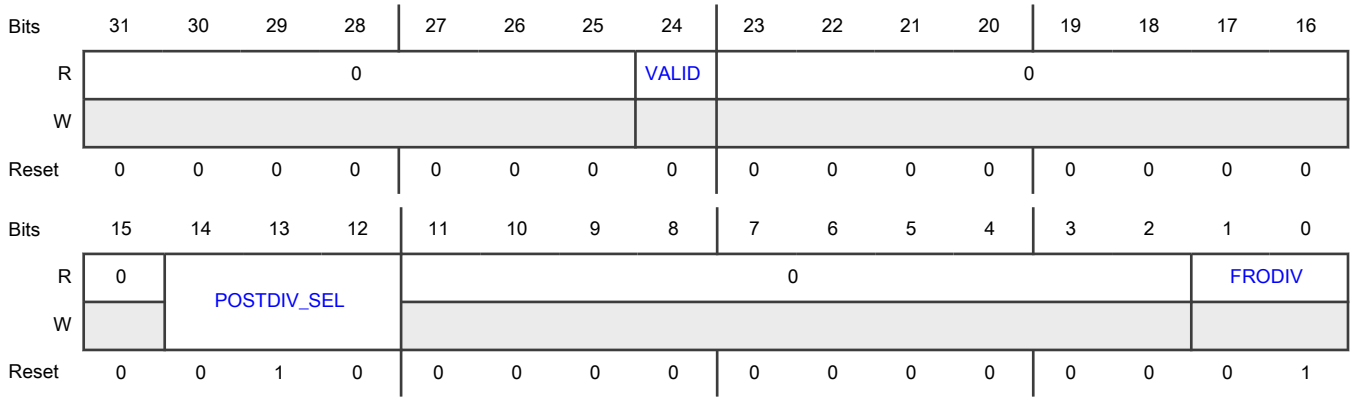
Offset

Register	Offset
FROCCSR	0h

Function

FRO192 control register is used to select the frequency range and post divider. It also shows the status of the FRO Clock.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 VALID	Clock Valid Flag This flag indicates if the status of FRO192 clock is valid or invalid 0b - FRO192 is not enabled or clock is not valid. 1b - FRO192 is enabled and output clock is valid.
23-15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-12 POSTDIV_SEL	Post Divider Clock Select This field selects the correct post divider to generate selected output frequenc clock. Output frequency options supported are 64 MHz, 48 MHz, 32 MHz, 24 MHz, and 16 MHz. Divide values can be changed while FRO192 is enabled 000b - FRO 16MHz Range selected. 001b - FRO 24MHz Range selected 010b - FRO 32MHz Range selected 011b - FRO 48MHz Range selected 100b - FRO 64MHz Range selected 101b - RESERVED. Not Supported 110b - RESERVED. Not Supported 111b - RESERVED. Not Supported
11-2 —	Reserved
1-0 FRODIV	FRO Clock Divide This field indicates the status of the FRODIV Divider. 00b - Divide by 1 01b - Divide by 2 10b - Divide by 3 11b - Divide by 4

59.5.1.3 FRO192 Divide Register (FRODIV)

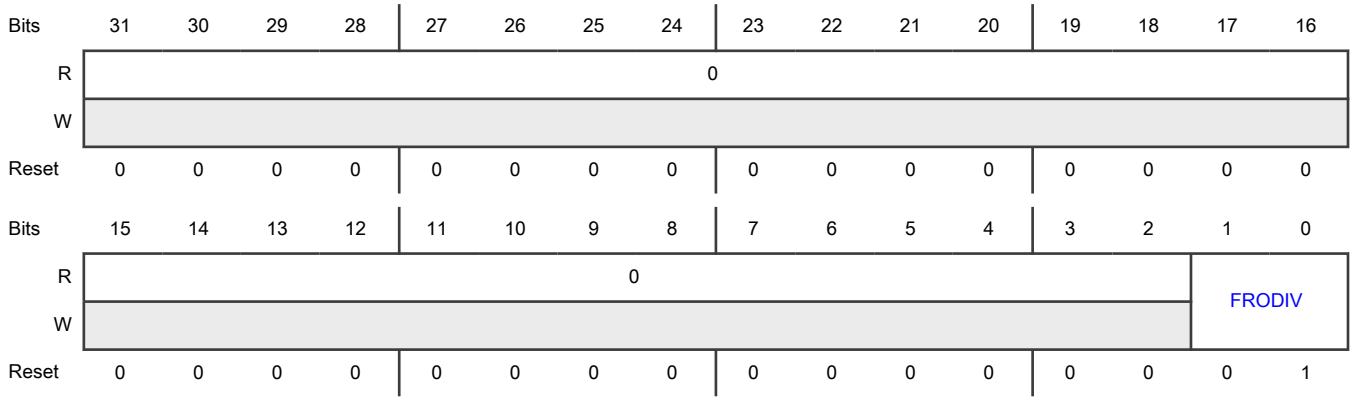
Offset

Register	Offset
FRODIV	4h

Function

This register divides the output of the POSTDIV_SEL to generate the a system clock source.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 FRODIV	FRO Clock Divide This read/write field select the clock divider ratio for POSTDIV_SEL output clock. Used to generate the system clock source. 00b - Divide by 1 01b - Divide by 2 10b - Divide by 3 11b - Divide by 4

Chapter 60
Release notes

60.1 Revision history

Table 494. Revision history

Rev. No.	Date	Substantial Changes
3.0	09/2024	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

Bluetooth — the Bluetooth wordmark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by NXP Semiconductors is under license.

Matter, Zigbee — are developed by the Connectivity Standards Alliance. The Alliance's Brands and all goodwill associated therewith, are the exclusive property of the Alliance.

For more information, please visit: https://www.nxp.com

Date of release: September 2024
Document identifier: MCXW71RM