NXP Semiconductors
Reference Manual

Document identifier: MCXAP100M96FS6RM
Rev. 7, 2024-12-09

MCX A156, A155, A154, A146, A145, A144 Reference Manual

Supports MCXA156x, A155x, A154x, A146x, A145x, and A144x.

Contents

Chapter 1
About This Manual

1.1 Audience

This reference manual (RM) is intended for system software, hardware developers, and applications programmers who need to develop products using this chip. It assumes that its users understand operating systems, microprocessor system design, and basic principles of software and hardware.

1.2 Organization

This manual has two main sets of chapters.

- Chapters in the first set contain information that applies to all components on the chip.
- Chapters in the second set are organized into functional groupings that detail particular areas of functionality.
  - Examples of these groupings are clocking, timers, and communication interfaces.
  - Each grouping includes chapters that provide a technical description of individual modules.

1.2.1 Attachments

This manual includes key information in the files attached to it. For example, memory map and I/O details. Use the content in these attachments in conjunction with this manual's content.

NOTE

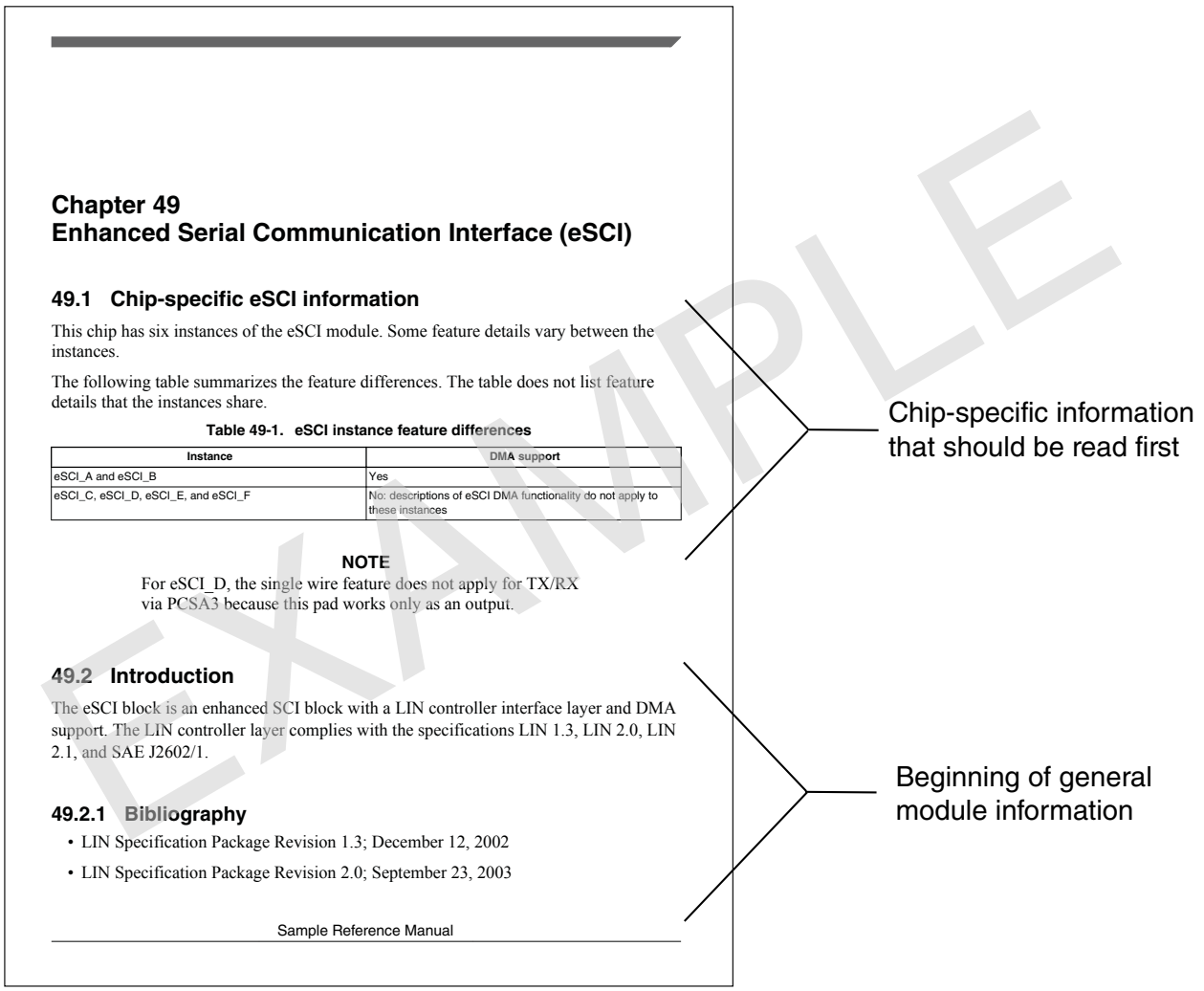
Select the paperclip icon on the left side of the PDF window to see the list of attachments.

1.3 Module descriptions

Each module chapter has two main parts:

- The first section, *chip-specific [module name]* information, provides details such as the number of module instances on the chip and connections between that module and the other ones. Read this section *first* because its content is crucial for understanding the information in the other sections of the chapter.
- The subsequent sections provide general information about the module, including its signals, registers, and functional description.

The following figure shows you an example of this demarcation.



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 controller interface layer and DMA support. The LIN controller 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

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

1.3.1 Chip-specific information that clarifies content in the same chapter

The following figure shows an example of chip-specific information that clarifies 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.

System Integration Unit Lite2 (SIUL2)

Chapter 9
System Integration Unit Lite2 (SIUL2)

9.1 Chip-specific SIUL2 information

9.1.1 Feature configurations

In this device, the SIUL2_0 module instance does not support the following features described in the generic description:

- Interrupts
- DMA channels

9.1.2 Notes for IMCR

Out of reset, PA_00, PA_04, and PA_05 pads have JTAG input functionality selected by default. It should be disabled in the corresponding IMCR registers (IMCR61, IMCR60, and IMCR50 respectively) in order to use other functionality such as GPIO.

9.2 Introduction

9.2.1 Overview

The System Integration Unit Lite2 provides control over all the electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. One of the most important functions of the SIUL2 is to enable the user to select the functions and electrical characteristics that appear on external device pins. It also controls the multiplexing of internal signals from one module to another and controls chip I/O. It supports as many as 32 external interrupts with trigger event configuration. The following figure is the block diagram of SIUL2 and its interfaces to other system components.

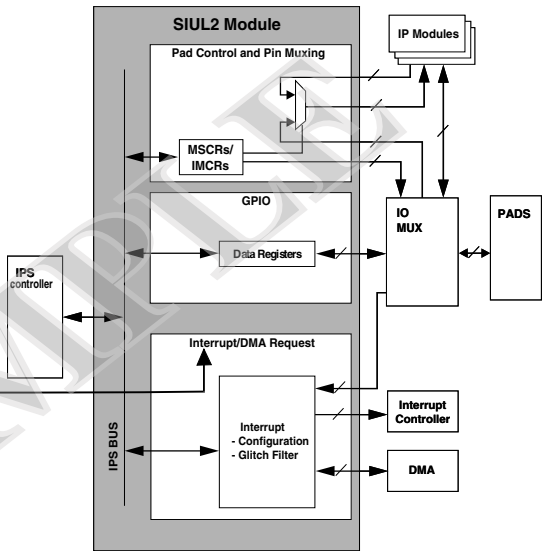


Figure 23. System Integration Unit Lite2 block diagram

This module provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. The SIUL2 module provides registers that enable user software to read values from GPIO pads configured as inputs, and write values to GPIO pads configured as outputs:

- When configured as output, you can write to an internal register to control the state driven on the associated output pad.
- When configured as input, you can detect the state of the associated pad by reading the value from an internal register.
- When configured as input and output, the pad value can be read back, which can be used as a method of checking if the written value appeared on the pad.

To assist software development, GPIO data registers can be accessed using various mechanisms. These differing mechanisms allow support for port access or for bit manipulation without the need to use read-modify-write operations:

- Access to two 16-bit ports in one access
- Read/write access to a single bit
- A 16-bit port write with a bit mask, using single 32-bit access.

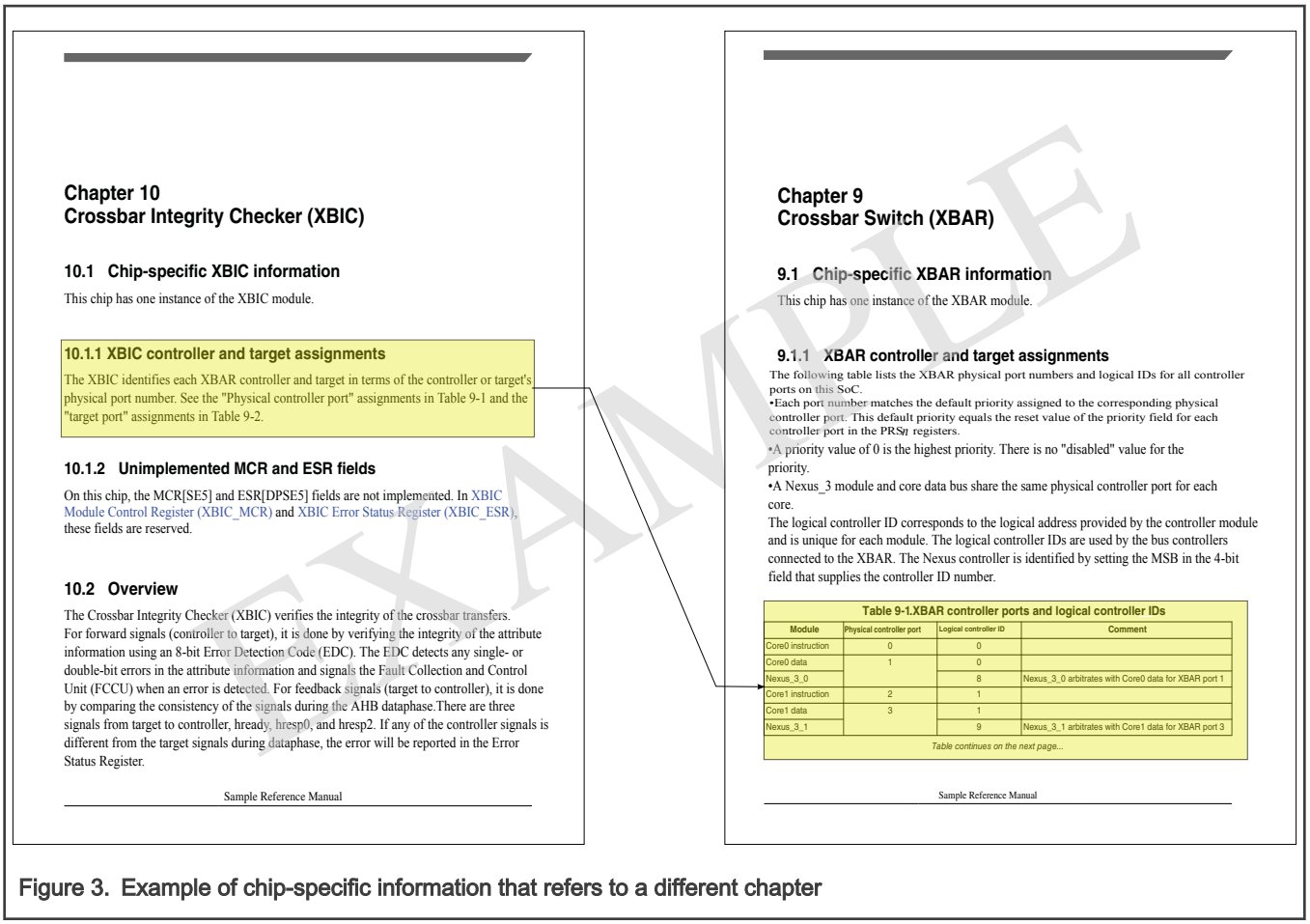
Sample Reference Manual
NXP Semiconductors

Sample Reference Manual
NXP Semiconductors

Figure 2. Example of chip-specific information that clarifies content in the same chapter

1.3.2 Chip-specific information that refers to a different chapter

Related chip-specific information may be provided in different chapters of the manual. The following figure shows an example of two such connected pieces of information. In this case, read both before you proceed.



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC controller and target assignments

The XBIC identifies each XBAR controller and target in terms of the controller or target's physical port number. See the "Physical controller port" assignments in Table 9-1 and the "target port" assignments in Table 9-2.

10.1.2 Unimplemented MCR and ESR fields

On this chip, the MCR[SE5] and ESR[DPSE5] fields are not implemented. In XBIC Module Control Register (XBIC_MCR) and XBIC Error Status Register (XBIC_ESR), these fields are reserved.

10.2 Overview

The Crossbar Integrity Checker (XBIC) verifies the integrity of the crossbar transfers. For forward signals (controller to target), it is done by verifying the integrity of the attribute information using an 8-bit Error Detection Code (EDC). The EDC detects any single- or double-bit errors in the attribute information and signals the Fault Collection and Control Unit (FCCU) when an error is detected. For feedback signals (target to controller), it is done by comparing the consistency of the signals during the AHB dataphase.There are three signals from target to controller, hready, hresp0, and hresp2. If any of the controller signals is different from the target signals during dataphase, the error will be reported in the Error Status Register.

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9.1 Chip-specific XBAR information

This chip has one instance of the XBAR module.

9.1.1 XBAR controller and target assignments

The following table lists the XBAR physical port numbers and logical IDs for all controller ports on this SoC.

- Each port number matches the default priority assigned to the corresponding physical controller port. This default priority equals the reset value of the priority field for each controller port in the PRSn registers.
- A priority value of 0 is the highest priority. There is no "disabled" value for the priority.
- A Nexus_3 module and core data bus share the same physical controller port for each core.

The logical controller ID corresponds to the logical address provided by the controller module and is unique for each module. The logical controller IDs are used by the bus controllers connected to the XBAR. The Nexus controller is identified by setting the MSB in the 4-bit field that supplies the controller ID number.

Table 9-1.XBAR controller ports and logical controller IDs

Module	Physical controller port	Logical controller ID	Comment
Core0 instruction	0	0	
Core0 data	1	0	
Nexus_3_0		8	Nexus_3_0 arbitrates with Core0 data for XBAR port 1
Core1 instruction	2	1	
Core1 data	3	1	
Nexus_3_1		9	Nexus_3_1 arbitrates with Core1 data for XBAR port 3

Table continues on the next page...

Sample Reference Manual

Figure 3. Example of chip-specific information that refers to a different chapter

1.4 Register descriptions

Module chapters present register information in the following:

- Memory maps, which contain:
  - An offset from the module's base address
  - The mnemonic and name of each register
  - The width of each register (in bits)
  - The reset value of each register
- Register figures
- Field-description tables
- Associated text

The following figure shows register figure conventions used throughout the manual.

Access type	Access description	DITA output	Effect of Write on Value	Readback Value
RW	Read/write	R / W: Mnemonic	Changes to Written Value	Current Value
RO	Read-only	R: Mnemonic / W:	No Effect	Current Value
RU	Reserved, unimplemented	R / W: Reserved	No Effect	Value Undefined
ROZ	Reserved, Read-only zero	R: 0 / W:	No Effect	Returns All Zero
ROO	Reserved, Read-only one	R: 1 / W:	No Effect	Returns All Ones
WO	Write-only	R: / W: Mnemonic	Changed to Written Value	Value Undefined
WOZ	Reserved, Write-only zero	R: / W: 0	Write Value Must Be All 0s.	Value Undefined
WOO	Reserved, Write-only one	R: / W: 1	Write Value Must Be All 1s.	Value Undefined
W0C	Write zero to clear	R: Mnemonic / W: w0c	If a Bit in the Written Value is a 0, the Corresponding Bit in the Field is Set to 0. Otherwise, the Field Bit is Not Affected.	Current Value
W1C	Write one to clear	R: Mnemonic / W: w1c	If a Bit in the Written Value is a 1, the Corresponding Bit in the Field is set to 0. Otherwise, the Field Bit is Not Affected.	Current Value
R2C	Read to clear		Value is cleared following the read operation	Current Value
ROWZ	Read-only, writes zero	R: Mnemonic / W: 0	Write Value Must Be All 0s	Current Value
ROWO	Read-only writes one	R: Mnemonic / W: 1	Write Value Must Be All 1s	Current Value
ROWU	Ready-only writes undefined	R: Mnemonic / W: —	Writes Operation Undefine	Current Value
WORZ	Write-only reads zero	R: 0 / W: Mnemonic	Changes to Written Value	Returns All Zero
WORO	Write-only reads one	R: 1 / W: Mnemonic	Changes to Written Value	Returns All Ones
WORU	Write-only reads undefined	R: — / W: Mnemonic	Changes to Written Value	Value Undefined

Figure 4. Register figure conventions

NOTE

Reset values of reserved locations documented in this manual are subject to change and must not be used for diagnostic purposes.

1.5 Conventions

1.5.1 Notes and cautions

Specific information is provided as part of notes and cautions throughout this manual.

NOTE

Emphasizes information that deserves extra attention.

CAUTION

Informs you of situations that could lead to highly undesirable outcomes—such as damage to the chip or irreversible malfunction.

1.5.2 Numbering systems

The following suffixes identify different numbering systems:

Table 1. Numbering systems

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

1.5.3 Typographic notation

The following typographic notations are used throughout this document:

Table 2. Typographic notation

Example	Description
x and other italicized text	The italicized, lowercase *x* is used as a placeholder for replaceable numbers. In general, italicized text is used for titles of publications and for emphasis. Additionally, italics could be used for metasymbols in syntax descriptions. Plain lowercase letters are used as placeholders for single letters and numbers.
`code font`	Fixed-width font (such as Courier) used for code. It is used for a letter, word, or phrase that you want the user to type. For example, "Type `Read` and press Enter." This type of font is also used for instruction mnemonics, directives, symbols, subcommands, parameters, operators, computer-language elements, code listings, commands that appear in running text, and for sample code. Instruction mnemonics and directives in text and tables are mentioned in all caps; for example, BSR.
SR[SCM]	A mnemonic in square brackets represents the name of a register field. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets that are separated by a colon represent either: • A subset of a register's named field

Table continues on the next page...

Table 2. Typographic notation (continued)

Example	Description
	For example, REVNO[6:4] refers to bits 6-4 that are part of the COREREV field occupying 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.
MOD.REG	A period separates the elements of a hierarchy: subsystem.module.register. For example: • SWT.TO means that the TO register is located in the SWT module. • SMU.XRDC.CR means that the CR register is located in the XRDC module within the SMU subsystem.

1.5.4 Special terms

The following terms have special meanings.

Table 3. Special terms

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 memory space, register, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. You must: • Before writing to a location which contain reserved bits user must make sure the write operation will write the reserved bit with value specified as the reset value in NXP reference manual. • Consider undefined locations in memory to be reserved as reset value 0 shall be assumed. You might get a BERR(transfer error) response on access to undefined locations in memory. • If user reads data from memory area containing reserved bit, the value of reserved bits should be ignored and not used for any functional purposes. **NOTE** BootROM could modify the reserved bit values after reset. Please refer to the BootROM settings attachment.
Write 1 to clear (w1c)	Refers to the access type of a register field that is used to clear the field by writing the value 1 to it.
Undefined (u)	Refers to undefined reset values

1.6 Editorial changes

Each new release of this document includes editorial improvements such as:

- Spelling
- Grammar
- Punctuation
- Voice
- Tense
- Capitalization
- Formatting
- Presentation
- Navigation

Chapter 2
Introduction

2.1 Overview

This group of products expands the MCX Arm® Cortex®-M33 product offerings with multiple high-speed connectivity, operating up to 96 MHz, serial peripherals, timers, analog and low power consumption.

2.2 Target applications

This group of product have following target applications:

- Energy Storage and Management System
- Smart Metering
- Factory Automation
- Industrial HMI
- Mobile Robotics Ecosystem
- Motion Control and Robotics
- Motor Drives
- Brushless DC Motor (BLDC) Control
- Permanent Magnet Synchronous Motor (PMSM)

Smart Home

- Home Control Panel
- Major Home Appliances
- Robotic Appliance
- Smart Speaker
- Soundbar
- Gaming Accessories
- Smart Lighting
- Smart Power Socket and Light Switch

2.3 Block diagram

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

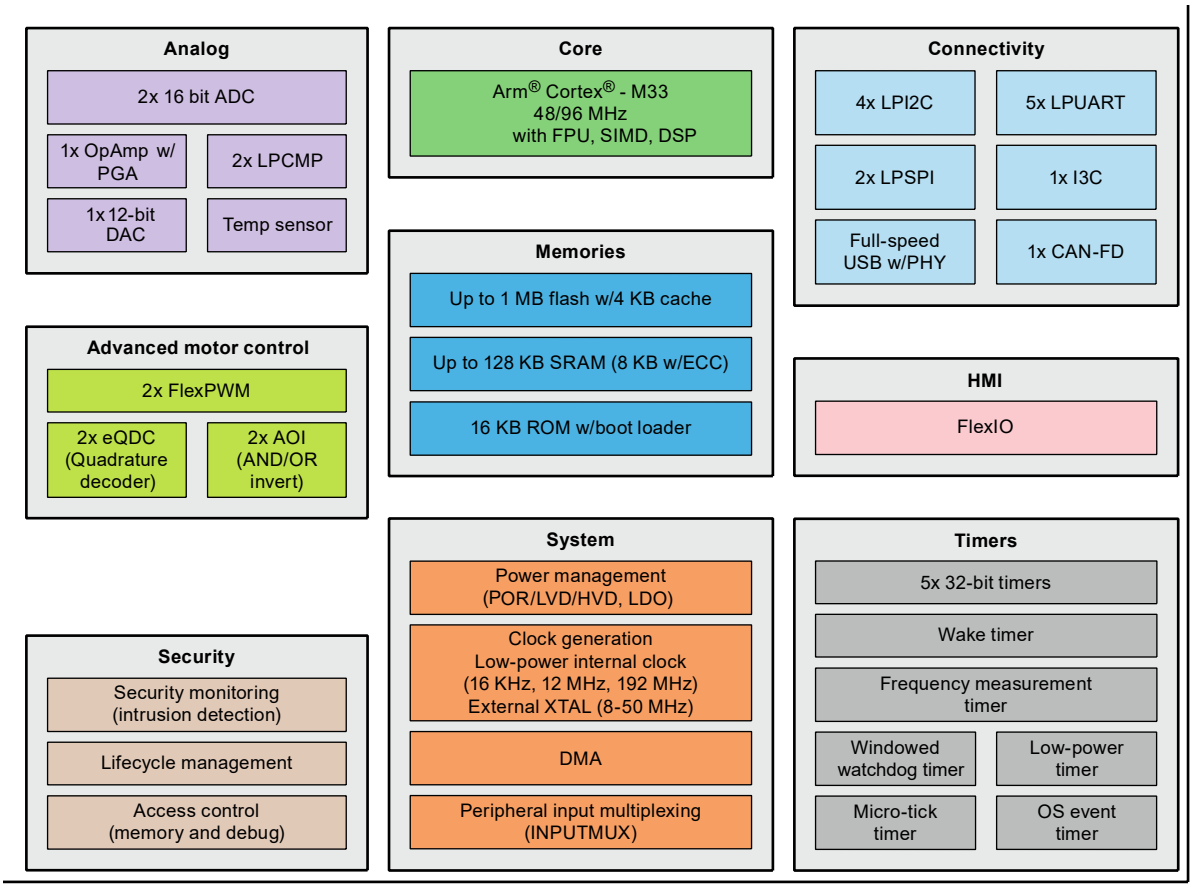


Figure 5. Features block diagram

2.4 Features

This group of product offers the following features:

- Core
  - Single Cortex-M33 core, operating at 48/96MHz (w/ FPU, DSP, no Trustzone, no MPU). 48 MHz at Mid Drive mode (1 V), 96 MHz at Standard Drive mode (1.1 V)
  - Nested Vectored Interrupt Controller
  - Clock gating for core processor
  - Multilayer AHB Bus Matrix
- Memories
  - 1 MB Flash array implemented as single array. The smallest programming phrase should be 16 bytes.
    - 128-bit + 9-bit ECC, 8 KB sector size
    - 32 KB User IFR + 8 KB Test IFR
  - Flash Memory Controller
    - Line buffer
    - Prefetch buffer
    - Swap

    - 3 wait states at 96 MHz (SD mode), 1 wait state at 48 MHz (MD mode)
  - Memory Block Checker (MBC)
    - Protect flash read, write and execute permission.
    - Main array granularity is 16 KB
    - IFR0 granularity is 8 KB.
    - IFR1 granularity is 4 KB
  - RAM
    - 4 KB LPCAC to support on-chip Flash caching. 4 KB LPCAC RAM can be used as Code SRAM when LPCAC is disabled.
    - 8 KB Code SRAM
    - Up to 120 KB System SRAM (8 KB with ECC)
  - ROM
    - 16 KB System ROM
- Timers
  - Five 32-bit standard general purpose asynchronous timers/counters, which support up to four capture inputs and four compare outputs, PWM mode, and external count input. Specific timer events can be selected to generate DMA requests.
  - Low power timer
  - Frequency measurement timer
  - Windowed watchdog timer
  - Wake timer
  - Micro-tick timer (UTICK) running from the watchdog oscillator can be used to wake-up the device from sleep and deep-sleep modes. Includes 4 capture registers with pin inputs.
  - OS event timer
  - 42-bit free running OS Timer as continuous time-base for the system
- Advanced Motor Control
  - 2x FlexPWM
    - 3 sub-modules per each FlexPWM, providing 6 complementary outputs of PWM
    - 3 fault inputs are supported
  - 2x Quadrature Decoder (eQDC)
  - 2x AOI (AND/OR/Invert) module
    - Implements four events output. Each event output represents a user-programmed combinational Boolean function based on four event inputs.
    - Four event inputs are selected from 16 input
- Communication Interfaces for Connectivity
  - 4x LPI2C, 2x LPSPI, 5x LPUART
  - 1x I3C
  - USB Full-speed (Device) with on-chip FS PHY
  - FlexCAN with FD

  - FlexIO
- Analog
  - 2x 16-bit ADC
    - Up to 3.2 Msps in 16-bit mode, and 4 Msps in 12-bit
    - Up to 32 ADC Input channels (depending on the package)
    - One integrated temperature sensor per ADC
  - Two High-speed Comparators with 8 input pins and 8-bit DAC as internal reference
    - 2x LPCMP is functional down to deep power-down mode
  - 1x OpAmp with PGA
  - 1x 12-bit DAC
- Operating characteristics
  - Operating voltage: 1.71 to 3.6 V
  - Temperature range: -40 to 125 °C
- Debug
  - Minimal debug – 4 breakpoint and 2 watch point, no ETM
  - Debug mail box for allow tools consistency during power up
  - SWD/JTAG
- Packages
  - LQFP100
  - LFBGA64
  - VFBGA112
  - HVQFN48
  - LQFP64

See the device data sheet for details on packages.

2.5 Functional overview

The following table shows the chip modules organized by functional category.

Table 4. 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, and interrupt driven environments. The Cortex-M33 processor is based on the Armv8-M Architecture and ThumbR-2 ISA and is upward compatible with the Cortex-M7, M4, M3, M1, and M0/M0+.
System	• Debug Mailbox • AHB Cross-Bar Switch (AXBS) Lite

Table continues on the next page...

Table 4. Module functional categories (continued)

Module category	Description
	• Enhanced Direct Memory Access (8-channel eDMA) • Wake-Up Unit (WUU) • Peripheral Input Mux (INPUTMUX) • Error Injection Module (EIM) • Error Reporting Module (ERM) • System Power Controller (SPC) • System Controller • RMC (part of CMC)
Memory	• Flash Memory Controller (FMC) • Flash Management Unit (FMU) • ROM-BOOT and ROM-CODE • Static Random Access Memory (SRAM) • AHB Low Power Cache Controller (LPCAC)
Clock	• VBAT Wrapper — Free Running Oscillator - 16 K (FRO_16K) • System Clock Generator (SCG) — Crystal Oscillator - System (OSC_SYS) — Free Running Oscillator - 192 M (FRO_192M) — Free Running Oscillator - 12 M (FRO_12M)
Security	• Cyclic Redundancy Check (CRC) • Code Watchdog • Glikey • Memory Block Checker (MBC)
Timer	• eFlexPWM • Quadrature Decoder (ENC) • Micro-Tick (UTICK) Timer • OS Event Timer (OSTIMER) • Low-power Timer (LPTMR) • Standard Counter/Timer (CTIMER) • Windowed Watchdog Timer (WWDT) • Frequency Measurement (FREQME) • Wake Timer

Table continues on the next page...

Table 4. Module functional categories (continued)

Module category	Description
Communication	• Improved Inter-Integrated Circuit (I3C) • Universal Serial Bus - Full Speed (USBFS) and Transceiver • USB FS Physical Layer Interface (USBFS PHY) • Low-Power UART (LPUART) • Low-Power Serial Peripheral Interface (LPSPI) • Low-Power I2C (LPI2C) • Flexible Data Rate CAN (FlexCAN) • Flexible I/O (FLEXIO)
Human Machine Interface (HMI)	• General Purpose Input/Output (GPIO) • Port Control (PORT)
Analog	• 16-bit Analog-to-Digital Converter (ADC) with 4 Msps conversion rate at 12-bit mode. • 12-bit DAC • Operational Amplifier (OPAMP) • Low Power analog Comparator (LPCMP)

2.5.1 Core

The following core modules are available on this chip.

Table 5. Core modules

Module	Description
CPU	Is an Arm Cortex-M33 processor that runs at a frequency of up to 96 MHz.
Nested Vectored Interrupt Controller (NVIC)	The Armv8M exception model and Nested-Vectored Interrupt Controller (NVIC) implement a relocatable vector table supporting external interrupts, a single non-maskable interrupt (NMI), and priority levels.
System Tick Timer (SysTick)	See the Armv8M Architecture Reference Manual for more information about this system timer.

2.5.2 Clock

The following clock modules are available on this chip.

Table 6. Clock sources

Module	Description
Free Running Oscillator - 16 K (FRO_16K)	Is an ultra-low power internal 16.384 kHz clock source. It is functional down to VBAT mode. The FRO_16K is trimmed to +/- 6% accuracy over the entire voltage and temperature range.

Table continues on the next page...

Table 6. Clock sources (continued)

Module	Description
System Clock Generator (SCG)	Provides the user with access to the configuration control registers for the system level clock sources.
Crystal Oscillator - System (OSC_SYS)	Generates, in conjunction with an external crystal or resonator, a reference clock for the chip.
Free Running Oscillator - 192 MHz (FRO_192M)	The free running oscillator can generate either 192 MHz for use by the chip.
Free Running Oscillator - 12 MHz (FRO12M)	Is an internal clock source that generates a 12 MHz frequency for use by the chip.

2.5.3 Communication

The following communication modules are available on this chip.

Table 7. Communication modules

Module	Description
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)	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.
Universal Serial Bus Full Speed Device Controller (USBFS)	In Device mode, the USB FS subsystem working together with the SCG module can use the FIRC to generate a 48 MHz USB controller clock tuned using the incoming Host USB data for crystal-less operation.
Flexible Data Rate CAN (FlexCAN)	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.
Flexible I/O (FLEXIO)	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 which allow 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 the CPU

2.5.4 Debug

The following debug modules are available on this chip.

Table 8. Debug modules

Module	Description
Data Watchpoint and Trace (DWT)	Is a generic name for modules that allow debug access of the Cortex-M33. The DWT comprises of the Debug Watchpoint and Trace (DWT) module and the Flash Patch and Breakpoint (FPB) unit.
Debug Access Port (DAP)	Enables real-time access to the chip registers from an external debugger without halting the processor cores.
Instruction Trace Macrocell (ITM)	Provides a memory-mapped register interface that applications can use to write logging or event words for profiling software.
Joint Test Action Group (JTAG)	Implements serial communication protocol for communication with the Test Access Port (TAP).
Serial Wire Debug (SWD)	Is a serial communication interface used for debugging devices with multiple cores while only requiring a single external interface.
Trace Port Interface Unit (TPIU)	Acts as a bridge between the on-chip trace data from the modules such as the ITM, which have separate system IDs, to the external world.

2.5.5 Memory

The following memory modules are available on this chip.

Table 9. Memory modules

Module	Description
Flash Memory Controller (FMC)	Is a programmable flash memory — non-volatile flash memory that can store executable program code or data.
Flash Management Unit (FMU)	Manages the interface between the chip and the on-chip flash memory.
Static Random Access Memory (SRAM)	Internal system SRAM memory. Each individual block of SRAM can be configured to be retained in low-power modes.
AHB Low Power Cache Controller (LPCAC)	Is a processor-local level 1 (L1) bus cache controller for use with cores using AMBA-AHB input/output buses.

2.5.6 System

The following system modules are available on this chip.

Table 10. System modules

Module	Description
Debug mailbox	Supports Arm Serial Wire Debug mode.

Table continues on the next page...

Table 10. System modules (continued)

Module	Description
Peripheral Bridge(PBRIDGE)	Converts an AMBA AHB interface to a peripheral interface that allows the Peripheral Bridge (PBRIDGE) controller to interface to multiple peripherals. This device has two peripheral bridges.
System Controller (SYSCON)	Provides controls and configurations of the system and peripherals for the multiple functions.
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.
Enhanced Direct Memory Access (eDMA)	Performs source/destination address calculations and data-movement operations. Capable of performing complex data transfers with minimal intervention from a host processor.
Wake-up Unit (WUU)	Allows selection of external pins and internal modules as interrupt wake-up sources from Deep Sleep and Power Down modes.
Peripheral Input Multiplexing (INPUTMUX)	Allows the trigger output of one peripheral to be connected to the trigger input of a second peripheral.
Error Injection Module (EIM)	Provides a method for diagnostic coverage of internal memories (for example, system RAM, cache RAMs, and peripheral memories).
Error Reporting Module (ERM)	Provides information and optional interrupt notification on memory error events associated with Error correction code (ECC) and parity.
System Power Controller (SPC)	Provides control over the operation and configuration of the system power generation modules to optimize power consumption for the level of functionality needed.
Low Drop Out Regulator - Core (LDO_CORE)	A voltage regulator for generating the core voltage for the chip.
RAM_RET_LDO	A voltage regulator for generating the voltage for SRAM retention.

2.5.7 Security

The following security modules are available on this chip.

Table 11. Security modules

Module	Description
Cyclic Redundancy Check (CRC)	Provides error detection for all single, double, and many multi-bit errors.
GLIKEY	It provides a mechanism to safely access security-sensitive registers.
Code Watch Dog	Helps protect the integrity of software by detecting unexpected changes (faults) in the code execution flow.

Table continues on the next page...

Table 11. Security modules (continued)

Module	Description
Memory Block Checker (MBC)	Provides domain-based access control for all system bus references targeted to on-chip internal memories and slave peripherals. It's a part of TRDC module.

2.5.8 Timer

The following timer modules are available on this chip.

Table 12. Timer modules

Module	Description
FlexPWM	Generates various switching patterns, including highly sophisticated waveforms. Controls different Switched Mode Power Supplies (SMPS) topologies.
Wake Timer	Provides time keeping and calendaring functions and additionally provides protection against spurious memory/register updates and battery operation.
Quadrature Decoder	Interfaces to position/speed sensors that are used in industrial motor control applications.
Micro-Tick (UTICK) Timer	Is a 31-bit timer that provides a fixed time interval between interrupts.
OS Event Timer (OSTIMER)	Is a 42-bit Gray code counter.
Low-power Timer (LPTMR)	Is a 16-bit timer or pulse counter with compare feature.
Standard Counter/Timer (CTimer)	Each Counter/timer is designed to count cycles of the CTIMER function clock.
Windowed Watchdog Timer (WWDT)	Helps reset or interrupt an erroneous microcontroller within a programmable time.
Frequency Measurement	Provides high-accuracy frequency measurement function for on-chip and off-chip clocks.

2.5.9 Human Machine Interface (HMI)

The following HMI modules are available on this chip.

Table 13. HMI modules

Module	Description
General Purpose Input/Output (GPIO)	All GPIO pins support interrupt and DMA request generation.
Port Control (PORT)	Provides support for pad control functions.

2.5.10 Analog

The following analog modules are available on this chip.

Table 14. Analog modules

Module	Description
Analog-to-Digital Converter (ADC)	Successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.
Low Power Comparator (LPCMP)	Provides a circuit for comparing two analog input voltages.
LPDAC	The Low Power Digital-to-Analog Converter(LPDAC) is a low power, 12-bit general-purpose digital-to-analog converter. The output of the LPDAC can be placed on an external pin or set as one of the inputs to the analog comparator or ADC.
Operational Amplifier (OPAMP)	The OPAMP supports PGA amplifier. Positive and negative inputs of amplifier connect to internal channels. Selecting different gains by configuring registers which contains optional non-inverting or inverting gain application. The module is applicable to the signal processing stage before SARADC.

Chapter 3
Core Overview

3.1 Introduction

This section covers the core modules included in this chip.

3.2 Cortex-M33 Code and System buses

This device has one Arm Cortex-M33 processor core, with Floating Point Unit (FPU) and without TrustZone-M, Memory Protection Unit (MPU).

The core 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 Cache Controller; code bus of the core goes to LPCAC. 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

All System bus accesses are routed to the target address in destination memories through multilayer AHB matrix slave port.

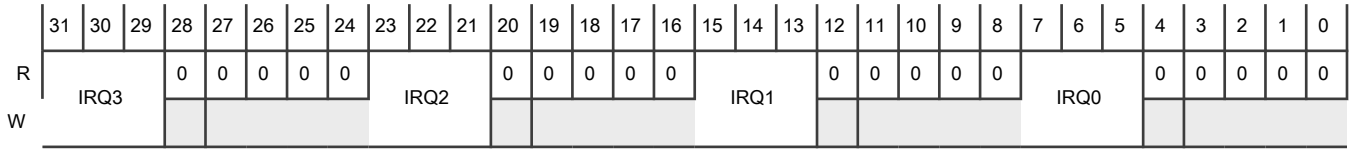
3.2.3 Access control

All core Code and System Bus accesses are checked by the core access control logic. All requests that miss or bypass the cache are checked by downstream secure AHB bus 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.

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) enable bit and source selection bits are implemented for each core in SYSCON register.

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.

See the attached NVIC_Configuration spreadsheet for the NVIC interrupt assignments.

3.4 System memory map

See the attached Memory Map spreadsheet (Overview tab) for the chip's high-level memory map.

3.5 Peripheral Bridge

The Peripheral Bridge (PBRG) 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.

3.5.1 Peripheral Bridge (PBRIDGE0-1) memory maps

See the attached Memory Map spreadsheet for PBRIDGE0, PBRIDGE1, and Fast Peripherals (Peripheral Memory Map tab) peripheral memory mapping on this device.

Chapter 4
Bus and Memory Architecture

4.1 Bus and Memory architecture

This chapter provides an overview of system bus and memory architecture of this device. The memory architecture supports a wide range of target applications and maximizes performance while maintaining low power consumption. The memory system of the device includes SRAM, ROM, internal flash, and external memory.

The following figure shows the system bus diagram of this chip.

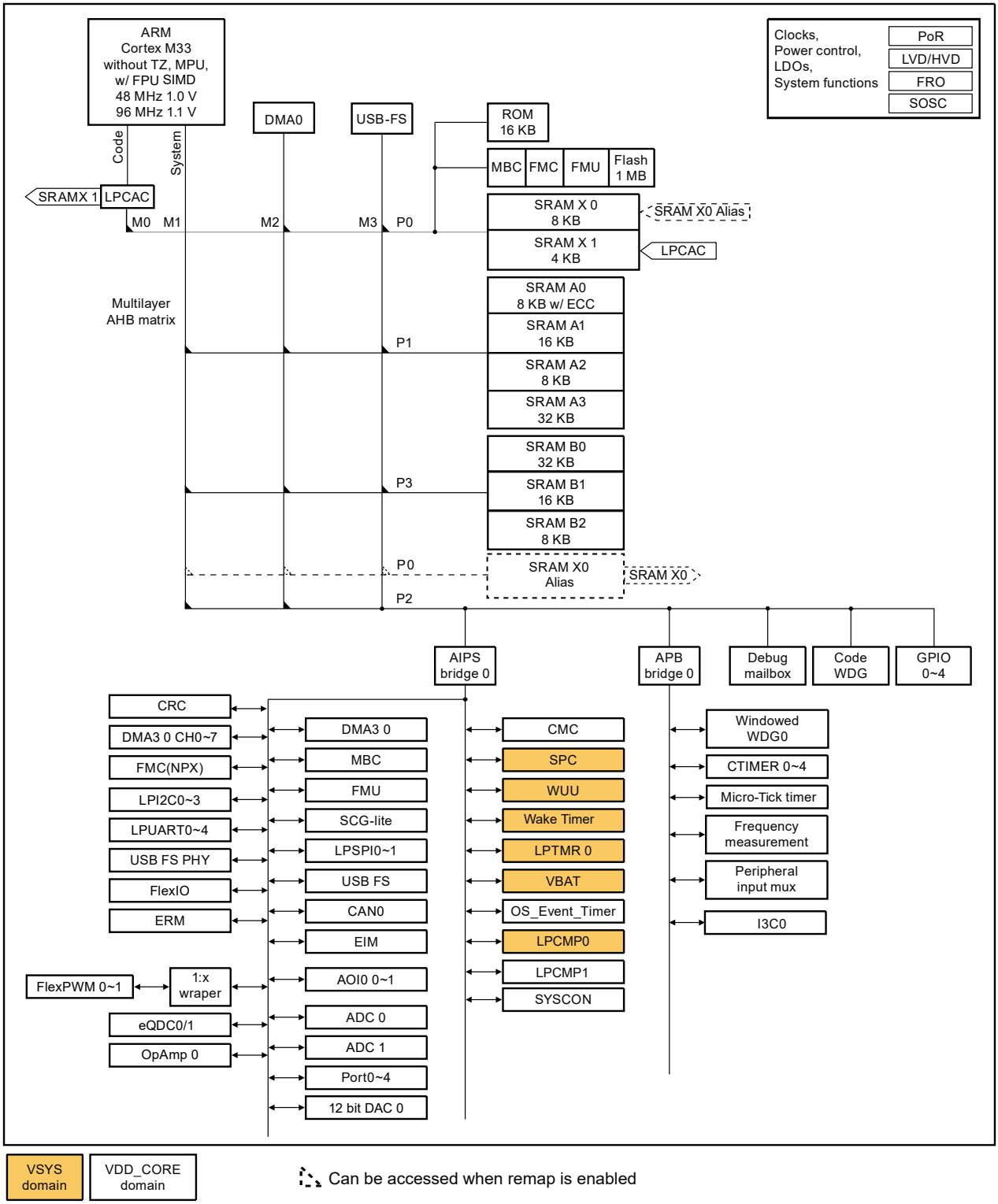


Figure 6. Bus matrix block diagram

4.2 System bus priority and arbitration

The multilayer AHB bus matrix allows for concurrent accesses when two masters are attempting to access to different slave ports. If two masters attempt an access to the same slave port at the same time then arbitration is required.

The SYSCON_AHBMATPRIO register is where the programmable priorities for each of the master ports can be configured. Masters are assigned a priority value between 0 and 3 with 3 being the highest priority. If two ports have the same priority, then the lowest port number is given priority.

There are some master ports that are shared between two masters. Where the port is shared, only one of the masters can have an active access at a time. The priority between two masters sharing a port uses a PARK_ON_LAST arbitration scheme. This means that the last of the two masters to use the port is given priority. Before either of the masters has used the port, there is a default priority scheme that determines which master is prioritized for the first access. The table below lists the master or masters for each of the ports.

Table 16. AHB Matrix priority control register

Bit	Master Name	Master AHB ID	Reset
1:0	CM33 Code Bus	0x0	0x0
3:2	CM33 System Bus	0x1	0x0
5:4	Reserved	0x2	0x0
7:6	Reserved	0x3	0x0
9:8	eDMA Controller AHB Bus	0x5	0x0
11:10	Reserved	0x4	0x0
13:12	Reserved	0xC	0x0
15:14	Reserved	0x7	0x0
17:16	Reserved	0x8	0x0
19:18	Reserved	0x9	0x0
21:20	Reserved	0xA	0x0
23:22	Reserved	0xB	0x0
25:24	USB FS AHB Bus	0x6	0x0
27:26	Reserved	0xD	0x0
29:28	Reserved	0xE	0x0
31:30	Reserved	Reserved	0x0

AHB to APB Bridge

The bridge connects legacy LPC APB/VPB interface peripherals.

AIPS bridge

The bridge connects IPS/APB interface peripherals.

4.3 SRAM configuration

SRAM is divided into Code TCM and System TCM.

CTCM: Mapped to CM33 code bus space

- RAM X0: 8 KB 32-bit RAM
- RAM X1: 4 KB 32-bit RAM

Can only be used as code RAM when LPCAC is disabled.

STCM: Mapped to CM33 system bus space

- RAM A0: 8KB 32+7bit ECC RAM
- RAM A1: 16KB 32bit RAM
- RAM A2: 8KB 32bit RAM
- RAM A3: 32KB 32bit RAM
- RAM B0: 32KB 32bit RAM
- RAM B1: 16KB 32bit RAM
- RAM B2: 8KB 32bit RAM

SRAM permission control

This chip implements a logic between RAM controller and AHB Slave port to control SRAM write permission. SRAM write permission control registers is implemented in SRAM_XEN[4:0] bits of the SYSCON as below

0: disable, default value

1: enable

- SRAM_XEN[0] controls RAM X0 8KB, offset 0x0~0x1FFF
- SRAM_XEN[1] controls RAM X1 4KB, offset 0x2000~above
- SRAM_XEN[2] controls RAM A0 8KB, offset 0x0~0x1FFF
- SRAM_XEN[3] controls RAM A1 16KB, RAM A2 8KB, RAM A3 32KB, offset 0x2000~0xFFFF
- SRAM_XEN[4] controls RAM B0 32KB, RAM B1 16KB, RAM B2 8KB, offset 0x1_0000~above
- A lock bit is also implemented in SRAM_XEN. Once locked, the register can't be programmed unless there is a reset.

GLIKEY is used to protect the access of SRAM_XEN register.

The figure below shows the SRAM configuration on this chip.

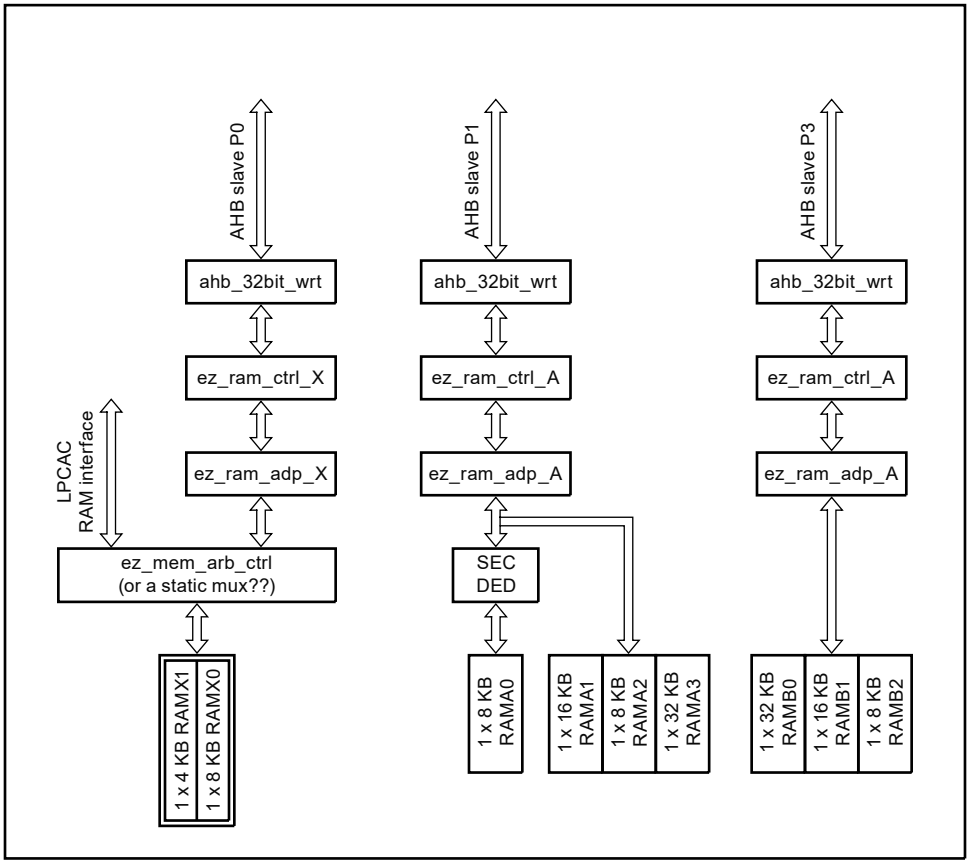


Figure 7. SRAM configuration

4.3.1 SRAM ECC

RAM A0 supports single-bit correction, and two bits detection ECC on every 32-bit aligned field. ECC supports byte and half-word operation. ECC can be disabled via SYSCON register.

4.3.2 RAM ECC Error

The Error Reporting Module (ERM) provides information and optional interrupt notification on SRAM error events associated with ECC. The syndrome and error address information is captured along with error event in ERM registers.

The Error Injection Module (EIM) can be used to induce artificial errors in the RAM ECC. EIM can inject single-bit and multi-bit inversions on data. Injecting faults on memory accesses can be used to exercise the SEC-DED ECC function of the related system.

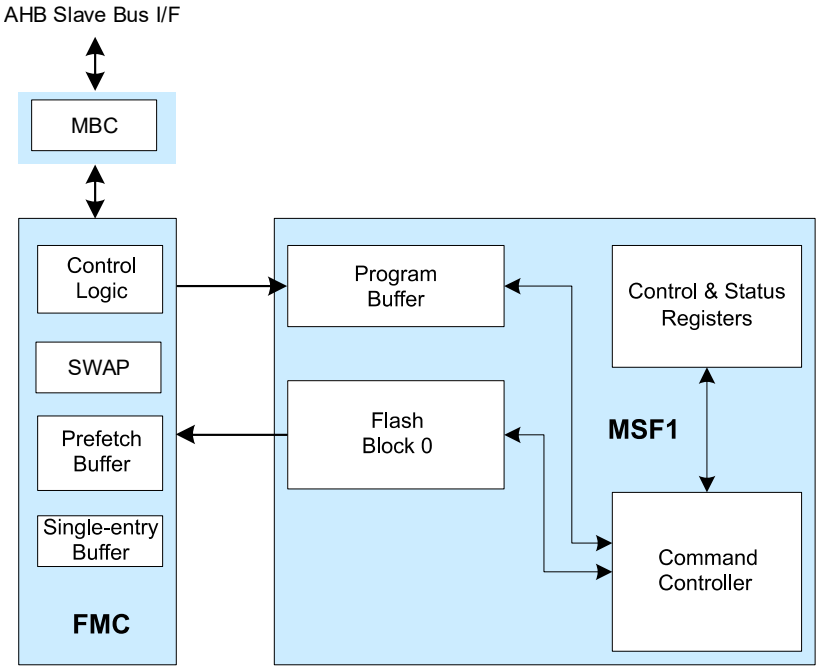
4.4 Read Only Memory (ROM)

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.

4.5 Flash subsystem

This device embeds up to 1 MB of single-array flash configuration. The Flash subsystem includes MBC, FMC and FMU (MSF1). FMC implements a 128-bit entry buffer and a 128-bit prefetch buffer. The control register for prefetch buffer is implemented in the System Controller (SYSCON) chapter.

The following figure shows the high-level block diagram of the on-chip flash.



Flash array configuration

This chip has up to 1 MB flash array, which includes up to 1 MB main array, 32 KB IFR0, and 8KB IFR1. The sector1 to sector3 IFR0 are used to store flash loader, which is programed by NXP. There are boot options provided for user in sector0 IFR0, which must be programmed by user. There is IFR0 table to define the details of boot options. See Boot chapter for details.

Flash ECC

Flash array supports ECC on every 128 bits. This chip implements a register in the SYSCON to disable Flash ECC error to generate bus fault, and use interrupt to handle ECC error.

Flash swap

FMC supports the swap function. The single flash array is divided to four regions. Two swappable regions and two static regions. When swap enables, swappable region 1 is mapped to logic address 0. Below is an example: Considering flash size is 128 KB:

- Swappable Region 0: flash physical address 0 ~ Upper limit
- Static Region 0: Upper limit ~ 64 KB
- Swappable Region 1: 64 KB ~ 64 KB + Upper limit
- Static Region 1: 64 KB + Upper limit ~ Flash End

SWAP enabled

- Map Swappable Region 1 to logic address starting at 0x0 ~ upper limit
- Static Region 0 logic address unchanged
- Map Swappable Region 0 to logic address starting at 0x10000 ~ 0x10000 + upper limit
- Static Region 1 unchanged

Swap granularity is 8 KB. The chip supports 1 MB flash array.

Flash performance

- 96 MHz SD mode, 2 wait states

- 48 MHz, MD mode, 1 wait state

4.5.1 Low-Power Cache Controller (LPCAC)

This chip implements 4 KB cache between CM33 code bus and AHB Matrix.

4.5.2 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 two 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

NOTE

The FMC has no visibility into flash memory erase and program cycles because the Flash Memory module manages them directly.

4.5.2.1 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.

4.5.2.2 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.

Chapter 5
Low Power Cache Controller (LPCAC)

5.1 Chip-specific LPCAC information

Table 17. Reference links to related information

Topic	Related module	Reference
Full description	LPCAC	LPCAC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

NOTE

See LPCAC Control (LPCAC_CTRL).

5.1.1 Module instances

This device has one instance of the LPCAC module.

5.1.2 LPCAC cacheable memory range

The LPCAC module has no registers. All configuration for the LPCAC module is handled through the LPCAC_CTRL register of SYSCON. The 4 KB LPCAC is connected to the Code bus of the primary CM33 core. Contents of this cache are only visible to the CM33 core. The cacheable memory range are shown in table below. For details, see the attached System memory map.

Table 18. LPCAC cacheable memory range

Start Address	End Address	Description	Size	Cached
0000_0000	000F_FFFF	Program Flash	1MB	LPCAC

5.2 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 data path AMBA-AHB input bus and a 32-bit data path AMBA-AHB output bus.

5.2.1 Block diagram

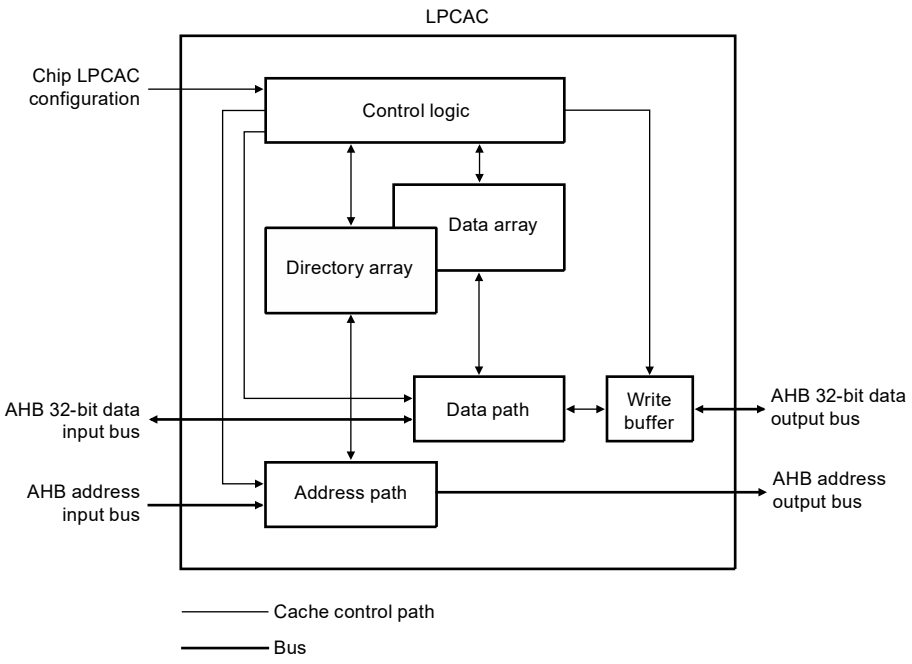


Figure 8. Block diagram

5.2.2 Features

LPCAC supports the following features:

- Nonblocking and write-through cache mode
- 4 KB total cache size
- Cache organization as an 8-way, 2-set-associative design based on 256-byte superpages
- Each way or superpage contains up to sixteen sequential 16-byte pages

5.3 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 sixteen 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, 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 LPCAC is enabled and available, cache write accesses with a bufferable attribute use the buffer, which allows write from the processor to receive an immediate (zero wait state) bus termination.

5.3.1 Cache functional description

The LPCAC cache is an 8-way, 2-set-associative, 256-byte per line write-through design. It supports a total cache capacity of 4 KB for a 32-bit wide cache miss data path.

5.3.1.1 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

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

5.3.2 Clocks

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

5.3.3 Reset

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

5.3.4 Interrupts

This module has no interrupts.

5.4 Signal descriptions

This module has no external signals.

5.5 Memory regions and input control description

5.5.1 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 region 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 19. Memory map

Cache memory range	Cache memory range	Cache total memory address space
0000_0000h–000F_FFFFh (1 MB)	--	1 MB

5.5.2 Input controls

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

Table 20. 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	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. If there is a software breakpoint, halting the processor. Then, using the debug port, cacheable memory and 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
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.

Chapter 6
Flash Memory Module (FMU)

6.1 Chip-specific MSF1 information

Table 21. Reference links to related information

Topic	Related module	Reference
Full description	MSF1	Flash Memory Module
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

6.1.1 Module instances

This device contains one instance of the FMU module, FMU0.

6.1.2 FCTRL[RWSC] Configuration

The configuration of Read Wait-State Control (RWSC) field of Flash Control Register (FCTRL) is shown in below table.

Table 22. FCTRL[RWSC] configuration in different run modes and speed

SD Mode (VDD_CORE = 1.1V)	MD Mode (VDD_CORE = 1.0V)	RWSC
96 MHz		0010b
48 MHz, 64 MHz	32 MHz, 48 MHz	0001b (default)
<=32 MHz	<=24MHz	0000b

NOTE

Erase All command and Mass Erase operation are not supported in this device.

6.2 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.

6.2.1 Block diagram

The block diagram of the flash module is shown in the following figure.

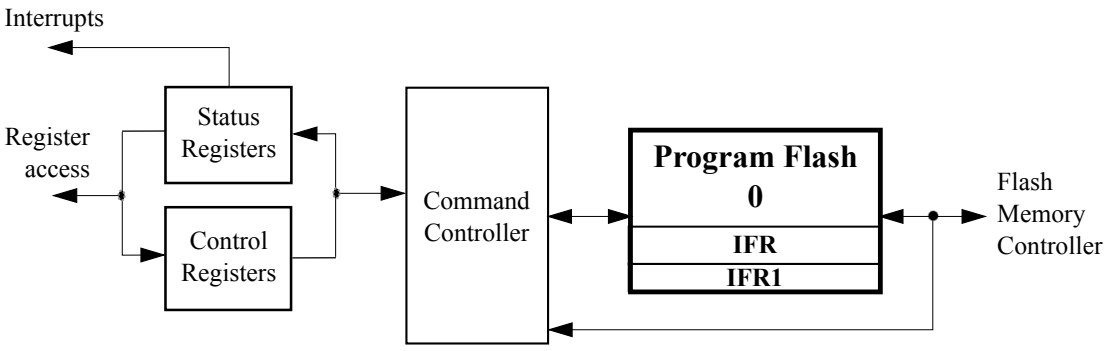


Figure 9. Flash block diagram

6.2.2 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.

6.3 Functional description

The following sections describe functional details of the flash module.

6.3.1 Operations

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

6.3.1.1 Command write sequence

Flash commands are accomplished using a command write sequence illustrated in Figure 10. 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.

While a flash command is executing, attempts to write to FCTRL[LSACTIVE] are ignored.

6.3.1.1.1 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.

6.3.1.1.2 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.

6.3.1.1.3 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.

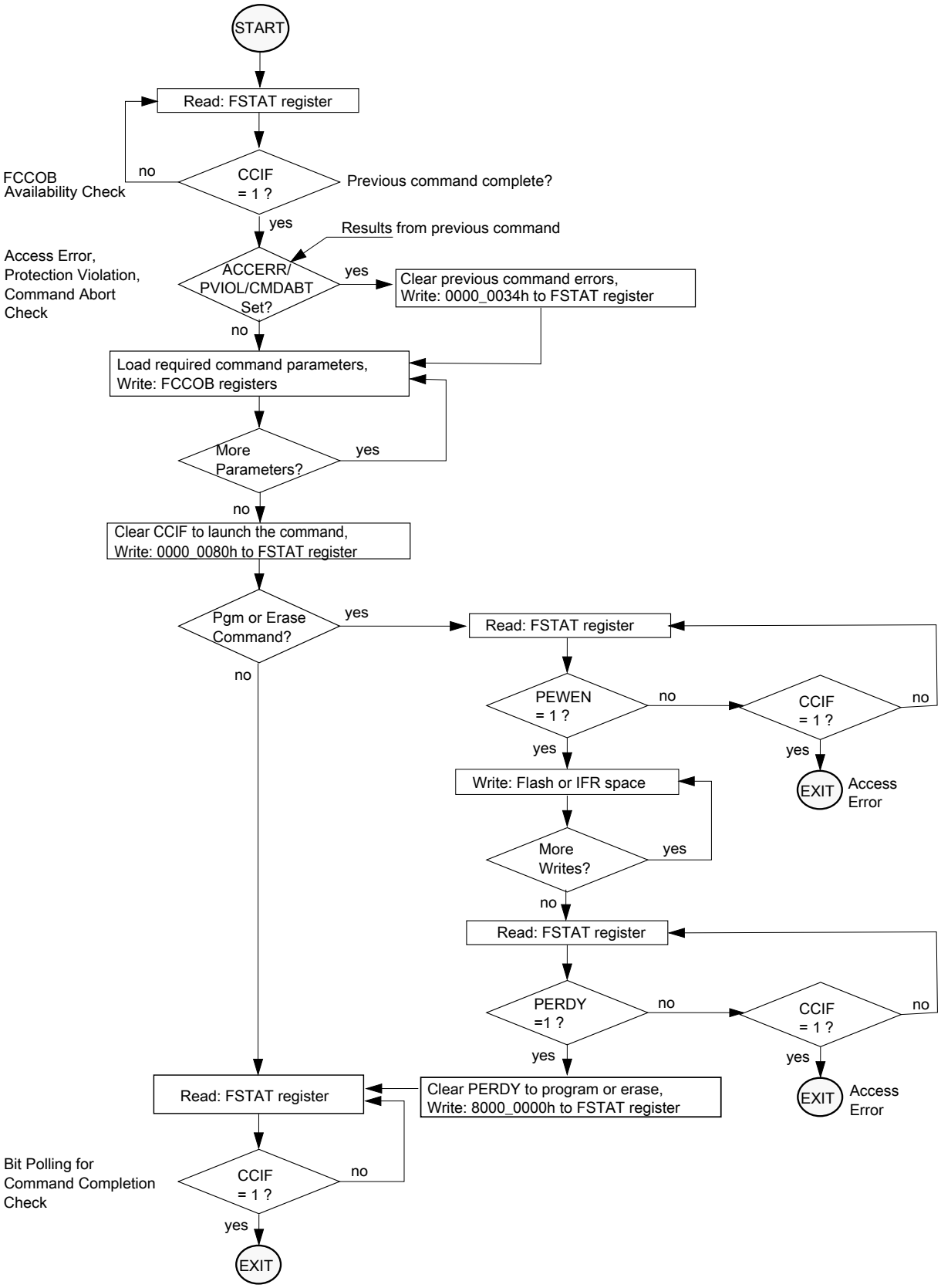


Figure 10. Generic Flash Command Write Sequence Flowchart

6.3.1.1.4 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.

6.3.1.2 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

6.3.1.3 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 23. 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

6.3.2 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.

6.3.2.1 Read 1s All Command

The Read 1s All command checks if all flash and IFR space are in the erased state.

Table 24. 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 25. 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]

6.3.2.2 Read 1s Block Command

The Read 1s Block command checks if a flash block is in the erased state.

Table 26. 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 27. 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 27. 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]

6.3.2.3 Read 1s Sector Command

The Read 1s Sector command checks if a flash sector is in the erased state.

Table 28. 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 29. 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]

6.3.2.4 Read 1s Page Command

The Read 1s Page command checks if a flash page is in the erased state.

Table 30. 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 31. 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]

6.3.2.5 Read 1s Phrase Command

The Read 1s Phrase command checks if a flash phrase is in the erased state.

Table 32. 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 33. 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]

6.3.2.6 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 34. 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 35. 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]

6.3.2.7 Read 1s IFR Sector Command

The Read 1s IFR Sector command checks if an IFR sector is in the erased state.

Table 36. 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 37. 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]

6.3.2.8 Read 1s IFR Page Command

The Read 1s IFR Page command checks if an IFR page is in the erased state.

Table 38. 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 39. 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 39. 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]

6.3.2.9 Read 1s IFR Phrase Command

The Read 1s IFR Phrase command checks if an IFR phrase is in the erased state.

Table 40. 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 41. 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]

6.3.2.10 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 42. 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 42. 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 43. 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]

6.3.2.11 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 44. 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.

Table 45. 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]

6.3.2.11.1 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.

6.3.2.12 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 46. 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.

Table 47. 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]

6.3.2.12.1 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.

6.3.2.13 Erase All command

The Erase All operation erases all flash and IFR space if enabled by the MCU.

Table 48. 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.

Table 49. 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]

6.3.2.14 Erase Sector command

The Erase Sector operation erases the IFR sector to prepare the sector for programming.

Table 50. 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.

Table 51. 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]

6.3.3 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].

6.3.4 Modes

6.3.4.1 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).

6.3.4.2 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.

6.3.4.3 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.

6.3.5 Clocking

This module has no clocking considerations.

6.3.6 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 52. 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.

6.4 External signals

The flash module contains no signals that connect off-chip.

6.5 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.

6.6 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.

6.6.1 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 53. IFR map

Offset Byte Address	IFR Field Size (Bytes)
Block 0 IFR Sector 0	
0000h-1FFFh	8,192
Block 0 IFR Sector 1	
2000h-3FFFh	8,192
Block 0 IFR Sector 2	
4000h-5FFFh	8,192
Block 0 IFR Sector 3	
6000h-7FFFh	8,192

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 54. IFR1 map

Offset Byte Address	IFR1 Field Size (Bytes)
Block 0 IFR1 Sector	
0000h-15FFh	5,632
1600h-16FFh	256
1700h-1FFFh	2,304

6.6.2 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].

6.6.2.1 Flash register descriptions

6.6.2.1.1 Flash memory map

FMU0 base address: 4009_5000h

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
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

6.6.2.1.2 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
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 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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
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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Error not detected 1b - Error detected

6.6.2.1.3 Flash Configuration Register (FCNFG)

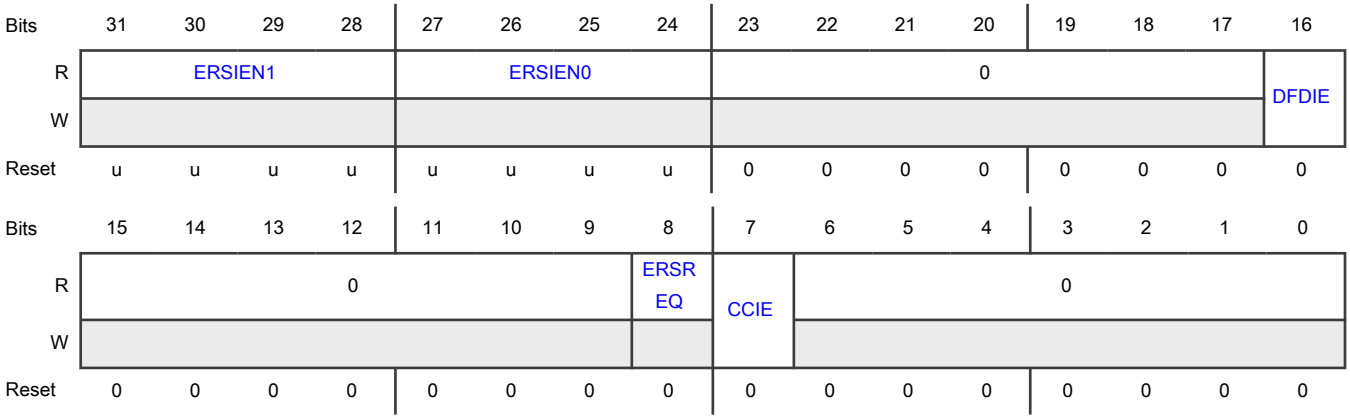
Offset

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 and at launch of ERSSCR command. 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	Erase IFR Sector Enable - Block 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERSIEN0	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 and at launch of ERSSCR command. 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.
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

6.6.2.1.4 Flash Control Register (FCTRL)

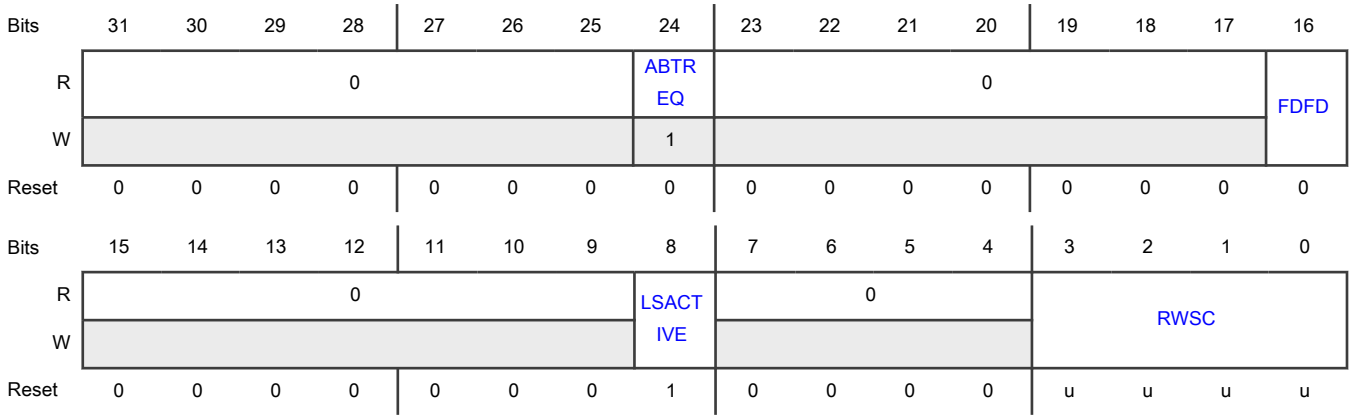
Offset

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 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.

6.6.2.1.5 Flash Common Command Object Registers (FCCOB0 - FCCOB7)

Offset

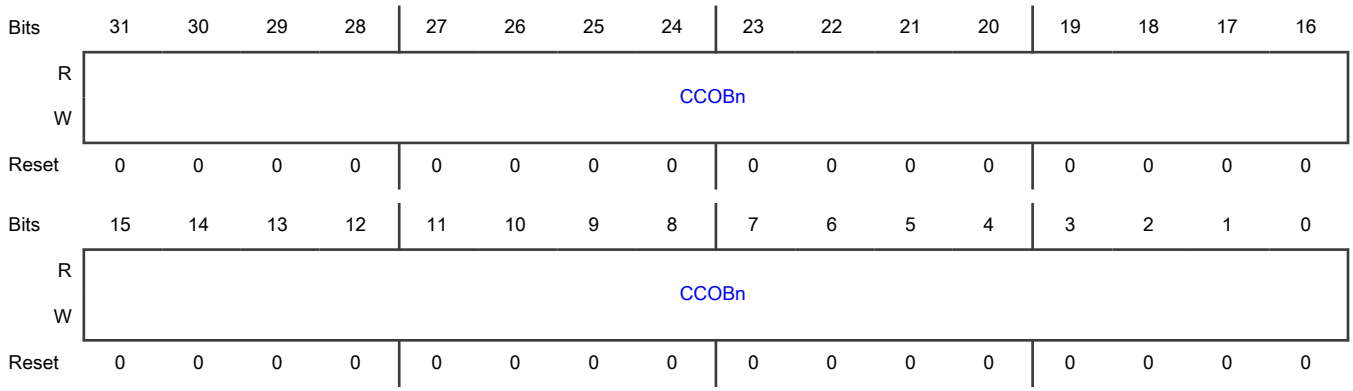
For a = 0 to 7:

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.

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
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.
<table>
<tr><th>FCCOB Number[1]</th><th>Typical Command Parameter Contents [31:0]</th></tr>
<tr><td>0</td><td>FCMD [7:0] (flash command code, bits [31:8] are not writable)</td></tr>
<tr><td>1</td><td>FCMDOPT [7:0] (flash command options, bits [31:8] are not writable)</td></tr>
<tr><td>2</td><td>Flash address 1 (start)</td></tr>
<tr><td>3</td><td>Flash address 2 (end)</td></tr>
<tr><td>4</td><td>Data Word 0</td></tr>
<tr><td>5</td><td>Data Word 1</td></tr>
<tr><td>6</td><td>Data Word 2</td></tr>
<tr><td>7</td><td>Data Word 3</td></tr>
<tr><td colspan="2">1. Refers to FCCOB register name, not register address</td></tr>
</table>
</td></tr>
</table>

6.7 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

Chapter 7
Flash Memory Controller (FMC)

7.1 Chip-specific FMC information

Table 55. Reference links to related information

Topic	Related module	Reference
Full description	FMC	FMC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

See NVM Control (NVM_CTRL) for details on flash speculation and flash cache control.

7.1.1 Module instances

This device has one instance of the FMC module, FMC0. It supports 1MB, 512KB, and 256KB phantom option.

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
  - A flash-phrase-sized buffer (128 bits) holds the most recently accessed flash phrase.
  - An optional flash-phrase-sized speculation buffer can prefetch the next flash phrase.

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 following table shows the supported read and 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 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 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. This buffer is used once for aligned 16-byte flash write phrases or eight 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

    See the chip-specific section for details about controls.

NOTE

Clear the speculation buffer before accessing recently modified flash addresses. To clear the speculation buffer, first disable then re-enable the speculation via other modules.

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, FMC is disabled automatically.

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.

7.3.3 Configuration options

The default configuration provides a high degree of flash acceleration, but 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 buffering controls allow other modules to tune resources to suit specific application requirements. The buffer are each controlled individually. The controls enable buffering and prefetching per access type (instruction fetch or data reference).

See the chip-specific section for details about the registers used to configure FMC.

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 single-entry buffer. Likewise, speculation can be enabled or disabled for either type of access.

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 usually require wait states.

FMC does not allow the configuration of wait states directly. Wait states can be controlled via FMU FCTRL[RWSC].

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 data accesses. Because many code accesses are sequential, using the speculative prefetch buffer improves performance in most cases.

See the chip-specific section for information about controlling speculative reads.

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 Remapping flash addresses

An address remap mechanism allows for the swapping (both ways) of a specified flash address range between the lower half and the upper half of flash memory. This remapping can facilitate software updates in the field without the need for address modifications in software. For example, to use the remap mechanism to update software in the lower portion of bank 0, follow these steps:

1. Load the updated software into the upper half of flash.
2. Enable remapping to automatically use the updated software.

In REMAP, remapping is enabled when LIM[22:16] = LIMDP[30:24], and LIM[22:16] is nonzero. LIM[22:16] and LIMDP[30:24] define the *remap_address*[19:13] for the remapping, providing an address range granularity of 8 KB. For a remapped FMC access to address *access_address*, see Table 56.

Table 56. Remapping *access_address*

When *access_address* originates in...	And is less than or equal to...	The access remaps to...	In...
flash lower half	*remap_address*[19:13]	flash upper half base + access_address	flash upper half
flash upper half	flash upper half base + *remap_address*[19:13]	*access_address* – flash upper half base	flash lower half

The address ranges that may be swapped depends on the total flash size available. For non-power-of-2 total flash sizes, the upper half of flash bank 0 cannot be swapped, as shown in Table 57.

Table 57. Remapping address ranges

For a flash size of...	This lower half flash range...	Remaps to this upper half flash range...	This lower half flash range cannot be remapped...
128 KB	0x0_0000 – 0x0_FFFF	0x1_0000 – 0x1_FFFF	—
96 KB	0x0_0000 – 0x0_7FFF	0x1_0000 – 0x1_7FFF	0x0_8000 – 0x0_FFFF
64 KB	0x0_0000 – 0x0_7FFF	0x0_8000 – 0x0_FFFF	—
48 KB	0x0_0000 – 0x0_3FFF	0x0_8000 – 0x0_BFFF	0x0_4000 – 0x0_7FFF
32 KB	0x0_0000 – 0x0_3FFF	0x0_4000 – 0x0_7FFF	—

When LIM = LIMDP = 0, FMC disables the remap function. When LIM = LIMDP = a nonzero value, the remapping address range is (LIM + 1) × 8 KB. For example:

- When LIM = LIMDP = 1, the range is ≤ 16 KB.
- When LIM = LIMDP = 2, the range is ≤ 32 KB.

7.3.7 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. To prevent the possibility of returning stale data when an application is executing flash memory commands, disable and/or flush FMC's current buffer and speculation buffer. While you can disable these features individually, we recommend disabling all of them when executing flash memory commands.

See the chip-specific section for details about the registers used to disable features.

7.6 Register descriptions

It has a Flash Remap Control register.

NOTE

See the chip-specific information for FMC controls.

NOTE

Any access to an undefined memory area results in a bus error.

7.6.1 NPX register descriptions

7.6.1.1 NPX memory map

FMC0 base address: 4009_4000h

Offset	Register	Width (In bits)	Access	Reset value
20h	Data Remap (REMAP)	32	RW	0000_0000h

7.6.1.2 Data Remap (REMAP)

Offset

Register	Offset
REMAP	20h

Function

Provides a data remapping mechanism.

To write to REMAP, the remap lock must be disabled (REMAPLK = 0). To control the remapping, follow these options:

- To write to LIMDP, write 0x*NNNN*A5A5 (where *NNNN* is data) to REMAP. In this case, bits 30 to 24 are written to LIMDP. All other fields of REMAP remain unchanged.
- To write to LIM, write 0x*NNNN*5A5A (where *NNNN* is data) to REMAP. In this case, bits 22 to 16 are written to LIM.
- Any other writes to REMAP are ignored.

Diagram

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

Fields

Field	Function
31 —	Always reads as 0. When writing, see the register description above.
30-24 LIMDP	LIMDP Remapping Address Defines remap_address[19:13] for remapping. Its value should be lower than the lower-half flash range.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 —	Always reads as 0. When writing, see the register description above.
22-16 LIM	LIM Remapping Address Defines remap_address[19:13] for remapping. Its value should be lower than the lower-half flash range.
15-1 —	Always reads as 0. When writing, see the register description above.
0 REMAPLK	Remap Lock Enable To set the remap lock, write 0xXXXXC3C3 (X = don't care) to REMAP. All other fields of REMAP remain unchanged. Set by software (sticky); cleared only by a reset. 0b - Lock disabled: can write to REMAP 1b - Lock enabled: cannot write to REMAP

Chapter 8
Error Injection Module (EIM)

8.1 Chip-specific Error Injection information

Table 58. Reference links to related information

Topic	Related module	Reference
Full description	EIM	Error Injection
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

8.1.1 Module instances

This device contains one instance of the Error Injection module, EIM0.

8.1.2 EIM channel mapping

This chip implements the EIM module for ECC RAM. It can inject error when read ECC RAM. It can be used for self-test. Only RAM A0 supports ECC on this device. EIM channel 0 is used to inject ECC error to RAM A0.See the functional block diagram of EIM in the following figure.

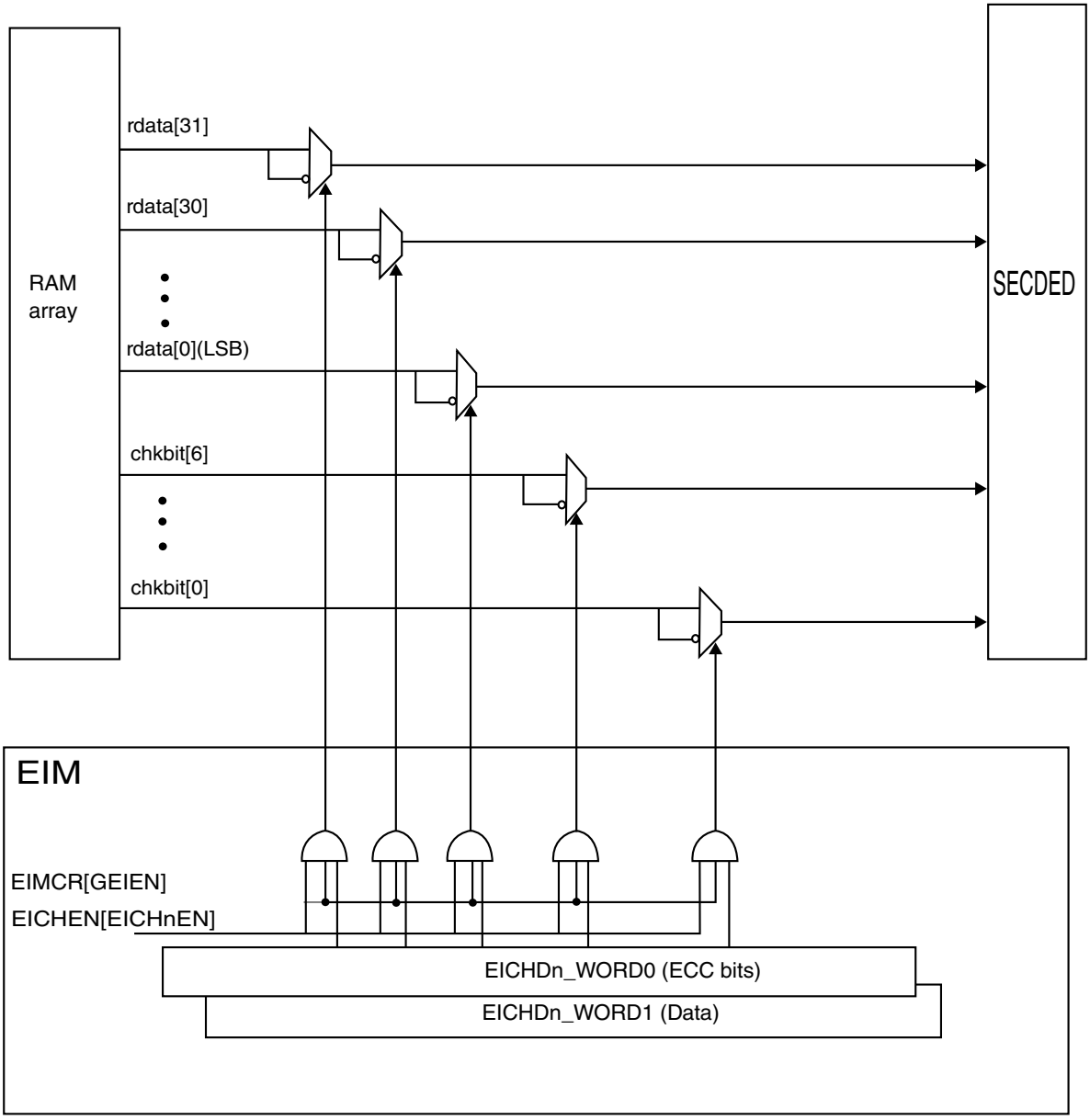


Figure 11. EIM functional block diagram

8.1.3 Access modes

If the privilege access is unavailable, the peripheral could be access in the user mode.

8.2 Overview

The Error Injection Module (EIM) is mainly used for diagnostic purposes. It provides a method to test the diagnostics (memory ECC, interconnect parity) by error injection in the field. See the chip-specific EIM information to determine which functional safety features are supported by this method.

EIM enables you to inject artificial errors on error-checking mechanisms of a system, such as ECC for RAM read data and parity bits. For each such mechanism that EIM supports on the chip, EIM can inject single-bit and multi-bit inversions on data in the applicable target bus. Injecting faults on memory accesses can be used to exercise the SEC-DED ECC function of the related system.

NOTE

Terminology in this chapter has been updated as follows:

Table 59. Updated terms

Updated term	Deprecated term
Controller	Master

8.2.1 Features

The EIM includes these features:

- Supports 1 error injection channel. See the chip-specific EIM information for channel assignment details.
- Protection against accidental enable and reconfiguration error injection function via two-stage enable mechanism

8.2.2 Block diagram

The following diagram shows an example of EIM implementation with a 64-bit read data bus and an 8-bit checkbit bus.

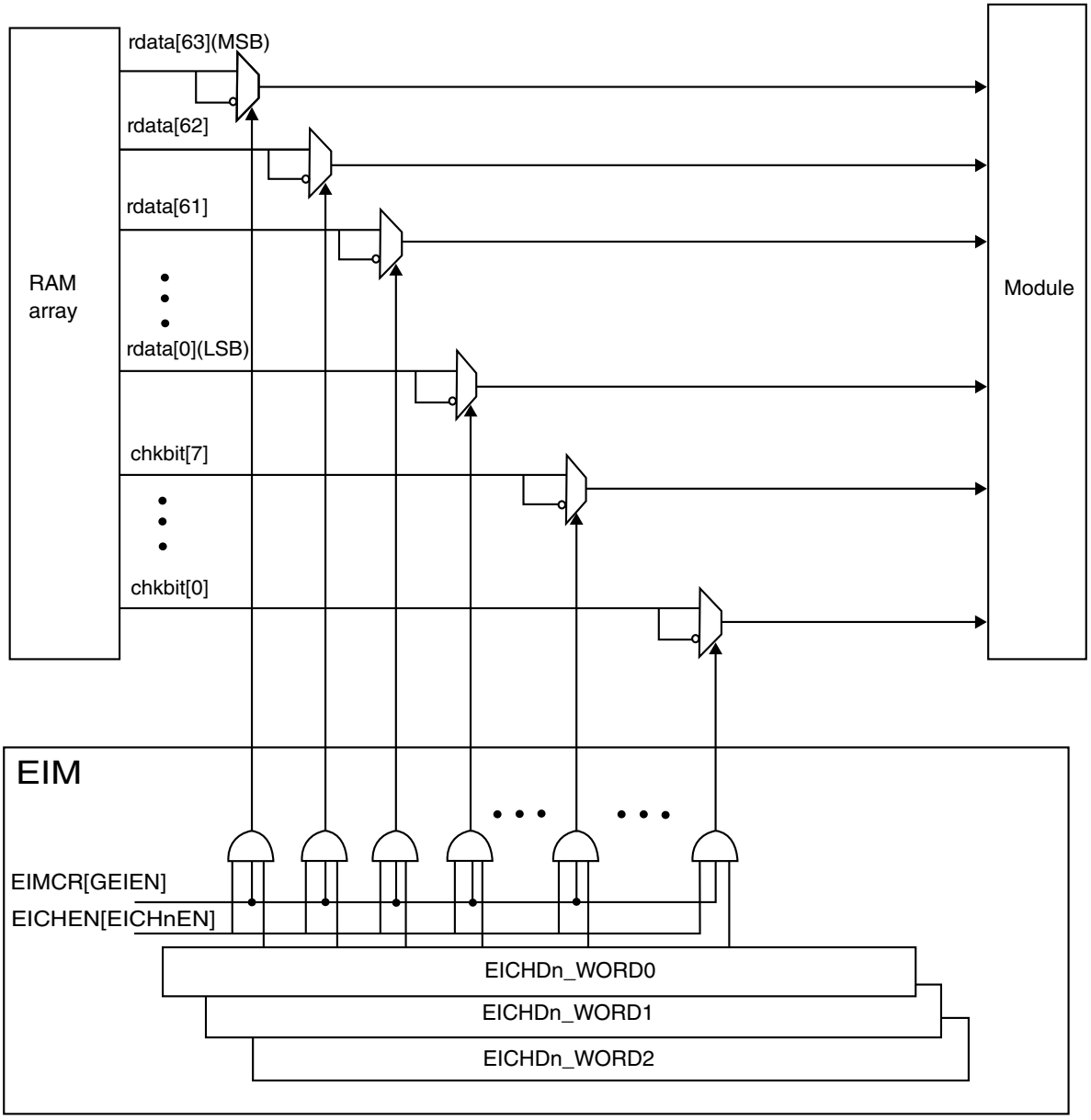


Figure 12. EIM functional block diagram (64-bit read data bus and 8-bit check bit bus)

Several memory elements are implemented within a device, which may not only be the large memory blocks (Flash and SRAM) but also smaller memories like caches, the TCD blocks, and the embedded peripheral memories. Some larger memories may actually be built from multiple memory elements, dependent on their size or function. Each of these memory elements implements its own control logic, the memory controller, that performs the accesses to the actual memory, the memory array. An EIM channel is associated with a memory controller and provides the capability to alter one or multiple signals in the read access path from the corresponding memory array(s). Only memory controllers controlling a safety related memory may be associated with an EIM channel.

8.3 Functional description

The EIM provides protection against accidental enabling and reconfiguration of the error injection function by enforcing a two-stage enablement mechanism. To properly enable the error injection mechanism for a channel:

- Write 1 to the EICHEN[EICH*n*EN] field, where *n* denotes the channel number.

- Write 1 to EIMCR[GEIEN].

NOTE

When the use case for a channel requires writing any EICHD*n*_WORD register, write the EICHD*n*_WORD register before executing the two-stage enablement mechanism. A successful write to any EICHD*n*_WORD register clears the corresponding EICHEN[EICH*n*EN] field.

The EIM supports 1 error injection channel. See the chip-specific EIM information for channel assignment details. Each channel:

- Can be assigned to a single memory array interface by intercepting the assigned memory read data bus and checkbit bus, and injects errors by inverting the value transmitted for selected bits on each bus line.
- Can be assigned to a redundant comparison unit by intercepting the signals being compared, and injecting errors by inverting the value transmitted for selected bits on each bus line.

On a memory read access, the applicable EICHD*n*_WORD registers define which bits of the read data and/or checkbit bus to invert.

Figure 12 depicts the interception and override of a 64-bit read data bus and an 8-bit checkbit data bus for an example memory array.

Error injection scenarios

The EIM supports these cases of error injection:

- To generate a single-bit error, invert only 1 bit of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.
- To generate a multi-bit error, invert only 2 bits of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.

NOTE

An attempt to invert more than 2 bits in one operation might result in undefined behavior.

To enable error injection:

1. Set the EICHD*n*_WORD*m*[CHKBIT_MASK] and EICHD*n*_WORD*m*[B*a*_*b*DATA_MASK] fields for each channel that will be driving an injection.
2. Program the EICHEN register to enable the channels that will be injecting errors.
3. Set the EIMCR[GEIEN] field to globally allow all enabled channels to actively inject errors.

To disable error injection, either disable the EIMCR[GEIEN] field or disable the individual channel enable fields of the EICHEN register.

8.4 Initialization

This module does not require initialization.

8.6 EIM register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus controllers can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 128-bit (16-byte) structure, composed of four 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1-3 (EICHD*n*_WORD1-3), if present, define the data mask. Word2 and Word3 are present only when required by the total width of the channel's data mask. See Error injection channel descriptor: DATA_MASK details.

Error injection channel descriptor: DATA_MASK details

The following table shows the total width of DATA_MASK and the distribution of its bits across the WORD registers.

Table 60. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	32	31-0	—	—

8.6.1 EIM memory map

EIM0 base address: 4008_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h

8.6.2 Error Injection Module Configuration Register (EIMCR)

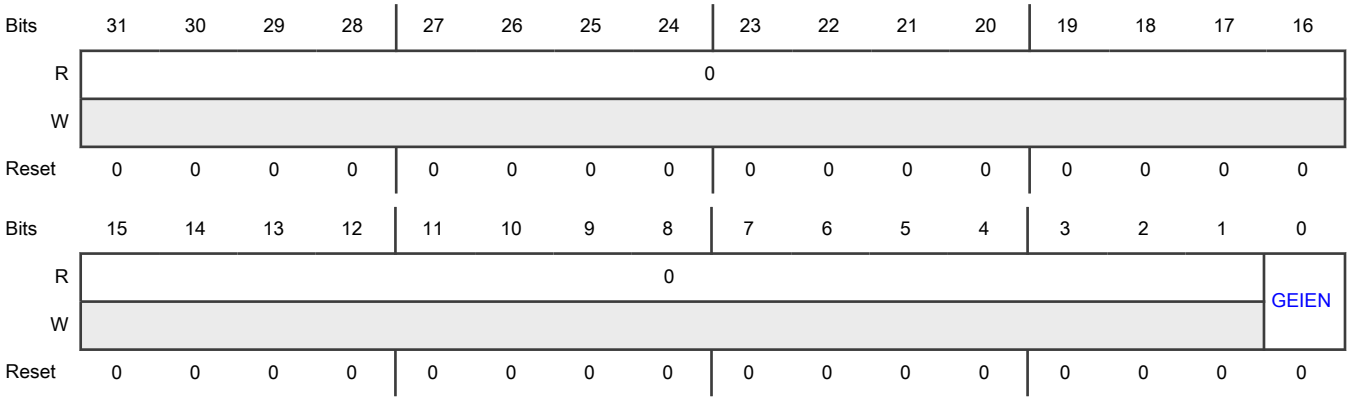
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

8.6.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	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	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 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

8.6.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

Offset

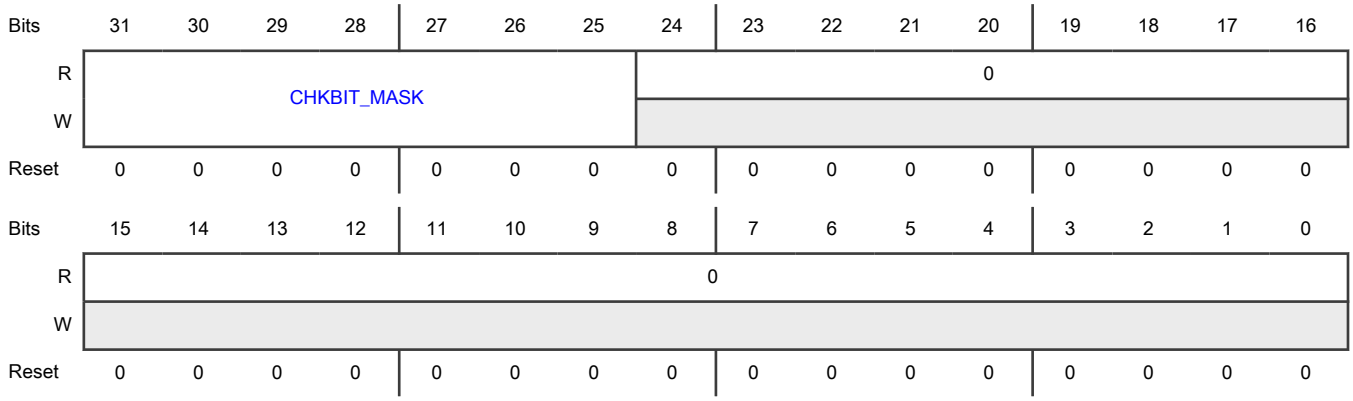
Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

8.6.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

Offset

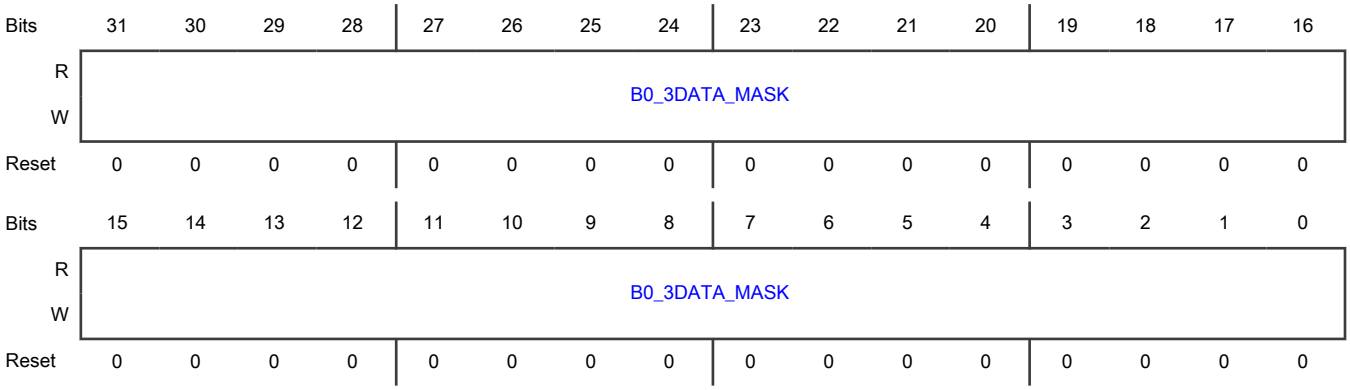
Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the target bus. Each bit specifies whether the corresponding bit of the target bus should be inverted

or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Chapter 9
Error Reporting Module (ERM)

9.1 Chip-specific Error Recording information

Table 61. Reference links to related information

Topic	Related module	Reference
Full description	ERM	Error Recording
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

9.1.1 Module instances

This device contains one instance of the Error Recording module, ERM0.

9.1.2 ERM channel mapping

ERM can record the address and ECC check bits (errsyndrome), when ECC error happens. ERM can be configured to generate interrupt when single bit or multi bit error happens. The error address, ECC check bits and ECC error outputs of SECDED should be connected to ERM.

Table 62. ERM channel mapping

Channel no.	Module	Captured status
0	RAM A0	Single-bit error, multiple bit error, error address syndrome
1	FMU0	ECC double-bit error detected on read data from array to the FMC

9.1.3 Access modes

If the privilege access is unavailable, the peripheral could be access in the user mode.

9.2 Overview

The Error Reporting Module (ERM) provides information and optional interrupt notification on memory error events associated with ECC and parity. The ERM collects error events on memory accesses for memory arrays, such as flash memory, system RAM, or peripheral RAMs. ERM supports various channels for memory sources where each ERM channel is associated with a different memory module. See the chip-specific ERM information for details about supported memory sources and specific memory channel assignments. If the memory supports ECC, then ERM syndrome and error address information is captured along with the error event. ERM does not capture syndrome or error address for cache memories or memory with parity instead of ECC.

9.2.1 Features

The ERM includes these features:

- Optional interrupt notification on captured error events
- Capturing of address and syndrome information on single-bit correction and non-correctable ECC events
- Support for error event capturing for memory sources, with individual reporting fields and interrupt configuration per memory channel
- Recording the count value of the number of corrected error events

9.2.2 Block diagram

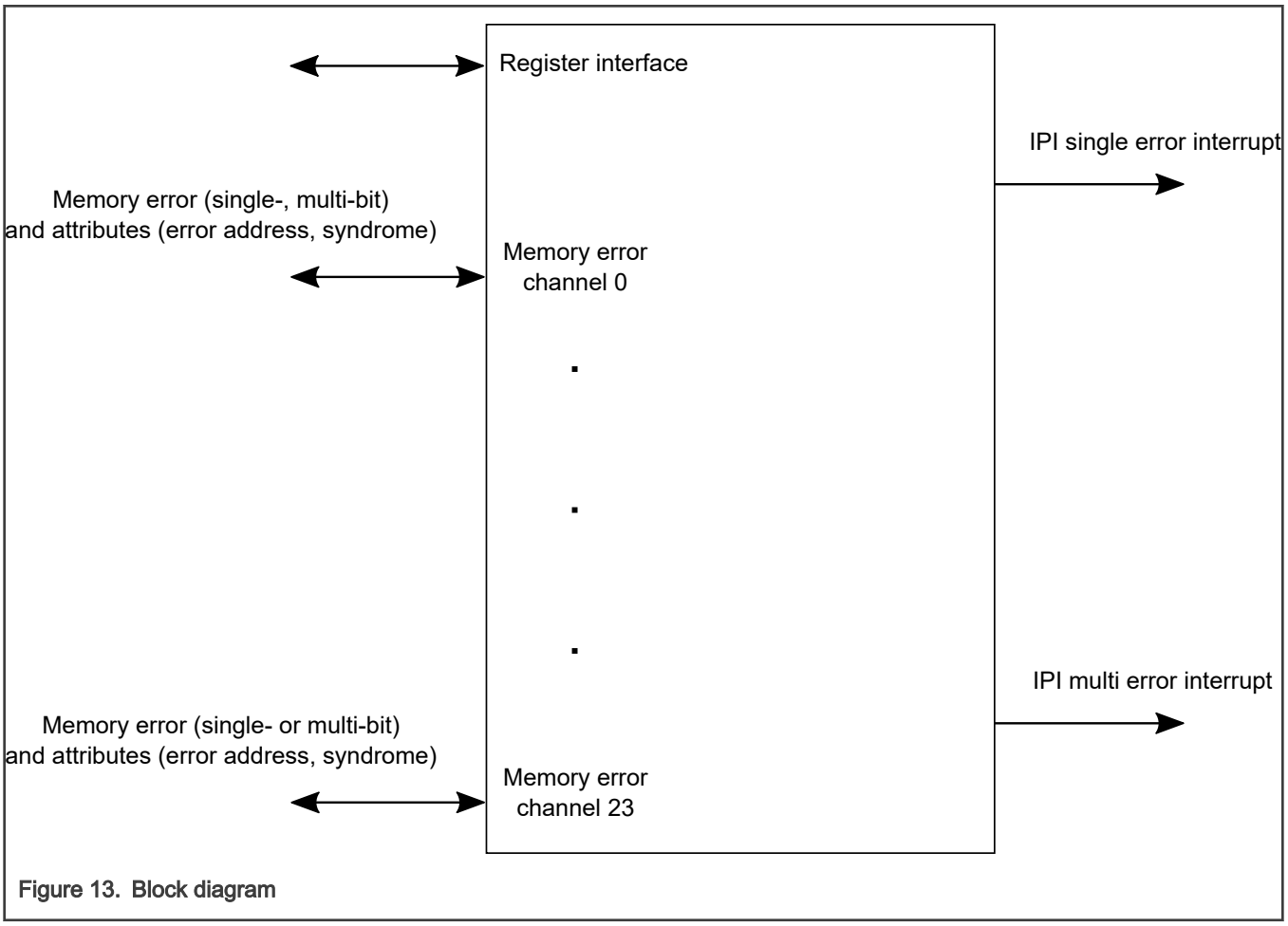


Figure 13. Block diagram

9.3 Functional description

9.3.1 Single-bit correction events

When a single-bit correction event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit SR*x*[SBC*n*] to 1.
- Increments the correctable error count value (until the counter reaches its maximum value): CORR_ERR_CNT*n*[COUNT].
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel). This register identifies the bit position of the corrected data on single-bit data inversion.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[SBC*n*] to change its value to 0.

To reset the correctable error count value, write all zeros to CORR_ERR_CNT*n*[COUNT].

Optional interrupt notification for single-bit correction events

The ERM provides an option to generate an interrupt notification upon the report of a single-bit correction event. To enable single-bit correction interrupts for a channel:

1. To enable interrupt notification for single-bit correction events on Memory *n*, set CR*x*[ESCIE*n*] to 1.
2. Subsequently, when a single-bit correction event on Memory *n* is detected, the ERM:
   - Records the event and address, and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[SBC*n*] to change its value to 0.

9.3.2 Non-correctable error events

When a non-correctable ECC error event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit: SR*x*[NCE*n*] to 1.
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel).
  - In the event of a non-correctable address bit inversion, SYN*n* identifies the pertinent address bit position.
  - In the event of a non-correctable, multi-bit data inversion, the syndrome value does not provide any additional diagnostic information.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[NCE*n*] to change its value to 0.

Optional interrupt notification for non-correctable error events

The ERM provides an option to generate an interrupt notification upon the report of a non-correctable ECC event. To enable non-correctable error interrupts for a channel:

1. To enable interrupt notifications for non-correctable error events on Memory *n*, set CR*x*[ENCIE*n*] to 1.
2. Subsequently, when a non-correctable error event on Memory *n* is detected, the ERM:
   - Records the event and address and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[NCE*n*] to change its value to 0.

NOTE

Parity errors can be mapped to non-correctable errors where error attributes like SYNDROME, ADDRESS are not provided.

9.4 Initialization

For each ERM channel supporting memory with ECC, prepare the corresponding memory array before enabling ERM interrupts about errors for that memory.

1. Initialize the memory to a known value so that the correct corresponding ECC codeword is stored.
2. During the memory's initialization, if the ERM captures information about any ECC error event, clear the corresponding SR*x*[SBC*n*] or SR*x*[NCE*n*] field that stores the record of the event.
3. Program the applicable CR*x*[ESCIE*n*] and CR*x*[ENCIE*n*] fields to enable ERM interrupts as desired.

9.5 ERM register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Based on the design implementation, the following XFR error behavior is evident at the IPS interface.

- Within the ERM memory map, an XFR error is evident at reserved addresses from location 20h to FFh.
- No XFR error is evident at reserved addresses in memory spaces allocated to each channel. For example: For channel 0, for read/write accesses to reserved address 10Ch, the XFR error is 0.
- For accesses to locations beyond the addresses allocated for the final channel, the XFR error is 1.

NOTE

- See the chip-specific ERM information at the beginning of this chapter for details on Memory channel mapping.
- To access the channel registers, corresponding memory channel clock must be enabled.

9.5.1 ERM memory map

ERM0 base address: 4008_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	RW	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	R	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	R	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	RW	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT1)	32	RW	0000_0000h

9.5.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		Reserved		0		Reserved		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	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification 0b - Interrupt notification of Memory 1 single-bit correction events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1-0 —	Reserved

9.5.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register reports error events for available channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		0				0			
W	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	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 SBC0	SBC0 Memory 0 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event Write 1 to clear this field.This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

9.5.4 ERM Memory 0 Error Address Register (EAR0)

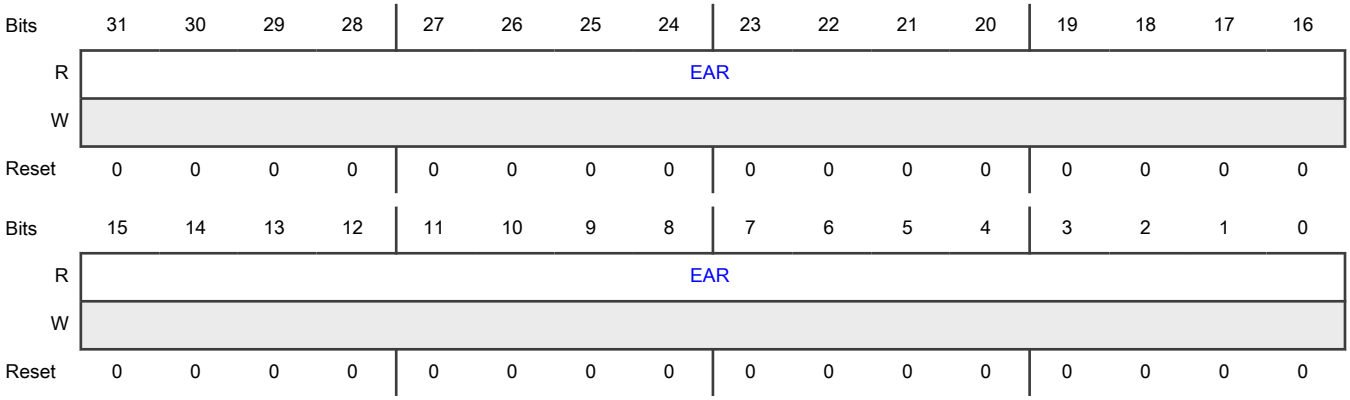
Offset

Register	Offset
EAR0	100h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory*n*, where*n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory*n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory*n*.

9.5.5 ERM Memory 0 Syndrome Register (SYN0)

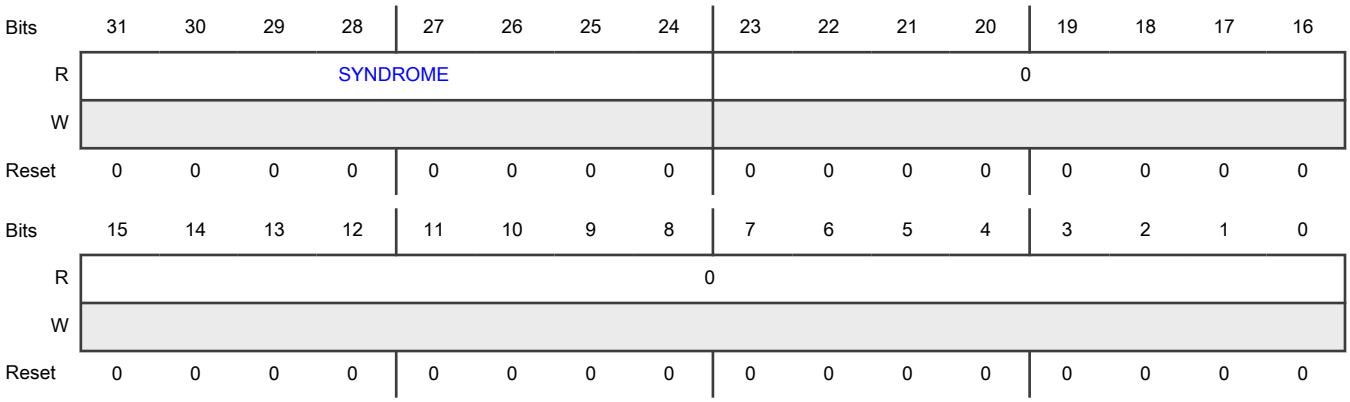
Offset

Register	Offset
SYN0	104h

Function

The ERM Memory *n* Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory*n*, where*n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory*n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory*n*.
23-0 —	Reserved

9.5.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT1)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h

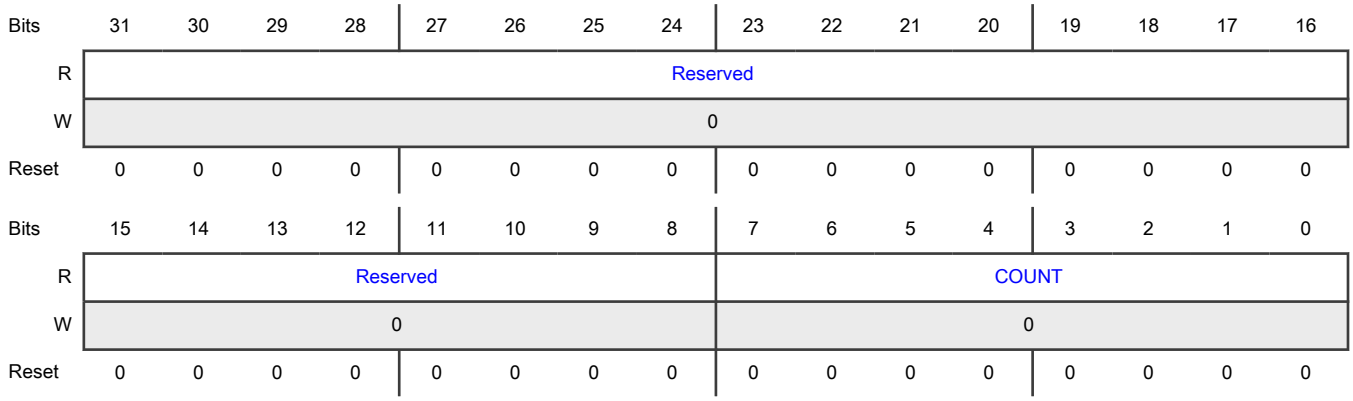
Function

Each 32-bit ERM Memory *n* Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory*n*, where*n* denotes the memory channel.

NOTE

Non-correctable errors are considered a serious fault, so the ERM does not provide any mechanism to count non-correctable errors. Only correctable errors are counted.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT value will stop when it reaches maximum value FFh and will not wrap even if additional errors occur. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing a non-zero value has no effect.

Chapter 10
Signal Multiplexing

10.1 Introduction

Pins have several functions available via signal multiplexing to optimize functionality in small packages. This chapter illustrates which device's signals are multiplexed on which external pin.

See the attached pinout spreadsheet to know how the pins available on this device are configured. The attached spreadsheet also provides the available device packages and pinout diagrams.

10.1.1 Signal multiplexing constraints

- A given module signal must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
- To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.

10.2 Pinout

See the attached pinout spreadsheet for the pinouts and ballmaps for each package offered on this device.

10.3 Pin Assignments

Table 63. Pin Assignments

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
P1_8	A1	1	B2	2	1	**ALT0** - P1_8 **ALT1** - FREQME_CLK_IN0 **ALT2** - LPUART1_RXD **ALT3** - LPI2C2_SDA **ALT4** - CT_INP8 **ALT5** - CT0_MAT2 **ALT6** - FLEXIO0_D16 **ALT10** - I3C0_SDA	**IO Supply** - VDD **Pad type** - HD+I3C **Default** - DIS	**VDD SYS** - WUU0_IN10
P1_9	B1	2	C2	3	2	**ALT0** - P1_9 **ALT1** - FREQME_CLK_IN1 **ALT2** - LPUART1_TXD **ALT3** - LPI2C2_SCL **ALT4** - CT_INP9 **ALT5** - CT0_MAT3 **ALT6** - FLEXIO0_D17	**IO Supply** - VDD **Pad type** - HD **Default** - DIS	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT10 - I3C0_SCL		
P1_10	C3	3	D2	4	3	**ALT0** - P1_10 **ALT2** - LPUART1_RTS_B **ALT3** - LPI2C2_SDAS **ALT4** - CT2_MAT0 **ALT6** - FLEXIO0_D18 **ALT11** - CAN0_TXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A8
P1_11	D3	4	D1	5	4	**ALT0** - P1_11 **ALT1** - TRIG_OUT2 **ALT2** - LPUART1_CTS_B **ALT3** - LPI2C2_SCLS **ALT4** - CT2_MAT1 **ALT6** - FLEXIO0_D19 **ALT10** - I3C0_PUR **ALT11** - CAN0_RXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A9 **VDD SYS** - WUU0_IN11
P1_12	D2	5	--	6	--	**ALT0** - P1_12 **ALT2** - LPI2C1_SDA **ALT3** - LPUART2_RXD **ALT4** - CT2_MAT2 **ALT6** - FLEXIO0_D20 **ALT11** - CAN0_RXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A10 **VDD SYS** - WUU0_IN12
P1_13	D1	6	--	7	--	**ALT0** - P1_13 **ALT1** - TRIG_IN3 **ALT2** - LPI2C1_SCL **ALT3** - LPUART2_TXD **ALT4** - CT2_MAT3 **ALT6** - FLEXIO0_D21 **ALT11** - CAN0_TXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A11
P1_14	E4	7	--	--	--	**ALT0** - P1_14 **ALT2** - LPI2C1_SCLS **ALT3** - LPUART2_RTS_B **ALT4** - CT_INP10	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A12

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT5 - CT3_MAT0 **ALT6** - FLEXIO0_D22		
P1_15	F4	8	--	--	--	**ALT0** - P1_15 **ALT2** - LPI2C1_SDAS **ALT3** - LPUART2_CTS_B **ALT4** - CT_INP11 **ALT5** - CT3_MAT1 **ALT6** - FLEXIO0_D23	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A13
P1_29	F3	9	E1	8	5	**ALT0** - P1_29 **ALT1** - RESET_B **ALT2** - SPC_LPREQ	**IO Supply** - VDD **Pad type** - RST **Default** - ALT1	**VDD SYS** - RESET_B
P1_30	F1	10	E2	9	6	**ALT0** - P1_30 **ALT1** - TRIG_OUT3 **ALT3** - LPI2C0_SDA **ALT4** - CT_INP16 **ALT6** - FLEXIO0_D30 **ALT10** - I3C0_SDA	**IO Supply** - VDD **Pad type** - HD+I3C **Default** - DIS	**ANALOG** - XTAL48M
P1_31	F2	11	F2	10	7	**ALT0** - P1_31 **ALT1** - TRIG_IN4 **ALT3** - LPI2C0_SCL **ALT4** - CT_INP17 **ALT6** - FLEXIO0_D31 **ALT10** - I3C0_SCL	**IO Supply** - VDD **Pad type** - HD **Default** - DIS	**ANALOG** - EXTAL48M
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	12	D4,D5,D6,E6, F5,F6	11	--		**IO Supply** - VDD	
VREFL	H7	12	E4	11	--		**IO Supply** - VDD	
VREFH	H6	13	F4	12	8		**IO Supply** - VDD **Pad type** - ANA	
VDD_ANA	J5	14	E3	12	8		**IO Supply** - VDD	
VDD	E5,F6,H8,J9	15	G3	13	9		**IO Supply** - VDD	
P4_2	H1	16	--	--	--	**ALT0** - P4_2 **ALT1** - CLKOUT **ALT2** - LPI2C2_SDAS	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**VDD SYS** - WUU0_IN16

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT3 - LPUART3_RXD **ALT4** - CT4_MAT0 **ALT5** - PWM0_A2 **ALT6** - FLEXIO0_D10		
P4_3	H3	17	--	--	--	**ALT0** - P4_3 **ALT2** - LPI2C2_SCL **ALT3** - LPUART4_TXD **ALT4** - CT4_MAT1 **ALT5** - PWM0_B2 **ALT6** - FLEXIO0_D11	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	
P4_4	J3	18	--	--	--	**ALT0** - P4_4 **ALT2** - LPI2C2_SDA **ALT3** - LPUART4_RXD **ALT4** - CT4_MAT2 **ALT5** - PWM0_A1 **ALT6** - FLEXIO0_D12	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**VDD SYS** - WUU0_IN17
P4_5	K3	19	--	--	--	**ALT0** - P4_5 **ALT1** - TRIG_OUT3 **ALT2** - LPI2C2_SCLS **ALT3** - LPUART3_TXD **ALT4** - CT4_MAT3 **ALT5** - PWM0_B1 **ALT6** - FLEXIO0_D13	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	
P4_6	K1	20	--	--	--	**ALT0** - P4_6 **ALT1** - TRIG_IN4 **ALT2** - LPI2C2_HREQ **ALT3** - LPUART3_CTS_B **ALT4** - CT_INP6 **ALT5** - PWM0_A0 **ALT6** - FLEXIO0_D14	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	
P4_7	K2	21	--	--	--	**ALT0** - P4_7 **ALT1** - TRIG_IN5	**IO Supply** - VDD **Pad type** - MED	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT3 - LPUART3_RTS_B **ALT4** - CT_INP7 **ALT5** - PWM0_B0 **ALT6** - FLEXIO0_D15	**Default** - DIS	
P2_0	L2	22	G2	14	10	**ALT0** - P2_0 **ALT1** - TRIG_IN6 **ALT2** - LPUART0_RXD **ALT3** - LPUART4_CTS_B **ALT4** - CT_INP16 **ALT5** - CT2_MAT0 **ALT6** - FLEXIO0_D8	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A0 **VDD SYS** - WUU0_IN18
P2_1	M2	23	G1	15	11	**ALT0** - P2_1 **ALT1** - TRIG_IN7 **ALT2** - LPUART0_TXD **ALT3** - LPUART4_RTS_B **ALT4** - CT_INP17 **ALT5** - CT2_MAT1 **ALT6** - FLEXIO0_D9	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A1
P2_2	M1	24	H1	16	12	**ALT0** - P2_2 **ALT1** - TRIG_IN6 **ALT2** - LPUART0_RTS_B **ALT3** - LPUART2_TXD **ALT4** - CT_INP12 **ALT5** - CT2_MAT2 **ALT6** - FLEXIO0_D10	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A4/ CMP0_IN0/ DAC0_OUT
P2_3	N1	25	J1	17	13	**ALT0** - P2_3 **ALT1** - TRIG_IN7 **ALT2** - LPUART0_CTS_B **ALT3** - LPUART2_RXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - CMP1_IN0/ADC1_A4 **VDD SYS** - WUU0_IN19

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT4 - CT_INP13 **ALT5** - CT2_MAT3 **ALT6** - FLEXIO0_D11		
P2_4	N2	26	J2	18	--	**ALT0** - P2_4 **ALT3** - LPUART2_CTS_B **ALT4** - CT_INP14 **ALT5** - CT1_MAT0 **ALT6** - FLEXIO0_D12	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A0
P2_5	L3	27	H2	19	--	**ALT0** - P2_5 **ALT3** - LPUART2_RTS_B **ALT4** - CT_INP15 **ALT5** - CT1_MAT1 **ALT6** - FLEXIO0_D13	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A1
P2_6	M4	28	H3	20	14	**ALT0** - P2_6 **ALT1** - TRIG_OUT4 **ALT2** - LPSPI1_PCS1 **ALT3** - LPUART4_RXD **ALT4** - CT_INP18 **ALT5** - CT1_MAT2 **ALT6** - FLEXIO0_D14	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A3
P2_7	N4	29	H4	21	15	**ALT0** - P2_7 **ALT1** - TRIG_IN5 **ALT3** - LPUART4_TXD **ALT4** - CT_INP19 **ALT5** - CT1_MAT3 **ALT6** - FLEXIO0_D15	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - VREFI/ ADC0_A7/ADC1_A7
P2_10	L4	30	--	--	--	**ALT0** - P2_10 **ALT1** - TRIG_OUT5 **ALT3** - LPUART2_TXD **ALT4** - CT3_MAT2 **ALT6** - FLEXIO0_D18	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	
P2_11	K5	31	--	--	--	**ALT0** - P2_11	**IO Supply** - VDD	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT1 - TRIG_IN4 **ALT3** - LPUART2_RXD **ALT4** - CT3_MAT3 **ALT6** - FLEXIO0_D19	**Pad type** - SLOW **Default** - DIS	
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	32	D4,D5,D6,E6, F5,F6	--	--		**IO Supply** - VDD	
VDD	E5,F6,H8,J9	33	G3	--	--		**IO Supply** - VDD	
P2_12	K6	34	J4	22	16	**ALT0** - P2_12 **ALT1** - USB0_VBUS_DET **ALT2** - LPSPI1_SCK **ALT3** - LPUART1_RXD **ALT4** - CT4_MAT0 **ALT5** - CT0_MAT0 **ALT6** - FLEXIO0_D20 **ALT11** - CAN0_RXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ISP** - USB0_VBUS_DET **ANALOG** - ADC0_A5/ OPAMP0_INP0 **VDD SYS** - WUU0_IN20
P2_13	L6	35	J5	23	17	**ALT0** - P2_13 **ALT1** - TRIG_IN8 **ALT2** - LPSPI1_SDO **ALT3** - LPUART1_TXD **ALT4** - CT4_MAT1 **ALT5** - CT0_MAT1 **ALT6** - FLEXIO0_D21 **ALT11** - CAN0_TXD	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A5/ OPAMP0_INN
P2_15	N6	36	H5	24	18	**ALT0** - P2_15 **ALT1** - TRIG_OUT4 **ALT2** - LPSPI1_SDI **ALT3** - LPUART1_RTS_B **ALT4** - CT4_MAT3 **ALT5** - CT0_MAT2 **ALT6** - FLEXIO0_D23	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - OPAMP0_OUT/ ADC0_A2
P2_16	M6	37	--	--	--	**ALT0** - P2_16	**IO Supply** - VDD	**ANALOG** - ADC0_A6

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT2 - LPSPI1_SDI **ALT3** - LPUART1_RTS_B **ALT4** - CT3_MAT0 **ALT5** - CT0_MAT2 **ALT6** - FLEXIO0_D24	**Pad type** - SLOW **Default** - DIS	
P2_17	M7	38	--	--	--	**ALT0** - P2_17 **ALT1** - TRIG_IN9 **ALT2** - LPSPI1_PCS0 **ALT3** - LPUART1_CTS_B **ALT4** - CT3_MAT1 **ALT5** - CT0_MAT3 **ALT6** - FLEXIO0_D25	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A6
P2_19	M8	39	--	--	--	**ALT0** - P2_19 **ALT1** - TRIG_OUT5 **ALT4** - CT3_MAT3 **ALT6** - FLEXIO0_D27	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A2
P2_20	N8	40	--	--	--	**ALT0** - P2_20 **ALT1** - TRIG_IN8 **ALT2** - LPSPI1_PCS2 **ALT4** - CT2_MAT0 **ALT6** - FLEXIO0_D28	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	
P2_21	L8	41	--	--	--	**ALT0** - P2_21 **ALT1** - TRIG_IN9 **ALT2** - LPSPI1_PCS3 **ALT4** - CT2_MAT1 **ALT6** - FLEXIO0_D29	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	
P2_23	L9	42	--	--	--	**ALT0** - P2_23 **ALT1** - TRIG_OUT5 **ALT4** - CT2_MAT3 **ALT6** - FLEXIO0_D31	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	
VDD_USB	M11	43	H6	25	19		**IO Supply** - VDD_USB	
USB0_DM	M10	44	H7	26	20		**IO Supply** - VDD_USB **Pad type** - ANA	**ISP** - USB0_DM **ANALOG** - USB0_DM

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
								VDD SYS - WUU0_IN28
USB0_DP	N10	45	J7	27	21		**IO Supply** - VDD_USB **Pad type** - ANA	**ISP** - USB0_DP **ANALOG** - USB0_DP **VDD SYS** - WUU0_IN29
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	46	D4,D5,D6,E6, F5,F6	28	--		**IO Supply** - VDD_P3	
VDD_P3	E9,F8	47	G5	29	22		**IO Supply** - VDD_P3	
P3_31	N12	48	J8	30	23	**ALT0** - P3_31 **ALT1** - TRIG_IN10 **ALT2** - LPI2C3_SDAS **ALT3** - LPUART4_CTS_B **ALT4** - CT0_MAT3 **ALT6** - FLEXIO0_D31	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A20 **VDD SYS** - LPTMR0_ALT2
P3_30	N13	49	J9	31	24	**ALT0** - P3_30 **ALT1** - TRIG_OUT6 **ALT2** - LPI2C3_SCLS **ALT3** - LPUART4_RTS_B **ALT4** - CT0_MAT2 **ALT6** - FLEXIO0_D30	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC1_A21
P3_29	M13	50	H9	32	25	**ALT0** - P3_29 **ALT1** - ISPMODE_N **ALT2** - LPI2C3_HREQ **ALT4** - CT_INP3 **ALT5** - CT3_MAT3 **ALT6** - FLEXIO0_D29	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - ALT1	**ISP** - ISPMODE_N **ANALOG** - ADC1_A22 **VDD SYS** - WUU0_IN27
P3_28	L11	51	G7	33	26	**ALT0** - P3_28 **ALT1** - TRIG_IN11 **ALT2** - LPI2C3_SDA **ALT3** - LPUART4_RXD **ALT4** - CT_INP12 **ALT5** - CT3_MAT2	**IO Supply** - VDD_P3 **Pad type** - 5VTOL **Default** - DIS	**VDD SYS** - WUU0_IN26

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT6 - FLEXIO0_D28		
P3_27	K11	52	E7	34	27	**ALT0** - P3_27 **ALT1** - TRIG_OUT7 **ALT2** - LPI2C3_SCL **ALT3** - LPUART4_TXD **ALT4** - CT_INP13 **ALT5** - CT3_MAT1 **ALT6** - FLEXIO0_D27	**IO Supply** - VDD_P3 **Pad type** - 5VTOL **Default** - DIS	**VDD SYS** - WUU0_IN30
P3_22	K13	53	--	--	--	**ALT0** - P3_22 **ALT3** - LPUART1_RTS_B **ALT4** - CT_INP10 **ALT6** - FLEXIO0_D30 **ALT7** - PWM1_X2	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_21	J10	54	--	--	--	**ALT0** - P3_21 **ALT1** - TRIG_OUT1 **ALT2** - LPI2C3_SCL **ALT3** - LPUART1_TXD **ALT4** - CT2_MAT3 **ALT6** - FLEXIO0_D29	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_20	H10	55	--	--	--	**ALT0** - P3_20 **ALT1** - TRIG_OUT0 **ALT2** - LPI2C3_SDA **ALT3** - LPUART1_RXD **ALT4** - CT2_MAT2 **ALT5** - PWM0_X2 **ALT6** - FLEXIO0_D28	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_19	G11	56	--	--	--	**ALT0** - P3_19 **ALT2** - LPUART4_TXD **ALT4** - CT2_MAT1 **ALT5** - PWM0_X1 **ALT6** - FLEXIO0_D27 **ALT7** - PWM1_X1	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
P3_18	F10	57	--	--	--	**ALT0** - P3_18 **ALT2** - LPUART4_RXD **ALT4** - CT2_MAT0 **ALT5** - PWM0_X0 **ALT6** - FLEXIO0_D26 **ALT7** - PWM1_X0	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_17	H11	58	--	--	--	**ALT0** - P3_17 **ALT2** - LPUART4_CTS_B **ALT4** - CT_INP9 **ALT6** - FLEXIO0_D25 **ALT7** - PWM1_B0	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_16	H13	59	--	--	--	**ALT0** - P3_16 **ALT2** - LPUART4_RTS_B **ALT4** - CT_INP8 **ALT6** - FLEXIO0_D24 **ALT7** - PWM1_A0	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_15	H12	60	H8	35	--	**ALT0** - P3_15 **ALT2** - LPUART2_TXD **ALT3** - LPUART3_RTS_B **ALT4** - CT_INP7 **ALT6** - FLEXIO0_D23 **ALT7** - PWM1_B1	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
P3_14	G12	61	G8	36	28	**ALT0** - P3_14 **ALT2** - LPUART2_RXD **ALT3** - LPUART3_CTS_B **ALT4** - CT_INP6 **ALT5** - PWM0_X2 **ALT6** - FLEXIO0_D22 **ALT7** - PWM1_A1	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	**VDD SYS** - WUU0_IN25
P3_13	F12	62	F8	37	29	**ALT0** - P3_13	**IO Supply** - VDD_P3	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT2 - LPUART2_CTS_B **ALT3** - LPUART3_RXD **ALT4** - CT1_MAT3 **ALT5** - PWM0_X1 **ALT6** - FLEXIO0_D21 **ALT7** - PWM1_B2	**Pad type** - SLOW **Default** - DIS	
P3_12	F13	63	F9	38	30	**ALT0** - P3_12 **ALT2** - LPUART2_RTS_B **ALT3** - LPUART3_TXD **ALT4** - CT1_MAT2 **ALT5** - PWM0_X0 **ALT6** - FLEXIO0_D20 **ALT7** - PWM1_A2	**IO Supply** - VDD_P3 **Pad type** - SLOW **Default** - DIS	
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	64	D4,D5,D6,E6, F5,F6	--	--		**IO Supply** - VDD_P3	
VDD_P3	E9,F8	65	G5	--	--		**IO Supply** - VDD_P3	
P3_11	F11	66	E9	39	31	**ALT0** - P3_11 **ALT1** - TRIG_IN6 **ALT2** - LPSPI1_PCS0 **ALT3** - LPUART1_CTS_B **ALT4** - CT1_MAT1 **ALT5** - PWM0_B2 **ALT6** - FLEXIO0_D19	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	**VDD SYS** - WUU0_IN24
P3_10	E11	67	E8	40	32	**ALT0** - P3_10 **ALT1** - TRIG_IN5 **ALT2** - LPSPI1_SCK **ALT3** - LPUART1_RTS_B **ALT4** - CT1_MAT0 **ALT5** - PWM0_A2 **ALT6** - FLEXIO0_D18	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
P3_9	D11	68	D8	41	33	**ALT0** - P3_9 **ALT1** - TRIG_IN4 **ALT2** - LPSPI1_SDI **ALT3** - LPUART1_TXD **ALT4** - CT_INP5 **ALT5** - PWM0_B1 **ALT6** - FLEXIO0_D17	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	
P3_8	D13	69	C8	42	34	**ALT0** - P3_8 **ALT1** - TRIG_IN3 **ALT2** - LPSPI1_SDO **ALT3** - LPUART1_RXD **ALT4** - CT_INP4 **ALT5** - PWM0_A1 **ALT6** - FLEXIO0_D16 **ALT12** - CLKOUT	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	**VDD SYS** - WUU0_IN23
P3_7	D12	70	C9	43	--	**ALT0** - P3_7 **ALT1** - TRIG_IN2 **ALT2** - LPSPI1_PCS2 **ALT3** - LPUART3_CTS_B **ALT4** - CT4_MAT3 **ALT5** - PWM0_B0 **ALT6** - FLEXIO0_D15 **ALT7** - PWM1_B0	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	
P3_6	C12	71	B9	44	--	**ALT0** - P3_6 **ALT1** - CLKOUT **ALT2** - LPSPI1_PCS3 **ALT3** - LPUART3_RTS_B **ALT4** - CT4_MAT2 **ALT5** - PWM0_A0 **ALT6** - FLEXIO0_D14 **ALT7** - PWM1_A0 **ALT12** - FREQME_CLK_OUT1	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
P3_2	B12	--	--	--	--	**ALT0** - P3_2 **ALT2** - LPSPI1_PCS1 **ALT4** - CT4_MAT0 **ALT6** - FLEXIO0_D10 **ALT7** - PWM1_X2	**IO Supply** - VDD_P3 **Pad type** - MED **Default** - DIS	
P3_1	B13	72	A9	45	35	**ALT0** - P3_1 **ALT1** - TRIG_IN1 **ALT3** - LPUART3_TXD **ALT4** - CT_INP17 **ALT5** - PWM0_B0 **ALT6** - FLEXIO0_D9 **ALT7** - PWM1_X1 **ALT12** - FREQME_CLK_OUT0	**IO Supply** - VDD_P3 **Pad type** - HD **Default** - DIS	
P3_0	A13	73	A8	46	36	**ALT0** - P3_0 **ALT1** - TRIG_IN0 **ALT3** - LPUART3_RXD **ALT4** - CT_INP16 **ALT5** - PWM0_A0 **ALT6** - FLEXIO0_D8 **ALT7** - PWM1_X0	**IO Supply** - VDD_P3 **Pad type** - HD **Default** - DIS	**VDD SYS** - WUU0_IN22
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	74	D4,D5,D6,E6, F5,F6	47	--		**IO Supply** - VDD_P3	
VDD_P3	E9,F8	75	G5	48	37		**IO Supply** - VDD_P3	
VDD	E5,F6,H8,J9	--	G3	48	37		**IO Supply** - VDD	
P0_0	A12	76	B8	49	38	**ALT0** - P0_0 **ALT1** - TMS/SWDIO **ALT2** - LPUART0_RTS_B **ALT3** - LPSPI0_PCS0 **ALT4** - CT_INP0 **ALT6** - FLEXIO0_D0	**IO Supply** - VDD **Pad type** - SLOW **Default** - ALT1	
P0_1	C11	77	B7	50	39	**ALT0** - P0_1 **ALT1** - TCLK/SWCLK	**IO Supply** - VDD **Pad type** - SLOW	

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT2 - LPUART0_CTS_B **ALT3** - LPSPI0_SDI **ALT4** - CT_INP1 **ALT6** - FLEXIO0_D1	**Default** - ALT1	
P0_2	C10	78	B6	51	40	**ALT0** - P0_2 **ALT1** - TDO/SWO **ALT2** - LPUART0_RXD **ALT3** - LPSPI0_SCK **ALT4** - CT0_MAT0 **ALT5** - UTICK_CAP0 **ALT6** - FLEXIO0_D2 **ALT10** - I3C0_PUR	**IO Supply** - VDD **Pad type** - SLOW **Default** - ALT1	**ISP** - UART_RXD
P0_3	B10	79	A6	52	41	**ALT0** - P0_3 **ALT1** - TDI **ALT2** - LPUART0_TXD **ALT3** - LPSPI0_SDO **ALT4** - CT0_MAT1 **ALT5** - UTICK_CAP1 **ALT6** - FLEXIO0_D3 **ALT8** - CMP0_OUT	**IO Supply** - VDD **Pad type** - SLOW **Default** - ALT1	**ISP** - UART_TXD **ANALOG** - CMP1_IN1/ADC0_A14
P0_6	A10	80	C7	53	42	**ALT0** - P0_6 **ALT1** - ISPMODE_N **ALT2** - LPI2C0_HREQ **ALT3** - LPSPI0_PCS1 **ALT4** - CT_INP2 **ALT6** - FLEXIO0_D6 **ALT8** - CMP1_OUT **ALT12** - CLKOUT	**IO Supply** - VDD **Pad type** - SLOW **Default** - ALT1	**ISP** - ISPMODE_N **ANALOG** - ADC0_A15
VDD	E5,F6,H8,J9	81	G3	--	--		**IO Supply** - VDD	
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	82	D4,D5,D6,E6, F5,F6	--	--		**IO Supply** - VDD	
P0_16	D9	83	C5	54	43	**ALT0** - P0_16 **ALT2** - LPI2C0_SDA	**IO Supply** - VDD **Pad type** - HD+I3C	**VDD SYS** - WUU0_IN2

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT3 - LPSPI0_PCS2 **ALT4** - CT0_MAT0 **ALT5** - UTICK_CAP2 **ALT6** - FLEXIO0_D0 **ALT10** - I3C0_SDA	**Default** - DIS	
P0_17	D8	84	C3	55	44	**ALT0** - P0_17 **ALT2** - LPI2C0_SCL **ALT3** - LPSPI0_PCS3 **ALT4** - CT0_MAT1 **ALT5** - UTICK_CAP3 **ALT6** - FLEXIO0_D1 **ALT10** - I3C0_SCL	**IO Supply** - VDD **Pad type** - HD **Default** - DIS	
P0_18	C8	85	--	--	--	**ALT0** - P0_18 **ALT2** - LPI2C0_SCLS **ALT4** - CT0_MAT2 **ALT6** - FLEXIO0_D2 **ALT8** - CMP0_OUT	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A8
P0_19	A8	86	--	--	--	**ALT0** - P0_19 **ALT2** - LPI2C0_SDAS **ALT4** - CT0_MAT3 **ALT6** - FLEXIO0_D3 **ALT8** - CMP1_OUT	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A9
P0_20	B8	87	--	--	--	**ALT0** - P0_20 **ALT3** - LPUART0_RXD **ALT4** - CT_INP0 **ALT6** - FLEXIO0_D4	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A10
P0_21	B7	88	--	--	--	**ALT0** - P0_21 **ALT3** - LPUART0_TXD **ALT4** - CT_INP1 **ALT6** - FLEXIO0_D5	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A11
P0_22	B6	89	--	--	--	**ALT0** - P0_22 **ALT3** - LPUART0_RTS_B **ALT4** - CT_INP2	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A12

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT5 - CT0_MAT0 **ALT6** - FLEXIO0_D6		
P0_23	A6	90	--	--	--	**ALT0** - P0_23 **ALT3** - LPUART0_CTS_B **ALT4** - CT_INP3 **ALT5** - CT0_MAT1 **ALT6** - FLEXIO0_D7	**IO Supply** - VDD **Pad type** - SLOW **Default** - DIS	**ANALOG** - ADC0_A13
P1_0	C6	91	A5	56	45	**ALT0** - P1_0 **ALT1** - TRIG_IN0 **ALT2** - LPSPI0_SDO **ALT3** - LPI2C1_SDA **ALT4** - CT_INP4 **ALT5** - CT0_MAT2 **ALT6** - FLEXIO0_D8	**IO Supply** - VDD **Pad type** - MED+I2C_FILT **Default** - DIS	**ANALOG** - ADC0_A16/CMP0_IN3 **VDD SYS** - WUU0_IN6/ LPTMR0_ALT3
P1_1	C5	92	B5	57	46	**ALT0** - P1_1 **ALT1** - TRIG_IN1 **ALT2** - LPSPI0_SCK **ALT3** - LPI2C1_SCL **ALT4** - CT_INP5 **ALT5** - CT0_MAT3 **ALT6** - FLEXIO0_D9	**IO Supply** - VDD **Pad type** - MED+I2C_FILT **Default** - DIS	**ANALOG** - ADC0_A17/CMP1_IN3
P1_2	C4	93	B4	58	47	**ALT0** - P1_2 **ALT1** - TRIG_OUT0 **ALT2** - LPSPI0_SDI **ALT3** - LPI2C1_SDAS **ALT4** - CT1_MAT0 **ALT5** - CT_INP0 **ALT6** - FLEXIO0_D10 **ALT11** - CAN0_TXD	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**ANALOG** - ADC0_A18
P1_3	A4	94	B3	59	48	**ALT0** - P1_3 **ALT1** - TRIG_OUT1 **ALT2** - LPSPI0_PCS0 **ALT3** - LPI2C1_SCLS **ALT4** - CT1_MAT1 **ALT5** - CT_INP1	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**ANALOG** - ADC0_A19/CMP0_IN1 **VDD SYS** - WUU0_IN7

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT6 - FLEXIO0_D11 **ALT11** - CAN0_RXD		
VDD	E5,F6,H8,J9	95	G3	60	--		**IO Supply** - VDD	
VSS	D4,D10,F7,G2, G6,G8,K4,K10 ,M12	96	D4,D5,D6,E6, F5,F6	61	--		**IO Supply** - VDD	
P1_4	B4	97	A3	62	--	**ALT0** - P1_4 **ALT1** - FREQME_CLK_IN0 **ALT2** - LPSPI0_PCS3 **ALT3** - LPUART2_RXD **ALT4** - CT1_MAT2 **ALT6** - FLEXIO0_D12	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**ANALOG** - ADC0_A20/CMP0_IN2 **VDD SYS** - WUU0_IN8
P1_5	B3	98	A2	63	--	**ALT0** - P1_5 **ALT1** - FREQME_CLK_IN1 **ALT2** - LPSPI0_PCS2 **ALT3** - LPUART2_TXD **ALT4** - CT1_MAT3 **ALT6** - FLEXIO0_D13	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**ANALOG** - ADC0_A21/CMP1_IN2
P1_6	B2	99	A1	64	--	**ALT0** - P1_6 **ALT1** - TRIG_IN2 **ALT2** - LPSPI0_PCS1 **ALT3** - LPUART2_RTS_B **ALT4** - CT_INP6 **ALT5** - CT4_MAT0 **ALT6** - FLEXIO0_D14 **ALT11** - CAN0_TXD	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**ANALOG** - ADC0_A22
P1_7	A2	100	B1	1	--	**ALT0** - P1_7 **ALT1** - TRIG_OUT2 **ALT3** - LPUART2_CTS_B **ALT4** - CT_INP7 **ALT5** - CT4_MAT1	**IO Supply** - VDD **Pad type** - MED **Default** - DIS	**ANALOG** - ADC0_A23 **VDD SYS** - WUU0_IN9

Table continues on the next page...

Table 63. Pin Assignments (continued)

Pin Name	MCXA14xA15 x BGA112	MCXA14xA15 x LQFP100	MCXA14xA15 x BGA64	MCXA14xA15 x LQFP64	MCXA14xA15 x QFN48	Pinmux Assignment	Pad Settings	Alternate Functions
						ALT6 - FLEXIO0_D15 **ALT11** - CAN0_RXD		

Note:

1. +I3C in Pad Type represents that strong pull up resistor is implemented on the pin. PV bit is implemented in Pin Control register of the pin.

2. +I2C_FILT in Pad Type represents that I2C filter is implemented on the pin. PFE bit is implemented in Pin Control register of the pin.

3. HD in Pad Type represents that the pin can support up to 20mA drive strength. I2C filter is implemented on the pin. PFE bit is implemented in Pin Control register of the pin.

4. 5VTOL in Pad Type represents that the pin is 5V tolerant

5. DIS in default column represents that the pin's input buffer is disabled by default

6. RST pads support passive filter and 1M ohm pull resistor. PFE and PV bits are implemented in Pin Control register of the pin.

7. PE, PS, SRE, ODE and DSE are supported in Pin Control register of all types of IO.

8. 5VTol and HD pads support two DSE bits in Pin Control register of the pin.

9. PWM1, OpAMP and DAC are not available in MCXA145 and MCXA146.

10. In 100LQFP, 112BGA and 64BGA, the ISPMODE_n pin is on P0_6. In 64LQFP and smaller pin count package, ISPMODE_n pin is on P3_29.

11. SLOW in Pad Type represents the IO supports 25MHz. MED in Pad Type represents the IO supports 50MHz.

Chapter 11
Port Control (PORT)

11.1 Chip-specific PORT information

Table 64. Reference links to related information

Topic	Related module	Reference
Full description	PORT	PORT
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

11.1.1 Module instances

This device has five instances of the port module, PORT0, PORT1, PORT2, PORT3 and PORT4.

11.1.2 Normal I/O pin structure

The following diagram shows the structure of normal I/O pin.

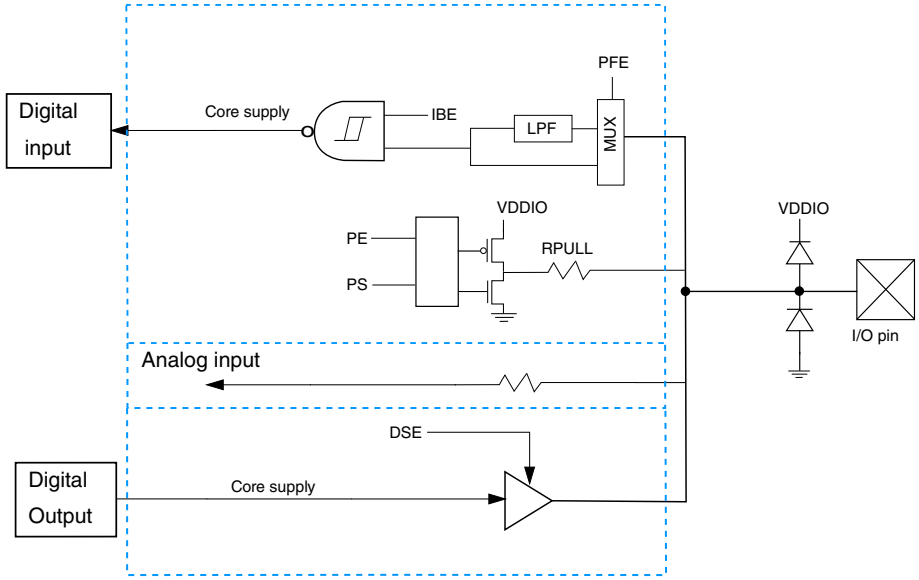


Figure 14. Basic structure of an I/O port

11.2 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.

11.2.1 Block diagram

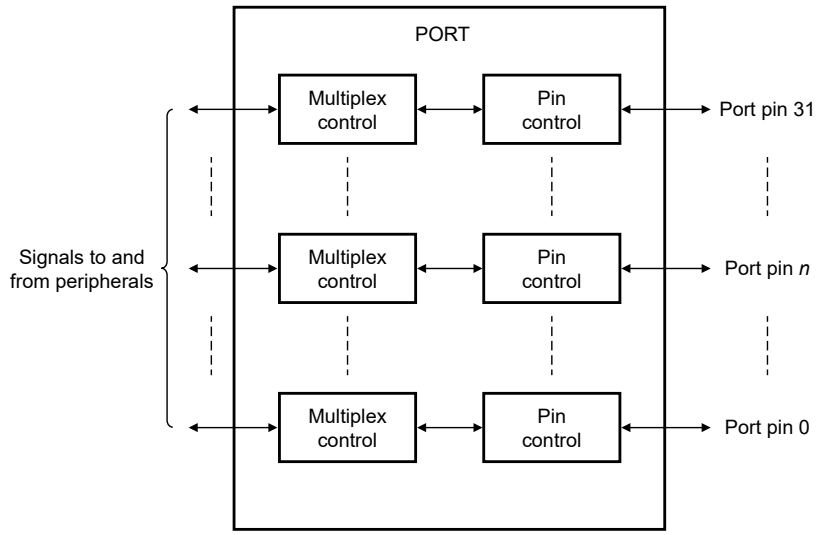


Figure 15. Block diagram

11.2.2 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
- Digital input inversion to optionally invert the digital input
- Digital PCR*n*[IBE] fields to configure between analog or disabled functions and digital functions
- Individual PCR*n*[MUX] fields supporting GPIO and up to 13 chip-specific digital functions

11.3 Functional description

11.3.1 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
- Digital input inversion
- 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.

11.3.2 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.

11.3.3 Clocking

This module has no clocking considerations.

11.3.4 Calibration

The combination of Calibration 0 (CALIB0) and Calibration 1 (CALIB1) controls the drive strength of a pin for the port and PCR*n*[DSE] for the pin.

Calibration 0 (CALIB0) and Calibration 1 (CALIB1) represent two driver configurations. If PCR*n*[DSE] = 0, the configuration in Calibration 0 (CALIB0) is used. Likewise, if PCR*n*[DSE] = 1, the configuration in Calibration 1 (CALIB1) is used.

The pulldown and pullup drivers have eight segments. The value of NCAL[5:3] represents the number of pulldown segments that are enabled when driving low, and the value of PCAL[5:3] represents the number of pullup segments that turn on when driving high. Writing 000b to these fields causes one segment to turn on, while writing 111b causes all eight segments to turn on. Upon reset, Calibration 0 (CALIB0) becomes 1 to create approximately 50 Ω driver at 3.3 V and 25 °C (3 is a typical value) and Calibration 1 (CALIB1) becomes 1 to create approximately 50 Ω driver at 1.8 V and 25 °C (7 is the typical value).

The three LSBs of CALIB0[NCAL], CALIB1[NCAL], CALIB0[PCAL], and CALIB1[PCAL] do not affect the driver's strength. Instead, they are included to improve precision when calibrating to a different impedance value. This can be done by using the following formula:

Calibration = (G × (CAL0 + 4) – 1024) >> 8, where G is a binary code representing the desired conductance and CAL0 is one of the 6-bit fields in Calibration 0 (CALIB0) and Calibration 1 (CALIB1). The value of G scales linearly with the intended conductance; G = 0 corresponds to 0 S (an open circuit), and G = 256 corresponds to 0.02 S (50 Ω). For example, the following operation configures the driver to be approximately 0.03 S (33 Ω) at 3.3 V when PCR*n*[DSE] = 1:

The value of CALIB1[NCAL] = (384 × (the value of CALIB0[NCAL] + 4) – 1024) >> 8;

The value of CALIB1[PCAL] = (384 × (the value of CALIB0[PCAL] + 4) – 1024) >> 8;

Do not overflow or underflow NCAL[5:3] or PCAL[5:3]; otherwise, this operation may have an unintended result. You can configure the drive strength for 50 Ω at 2.5 V and 25 °C by averaging the initial NCAL codes in Calibration 0 (CALIB0) and Calibration 1 (CALIB1), and averaging the initial PCAL codes in Calibration 0 (CALIB0) and Calibration 1 (CALIB1). However, the 2.5 V

configuration is not as accurate or precise as the original 1.8 V and 3.3 V configuration because the drive strength is not perfectly linear with supply voltage.

NOTE

This calibration feature may not apply to all PORT pins. See the chip-specific PORT information for pins that support calibration.

11.4 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.

11.5 Application information

11.6 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.

11.6.1 PORT register descriptions

11.6.1.1 PORT memory map

PORT0 base address: 400B_C000h

PORT1 base address: 400B_D000h

PORT2 base address: 400B_E000h

PORT3 base address: 400B_F000h

PORT4 base address: 400C_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0000h
10h	Global Pin Control Low (GPCLR)	32	RW	0000_0000h
14h	Global Pin Control High (GPCHR)	32	RW	0000_0000h
20h	Configuration (CONFIG)	32	RW	0000_0000h
60h	Calibration 0 (CALIB0)	32	RW	See section
64h	Calibration 1 (CALIB1)	32	RW	See section
80h	Pin Control 0 (PCR0)	32	RW	See section
84h	Pin Control 1 (PCR1)	32	RW	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
88h	Pin Control 2 (PCR2)	32	RW	See section
8Ch	Pin Control 3 (PCR3)	32	RW	See section
90h - 94h	Pin Control a (PCR4 - PCR5)	32	RW	0000_0000h
98h	Pin Control 6 (PCR6)	32	RW	See section
9Ch	Pin Control 7 (PCR7)	32	RW	0000_0000h
A0h	Pin Control 8 (PCR8)	32	RW	0000_0000h
A4h	Pin Control 9 (PCR9)	32	RW	0000_0000h
A8h - ACh	Pin Control a (PCR10 - PCR11)	32	RW	0000_0000h
B0h - BCh	Pin Control a (PCR12 - PCR15)	32	RW	See section
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	See section
D0h - DCh	Pin Control a (PCR20 - PCR23)	32	RW	See section
ECh - F0h	Pin Control a (PCR27 - PCR28)	32	RW	0000_0000h
F4h	Pin Control 29 (PCR29)	32	RW	See section
F8h	Pin Control 30 (PCR30)	32	RW	0000_0000h
FCh	Pin Control 31 (PCR31)	32	RW	0000_0000h

11.6.1.2 Version ID (VERID)

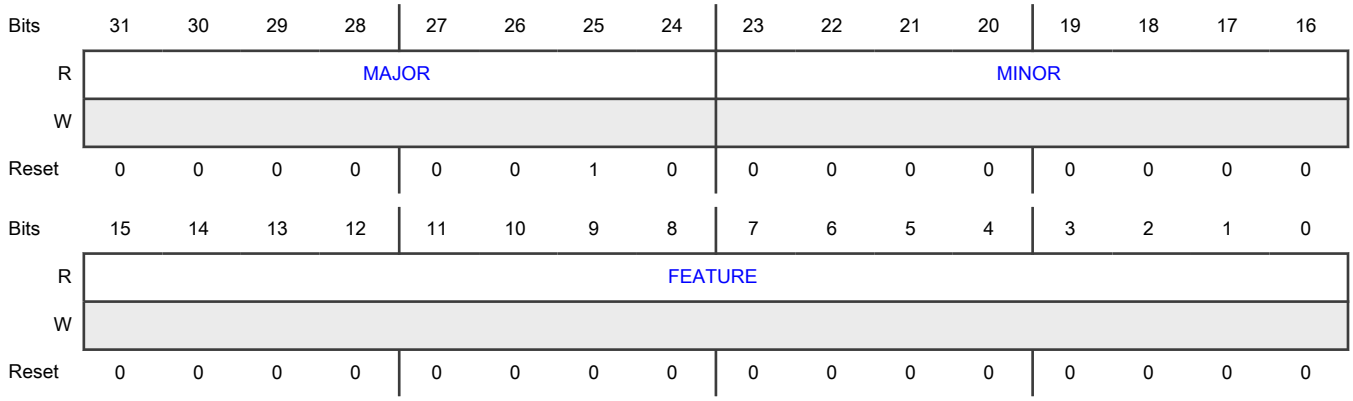
Offset

Register	Offset
VERID	0h

Function

Specifies the version number and feature number of the chip.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the specification.
23-16 MINOR	Minor Version Number 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

11.6.1.3 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.

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 1b - Updated
15-0 GPWD	Global Pin Write Data Is written to PCR*n*[15:0] if GPWE*n* = 1.

11.6.1.4 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.

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
15-0 GPWD	Global Pin Write Data Is written to PCR*n*[15:0] if GPWE*n* = 1.

11.6.1.5 Configuration (CONFIG)

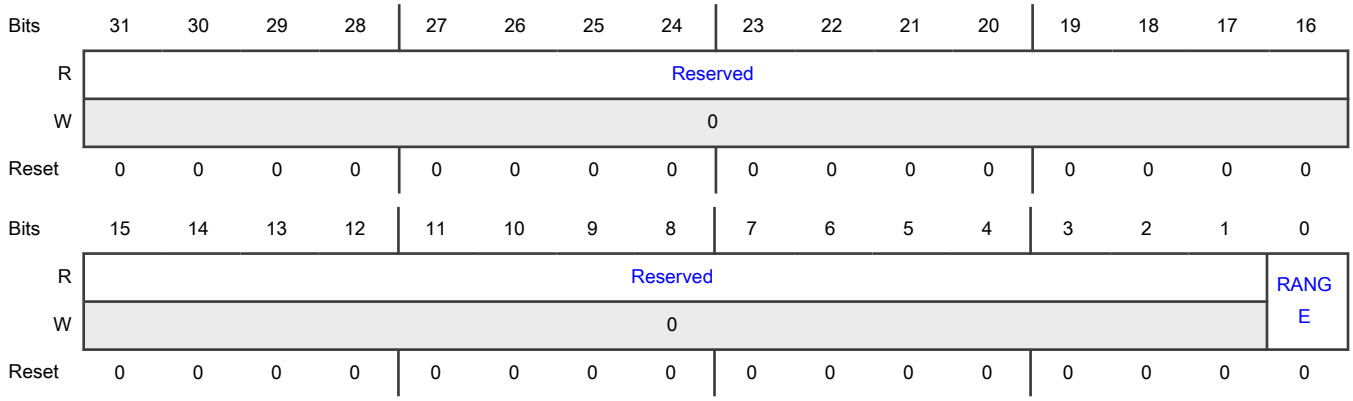
Offset

Register	Offset
CONFIG	20h

Function

Configures the port voltage range.

Diagram



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

11.6.1.6 Calibration 0 (CALIB0)

Offset

Register	Offset
CALIB0	60h

Function

Stores calibration values for the PMOS and NMOS output drivers when PCR*n*[DSE] = 0.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	CALIB0	—
PORT1	CALIB0	—
PORT2	—	CALIB0

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PORT3	CALIB0	—
PORT4	CALIB0	—

Diagram

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

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

1. Reset values are loaded out of IFR.

Fields

Field	Function
31-22 —	Reserved
21-16 PCAL	Calibration of PMOS Output Driver
15-6 —	Reserved
5-0 NCAL	Calibration of NMOS Output Driver

11.6.1.7 Calibration 1 (CALIB1)

Offset

Register	Offset
CALIB1	64h

Function

Stores calibration values for the PMOS and NMOS output drivers when PCR*n*[DSE] = 1.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	CALIB1	—
PORT1	CALIB1	—
PORT2	—	CALIB1
PORT3	CALIB1	—
PORT4	CALIB1	—

Diagram

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

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

1. Reset values are loaded out of IFR.

Fields

Field	Function
31-22 —	Reserved
21-16 PCAL	Calibration of PMOS Output Driver
15-6 —	Reserved
5-0 NCAL	Calibration of NMOS Output Driver

11.6.1.8 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.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	PCR0	—
PORT1	PCR0	—
PORT2	PCR0	—
PORT3	PCR0	—
PORT4	—	PCR0

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	INV	IBE	MUX				DSE1	DSE	ODE	PFE	SRE	0	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR0	PORT0: 0000_1143h PORT1–PORT3: 0000_0000h PORT4: Register not supported

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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: **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Field	Function

Instance	Field supported in	Field not supported in
PORT0	PCR0[10–8]	PCR0[11]
PORT1	PCR0	—
PORT2	PCR0[10–8]	PCR0[11]
PORT3	PCR0	—

NOTE

The descriptions of the field settings vary by module instance.

Instance	Field value and description
PORT0	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 0010b - Alternative 2 (chip-specific) 0011b - Alternative 3 (chip-specific)

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT3	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (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>PORT0</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORT1</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORT2</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORT3</td><td>PCR0</td><td>—</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.
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.</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>

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>PORT0</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORT1</td><td>PCR0</td><td>—</td></tr>
<tr><td>PORT2</td><td>—</td><td>PCR0</td></tr>
<tr><td>PORT3</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.
0b - Fast
1b - Slow</td></tr>
<tr><td>2
—</td><td>Reserved</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</td><td>Pull Select</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
PS	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

11.6.1.9 Pin Control 1 (PCR1)

Offset

Register	Offset
PCR1	84h

Function

Configures pin control features on each pin.

NOTE

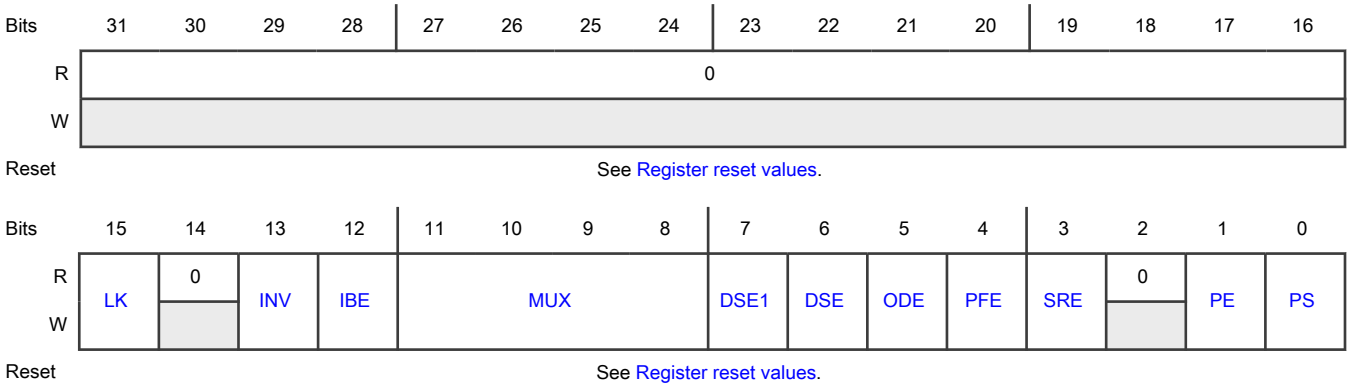
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	PCR1	—
PORT1	PCR1	—
PORT2	PCR1	—
PORT3	PCR1	—
PORT4	—	PCR1

Diagram



Register reset values

Register	Reset value
PCR1	PORT0: 0000_1102h PORT1–PORT3: 0000_0000h PORT4: Register not supported

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions.

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>11-8
MUX</td><td>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:
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>PORT0</td><td>PCR1[10–8]</td><td>PCR1[11]</td></tr>
<tr><td>PORT1</td><td>PCR1</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR1[10–8]</td><td>PCR1[11]</td></tr>
<tr><td>PORT3</td><td>PCR1</td><td>—</td></tr>
</table>
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>PORT0</td><td>000b - Alternative 0 (GPIO)
001b - Alternative 1 (chip-specific)
010b - Alternative 2 (chip-specific)
011b - Alternative 3 (chip-specific)
100b - Alternative 4 (chip-specific)
101b - Alternative 5 (chip-specific)
110b - Alternative 6 (chip-specific)
111b - Alternative 7 (chip-specific)</td></tr>
<tr><td>PORT2</td><td>000b - Alternative 0 (GPIO)
001b - Alternative 1 (chip-specific)</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
	PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT3	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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)

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></td><td>1010b - Alternative 10 (chip-specific)
1011b - Alternative 11 (chip-specific)
1100b - Alternative 12 (chip-specific)
1101b - Alternative 13 (chip-specific)</td></tr>
</table>
</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>PORT0</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORT1</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORT2</td><td>—</td><td>PCR1</td></tr>
<tr><td>PORT3</td><td>PCR1</td><td>—</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.
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>PORT0</td><td>—</td><td>PCR1</td></tr><tr><td>PORT1</td><td>PCR1</td><td>—</td></tr><tr><td>PORT2</td><td>—</td><td>PCR1</td></tr><tr><td>PORT3</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.
0b - Fast
1b - Slow</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
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

11.6.1.10 Pin Control 2 (PCR2)

Offset

Register	Offset
PCR2	88h

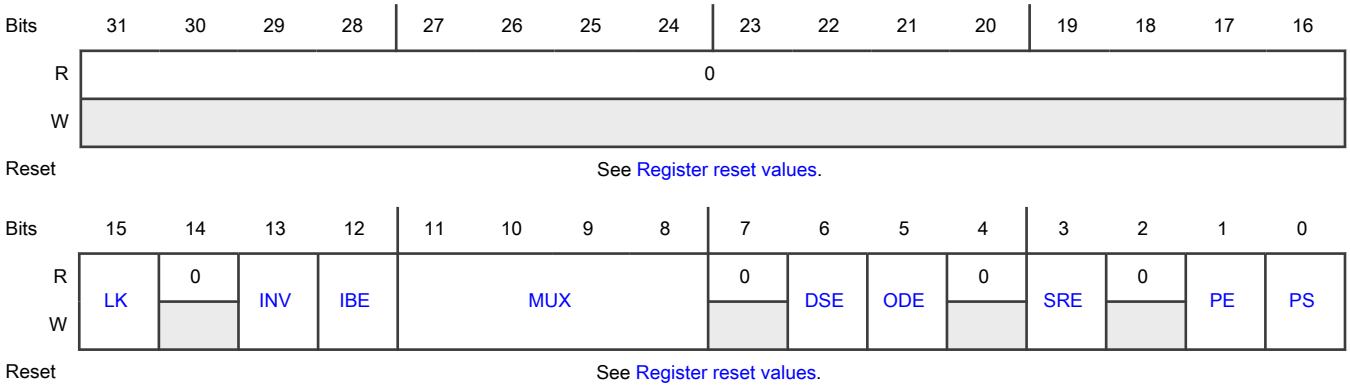
Function

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



Register reset values

Register	Reset value
PCR2	PORT0: 0000_0140h PORT1–PORT4: 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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables

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>1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT0</td><td>PCR2</td><td>—</td></tr>
<tr><td>PORT1</td><td>PCR2</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR2[10–8]</td><td>PCR2[11]</td></tr>
<tr><td>PORT3</td><td>PCR2[10–8]</td><td>PCR2[11]</td></tr>
<tr><td>PORT4</td><td>PCR2[10–8]</td><td>PCR2[11]</td></tr>
</table>
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>PORT0</td><td>0000b - Alternative 0 (GPIO)
0001b - Alternative 1 (chip-specific)
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)</td></tr>
</table>
</td></tr>
</table>

Field	Function	
	Instance	Field value and description
		1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
	PORT3	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific)

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></td><td>100b - Alternative 4 (chip-specific)
101b - Alternative 5 (chip-specific)
110b - Alternative 6 (chip-specific)
111b - Alternative 7 (chip-specific)</td></tr>
<tr><td>PORT4</td><td>000b - Alternative 0 (GPIO)
001b - Alternative 1 (chip-specific)
010b - Alternative 2 (chip-specific)
011b - Alternative 3 (chip-specific)
100b - Alternative 4 (chip-specific)
101b - Alternative 5 (chip-specific)
110b - Alternative 6 (chip-specific)
111b - Alternative 7 (chip-specific)</td></tr>
</table>
</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.
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</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

11.6.1.11 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.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	PCR3	—
PORT1	PCR3	—
PORT2	PCR3	—
PORT3	—	PCR3
PORT4	PCR3	—

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	INV	IBE	MUX				0	DSE	ODE	0	SRE	0	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR3	PORT0: 0000_1103h PORT1,PORT2: 0000_0000h PORT3: Register not supported PORT4: 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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>11-8
MUX</td><td>
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:
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>PORT0</td><td>PCR3</td><td>—</td></tr>
<tr><td>PORT1</td><td>PCR3</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR3</td><td>—</td></tr>
<tr><td>PORT4</td><td>PCR3[10–8]</td><td>PCR3[11]</td></tr>
</table>
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>PORT0</td><td>0000b - Alternative 0 (GPIO)
0001b - Alternative 1 (chip-specific)
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>
</table>
</td></tr>
</table>

Field	Function	
	Instance	Field value and description
		1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td>PORT4</td><td>000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific)</td></tr></table>
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. 0b - Disables 1b - Enables
4 —	Reserved
3 SRE	Slew Rate Enable Configures the slew rate feature, fast or slow, on each corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

11.6.1.12 Pin Control a (PCR4 - PCR5)

Offset

Register	Offset
PCR4	90h
PCR5	94h

Function

Configures pin control features on each pin.

NOTE

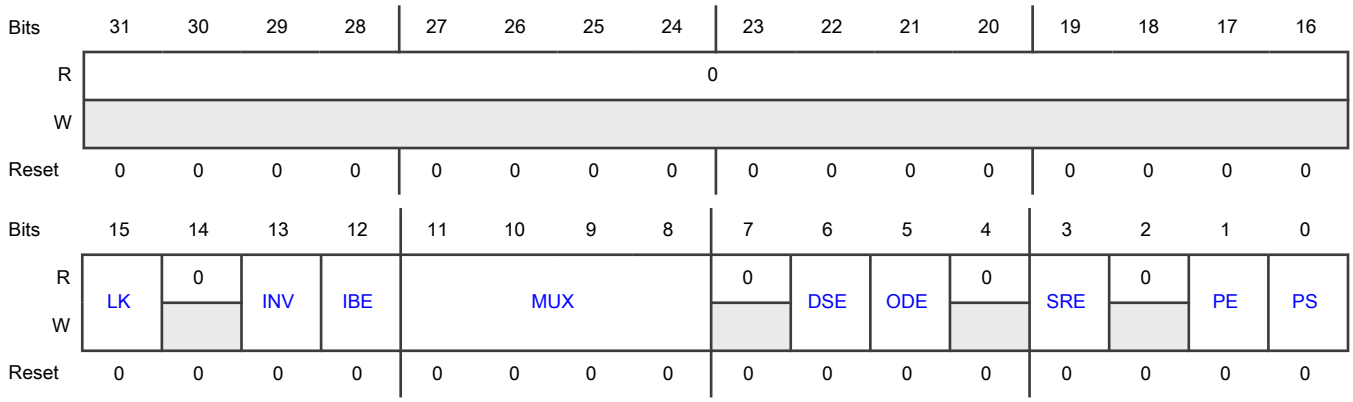
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR4–PCR5
PORT1	PCR4–PCR5	—
PORT2	PCR4–PCR5	—
PORT3	—	PCR4–PCR5
PORT4	PCR4–PCR5	—

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input.

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 - Does not invert
1b - Inverts</td></tr>
<tr><td>12
IBE</td><td>Input Buffer Enable
Enables digital input buffer. When disabled, the digital input is required for analog functions.
0b - Disables
1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT1</td><td>PCR4–PCR5</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR4–PCR5[10–8]</td><td>PCR4–PCR5[11]</td></tr>
<tr><td>PORT4</td><td>PCR4–PCR5[10–8]</td><td>PCR4–PCR5[11]</td></tr>
</table>
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>PORT1</td><td>0000b - Alternative 0 (GPIO)
0001b - Alternative 1 (chip-specific)
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)</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
	PORT4	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (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.	

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 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 —	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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 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

11.6.1.13 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.

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	INV	IBE	MUX				0	DSE	ODE	0	SRE	0	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR6	PORT0: 0000_1103h PORT1–PORT4: 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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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: **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
PORT0	PCR6	—
PORT1	PCR6	—
PORT2	PCR6[10–8]	PCR6[11]
PORT3	PCR6	—

Field	Function

Instance	Field supported in	Field not supported in
PORT4	PCR6[10–8]	PCR6[11]

NOTE

The descriptions of the field settings vary by module instance.

Instance	Field value and description
PORT0	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific)

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		1101b - Alternative 13 (chip-specific)
	PORT3	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
	PORT4	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific)

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td></td><td>111b - Alternative 7 (chip-specific)</td></tr></table>
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. 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	Pull Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
PE	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

11.6.1.14 Pin Control 7 (PCR7)

Offset

Register	Offset
PCR7	9Ch

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.

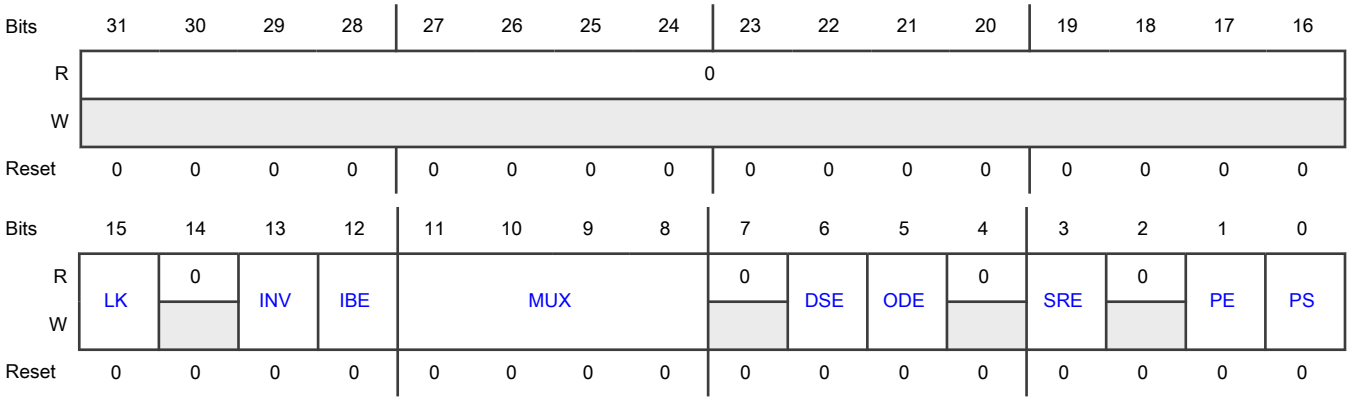
Instance	Register supported	Register not supported
PORT0	—	PCR7

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PORT1	PCR7	—
PORT2	PCR7	—
PORT3	PCR7	—
PORT4	PCR7	—

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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:

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
PORT1	PCR7	—
PORT2	PCR7	—
PORT3	PCR7	—
PORT4	PCR7[10–8]	PCR7[11]

NOTE

The descriptions of the field settings vary by module instance.

Instance	Field value and description
PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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)

Field	Function	
	Instance	Field value and description
		1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT3	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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)

Table continued from the previous page...

Field	Function
	<table><tr><th>Instance</th><th>Field value and description</th></tr><tr><td></td><td>1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)</td></tr><tr><td>PORT4</td><td>000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific)</td></tr></table>
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. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
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

11.6.1.15 Pin Control 8 (PCR8)

Offset

Register	Offset
PCR8	A0h

Function

Configures pin control features on each pin.

NOTE

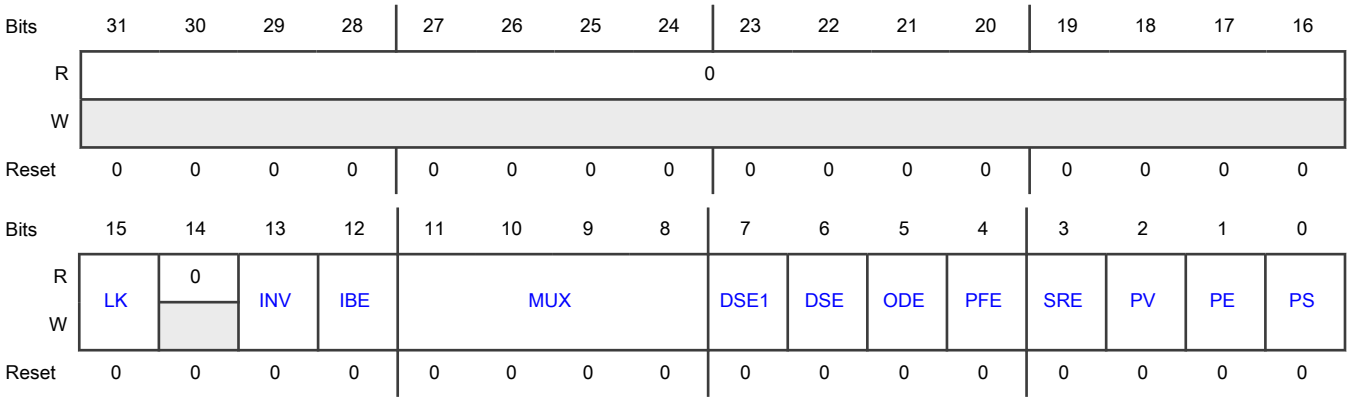
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR8
PORT1	PCR8	—
PORT2	—	PCR8
PORT3	PCR8	—
PORT4	—	PCR8

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 - Does not lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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 - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
7	Drive Strength Enable

Table continues on the next page...

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>PORT1</td><td>PCR8</td><td>—</td></tr><tr><td>PORT3</td><td>—</td><td>PCR8</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.
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.</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>

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>PORT1</td><td>PCR8</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR8</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.
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>PORT1</td><td>PCR8</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR8</td></tr>
</table>
0b - Low
1b - High</td></tr>
<tr><td>1</td><td>Pull Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
PE	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

11.6.1.16 Pin Control 9 (PCR9)

Offset

Register	Offset
PCR9	A4h

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.

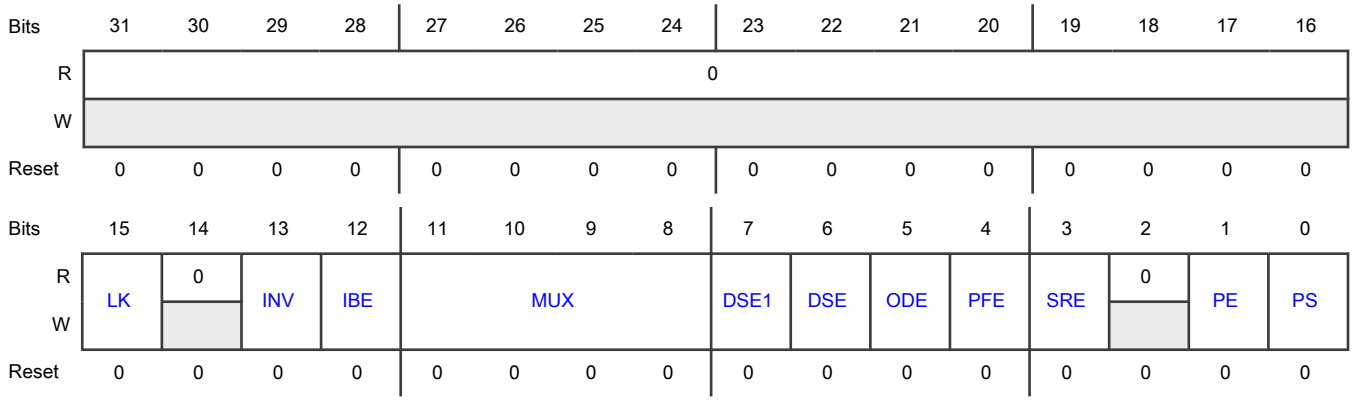
Instance	Register supported	Register not supported
PORT0	—	PCR9

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PORT1	PCR9	—
PORT2	—	PCR9
PORT3	PCR9	—
PORT4	—	PCR9

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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 - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (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. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

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><tr><td>PORT1</td><td>PCR9</td><td>—</td></tr><tr><td>PORT3</td><td>—</td><td>PCR9</td></tr></table> 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. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

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>PORT1</td><td>PCR9</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR9</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.
0b - Fast
1b - Slow</td></tr>
<tr><td>2
—</td><td>Reserved</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>

11.6.1.17 Pin Control a (PCR10 - PCR11)

Offset

Register	Offset
PCR10	A8h
PCR11	ACh

Function

Configures pin control features on each pin.

NOTE

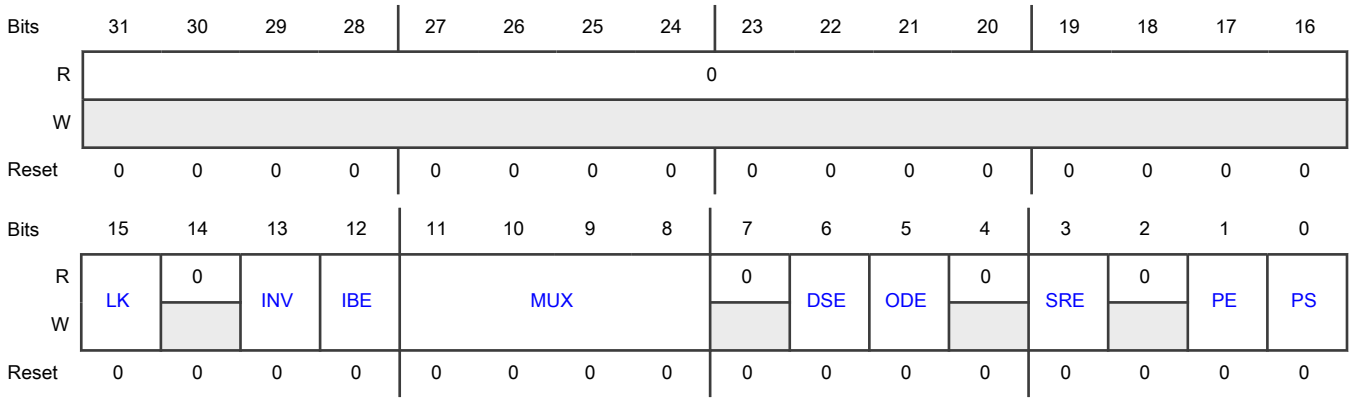
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR10–PCR11
PORT1	PCR10–PCR11	—
PORT2	PCR10–PCR11	—
PORT3	PCR10–PCR11	—
PORT4	—	PCR10–PCR11

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables

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: **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
PORT1	PCR10–PCR11	—
PORT2	PCR10–PCR11[10–8]	PCR10–PCR11[11]
PORT3	PCR10–PCR11	—

NOTE
The descriptions of the field settings vary by module instance.

Instance	Field value and description
PORT1	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
	PORT3	0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (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.	

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 —	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.

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

11.6.1.18 Pin Control a (PCR12 - PCR15)

Offset

Register	Offset
PCR12	B0h
PCR13	B4h
PCR14	B8h
PCR15	BCh

Function

Configures pin control features on each pin.

NOTE

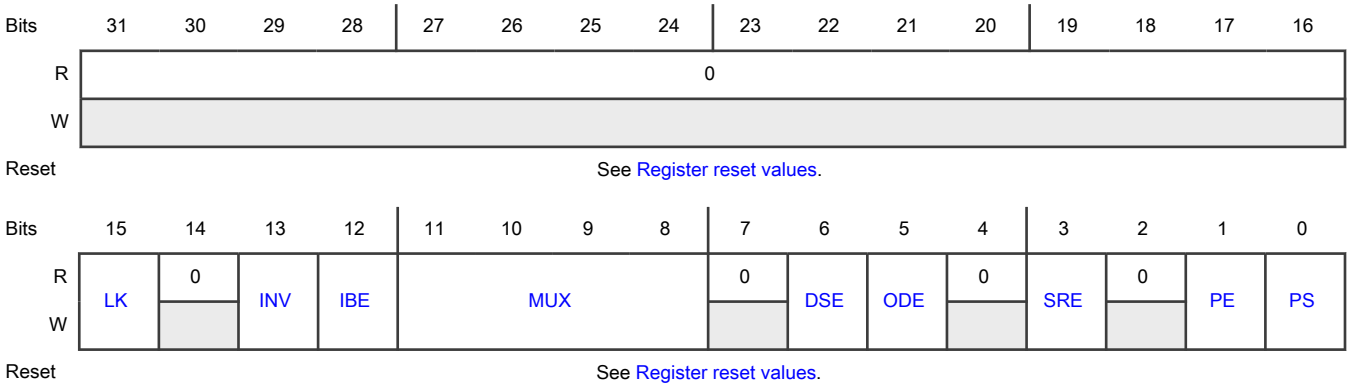
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR12–PCR15
PORT1	PCR12–PCR15	—
PORT2	PCR12–PCR13 PCR15	PCR14
PORT3	PCR12–PCR15	—
PORT4	—	PCR12–PCR15

Diagram



Register reset values

Register	Reset value
PCR12–PCR13	PORT0: Register not supported PORT1–PORT3: 0000_0000h PORT4: Register not supported
PCR14	PORT0: Register not supported PORT1: 0000_0000h PORT2: Register not supported PORT3: 0000_0000h PORT4: Register not supported
PCR15	PORT0: Register not supported PORT1–PORT3: 0000_0000h PORT4: Register not supported

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 - Does not lock 1b - Locks
14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>13
INV</td><td>Invert Input
Inverts the digital input.
0b - Does not invert
1b - Inverts</td></tr>
<tr><td>12
IBE</td><td>Input Buffer Enable
Enables digital input buffer. When disabled, the digital input is required for analog functions.
0b - Disables
1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT1</td><td>PCR12–PCR13
PCR14–PCR15[10–8]</td><td>PCR14–PCR15[11]</td></tr><tr><td>PORT2</td><td>PCR12–PCR13
PCR15[10–8]</td><td>PCR15[11]</td></tr><tr><td>PORT3</td><td>PCR12–PCR15</td><td>—</td></tr></table>000b - Alternative 0 (GPIO)
0000b - Alternative 0 (GPIO)
0001b - Alternative 1 (chip-specific)
001b - Alternative 1 (chip-specific)
0010b - Alternative 2 (chip-specific)
0011b - Alternative 3 (chip-specific)
010b - Alternative 2 (chip-specific)
0100b - Alternative 4 (chip-specific)</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Alternative 5 (chip-specific) 011b - Alternative 3 (chip-specific) 0110b - Alternative 6 (chip-specific) 0111b - Alternative 7 (chip-specific) 100b - Alternative 4 (chip-specific) 1000b - Alternative 8 (chip-specific) 1001b - Alternative 9 (chip-specific) 101b - Alternative 5 (chip-specific) 1010b - Alternative 10 (chip-specific) 1011b - Alternative 11 (chip-specific) 110b - Alternative 6 (chip-specific) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific) 111b - Alternative 7 (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. 0b - Disables 1b - Enables

Table continues on the next page...

Table continued from the previous page...

Field	Function
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

11.6.1.19 Pin Control 16 (PCR16)

Offset

Register	Offset
PCR16	C0h

Function

Configures pin control features on each pin.

NOTE

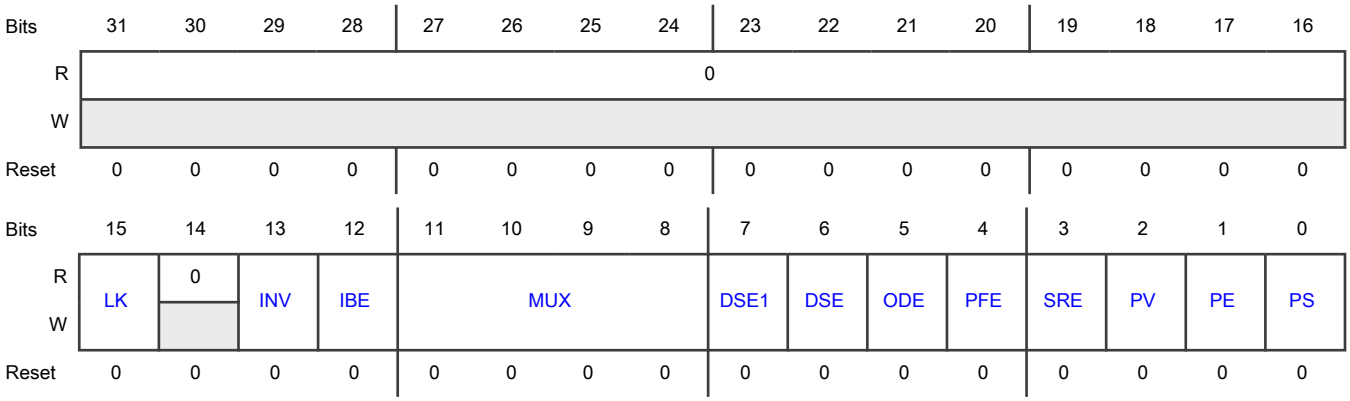
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	PCR16	—
PORT1	—	PCR16
PORT2	PCR16	—
PORT3	PCR16	—
PORT4	—	PCR16

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 - Does not lock

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>1b - Locks</td></tr>
<tr><td>14
—</td><td>Reserved</td></tr>
<tr><td>13
INV</td><td>Invert Input
Inverts the digital input.
0b - Does not invert
1b - Inverts</td></tr>
<tr><td>12
IBE</td><td>Input Buffer Enable
Enables digital input buffer. When disabled, the digital input is required for analog functions.
0b - Disables
1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT0</td><td>PCR16</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR16[10–8]</td><td>PCR16[11]</td></tr>
<tr><td>PORT3</td><td>PCR16[10–8]</td><td>PCR16[11]</td></tr>
</table>
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>PORT0</td><td>0000b - Alternative 0 (GPIO)</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
	PORT3	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
7	Drive Strength Enable	

Table continues on the next page...

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>PORT0</td><td>PCR16</td><td>—</td></tr><tr><td>PORT2</td><td>—</td><td>PCR16</td></tr><tr><td>PORT3</td><td>—</td><td>PCR16</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.
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</td><td>Passive Filter 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>PFE</td><td>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>PORT0</td><td>PCR16</td><td>—</td></tr><tr><td>PORT2</td><td>—</td><td>PCR16</td></tr><tr><td>PORT3</td><td>—</td><td>PCR16</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.
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>PORT0</td><td>PCR16</td><td>—</td></tr><tr><td>PORT2</td><td>—</td><td>PCR16</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><tr><td>PORT3</td><td>—</td><td>PCR16</td></tr></table> 0b - Low 1b - High
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

11.6.1.20 Pin Control 17 (PCR17)

Offset

Register	Offset
PCR17	C4h

Function

Configures pin control features on each pin.

NOTE

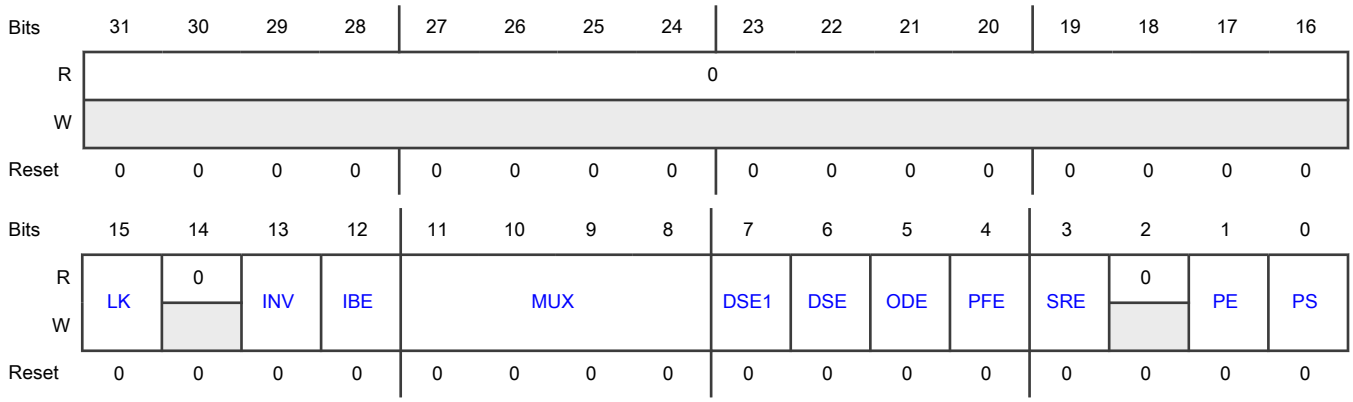
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	PCR17	—
PORT1	—	PCR17
PORT2	PCR17	—
PORT3	PCR17	—
PORT4	—	PCR17

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input.

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 - Does not invert
1b - Inverts</td></tr>
<tr><td>12
IBE</td><td>Input Buffer Enable
Enables digital input buffer. When disabled, the digital input is required for analog functions.
0b - Disables
1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT0</td><td>PCR17</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR17[10–8]</td><td>PCR17[11]</td></tr>
<tr><td>PORT3</td><td>PCR17[10–8]</td><td>PCR17[11]</td></tr>
</table>
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>PORT0</td><td>0000b - Alternative 0 (GPIO)
0001b - Alternative 1 (chip-specific)
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)</td></tr>
</table>
</td></tr>
</table>

Table continued from the previous page...

Field	Function	
	Instance	Field value and description
		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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)
	PORT2	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (chip-specific)
	PORT3	000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (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>PORT0</td><td>PCR17</td><td>—</td></tr><tr><td>PORT2</td><td>—</td><td>PCR17</td></tr><tr><td>PORT3</td><td>—</td><td>PCR17</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.
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.</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>
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>PORT0</td><td>PCR17</td><td>—</td></tr>
<tr><td>PORT2</td><td>—</td><td>PCR17</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR17</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.
0b - Fast
1b - Slow</td></tr>
<tr><td>2
—</td><td>Reserved</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.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 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

11.6.1.21 Pin Control a (PCR18 - PCR19)

Offset

Register	Offset
PCR18	C8h
PCR19	CCh

Function

Configures pin control features on each pin.

NOTE

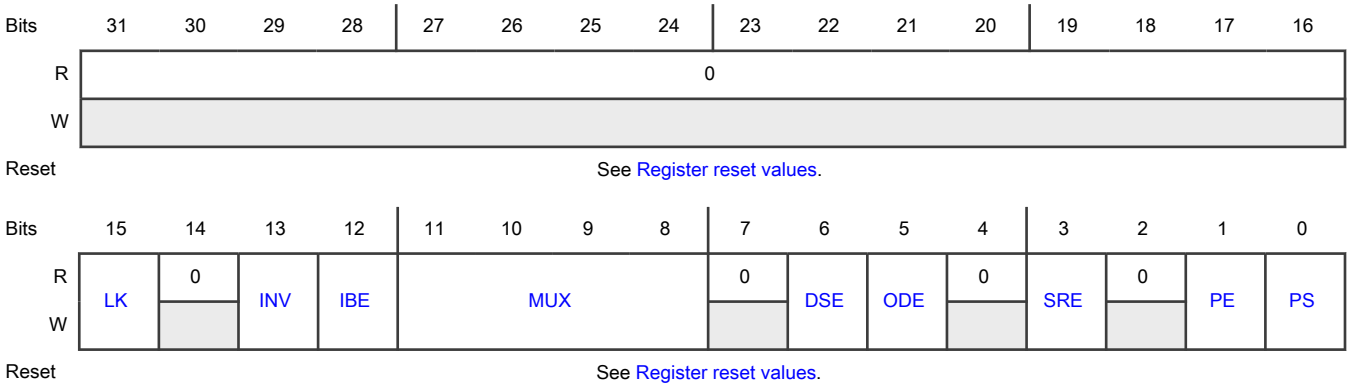
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	PCR18–PCR19	—
PORT1	—	PCR18–PCR19
PORT2	PCR19	PCR18
PORT3	PCR18–PCR19	—
PORT4	—	PCR18–PCR19

Diagram



Register reset values

Register	Reset value
PCR18	PORT0: 0000_0000h PORT1,PORT2: Register not supported PORT3: 0000_0000h PORT4: Register not supported
PCR19	PORT0: 0000_0000h PORT1: Register not supported PORT2,PORT3: 0000_0000h PORT4: Register not supported

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input.

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 - Does not invert
1b - Inverts</td></tr>
<tr><td>12
IBE</td><td>Input Buffer Enable
Enables digital input buffer. When disabled, the digital input is required for analog functions.
0b - Disables
1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT0</td><td>PCR18–PCR19</td><td>—</td></tr>
<tr><td>PORT2</td><td>PCR19[10–8]</td><td>PCR19[11]</td></tr>
<tr><td>PORT3</td><td>PCR18–PCR19[10–8]</td><td>PCR18–PCR19[11]</td></tr>
</table>
NOTE
The descriptions of the field settings vary by module instance.
<table>
<tr><th>Instance</th><th>Register</th><th>Field value and description</th></tr>
<tr><td>PORT0</td><td>PCR18–PCR19</td><td rowspan="2">0000b - Alternative 0 (GPIO)
0001b - Alternative 1 (chip-specific)
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)</td></tr>
<tr><td>PORT2</td><td>PCR18</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>Register</th><th>Field value and description</th></tr>
<tr><td></td><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)
1100b - Alternative 12 (chip-specific)
1101b - Alternative 13 (chip-specific)</td></tr>
<tr><td>PORT2</td><td>PCR19</td><td rowspan="2">000b - Alternative 0 (GPIO)
001b - Alternative 1 (chip-specific)
010b - Alternative 2 (chip-specific)
011b - Alternative 3 (chip-specific)
100b - Alternative 4 (chip-specific)
101b - Alternative 5 (chip-specific)
110b - Alternative 6 (chip-specific)
111b - Alternative 7 (chip-specific)</td></tr>
<tr><td>PORT3</td><td>PCR18–PCR19</td></tr>
</table>
</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.
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.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 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

11.6.1.22 Pin Control a (PCR20 - PCR23)

Offset

Register	Offset
PCR20	D0h
PCR21	D4h
PCR22	D8h
PCR23	DCh

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
PORT0	PCR20–PCR23	—
PORT1	—	PCR20–PCR23
PORT2	PCR20–PCR21 PCR23	PCR22
PORT3	PCR20–PCR22	PCR23
PORT4	—	PCR20–PCR23

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	INV	IBE	0	MUX			0	DSE	ODE	0	SRE	0	PE	PS
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
PCR20–PCR21	PORT0: 0000_0000h PORT1: Register not supported PORT2,PORT3: 0000_0000h PORT4: Register not supported
PCR22	PORT0: 0000_0000h PORT1,PORT2: Register not supported PORT3: 0000_0000h PORT4: Register not supported
PCR23	PORT0: 0000_0000h PORT1: Register not supported PORT2: 0000_0000h PORT3,PORT4: Register not supported

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10-8 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: 000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific) 110b - Alternative 6 (chip-specific) 111b - Alternative 7 (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. 0b - Disables

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

11.6.1.23 Pin Control a (PCR27 - PCR28)

Offset

Register	Offset
PCR27	ECh
PCR28	F0h

Function

Configures pin control features on each pin.

NOTE

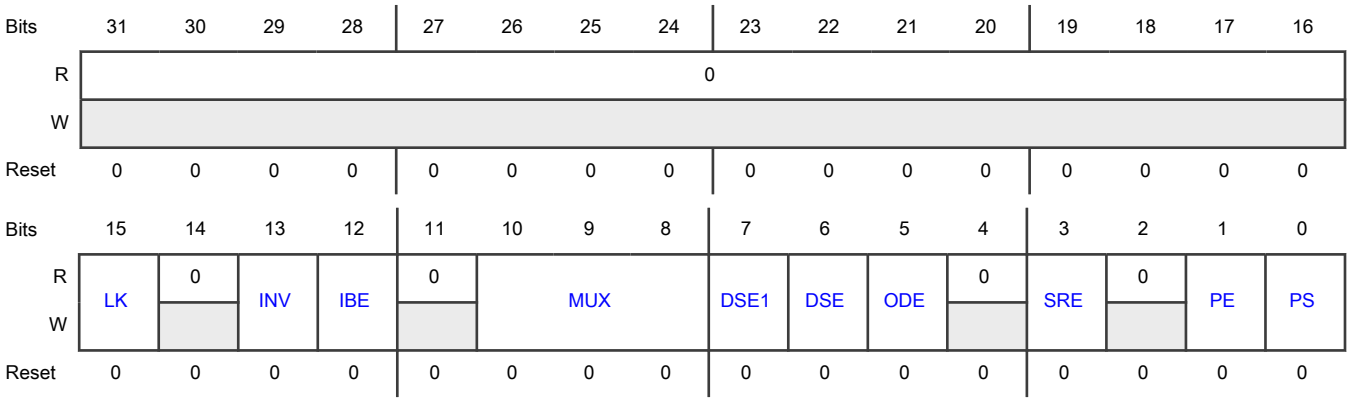
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR27–PCR28
PORT1	—	PCR27–PCR28
PORT2	—	PCR27–PCR28
PORT3	PCR27–PCR28	—
PORT4	—	PCR27–PCR28

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
11 —	Reserved
10-8 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: 000b - Alternative 0 (GPIO) 001b - Alternative 1 (chip-specific) 010b - Alternative 2 (chip-specific) 011b - Alternative 3 (chip-specific) 100b - Alternative 4 (chip-specific) 101b - Alternative 5 (chip-specific)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Alternative 6 (chip-specific) 111b - Alternative 7 (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. 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 —	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

Table continues on the next page...

Table continued from the previous page...

Field	Function
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

11.6.1.24 Pin Control 29 (PCR29)

Offset

Register	Offset
PCR29	F4h

Function

Configures pin control features on each pin.

NOTE

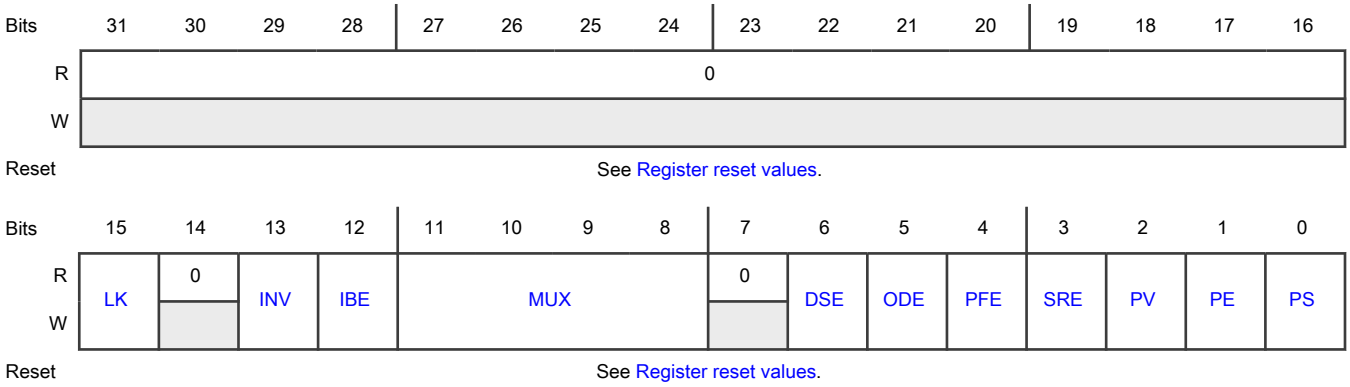
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR29
PORT1	PCR29	—
PORT2	—	PCR29
PORT3	PCR29	—
PORT4	—	PCR29

Diagram



Register reset values

Register	Reset value
PCR29	PORT0: Register not supported PORT1: 0000_0133h PORT2: Register not supported PORT3: 0000_0103h PORT4: Register not supported

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...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Does not lock
1b - Locks</td></tr>
<tr><td>14
—</td><td>Reserved</td></tr>
<tr><td>13
INV</td><td>Invert Input
Inverts the digital input.
0b - Does not invert
1b - Inverts</td></tr>
<tr><td>12
IBE</td><td>Input Buffer Enable
Enables digital input buffer. When disabled, the digital input is required for analog functions.
0b - Disables
1b - Enables</td></tr>
<tr><td>11-8
MUX</td><td>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:
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>PORT1</td><td>PCR29[9–8]</td><td>PCR29[11–10]</td></tr>
<tr><td>PORT3</td><td>PCR29</td><td>—</td></tr>
</table>
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>PORT1</td><td>00b - Alternative 0 (GPIO)
01b - Alternative 1 (chip-specific)</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></td><td>10b - Alternative 2 (chip-specific) 11b - Alternative 3 (chip-specific)</td></tr><tr><td>PORT3</td><td>0000b - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (chip-specific)</td></tr></table>
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...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>• 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>PORT1</td><td>PCR29</td><td>—</td></tr><tr><td>PORT3</td><td>—</td><td>PCR29</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.
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.</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><tr><td>PORT1</td><td>PCR29</td><td>—</td></tr><tr><td>PORT3</td><td>—</td><td>PCR29</td></tr></table> 0b - Low 1b - High
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

11.6.1.25 Pin Control 30 (PCR30)

Offset

Register	Offset
PCR30	F8h

Function

Configures pin control features on each pin.

NOTE

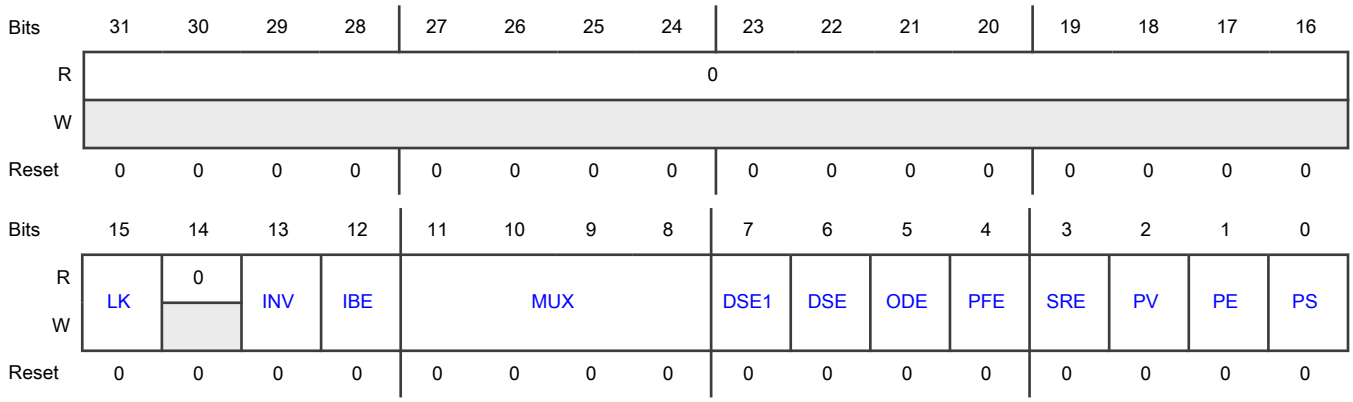
Do not modify pin configuration registers associated with pins that are unavailable in your selected package. All unbonded pins unavailable in your package default to the Disabled state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
PORT0	—	PCR30
PORT1	PCR30	—
PORT2	—	PCR30
PORT3	PCR30	—
PORT4	—	PCR30

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 - Does not lock 1b - Locks
14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12 IBE	Input Buffer Enable Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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 - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (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.

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>

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>PORT1</td><td>PCR30</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR30</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.

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.
</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>
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>PORT1</td><td>PCR30</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR30</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.
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>PORT1</td><td>PCR30</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR30</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

11.6.1.26 Pin Control 31 (PCR31)

Offset

Register	Offset
PCR31	FCh

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.

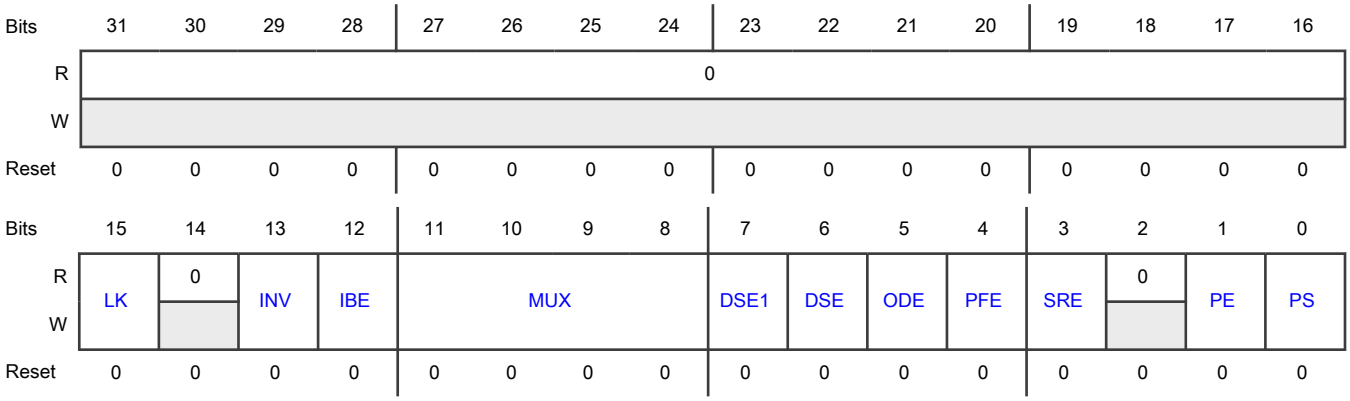
Instance	Register supported	Register not supported
PORT0	—	PCR31
PORT1	PCR31	—

Table continues on the next page...

Table continued from the previous page...

Instance	Register supported	Register not supported
PORT2	—	PCR31
PORT3	PCR31	—
PORT4	—	PCR31

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 - Does not lock 1b - Locks
14 —	Reserved
13 INV	Invert Input Inverts the digital input. 0b - Does not invert 1b - Inverts
12	Input Buffer Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
IBE	Enables digital input buffer. When disabled, the digital input is required for analog functions. 0b - Disables 1b - Enables
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 - Alternative 0 (GPIO) 0001b - Alternative 1 (chip-specific) 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) 1100b - Alternative 12 (chip-specific) 1101b - Alternative 13 (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. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

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><tr><td>PORT1</td><td>PCR31</td><td>—</td></tr><tr><td>PORT3</td><td>—</td><td>PCR31</td></tr></table> 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. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

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>PORT1</td><td>PCR31</td><td>—</td></tr>
<tr><td>PORT3</td><td>—</td><td>PCR31</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.
0b - Fast
1b - Slow</td></tr>
<tr><td>2
—</td><td>Reserved</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>

Chapter 12
General Purpose Input/Output (GPIO)

12.1 Chip-specific GPIO information

Table 65. Reference links to related information

Topic	Related module	Reference
Full description	GPIO	GPIO
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

12.1.1 Module instances

This device has five instances of the GPIO module, GPIO0, GPIO1, GPIO2, GPIO3 and GPIO4.

12.1.2 Interrupt, DMA request, and trigger outputs for GPIO

GPIO modules can be used to trigger interrupts, DMA requests, or trigger outputs (see the INPUTMUX and WUU chapters for details). See the table below for details on the chip-level interrupt, DMA request, and trigger output capability for each GPIO instance.

Table 66. Interrupt, DMA request, and trigger outputs for GPIO

GPIO	Interrupt	DMA	Trigger
GPIO0	Yes	Yes	Yes
GPIO1	Yes	Yes	Yes
GPIO2	Yes	Yes	Yes
GPIO3	Yes	Yes	Yes
GPIO4	Yes	Yes	Yes

12.2 Overview

GPIO communicates to the processor core via a zero wait-state interface for maximum pin performance.

12.2.1 Block diagram

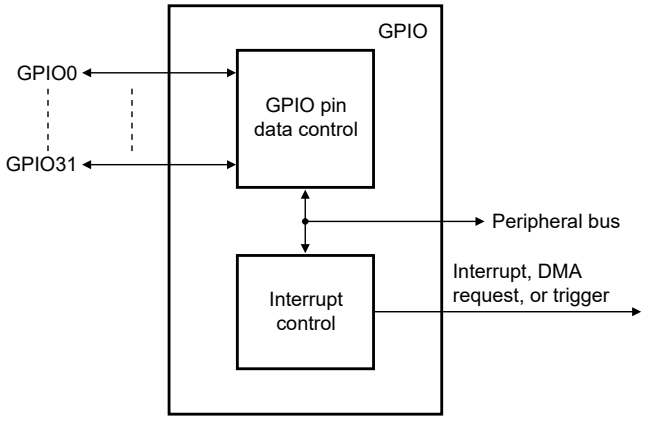


Figure 16. Block diagram

12.2.2 Features

- Port Data Input (PDIR) displays the logic value on each pin when the pin is configured for any digital function, provided the corresponding Port Control and Interrupt modules for that pin are enabled.
- Port Data Output (PDOR), with corresponding set, clear, and toggle registers, controls the output data of each pin when the pin is configured for the GPIO function.
- Port Data Direction (PDDR) controls the direction of each pin when the pin is configured for the GPIO function.
- Port Input Disable (PIDR) controls disabling of the input for each general-purpose pin.
- Pin interrupts:
  - Interrupt flags and enable registers for each pin are functional in all digital pin muxing modes.
  - Support for interrupt, peripheral trigger, or DMA request is configured for each pin.
  - Support for edge-sensitive (rising, falling, or both) or level-sensitive (low, high) interrupts is configured for each pin.
  - Asynchronous wake-up in Low-Power mode.
  - GPIO generates a total of 1 interrupts, 1 output triggers, and 1 DMA requests.

12.3 Functional description

12.3.1 Low-Power mode

You can configure GPIO to exit Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

12.3.2 Debug mode

GPIO remains functional in Debug mode.

12.3.3 General-purpose input

The logic state of each pin is available via Port Data Input (PDIR) if:

- The corresponding field in Port Input Disable (PIDR) is 0.
- The pin is configured for a digital function.

12.3.4 General-purpose output

The logic state of each pin is controlled via Port Data Output (PDOR) and Port Data Direction (PDDR), provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input or output.

Table 67. General-purpose output

If	Then
A pin is configured for the GPIO function and the corresponding PDDR field is 0	The pin is configured as an input.
A pin is configured for the GPIO function and the corresponding PDDR field is 1	The pin is configured as an output and the logic state of the pin is equal to the corresponding PDOR field.

For efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers allow 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.

12.3.5 Clocking

GPIO receives a single clock, which is used for register access and synchronization with external pin inputs. There are no special considerations.

12.3.6 Reset

GPIO receives a single reset, which resets the peripheral. There are no special considerations.

12.3.7 External interrupts

The external interrupt capability of GPIO is available in all digital pin muxing modes.

You can configure each pin individually for any of the external interrupt modes shown in the following table.

Table 68. Available pin configurations for external interrupts

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 GPIO is in Low-Power mode, the input is first synchronized to the system clock to detect the configured level or edge transition.

GPIO 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 either ISFR0[ISF*n*] or ICR0[ISF].

GPIO 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 Low-Power mode.

GPIO generates a peripheral trigger output that asserts if any pin configured for the active-high trigger is logic 1, or any pin triggered for the active-low trigger is logic 0. The peripheral trigger output asynchronously updates from the value on the configured pins.

12.3.8 Global interrupt control

The two global interrupt control registers (Global Interrupt Control Low (GICLR) and Global Interrupt Control High (GICHR)) allow a single register write to update the upper 16 bits of Interrupt Control a (ICR0 - ICR31) for up to 16 pins, all with the same value.

These global interrupt control registers allow you 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 0.

12.3.9 DMA

GPIO generates 1 requests. For each pin, you can configure DMA requests to assert for either rising, falling, or both edge detection. See External interrupts for more information.

12.4 External signals

Table 69. 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. 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 chip. See the "Signal Multiplexing" chapter for the number of GPIO pins within each port available on this chip.

12.5 Initialization

To initialize GPIO:

1. Initialize the GPIO pins for the output function:
   a. Configure the output logic value for each pin by using Port Data Output (PDOR).
   b. Configure the direction for each pin by using Port Data Direction (PDDR).
2. Initialize the interrupt function by writing to Interrupt Control a (ICR0 - ICR31) for the corresponding pins and desired configuration. If the pin is previously used for a different function, first write 0100_0000h to Interrupt Control a (ICR0 - ICR31) to disable the previous function and clear the flag.

12.6 Application information

GPIO includes the following applications:

- Reading the state of a single pin by performing a byte read of Pin Data (P0DR - P31DR).
- Updating the state of a single pin by performing a byte write to Pin Data (P0DR - P31DR).
- Reading the state of multiple pins by reading Port Data Input (PDIR).
- Updating the state of multiple pins by using the following ways:
  - Writing to Port Data Output (PDOR).
  - Writing to Port Set Output (PSOR) to write 1 to Port Data Output (PDOR).
  - Writing to Port Clear Output (PCOR) to write 0 to Port Data Output (PDOR).
  - Writing to Port Toggle Output (PTOR) to toggle Port Data Output (PDOR).

12.7 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" chapter for the exact control bits of each port.

12.7.1 GPIO register descriptions

12.7.1.1 GPIO memory map

RGPIO0 base address: 4010_2000h

RGPIO1 base address: 4010_3000h

RGPIO2 base address: 4010_4000h

RGPIO3 base address: 4010_5000h

RGPIO4 base address: 4010_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0201_0000h
4h	Parameter (PARAM)	32	R	0000_0001h
40h	Port Data Output (PDOR)	32	RW	0000_0000h
44h	Port Set Output (PSOR)	32	RW	0000_0000h
48h	Port Clear Output (PCOR)	32	RW	0000_0000h
4Ch	Port Toggle Output (PTOR)	32	RW	0000_0000h
50h	Port Data Input (PDIR)	32	R	0000_0000h
54h	Port Data Direction (PDDR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
58h	Port Input Disable (PIDR)	32	RW	0000_0000h
60h - 7Fh	Pin Data (P0DR - P31DR)	8	RW	00h
80h - FCh	Interrupt Control a (ICR0 - ICR31)	32	RW	0000_0000h
100h	Global Interrupt Control Low (GICLR)	32	RW	0000_0000h
104h	Global Interrupt Control High (GICHR)	32	RW	0000_0000h
120h	Interrupt Status Flag (ISFR0)	32	RW	0000_0000h

12.7.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Indicates the version ID number of each GPIO.

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	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 0000_0000_0000_0001b - Protection registers implemented

12.7.1.3 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	0	1

Fields

Field	Function
31-4 —	Reserved
3-0 IRQNUM	Interrupt Number Indicates the number of interrupt, trigger, or DMA request domains.

12.7.1.4 Port Data Output (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. By default, these unbonded pins are set to the 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 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 Configures the logic level on the pin if it is configured for general-purpose output. If this field is 0, logic level 0 is driven on the pin, if the pin is configured for general-purpose output. If this field is 1, logic level 1 is driven on the pin, if the pin is configured for general-purpose output. **NOTE** Reading the fields for unbonded pins returns an undefined value. 0b - Logic level 0 1b - Logic level 1

12.7.1.5 Port Set Output (PSOR)

Offset

Register	Offset
PSOR	44h

Function

Updates the corresponding fields of Port Data Output (PDOR) to become 1.

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 Updates the content of the corresponding field in Port Data Output (PDOR). If this field is 0, the corresponding PDOR field does not change. If this field is 1, the corresponding PDOR field becomes 1. 0b - No change 1b - Corresponding field in PDOR becomes 1

12.7.1.6 Port Clear Output (PCOR)

Offset

Register	Offset
PCOR	48h

Function

Updates the corresponding fields of Port Data Output (PDOR) to become 0.

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 Updates the content of the corresponding field in Port Data Output (PDOR). If this field is 0, the corresponding PDOR field does not change. If this field is 1, the corresponding PDOR field becomes 0. 0b - No change 1b - Corresponding field in PDOR becomes 0

12.7.1.7 Port Toggle Output (PTOR)

Offset

Register	Offset
PTOR	4Ch

Function

Updates the corresponding fields of Port Data Output (PDOR) to set to the inverse of their current logic states.

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 Updates the content of the corresponding field in Port Data Output (PDOR). If this field is 0, the corresponding PDOR field does not change. If this field is 1, the corresponding PDOR field is set to the inverse of its current logic state. 0b - No change 1b - Set to the inverse of its current logic state

12.7.1.8 Port Data Input (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. By default, these unbonded pins are set to the 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 Indicates the logic level of the pin that is configured for use by a digital function. If this field is 0, the pin logic level is logic 0 or is not configured or implemented for use by a digital function. If this field is 1, the pin logic level is logic 1. 0b - Logic 0 1b - Logic 1

12.7.1.9 Port Data Direction (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 Configures individual port pins for input or output. If this field is 0, the pin is configured as general-purpose input for the GPIO function. If this field is 1, the pin is configured as general-purpose output for the GPIO function. 0b - Input 1b - Output

12.7.1.10 Port Input Disable (PIDR)

Offset

Register	Offset
PIDR	58h

Function

Disables the input for each general-purpose pin, which prevents the value from being reported in Port Data Input (PDIR).

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 Disables a pin for general-purpose input. If this field is 0, the pin is configured for general-purpose input, provided the pin is configured for a digital function. If this field is 1, the pin is disabled for general-purpose input. 0b - Configured for general-purpose input 1b - Disabled for general-purpose input

12.7.1.11 Pin Data (P0DR - P31DR)

Offset

For a = 0 to 31:

Register	Offset
PaDR	60h + (a × 1h)

Function

Configures the data feature of a pin. Pins that are unimplemented or not configured for a digital function read zero.

NOTE

You must not modify Pin Data (P0DR - P31DR) associated with the pins that are not available in your selected package. These unbonded pins are, by default, set to the Disable state for lowest power consumption.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0							PD
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-1 —	Reserved
0 PD	Pin Data (I/O) Specifies the pin logic level. This field updates the corresponding field in Port Data Output (PDOR); reading this field returns the value in the corresponding field of Port Data Input (PDIR). If this field is 0, the pin logic level is logic zero or not configured for use by a digital function. If this field is 1, the pin logic level is logic one. 0b - Logic zero 1b - Logic one

12.7.1.12 Interrupt Control a (ICR0 - ICR31)

Offset

For a = 0 to 31:

Register	Offset
ICRa	80h + (a × 4h)

Function

Configures interrupt features on each pin.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							ISF	0	0			IRQC			
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															
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 ISF	Interrupt Status Flag Indicates whether the configured interrupt is detected. The pin interrupt configuration is valid in all digital pin muxing modes. The fields in Interrupt Status Flag (ISFR0) have the same function. ISF can be cleared with either register field. If the pin is configured to generate a DMA request, then the corresponding flag is 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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22-20 —	Reserved
19-16 IRQC	Interrupt Configuration Specifies the ISF and DMA request configuration. The pin interrupt configuration is valid in all digital pin muxing modes. When changing the interrupt configuration, it is recommended to first disable ISF and then write the new configuration. The corresponding pin is configured to generate interrupt, trigger, or DMA request. **NOTE** See the GPIO chip-specific information to determine the GPIO instances that support interrupt, DMA request, or trigger capabilities. 0000b - ISF is disabled 0001b - ISF and DMA request on rising edge 0010b - ISF and DMA request on falling edge 0011b - ISF and DMA request on either edge 0100b - Reserved 0101b - ISF sets on rising edge 0110b - ISF sets on falling edge 0111b - ISF sets on either edge 1000b - ISF and interrupt when logic 0 1001b - ISF and interrupt on rising edge 1010b - ISF and interrupt on falling edge 1011b - ISF and Interrupt on either edge 1100b - ISF and interrupt when logic 1 1101b - Enable active-high trigger output; ISF 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 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

12.7.1.13 Global Interrupt Control Low (GICLR)

Offset

Register	Offset
GICLR	100h

Function

Updates any combination of the lower 16 Interrupt Control registers with the same value. 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															
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 Indicates the write value that is written to the upper 16 bits of Interrupt Control a (ICR0 - ICR31), selected by GIWE.
15-0 GIWEn	Global Interrupt Write Enable Indicates whether the upper 16 bits of the corresponding Interrupt Control a (ICR0 - ICR31) are updated with the value in GIWD. 0b - Not updated 1b - Updated

12.7.1.14 Global Interrupt Control High (GICHR)

Offset

Register	Offset
GICHR	104h

Function

Updates any combination of the upper 16 Interrupt Control registers with the same value. 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															
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 Indicates the write value that is written to the upper 16 bits of Interrupt Control a (ICR0 - ICR31), selected by GIWE.
15-0 GIWEn	Global Interrupt Write Enable Indicates whether the upper 16 bits of the corresponding Interrupt Control a (ICR0 - ICR31) are updated with the value in GIWD. 0b - Not updated. 1b - Updated

12.7.1.15 Interrupt Status Flag (ISFR0)

Offset

Register	Offset
ISFR0	120h

Function

Indicates whether the related configured interrupt is detected on each pin. The pin interrupt configuration is valid in all digital pin muxing modes. The ISF for each pin is also visible in the corresponding Interrupt Control register, and each flag can be cleared in either location.

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 Indicates the detection of the configured interrupt on each pin of the same number. If this field is 0, the configured interrupt is not detected on the pin of the same number. If this field is 1, the 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 is 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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag

Chapter 13
Input Multiplexing (INPUTMUX)

13.1 Chip-specific INPUTMUX information

Table 70. Reference links to related information

Topic	Related module	Reference
Full description	INPUTMUX	INPUTMUX
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing

Once set up, no clocks are required for the input multiplexer to function. The system clock is needed only to write to, or read from the INPUTMUX registers. Once the input multiplexer is configured, disable the clock to the INPUTMUX module in registers of the SYSCON module.

13.1.1 Module instances

This device contains one instance of the INPUTMUX module.

13.2 Overview

The Input Multiplexing module (INPUTMUX) provides signal routing options for internal peripherals. Some peripheral inputs are multiplexed to multiple input sources. The sources can be external pins, interrupts, output signals of other peripherals, or other internal signals.

NOTE

Depending on the package, not all inputs from external pins may be available.

13.2.1 Block Diagram

Figure 17 shows a generic input multiplexer arrangement with n inputs.

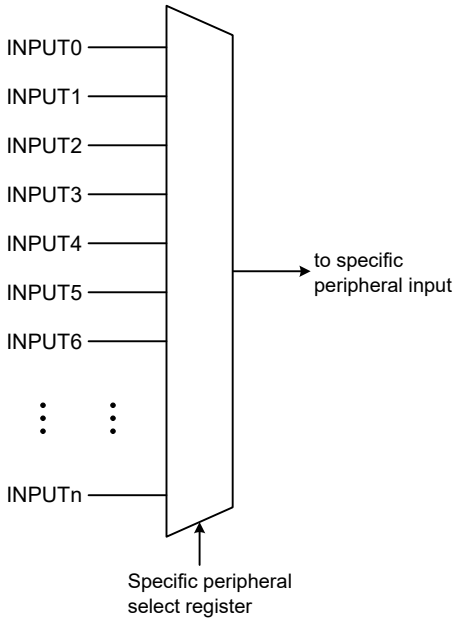


Figure 17. Generic input multiplexing

13.2.2 Features

- Configures the inputs to the SCTIMER.
- Configures the inputs to the asynchronous CTimers.
- Configures the inputs to the Pin Interrupt and Pattern Match (PINT).
- Configures the inputs to CMP, ADC, DAC, ENC, PWM, PWM EXT clock, AOI, external trigger.
- Configures the inputs to the Frequency Measurement (FREQME). This function is controlled by the FREQMECTRL register (See ANACTRL).

13.3 Functional description

The INPUTMUX implements a number of input multiplexers that select one of many inputs to be routed to a specific input signal for a given peripheral. This is used to allow user configuration of data paths between internal modules and/or external pins on the device. For every module input (output from the INPUTMUX), there is a register that selects the input to use, where the register name and description provide details on the module input controlled by each register. The input signal/pin options for each of the muxes are configurable, and can vary from mux to mux. Refer to the register descriptions for the details on the input signal/pin options used for each INPUTMUX output in Memory map and register definition.

13.4 External signals

The INPUTMUX has no dedicated pins. Multiplexer inputs from external pins work independently of any other function assigned to the pin as long as no analog function is enabled.

Table 71. INPUTMUX pin description

Pins	Peripheral	Section
Any existing pin on port 0 or 1	Pin interrupts 0 to 7	See INPUTMUX.

Table continues on the next page...

Table 71. INPUTMUX pin description (continued)

Pins	Peripheral	Section
PIO0_11, PIO0_12, PIO1_4	Frequency measure module	See INPUTMUX.
SCT0_GPI [0:7] pin functions selected from IOCON register	SCTimer/PWM	See SCTIMER.

13.5 Memory map and register definition

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

13.5.1 INPUTMUX register descriptions

13.5.1.1 INPUTMUX memory map

INPUTMUX0 base address: 4000_1000h

Offset	Register	Width (In bits)	Access	Reset value
20h - 2Ch	Capture select register for CTIMER inputs (CTIMER0CAP0 - CTIMER0CAP3)	32	RW	0000_007Fh
30h	Trigger register for TIMER0 (TIMER0TRIG)	32	RW	0000_007Fh
40h - 4Ch	Capture select register for CTIMER inputs (CTIMER1CAP0 - CTIMER1CAP3)	32	RW	0000_007Fh
50h	Trigger register for TIMER1 (TIMER1TRIG)	32	RW	0000_007Fh
60h - 6Ch	Capture select register for CTIMER inputs (CTIMER2CAP0 - CTIMER2CAP3)	32	RW	0000_007Fh
70h	Trigger register for TIMER2 inputs (TIMER2TRIG)	32	RW	0000_007Fh
180h	Selection for frequency measurement reference clock (FREQMEAS_REF)	32	RW	0000_003Fh
184h	Selection for frequency measurement target clock (FREQMEAS_TAR)	32	RW	0000_003Fh
1A0h - 1ACh	Capture select register for CTIMER inputs (CTIMER3CAP0 - CTIMER3CAP3)	32	RW	0000_007Fh
1B0h	Trigger register for TIMER3 (TIMER3TRIG)	32	RW	0000_007Fh
1C0h - 1CCh	Capture select register for CTIMER inputs (CTIMER4CAP0 - CTIMER4CAP3)	32	RW	0000_007Fh
1D0h	Trigger register for TIMER4 (TIMER4TRIG)	32	RW	0000_007Fh
200h - 23Ch	AOI1 trigger input connections 0 (AOI1_INPUT0 - AOI1_INPUT15)	32	RW	0000_007Fh
260h	CMP0 input connections (CMP0_TRIG)	32	RW	0000_003Fh
280h - 28Ch	ADC Trigger input connections (ADC0_TRIG0 - ADC0_TRIG3)	32	RW	0000_003Fh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2C0h - 2CCh	ADC Trigger input connections (ADC1_TRIG0 - ADC1_TRIG3)	32	RW	0000_003Fh
300h	This register selects the DAC0 trigger inputs. (DAC0_TRIG)	32	RW	0000_003Fh
360h	QDC0 Trigger Input Connections (QDC0_TRIG)	32	RW	0000_007Fh
364h	QDC0 Trigger Input Connections (QDC0_HOME)	32	RW	0000_007Fh
368h	QDC0 Trigger Input Connections (QDC0_INDEX)	32	RW	0000_007Fh
36Ch	QDC0 Trigger Input Connections (QDC0_PHASEB)	32	RW	0000_007Fh
370h	QDC0 Trigger Input Connections (QDC0_PHASEA)	32	RW	0000_007Fh
374h	QDC0 Trigger Input Connections (QDC0_ICAP1)	32	RW	0000_007Fh
378h	QDC0 Trigger Input Connections (QDC0_ICAP2)	32	RW	0000_007Fh
37Ch	QDC0 Trigger Input Connections (QDC0_ICAP3)	32	RW	0000_007Fh
380h	QDC1 Trigger Input Connections (QDC1_TRIG)	32	RW	0000_007Fh
384h	QDC1 Trigger Input Connections (QDC1_HOME)	32	RW	0000_007Fh
388h	QDC1 Trigger Input Connections (QDC1_INDEX)	32	RW	0000_007Fh
38Ch	QDC1 Trigger Input Connections (QDC1_PHASEB)	32	RW	0000_007Fh
390h	QDC1 Trigger Input Connections (QDC1_PHASEA)	32	RW	0000_007Fh
394h	QDC1 Trigger Input Connections (QDC1_ICAP1)	32	RW	0000_007Fh
398h	QDC1 Trigger Input Connections (QDC1_ICAP2)	32	RW	0000_007Fh
39Ch	QDC1 Trigger Input Connections (QDC1_ICAP3)	32	RW	0000_007Fh
3A0h	PWM0 input trigger connections (FlexPWM0_SM0_EXTA0)	32	RW	0000_003Fh
3A4h	PWM0 input trigger connections (FlexPWM0_SM0_EXTSYNC)	32	RW	0000_003Fh
3A8h	PWM0 input trigger connections (FlexPWM0_SM1_EXTA)	32	RW	0000_003Fh
3ACh	PWM0 input trigger connections (FlexPWM0_SM1_EXTSYNC)	32	RW	0000_003Fh
3B0h	PWM0 input trigger connections (FlexPWM0_SM2_EXTA)	32	RW	0000_003Fh
3B4h	PWM0 input trigger connections (FlexPWM0_SM2_EXTSYNC)	32	RW	0000_003Fh
3C0h - 3CCh	PWM0 Fault Input Trigger Connections (FlexPWM0_FAULT0 - FlexPWM0_FAULT3)	32	RW	0000_003Fh
3D0h	PWM0 input trigger connections (FlexPWM0_FORCE)	32	RW	0000_003Fh
3E0h	PWM1 input trigger connections (FlexPWM1_SM0_EXTA0)	32	RW	0000_003Fh
3E4h	PWM1 input trigger connections (FlexPWM1_SM0_EXTSYNC)	32	RW	0000_003Fh
3E8h	PWM1 input trigger connections (FlexPWM1_SM1_EXTA)	32	RW	0000_003Fh
3ECh	PWM1 input trigger connections (FlexPWM1_SM1_EXTSYNC)	32	RW	0000_003Fh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3F0h	PWM1 input trigger connections (FlexPWM1_SM2_EXTA)	32	RW	0000_003Fh
3F4h	PWM1 input trigger connections (FlexPWM1_SM2_EXTSYNC)	32	RW	0000_003Fh
400h - 40Ch	PWM1 Fault Input Trigger Connections (FlexPWM1_FAULT0 - FlexPWM1_FAULT3)	32	RW	0000_003Fh
410h	PWM1 input trigger connections (FlexPWM1_FORCE)	32	RW	0000_003Fh
420h	PWM0 external clock trigger (PWM0_EXT_CLK)	32	RW	0000_000Fh
424h	PWM1 external clock trigger (PWM1_EXT_CLK)	32	RW	0000_000Fh
440h - 47Ch	AOI0 trigger input connections 0 (AOI0_INPUT0 - AOI0_INPUT15)	32	RW	0000_007Fh
480h	USB-FS trigger input connections (USBFS_TRIG)	32	RW	0000_0007h
4C0h - 4DCh	EXT trigger connections (EXT_TRIG0 - EXT_TRIG7)	32	RW	0000_001Fh
4E0h	CMP1 input connections (CMP1_TRIG)	32	RW	0000_003Fh
540h	LPI2C2 trigger input connections (LPI2C2_TRIG)	32	RW	0000_003Fh
580h	OPAMP0 Trigger Input Connections (OPAMP0_TRIG)	32	RW	0000_003Fh
5A0h	LPI2C0 trigger input connections (LPI2C0_TRIG)	32	RW	0000_003Fh
5C0h	LPI2C1 trigger input connections (LPI2C1_TRIG)	32	RW	0000_003Fh
5E0h	LPSPI0 trigger input connections (LPSPI0_TRIG)	32	RW	0000_003Fh
600h	LPSPI1 trigger input connections (LPSPI1_TRIG)	32	RW	0000_003Fh
620h	LPUART0 trigger input connections (LPUART0)	32	RW	0000_003Fh
640h	LPUART1 trigger input connections (LPUART1)	32	RW	0000_003Fh
660h	LPUART2 trigger input connections (LPUART2)	32	RW	0000_003Fh
680h	LPUART3 trigger input connections (LPUART3)	32	RW	0000_003Fh
6A0h	LPUART4 trigger input connections (LPUART4)	32	RW	0000_003Fh
6E0h - 6ECh	FlexIO Trigger Input Connections (FLEXIO_TRIG0 - FLEXIO_TRIG3)	32	RW	0000_007Fh

13.5.1.2 Capture select register for CTIMER inputs (CTIMER0CAP0 - CTIMER0CAP3)

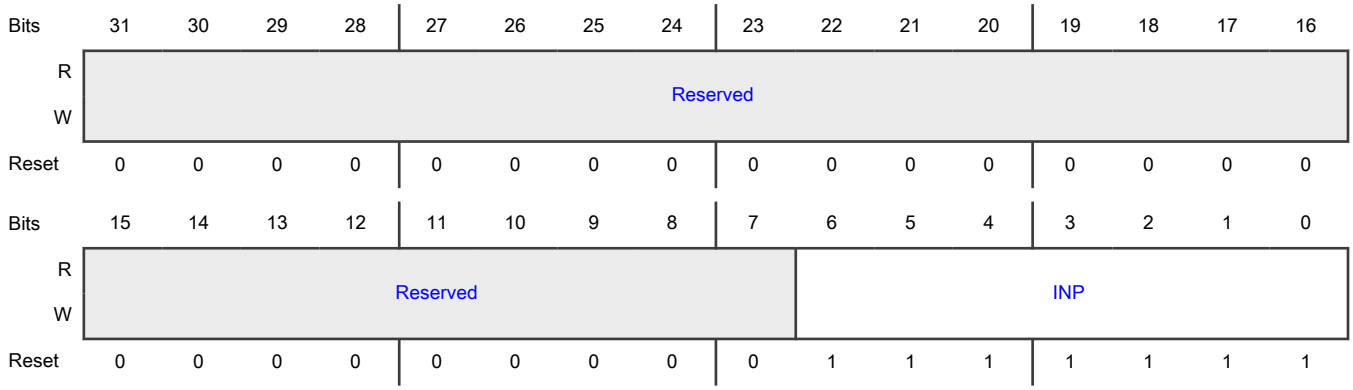
Offset

Register	Offset
CTIMER0CAP0	20h
CTIMER0CAP1	24h
CTIMER0CAP2	28h
CTIMER0CAP3	2Ch

Function

For each of the 5 standard timers, numbered i = 0 to 4 there are 4 TIMERiCAPTj, with j = 0 to 3, each allowing selecting between 25 external or internal input sources. The output of TIMER0CAPT0 Input multiplexing register 0 selects the source for TIMER0 capture input 0. The output of TIMER0CAPT1 Input multiplexing register 1 selects the source for TIMER0 capture input 1, and so forth up to TIMER4CAPT3. Input multiplexing register 3, which selects the input for TIMER4 capture input 3.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER0 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer1_MAT1 input is selected 010_0010b - CTimer1_MAT2 input is selected 010_0011b - CTimer1_MAT3 input is selected 010_0100b - CTimer2_MAT1 input is selected 010_0101b - CTimer2_MAT2 input is selected 010_0110b - CTimer2_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer3_MAT1 input is selected 101_0000b - CTimer3_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101_0001b - CTimer3_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.3 Trigger register for TIMER0 (TIMER0TRIG)

Offset

Register	Offset
TIMER0TRIG	30h

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									INP						
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 INP	Input number for CTIMER0 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer1_MAT1 input is selected 010_0010b - CTimer1_MAT2 input is selected 010_0011b - CTimer1_MAT3 input is selected 010_0100b - CTimer2_MAT1 input is selected 010_0101b - CTimer2_MAT2 input is selected 010_0110b - CTimer2_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer3_MAT1 input is selected 101_0000b - CTimer3_MAT2 input is selected 101_0001b - CTimer3_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.4 Capture select register for CTIMER inputs (CTIMER1CAP0 - CTIMER1CAP3)

Offset

Register	Offset
CTIMER1CAP0	40h
CTIMER1CAP1	44h
CTIMER1CAP2	48h
CTIMER1CAP3	4Ch

Function

For each of the 5 standard timers, numbered i = 0 to 4 there are 4 TIMERiCAPTj, with j = 0 to 3, each allowing selecting between 25 external or internal input sources. The output of TIMER0CAPT0 Input multiplexing register 0 selects the source for TIMER0 capture input 0. The output of TIMER0CAPT1 Input multiplexing register 1 selects the source for TIMER0 capture input 1, and so forth up to TIMER4CAPT3. Input multiplexing register 3, which selects the input for TIMER4 capture input 3.

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									INP						
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Fields

Field	Function
31-7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-0 INP	Input number for CTIMER1 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer2_MAT1 input is selected 010_0101b - CTimer2_MAT2 input is selected 010_0110b - CTimer2_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer3_MAT1 input is selected 101_0000b - CTimer3_MAT2 input is selected 101_0001b - CTimer3_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected

Table continues on the next page...

Table continued from the previous page...

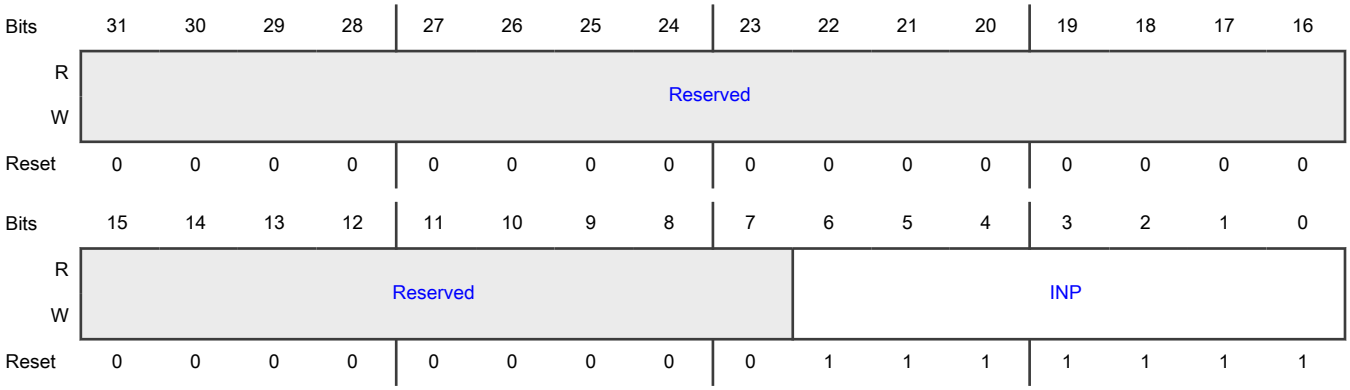
Field	Function
	All other values are reserved.

13.5.1.5 Trigger register for TIMER1 (TIMER1TRIG)

Offset

Register	Offset
TIMER1TRIG	50h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER1 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer2_MAT1 input is selected 010_0101b - CTimer2_MAT2 input is selected 010_0110b - CTimer2_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer3_MAT1 input is selected 101_0000b - CTimer3_MAT2 input is selected 101_0001b - CTimer3_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.6 Capture select register for CTIMER inputs (CTIMER2CAP0 - CTIMER2CAP3)

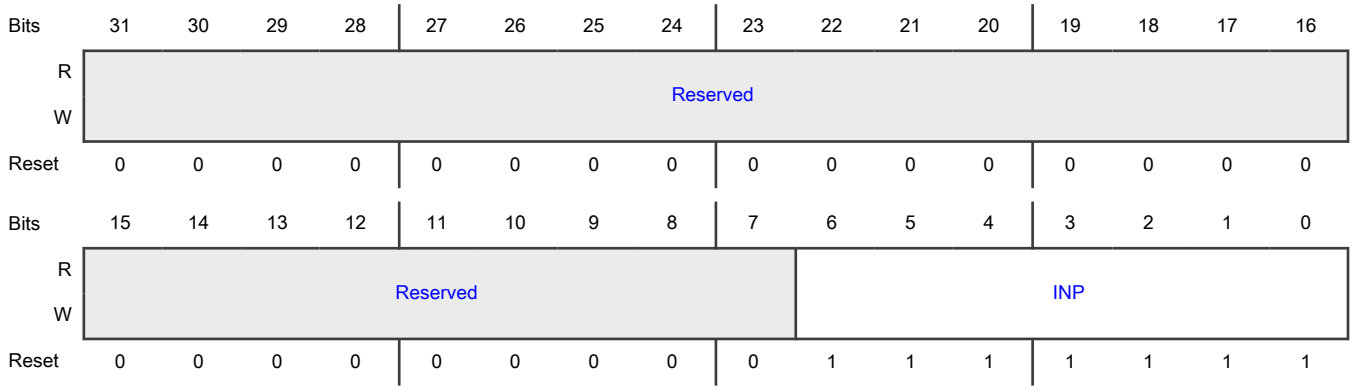
Offset

Register	Offset
CTIMER2CAP0	60h
CTIMER2CAP1	64h
CTIMER2CAP2	68h
CTIMER2CAP3	6Ch

Function

For each of the 5 standard timers, numbered i = 0 to 4 there are 4 TIMERiCAPTj, with j = 0 to 3, each allowing selecting between 25 external or internal input sources. The output of TIMER0CAPT0 Input multiplexing register 0 selects the source for TIMER0 capture input 0. The output of TIMER0CAPT1 Input multiplexing register 1 selects the source for TIMER0 capture input 1, and so forth up to TIMER4CAPT3. Input multiplexing register 3, which selects the input for TIMER4 capture input 3.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER2 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer1_MAT1 input is selected 010_0101b - CTimer1_MAT2 input is selected 010_0110b - CTimer1_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer3_MAT1 input is selected 101_0000b - CTimer3_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

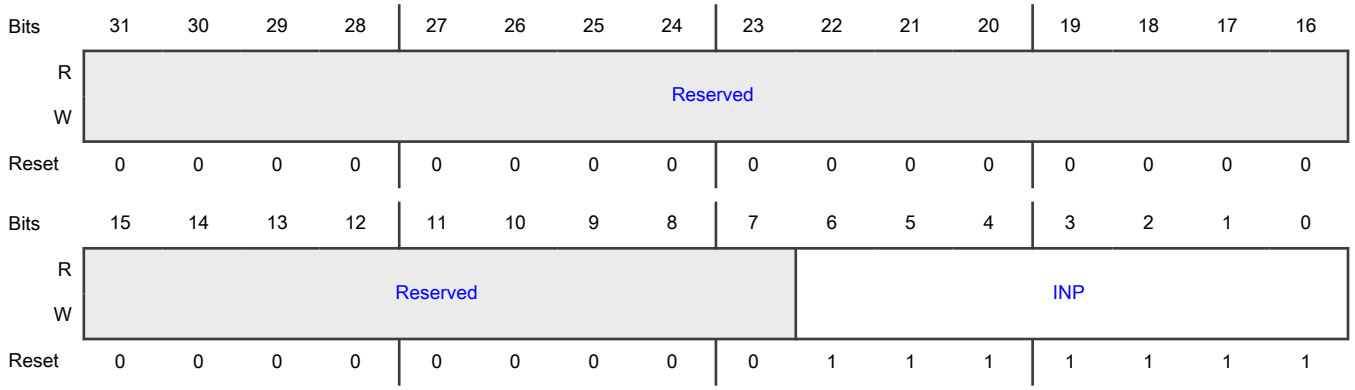
Field	Function
	101_0001b - CTimer3_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.7 Trigger register for TIMER2 inputs (TIMER2TRIG)

Offset

Register	Offset
TIMER2TRIG	70h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER2 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer1_MAT1 input is selected 010_0101b - CTimer1_MAT2 input is selected 010_0110b - CTimer1_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer3_MAT1 input is selected 101_0000b - CTimer3_MAT2 input is selected 101_0001b - CTimer3_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.8 Selection for frequency measurement reference clock (FREQMEAS_REF)

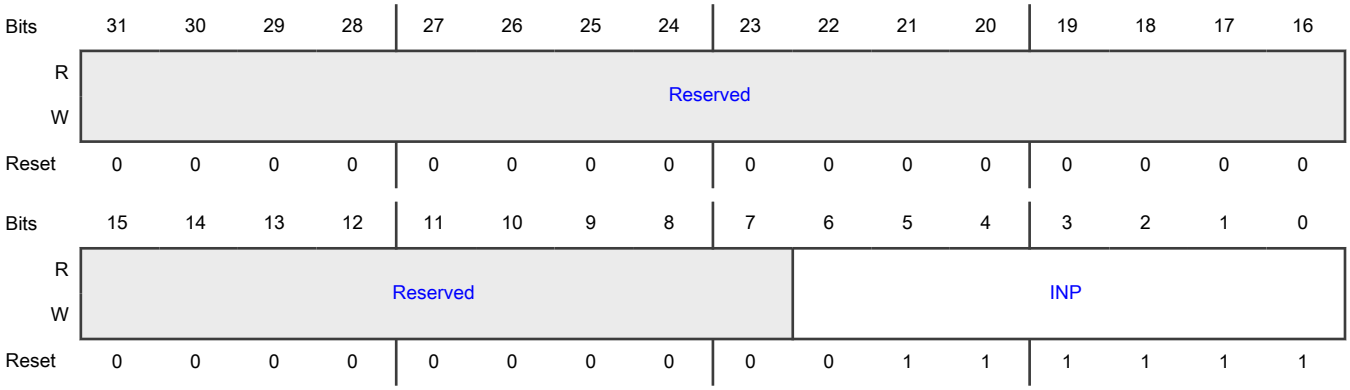
Offset

Register	Offset
FREQMEAS_REF	180h

Function

This register selects a clock for the reference clock of the frequency measure function. By default, no clock is selected.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Clock source number (binary value) for frequency measure function target clock. 000_0000b - Reserved 000_0001b - clk_in input is selected 000_0010b - FRO_OSC_12M input is selected 000_0011b - fro_hf_div input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_0100b - Reserved 000_0101b - clk_16k[1] input is selected 000_0110b - SLOW_CLK input is selected 000_0111b - FREQME_CLK_IN0 input is selected 000_1000b - FREQME_CLK_IN1 input is selected input is selected 000_1001b - AOI0_OUT0 input is selected 000_1010b - AOI0_OUT1 000_1011b - PWM0_SM0_MUX_TRIG0 000_1100b - PWM0_SM0_MUX_TRIG1 000_1101b - PWM0_SM1_MUX_TRIG0 000_1110b - PWM0_SM1_MUX_TRIG1 000_1111b - PWM0_SM2_MUX_TRIG0 001_0000b - PWM0_SM2_MUX_TRIG1 001_0001b - Reserved 001_0010b - Reserved 001_0011b - Reserved 001_0100b - Reserved 001_0101b - Reserved 001_0110b - Reserved 001_0111b - Reserved 001_1000b - Reserved 001_1001b - Reserved 001_1010b - Reserved 001_1011b - Reserved 001_1100b - Reserved 001_1101b - Reserved 001_1110b - Reserved 001_1111b - Reserved 010_0000b - AOI1_OUT0 input is selected 010_0001b - AOI1_OUT1 input is selected 010_0010b - PWM1_SM0_MUX_TRIG0 input is selected 010_0011b - PWM1_SM0_MUX_TRIG1 input is selected 010_0100b - PWM1_SM1_MUX_TRIG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_0101b - PWM1_SM1_MUX_TRIG1 input is selected 010_0110b - PWM1_SM2_MUX_TRIG0 input is selected 010_0111b - PWM1_SM2_MUX_TRIG1 input is selected 010_1000b - Reserved 010_1001b - Reserved All other values are reserved.

13.5.1.9 Selection for frequency measurement target clock (FREQMEAS_TAR)

Offset

Register	Offset
FREQMEAS_TAR	184h

Function

This register selects a clock for the target clock of the frequency measure function. By default, no clock is selected.

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								INP							
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-7 —	Reserved
6-0 INP	Clock source number (binary value) for frequency measure function target clock. 000_0000b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_0001b - clk_in input is selected 000_0010b - FRO_OSC_12M input is selected 000_0011b - fro_hf_div input is selected 000_0100b - Reserved 000_0101b - clk_16k[1] input is selected 000_0110b - SLOW_CLK input is selected 000_0111b - FREQME_CLK_IN0 input is selected 000_1000b - FREQME_CLK_IN1 input is selected input is selected 000_1001b - AOI0_OUT0 input is selected 000_1010b - AOI0_OUT1 000_1011b - PWM0_SM0_MUX_TRIG0 000_1100b - PWM0_SM0_MUX_TRIG1 000_1101b - PWM0_SM1_MUX_TRIG0 000_1110b - PWM0_SM1_MUX_TRIG1 000_1111b - PWM0_SM2_MUX_TRIG0 001_0000b - PWM0_SM2_MUX_TRIG1 001_0001b - Reserved 001_0010b - Reserved 001_0011b - Reserved 001_0100b - Reserved 001_0101b - Reserved 001_0110b - Reserved 001_0111b - Reserved 001_1000b - Reserved 001_1001b - Reserved 001_1010b - Reserved 001_1011b - Reserved 001_1100b - Reserved 001_1101b - Reserved 001_1110b - Reserved 001_1111b - Reserved 010_0000b - AOI1_OUT0 input is selected 010_0001b - AOI1_OUT1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_0010b - PWM1_SM0_MUX_TRIG0 input is selected 010_0011b - PWM1_SM0_MUX_TRIG1 input is selected 010_0100b - PWM1_SM1_MUX_TRIG0 input is selected 010_0101b - PWM1_SM1_MUX_TRIG1 input is selected 010_0110b - PWM1_SM2_MUX_TRIG0 input is selected 010_0111b - PWM1_SM2_MUX_TRIG1 input is selected 010_1000b - Reserved 010_1001b - Reserved All other values are reserved.

13.5.1.10 Capture select register for CTIMER inputs (CTIMER3CAP0 - CTIMER3CAP3)

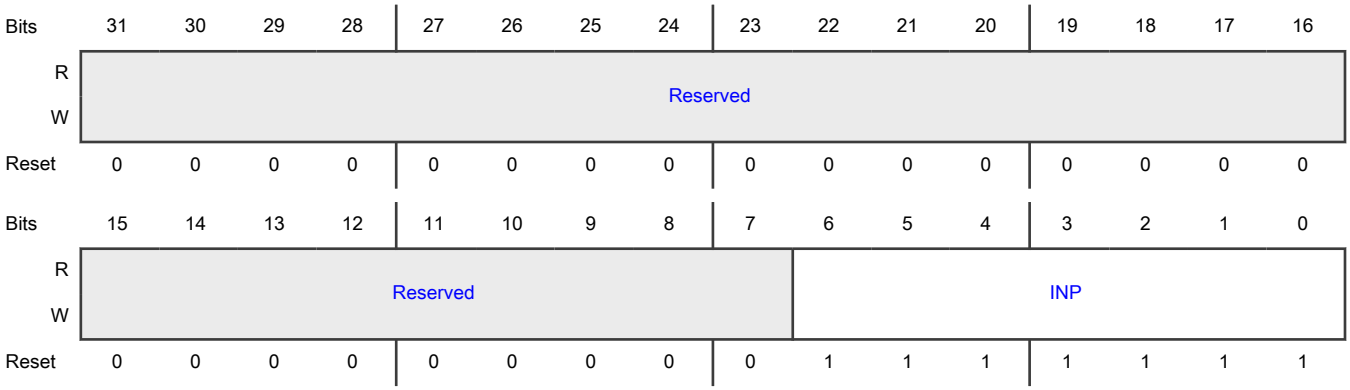
Offset

Register	Offset
CTIMER3CAP0	1A0h
CTIMER3CAP1	1A4h
CTIMER3CAP2	1A8h
CTIMER3CAP3	1ACh

Function

For each of the 5 standard timers, numbered i = 0 to 4 there are 4 TIMERiCAPTj, with j = 0 to 3, each allowing selecting between 25 external or internal input sources. The output of TIMER0CAPT0 Input multiplexing register 0 selects the source for TIMER0 capture input 0. The output of TIMER0CAPT1 Input multiplexing register 1 selects the source for TIMER0 capture input 1, and so forth up to TIMER4CAPT3. Input multiplexing register 3, which selects the input for TIMER4 capture input 3.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER3 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer1_MAT1 input is selected 010_0101b - CTimer1_MAT2 input is selected 010_0110b - CTimer1_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer2_MAT1 input is selected 101_0000b - CTimer2_MAT2 input is selected 101_0001b - CTimer2_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected

Table continues on the next page...

Table continued from the previous page...

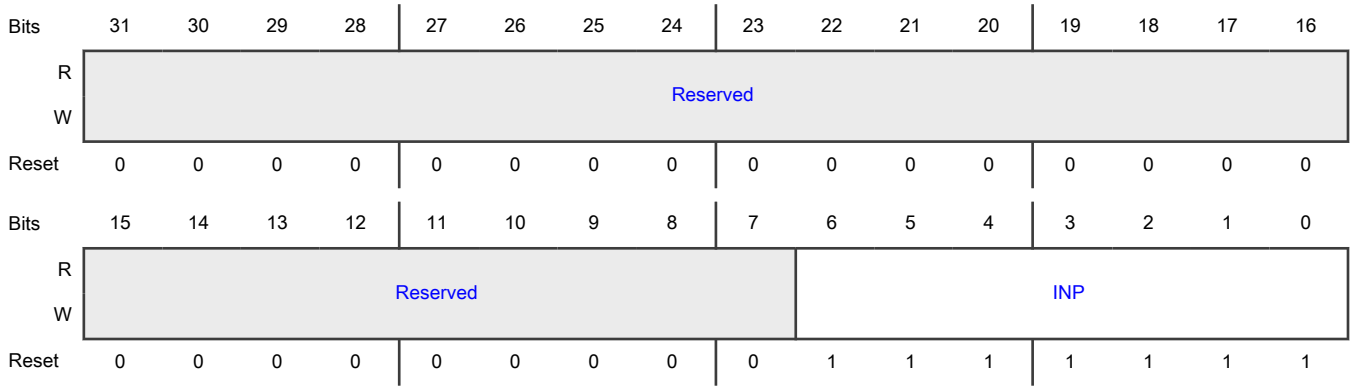
Field	Function
	110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.11 Trigger register for TIMER3 (TIMER3TRIG)

Offset

Register	Offset
TIMER3TRIG	1B0h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER3 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer1_MAT1 input is selected 010_0101b - CTimer1_MAT2 input is selected 010_0110b - CTimer1_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer2_MAT1 input is selected 101_0000b - CTimer2_MAT2 input is selected 101_0001b - CTimer2_MAT3 input is selected 101_0010b - CTimer4_MAT1 input is selected 101_0011b - CTimer4_MAT2 input is selected 101_0100b - CTimer4_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.12 Capture select register for CTIMER inputs (CTIMER4CAP0 - CTIMER4CAP3)

Offset

Register	Offset
CTIMER4CAP0	1C0h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CTIMER4CAP1	1C4h
CTIMER4CAP2	1C8h
CTIMER4CAP3	1CCh

Function

For each of the 5 standard timers, numbered i = 0 to 4 there are 4 TIMERiCAPTj, with j = 0 to 3, each allowing selecting between 25 external or internal input sources. The output of TIMER0CAPT0 Input multiplexing register 0 selects the source for TIMER0 capture input 0. The output of TIMER0CAPT1 Input multiplexing register 1 selects the source for TIMER0 capture input 1, and so forth up to TIMER4CAPT3. Input multiplexing register 3, which selects the input for TIMER4 capture input 3.

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									INP						
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 INP	Input number for CTIMER4 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected 001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer1_MAT1 input is selected 010_0101b - CTimer1_MAT2 input is selected 010_0110b - CTimer1_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected 011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected

Table continues on the next page...

Table continued from the previous page...

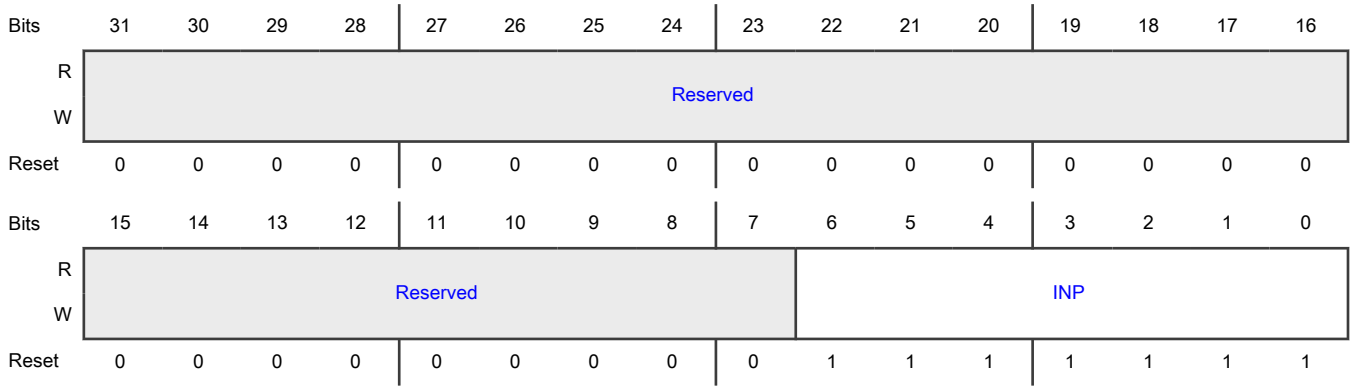
Field	Function
	100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer2_MAT1 input is selected 101_0000b - CTimer2_MAT2 input is selected 101_0001b - CTimer2_MAT3 input is selected 101_0010b - CTimer3_MAT1 input is selected 101_0011b - CTimer3_MAT2 input is selected 101_0100b - CTimer3_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected 101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.13 Trigger register for TIMER4 (TIMER4TRIG)

Offset

Register	Offset
TIMER4TRIG	1D0h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	Input number for CTIMER4 000_0000b - Reserved 000_0001b - CT_INP0 input is selected 000_0010b - CT_INP1 input is selected 000_0011b - CT_INP2 input is selected 000_0100b - CT_INP3 input is selected 000_0101b - CT_INP4 input is selected 000_0110b - CT_INP5 input is selected 000_0111b - CT_INP6 input is selected 000_1000b - CT_INP7 input is selected 000_1001b - CT_INP8 input is selected 000_1010b - CT_INP9 input is selected 000_1011b - CT_INP10 input is selected 000_1100b - CT_INP11 input is selected 000_1101b - CT_INP12 input is selected 000_1110b - CT_INP13 input is selected 000_1111b - CT_INP14 input is selected 001_0000b - CT_INP15 input is selected 001_0001b - CT_INP16 input is selected 001_0010b - CT_INP17 input is selected 001_0011b - CT_INP18 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - CT_INP19 input is selected 001_0101b - USB0 usb0 start of frame input is selected 001_0110b - AOI0_OUT0 input is selected 001_0111b - AOI0_OUT1 input is selected 001_1000b - AOI0_OUT2 input is selected 001_1001b - AOI0_OUT3 input is selected 001_1010b - ADC0_tcomp[0] 001_1011b - ADC0_tcomp[1] 001_1100b - ADC0_tcomp[2] 001_1101b - ADC0_tcomp[3] input is selected 001_1110b - CMP0_OUT is selected 001_1111b - CMP1_OUT is selected 010_0000b - Reserved 010_0001b - CTimer0_MAT1 input is selected 010_0010b - CTimer0_MAT2 input is selected 010_0011b - CTimer0_MAT3 input is selected 010_0100b - CTimer1_MAT1 input is selected 010_0101b - CTimer1_MAT2 input is selected 010_0110b - CTimer1_MAT3 input is selected 010_0111b - QDC0_CMP_FLAG0 is selected 010_1000b - QDC0_CMP_FLAG1 input is selected 010_1001b - QDC0_CMP_FLAG2 input is selected 010_1010b - QDC0_CMP_FLAG3 input is selected 010_1011b - QDC0_POS_MATCH0 input is selected 010_1100b - PWM0_SM0_MUX_TRIG0 input is selected 010_1101b - PWM0_SM1_MUX_TRIG0 input is selected 010_1110b - PWM0_SM2_MUX_TRIG0 input is selected 010_1111b - Reserved 011_0000b - LPI2C0 Master End of Packet input is selected 011_0001b - LPI2C0 Slave End of Packet input is selected 011_0010b - LPI2C1 Master End of Packet input is selected 011_0011b - LPI2C1 Slave End of Packet input is selected 011_0100b - LPSPI0 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - LPSPI0 Received Data Word input is selected 011_0110b - LPSPI1 End of Frame input is selected 011_0111b - LPSPI1 Received Data Word input is selected 011_1000b - LPUART0 Received Data Word input is selected 011_1001b - LPUART0 Transmitted Data Word input is selected 011_1010b - LPUART0 Receive Line Idle input is selected 011_1011b - LPUART1 Received Data Word input is selected 011_1100b - LPUART1 Transmitted Data Word input is selected 011_1101b - LPUART1 Receive Line Idle input is selected 011_1110b - LPUART2 Received Data Word input is selected 011_1111b - LPUART2 Transmitted Data Word input is selected 100_0000b - LPUART2 Receive Line Idle input is selected 100_0001b - LPUART3 Received Data Word input is selected 100_0010b - LPUART3 Transmitted Data Word input is selected 100_0011b - LPUART3 Receive Line Idle input is selected 100_0100b - LPUART4 Received Data Word input is selected 100_0101b - LPUART4 Transmitted Data Word input is selected 100_0110b - LPUART4 Receive Line Idle input is selected 100_0111b - AOI1_OUT0 input is selected 100_1000b - AOI1_OUT1 input is selected 100_1001b - AOI1_OUT2 input is selected 100_1010b - AOI1_OUT3 input is selected 100_1011b - ADC1_tcomp[0] input is selected 100_1100b - ADC1_tcomp[1] input is selected 100_1101b - ADC1_tcomp[2] input is selected 100_1110b - ADC1_tcomp[3] input is selected 100_1111b - CTimer2_MAT1 input is selected 101_0000b - CTimer2_MAT2 input is selected 101_0001b - CTimer2_MAT3 input is selected 101_0010b - CTimer3_MAT1 input is selected 101_0011b - CTimer3_MAT2 input is selected 101_0100b - CTimer3_MAT3 input is selected 101_0101b - QDC1_CMP_FLAG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101_0110b - QDC1_CMP_FLAG1 input is selected 101_0111b - QDC1_CMP_FLAG2 input is selected 101_1000b - QDC1_CMP_FLAG3 input is selected 101_1001b - QDC1_POS_MATCH0 input is selected 101_1010b - PWM1_SM0_MUX_TRIG0 input is selected 101_1011b - PWM1_SM1_MUX_TRIG0 input is selected 101_1100b - PWM1_SM2_MUX_TRIG0 input is selected 101_1101b - Reserved 101_1110b - LPI2C2 Master End of Packet input is selected 101_1111b - LPI2C2 Slave End of Packet input is selected 110_0000b - LPI2C3 Master End of Packet input is selected 110_0001b - LPI2C3 Slave End of Packet input is selected All other values are reserved.

13.5.1.14 AOI1 trigger input connections 0 (AOI1_INPUT0 - AOI1_INPUT15)

Offset

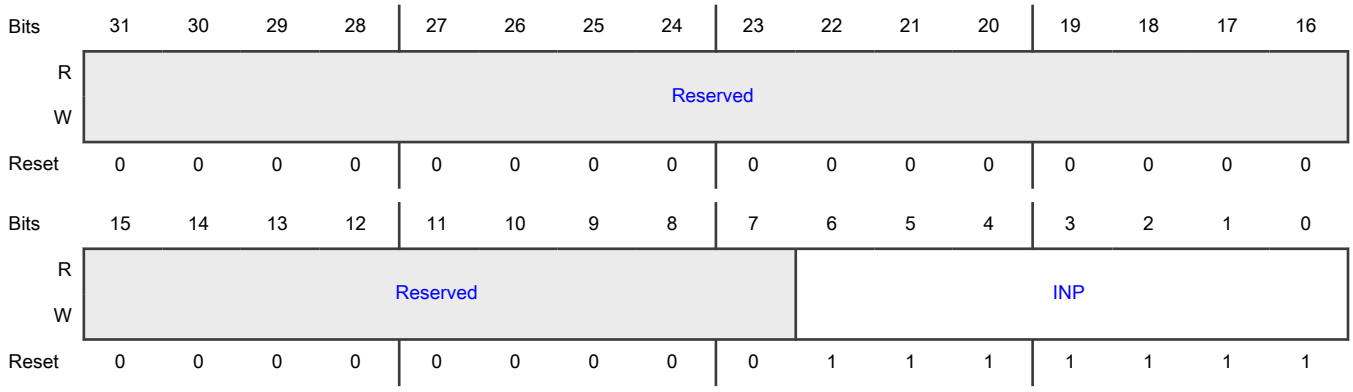
For a = 0 to 15:

Register	Offset
AOI1_INPUTa	200h + (a × 4h)

Function

This register is used to select the AOIO1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	AOI0 trigger input connections 000_0000b - Reserved 000_0001b - ADC0_tcomp[0] input is selected 000_0010b - ADC0_tcomp[1] input is selected 000_0011b - ADC0_tcomp[2] input is selected 000_0100b - ADC0_tcomp[3] input is selected 000_0101b - CMP0_OUT input is selected 000_0110b - CMP1_OUT input is selected 000_0111b - Reserved 000_1000b - CTimer0_MAT0 input is selected 000_1001b - CTimer0_MAT1 input is selected 000_1010b - CTimer0_MAT2 input is selected 000_1011b - CTimer0_MAT3 input is selected 000_1100b - CTimer1_MAT0 000_1101b - CTimer1_MAT1 input is selected 000_1110b - CTimer1_MAT2 input is selected 000_1111b - CTimer1_MAT3 input is selected 001_0000b - CTimer2_MAT0 input is selected 001_0001b - CTimer2_MAT1 input is selected 001_0010b - CTimer2_MAT2 input is selected 001_0011b - CTimer2_MAT3 input is selected 001_0100b - LPTMR0 input is selected 001_0101b - Reserved 001_0110b - QDC0_CMP_FLAG0 input is selected 001_0111b - QDC0_CMP_FLAG1 input is selected 001_1000b - QDC0_CMP_FLAG2 input is selected 001_1001b - QDC0_CMP_FLAG3 input is selected 001_1010b - QDC0_POS_MATCH input is selected 001_1011b - PWM0_SM0_MUX_TRIG0 0 input is selected 001_1100b - PWM0_SM0_MUX_TRIG1 input is selected 001_1101b - PWM0_SM1_MUX_TRIG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_1110b - PWM0_SM1_MUX_TRIG1 input is selected 001_1111b - PWM0_SM2_MUX_TRIG0 input is selected 010_0000b - PWM0_SM2_MUX_TRIG1 input is selected 010_0001b - Reserved 010_0010b - Reserved 010_0011b - TRIG_IN0 input is selected 010_0100b - TRIG_IN1 input is selected 010_0101b - TRIG_IN2 input is selected 010_0110b - TRIG_IN3 input is selected 010_0111b - TRIG_IN4 input is selected 010_1000b - TRIG_IN5 input is selected 010_1001b - TRIG_IN6 input is selected 010_1010b - TRIG_IN7 input is selected 010_1011b - TRIG_IN8 input is selected 010_1100b - TRIG_IN9 input is selected 010_1101b - TRIG_IN10 input is selected 010_1110b - TRIG_IN11 input is selected 010_1111b - GPIO0 Pin Event Trig 0 input is selected 011_0000b - GPIO1 Pin Event Trig 0 input is selected 011_0001b - GPIO2 Pin Event Trig 0 input is selected 011_0010b - GPIO3 Pin Event Trig 0 input is selected 011_0011b - GPIO4 Pin Event Trig 0 input is selected 011_0100b - ADC1_tcomp[0] input is selected 011_0101b - ADC1_tcomp[1] input is selected 011_0110b - ADC1_tcomp[2] input is selected 011_0111b - ADC1_tcomp[3] input is selected 011_1000b - CTimer3_MAT0 input is selected 011_1001b - CTimer3_MAT1 input is selected 011_1010b - CTimer3_MAT2 input is selected 011_1011b - CTimer3_MAT3 input is selected 011_1100b - CTimer4_MAT0 input is selected 011_1101b - CTimer4_MAT1 input is selected 011_1110b - CTimer4_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_1111b - CTimer4_MAT3 input is selected 100_0000b - FlexIO CH0 input is selected 100_0001b - FlexIO CH1 input is selected 100_0010b - FlexIO CH2 input is selected 100_0011b - FlexIO CH3 input is selected 100_0100b - QDC1_CMP_FLAG0 input is selected 100_0101b - QDC1_CMP_FLAG1 input is selected 100_0110b - QDC1_CMP_FLAG2 input is selected 100_0111b - QDC1_CMP_FLAG3 input is selected 100_1000b - QDC1_POS_MATCH0 input is selected 100_1001b - PWM1_SM0_MUX_TRIG0 input is selected 100_1010b - PWM1_SM0_MUX_TRIG1 input is selected 100_1011b - PWM1_SM1_MUX_TRIG0 input is selected 100_1100b - PWM1_SM1_MUX_TRIG1 input is selected 100_1101b - PWM1_SM2_MUX_TRIG0 input is selected 100_1110b - PWM1_SM2_MUX_TRIG1 input is selected 100_1111b - Reserved 101_0000b - Reserved All other values are reserved.

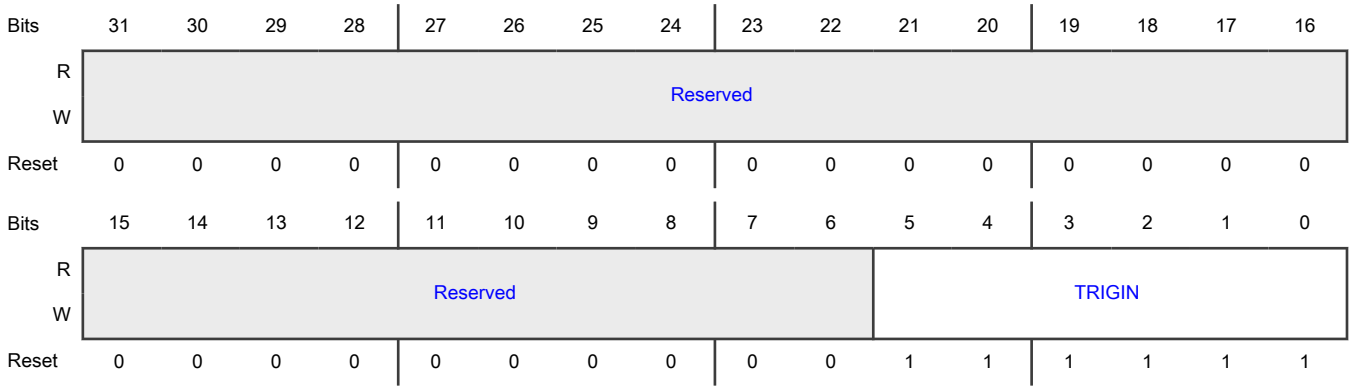
13.5.1.15 CMP0 input connections (CMP0_TRIG)

Offset

Register	Offset
CMP0_TRIG	260h

Function
This register selects the CMP0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	CMP0 input trigger 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP1_OUT input is selected 00_0111b - Reserved 00_1000b - CTimer0_MAT0 input is selected 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer1_MAT0 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer2_MAT0 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - LPTMR0 input is selected 00_1111b - Reserved 01_0000b - QDC0_POS_MATCH0 01_0001b - PWM0_SM0_MUX_TRIG0 input is selected 01_0010b - PWM0_SM0_MUX_TRIG1 input is selected 01_0011b - PWM0_SM1_MUX_TRIG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - PWM0_SM1_MUX_TRIG1 input is selected 01_0101b - PWM0_SM2_MUX_TRIG0 input is selected 01_0110b - PWM0_SM2_MUX_TRIG1 input is selected 01_0111b - Reserved 01_1000b - Reserved 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected 01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - Reserved 10_0100b - Reserved 10_0101b - Reserved 10_0110b - Reserved 10_0111b - CTimer3_MAT0 10_1000b - CTimer3_MAT1 10_1001b - CTimer4_MAT0 input is selected 10_1010b - CTimer4_MAT1 input is selected 10_1011b - Reserved 10_1100b - Reserved 10_1101b - Reserved 10_1110b - Reserved 10_1111b - QDC1_POS_MATCH0 input is selected 11_0000b - PWM1_SM0_MUX_TRIG0 input is selected 11_0001b - PWM1_SM0_MUX_TRIG1 input is selected 11_0010b - PWM1_SM1_MUX_TRIG0 input is selected 11_0011b - PWM1_SM1_MUX_TRIG1 input is selected 11_0100b - PWM1_SM2_MUX_TRIG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - PWM1_SM2_MUX_TRIG1 input is selected 11_0110b - Reserved 11_0111b - Reserved All other values are reserved.

13.5.1.16 ADC Trigger input connections (ADC0_TRIG0 - ADC0_TRIG3)

Offset

Register	Offset
ADC0_TRIG0	280h
ADC0_TRIG1	284h
ADC0_TRIG2	288h
ADC0_TRIG3	28Ch

Function

This register selects the ADC trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0	ADC0 trigger inputs

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIGIN	00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected 00_1011b - CTimer1_MAT0 input is selected 00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - QDC0_POS_MATCH0 input is selected 01_0010b - PWM0_SM0_OUT_TRIG0 input is selected 01_0011b - PWM0_SM0_OUT_TRIG1 input is selected 01_0100b - PWM0_SM1_OUT_TRIG0 input is selected 01_0101b - PWM0_SM1_OUT_TRIG1 input is selected 01_0110b - PWM0_SM2_OUT_TRIG0 input is selected 01_0111b - PWM0_SM2_OUT_TRIG1 input is selected 01_1000b - Reserved 01_1001b - Reserved 01_1010b - GPIO0 Pin Event Trig 0 input is selected 01_1011b - GPIO1 Pin Event Trig 0 input is selected 01_1100b - GPIO2 Pin Event Trig 0 input is selected 01_1101b - GPIO3 Pin Event Trig 0 input is selected 01_1110b - GPIO4 Pin Event Trig 0 input is selected 01_1111b - WUU 10_0000b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_0001b - AOI1_OUT0 input is selected 10_0010b - AOI1_OUT1 input is selected 10_0011b - AOI1_OUT2 input is selected 10_0100b - AOI1_OUT3 input is selected 10_0101b - ADC1_tcomp[0] input is selected 10_0110b - ADC1_tcomp[1] input is selected 10_0111b - ADC1_tcomp[2] input is selected 10_1000b - ADC1_tcomp[3] input is selected 10_1001b - CTimer3_MAT0 input is selected 10_1010b - CTimer3_MAT1 input is selected 10_1011b - CTimer4_MAT0 input is selected 10_1100b - CTimer4_MAT1 input is selected 10_1101b - FlexIO CH0 input is selected 10_1110b - FlexIO CH1 input is selected 10_1111b - FlexIO CH2 input is selected 11_0000b - FlexIO CH3 input is selected 11_0001b - QDC1_POS_MATCH0 input is selected 11_0010b - PWM1_SM0_MUX_TRIG0 input is selected 11_0011b - PWM1_SM0_MUX_TRIG1 input is selected 11_0100b - PWM1_SM1_MUX_TRIG0 input is selected 11_0101b - PWM1_SM1_MUX_TRIG1 input is selected 11_0110b - PWM1_SM2_MUX_TRIG0 input is selected 11_0111b - PWM1_SM2_MUX_TRIG1 input is selected 11_1000b - Reserved 11_1001b - Reserved All other values are reserved.

13.5.1.17 ADC Trigger input connections (ADC1_TRIG0 - ADC1_TRIG3)

Offset

Register	Offset
ADC1_TRIG0	2C0h
ADC1_TRIG1	2C4h

Table continues on the next page...

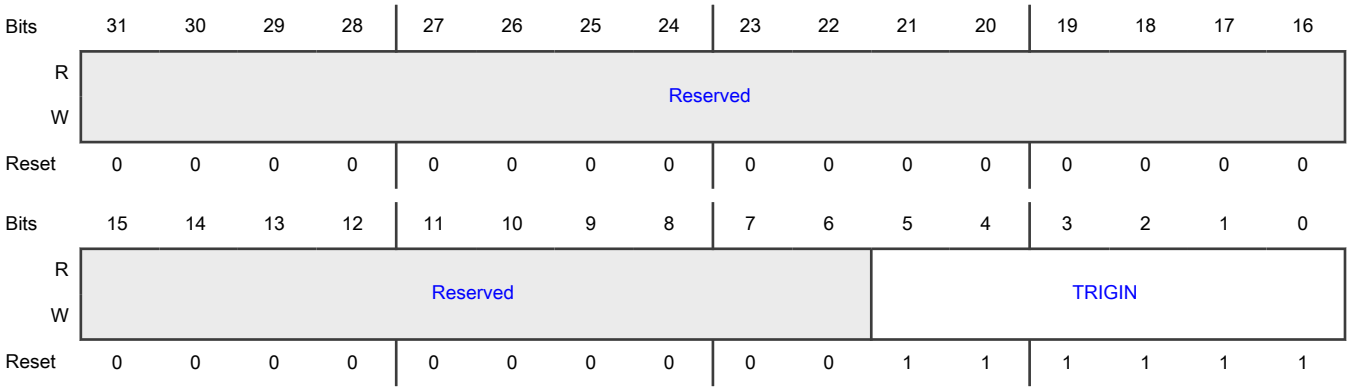
Table continued from the previous page...

Register	Offset
ADC1_TRIG2	2C8h
ADC1_TRIG3	2CCh

Function

This register selects the ADC trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	ADC1 trigger inputs 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected 00_1011b - CTimer1_MAT0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - QDC0_POS_MATCH0 input is selected 01_0010b - PWM0_SM0_OUT_TRIG0 input is selected 01_0011b - PWM0_SM0_OUT_TRIG1 input is selected 01_0100b - PWM0_SM1_OUT_TRIG0 input is selected 01_0101b - PWM0_SM1_OUT_TRIG1 input is selected 01_0110b - PWM0_SM2_OUT_TRIG0 input is selected 01_0111b - PWM0_SM2_OUT_TRIG1 input is selected 01_1000b - Reserved 01_1001b - Reserved 01_1010b - GPIO0 Pin Event Trig 0 input is selected 01_1011b - GPIO1 Pin Event Trig 0 input is selected 01_1100b - GPIO2 Pin Event Trig 0 input is selected 01_1101b - GPIO3 Pin Event Trig 0 input is selected 01_1110b - GPIO4 Pin Event Trig 0 input is selected 01_1111b - WUU 10_0000b - Reserved 10_0001b - AOI1_OUT0 input is selected 10_0010b - AOI1_OUT1 input is selected 10_0011b - AOI1_OUT2 input is selected 10_0100b - AOI1_OUT3 input is selected 10_0101b - ADC0_tcomp[0] input is selected 10_0110b - ADC0_tcomp[1] input is selected 10_0111b - ADC0_tcomp[2] input is selected 10_1000b - ADC0_tcomp[3] input is selected 10_1001b - CTimer3_MAT0 input is selected 10_1010b - CTimer3_MAT1 input is selected 10_1011b - CTimer4_MAT0 input is selected 10_1100b - CTimer4_MAT1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1101b - FlexIO CH0 input is selected 10_1110b - FlexIO CH1 input is selected 10_1111b - FlexIO CH2 input is selected 11_0000b - FlexIO CH3 input is selected 11_0001b - QDC1_POS_MATCH0 input is selected 11_0010b - PWM1_SM0_MUX_TRIG0 input is selected 11_0011b - PWM1_SM0_MUX_TRIG1 input is selected 11_0100b - PWM1_SM1_MUX_TRIG0 input is selected 11_0101b - PWM1_SM1_MUX_TRIG1 input is selected 11_0110b - PWM1_SM2_MUX_TRIG0 input is selected 11_0111b - PWM1_SM2_MUX_TRIG1 input is selected 11_1000b - Reserved 11_1001b - Reserved All other values are reserved.

13.5.1.18 This register selects the DAC0 trigger inputs. (DAC0_TRIG)

Offset

Register	Offset
DAC0_TRIG	300h

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	DAC0 trigger input 00_0000b - Reserved 00_0001b - ARM_TXEV 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected 00_1011b - CTimer1_MAT0 input is selected 00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - Reserved 01_0010b - Reserved 01_0011b - Reserved 01_0100b - Reserved 01_0101b - Reserved 01_0110b - Reserved 01_0111b - Reserved 01_1000b - Reserved 01_1001b - Reserved 01_1010b - GPIO0 Pin Event Trig 0 input is selected 01_1011b - GPIO1 Pin Event Trig 0 input is selected 01_1100b - GPIO2 Pin Event Trig 0 input is selected 01_1101b - GPIO3 Pin Event Trig 0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1110b - GPIO4 Pin Event Trig 0 input is selected 01_1111b - WUU input is selected 10_0000b - Reserved 10_0001b - AOI1_OUT0 input is selected 10_0010b - AOI1_OUT1 input is selected 10_0011b - AOI1_OUT2 input is selected 10_0100b - AOI1_OUT3 input is selected 10_0101b - ADC0_tcomp[0] input is selected 10_0110b - ADC0_tcomp[1] input is selected 10_0111b - ADC1_tcomp[0] input is selected 10_1000b - ADC1_tcomp[1] input is selected 10_1001b - CTimer3_MAT0 input is selected 10_1010b - CTimer3_MAT1 input is selected 10_1011b - CTimer4_MAT0 input is selected 10_1100b - CTimer4_MAT1 input is selected 10_1101b - Reserved 10_1110b - Reserved 10_1111b - Reserved 11_0000b - Reserved 11_0001b - Reserved 11_0010b - Reserved 11_0011b - Reserved 11_0100b - Reserved 11_0101b - Reserved 11_0110b - Reserved 11_0111b - Reserved 11_1000b - Reserved 11_1001b - Reserved All other values are reserved.

13.5.1.19 QDC0 Trigger Input Connections (QDC0_TRIG)

Offset

Register	Offset
QDC0_TRIG	360h

Function
This register selects the QDC0 trigger inputs.

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								INP							
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 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected 001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected 011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.20 QDC0 Trigger Input Connections (QDC0_HOME)

Offset

Register	Offset
QDC0_HOME	364h

Function

This register selects the QDC0 trigger inputs.

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								INP							
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 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected 001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected 011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

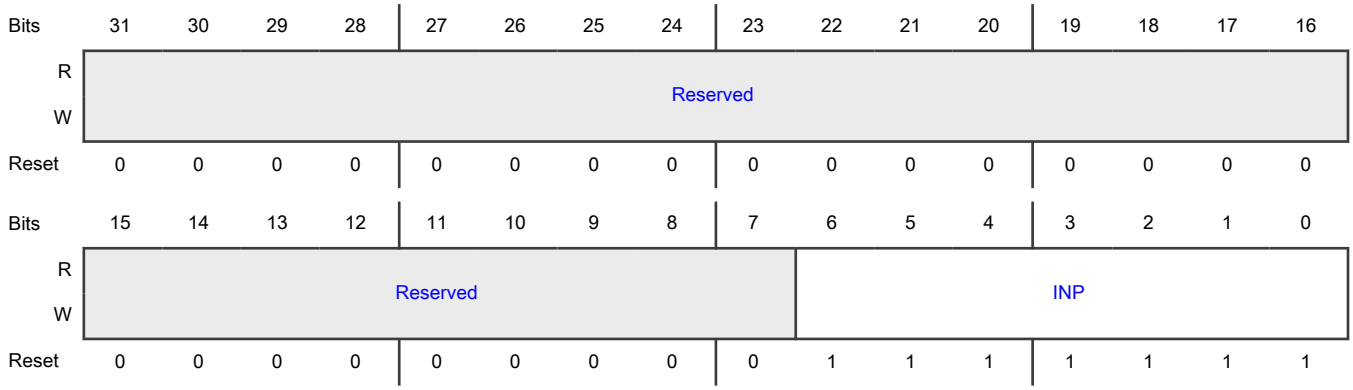
13.5.1.21 QDC0 Trigger Input Connections (QDC0_INDEX)

Offset

Register	Offset
QDC0_INDEX	368h

Function
This register selects the QDC0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.22 QDC0 Trigger Input Connections (QDC0_PHASEB)

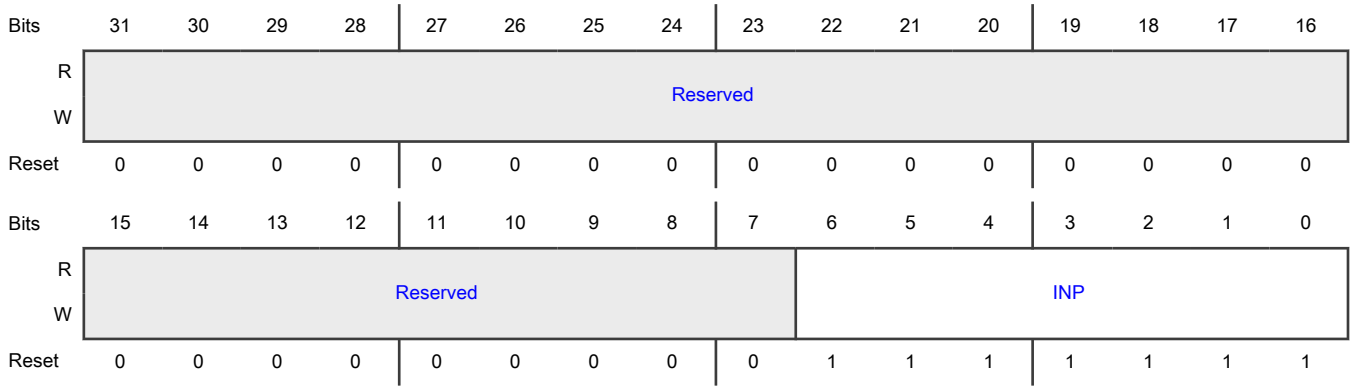
Offset

Register	Offset
QDC0_PHASEB	36Ch

Function

This register selects the QDC0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

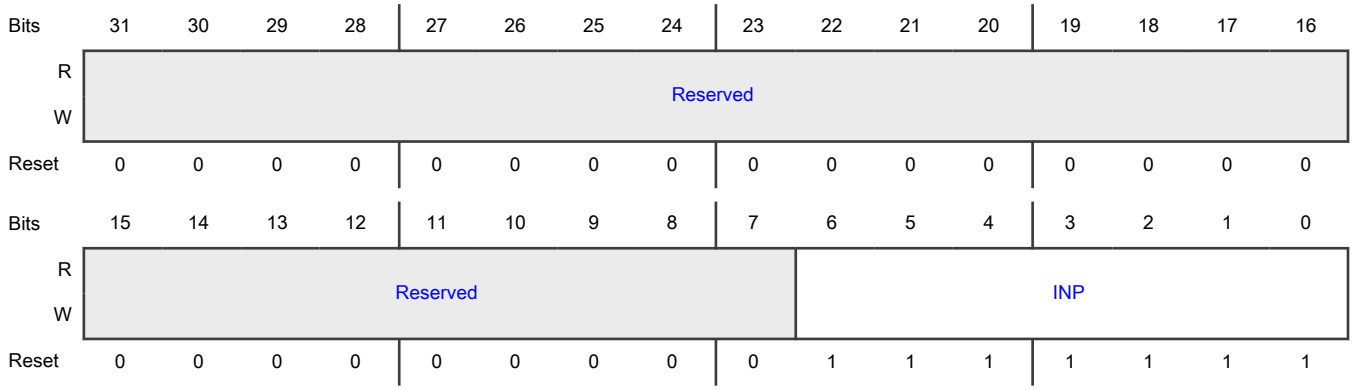
13.5.1.23 QDC0 Trigger Input Connections (QDC0_PHASEA)

Offset

Register	Offset
QDC0_PHASEA	370h

Function
This register selects the QDC0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.24 QDC0 Trigger Input Connections (QDC0_ICAP1)

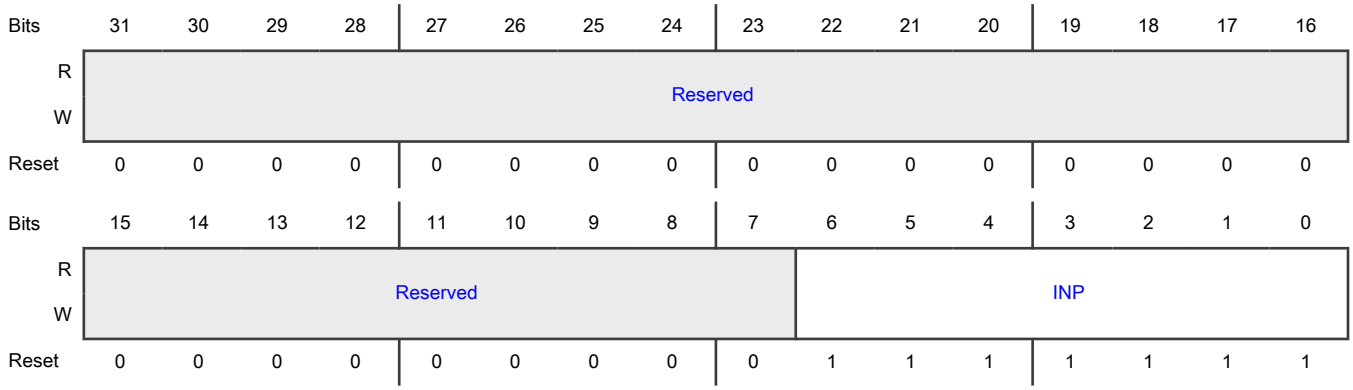
Offset

Register	Offset
QDC0_ICAP1	374h

Function

This register selects the QDC0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.25 QDC0 Trigger Input Connections (QDC0_ICAP2)

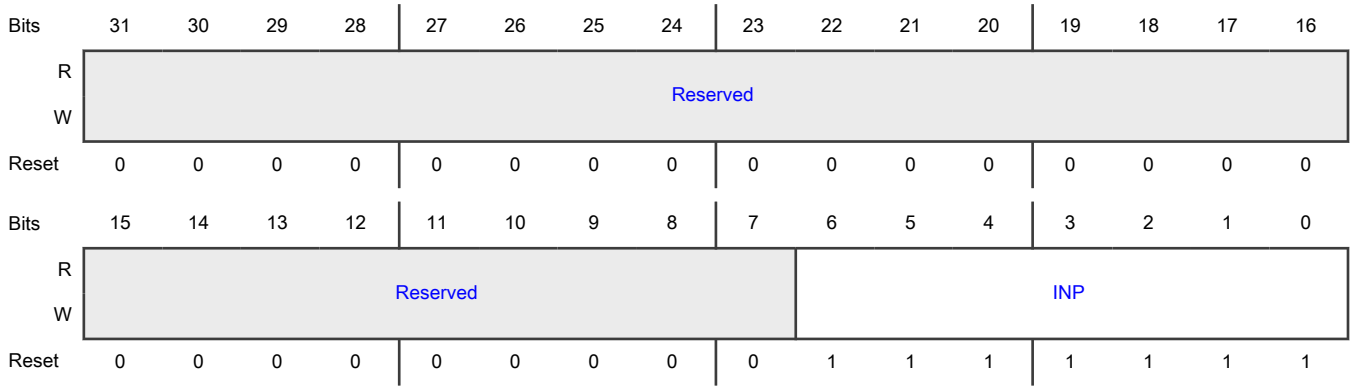
Offset

Register	Offset
QDC0_ICAP2	378h

Function

This register selects the QDC0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.26 QDC0 Trigger Input Connections (QDC0_ICAP3)

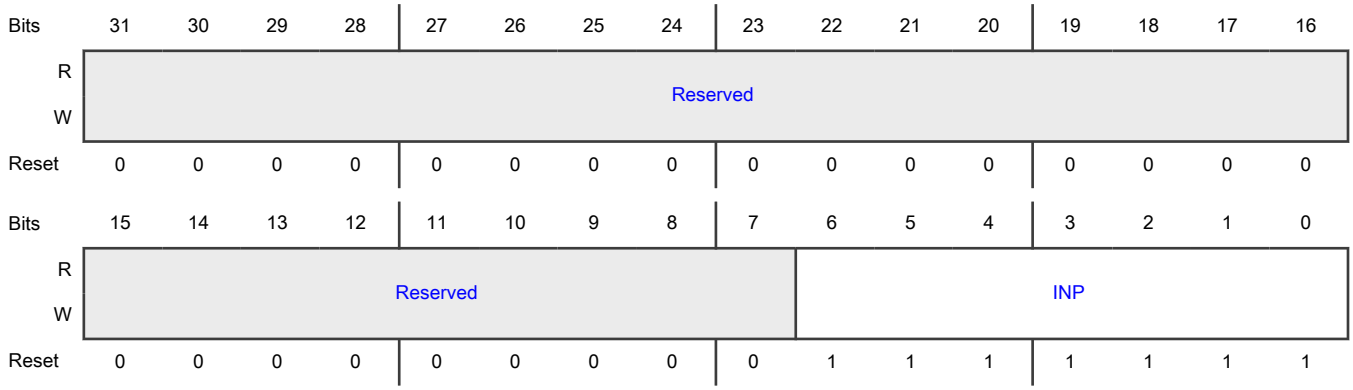
Offset

Register	Offset
QDC0_ICAP3	37Ch

Function

This register selects the QDC0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.27 QDC1 Trigger Input Connections (QDC1_TRIG)

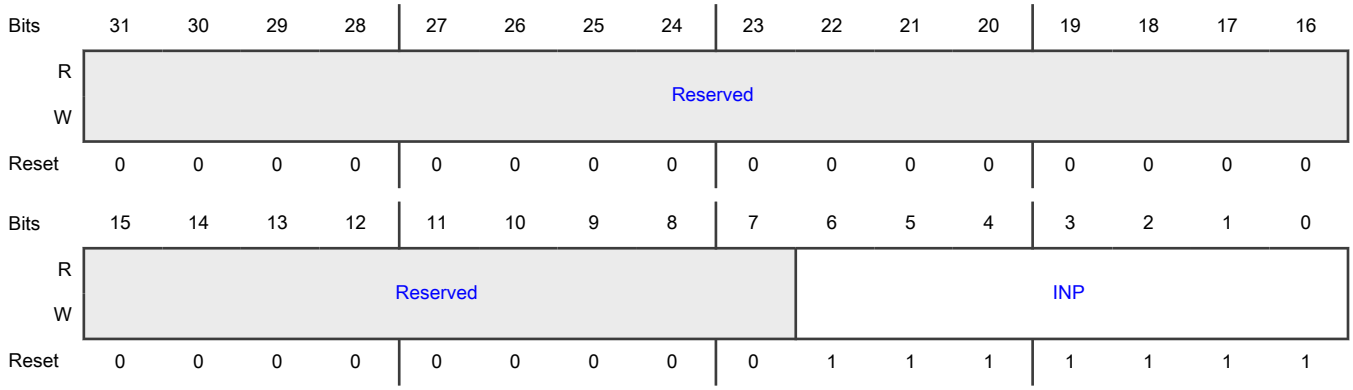
Offset

Register	Offset
QDC1_TRIG	380h

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.28 QDC1 Trigger Input Connections (QDC1_HOME)

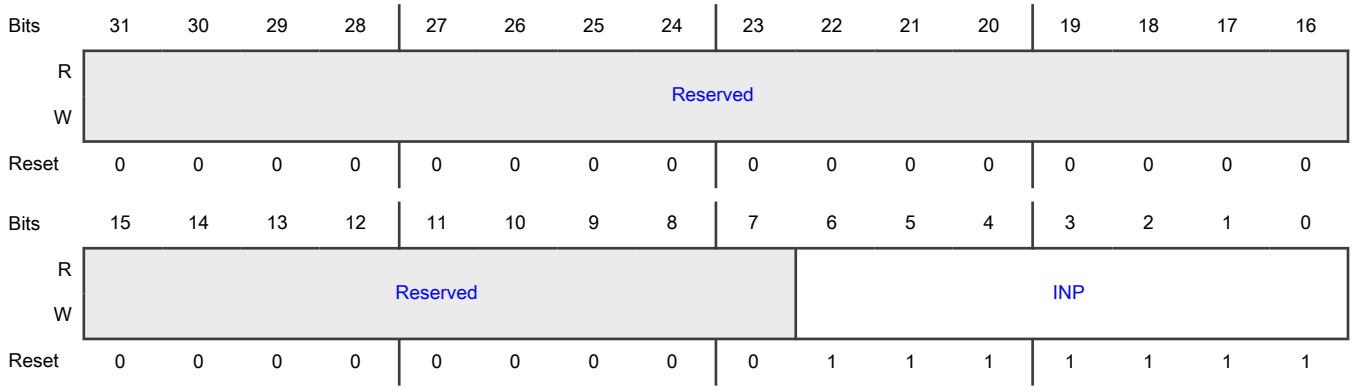
Offset

Register	Offset
QDC1_HOME	384h

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.29 QDC1 Trigger Input Connections (QDC1_INDEX)

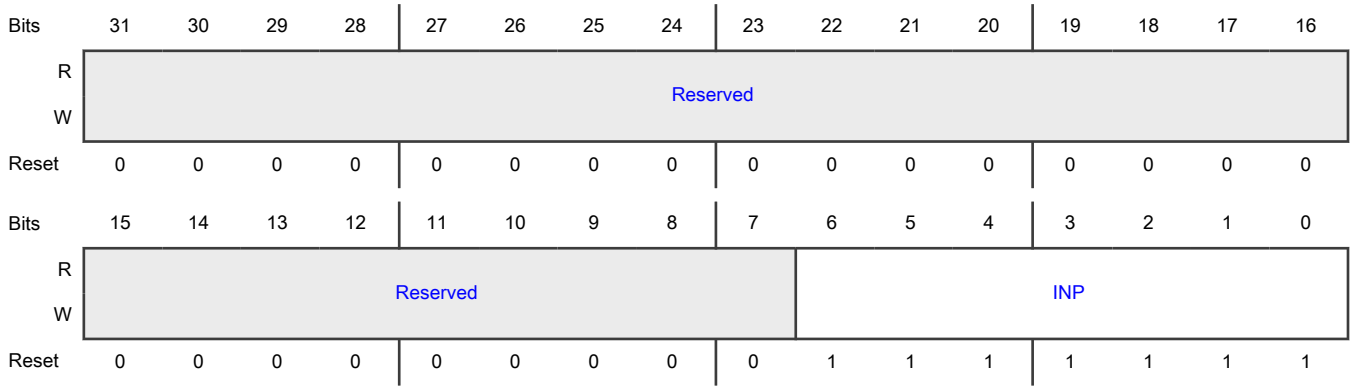
Offset

Register	Offset
QDC1_INDEX	388h

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.30 QDC1 Trigger Input Connections (QDC1_PHASEB)

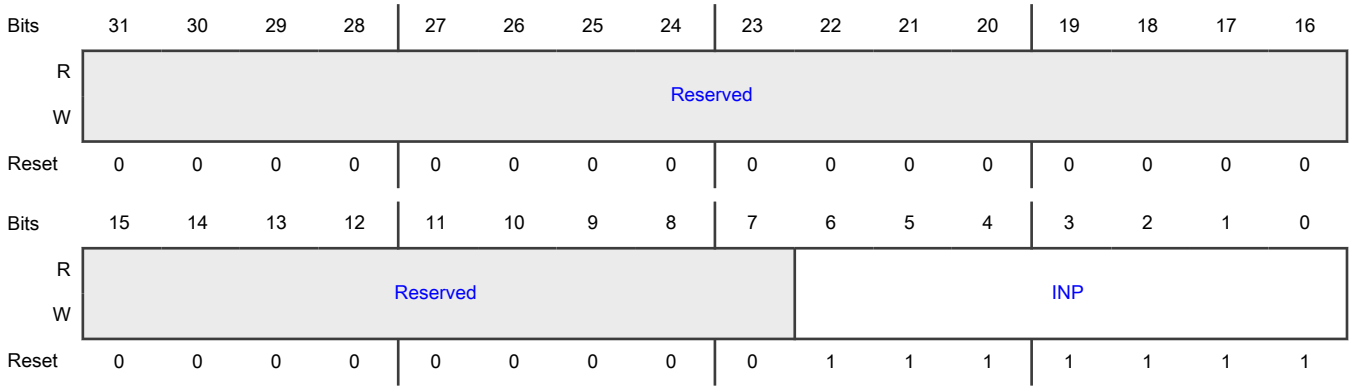
Offset

Register	Offset
QDC1_PHASEB	38Ch

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.31 QDC1 Trigger Input Connections (QDC1_PHASEA)

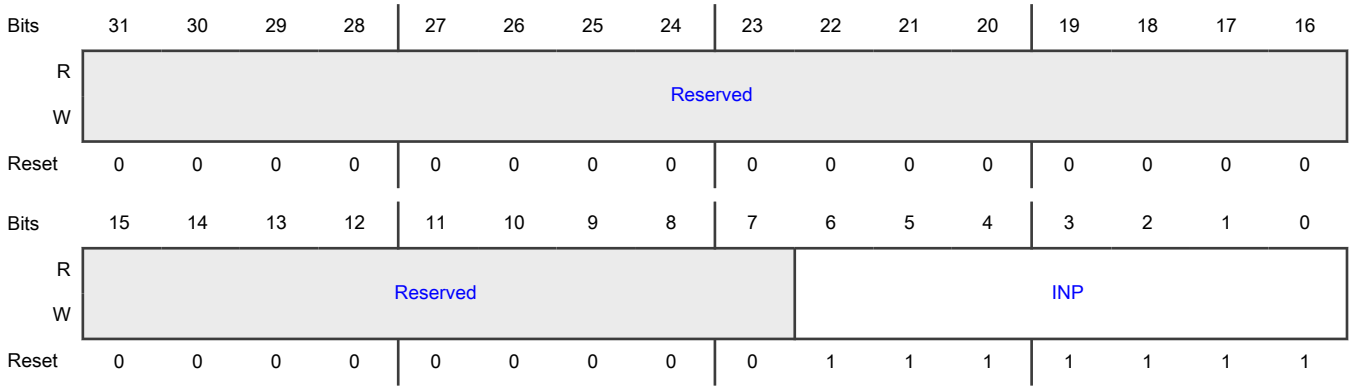
Offset

Register	Offset
QDC1_PHASEA	390h

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC0 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.32 QDC1 Trigger Input Connections (QDC1_ICAP1)

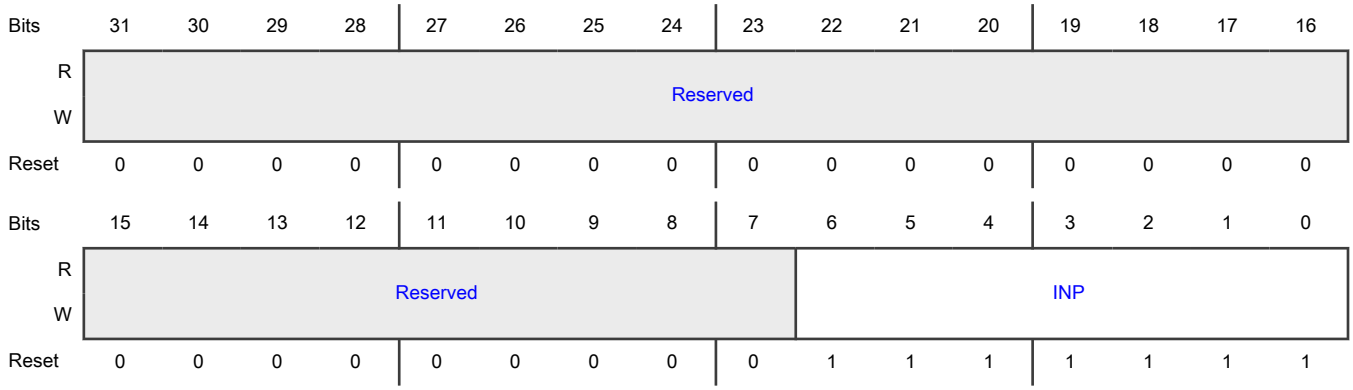
Offset

Register	Offset
QDC1_ICAP1	394h

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.33 QDC1 Trigger Input Connections (QDC1_ICAP2)

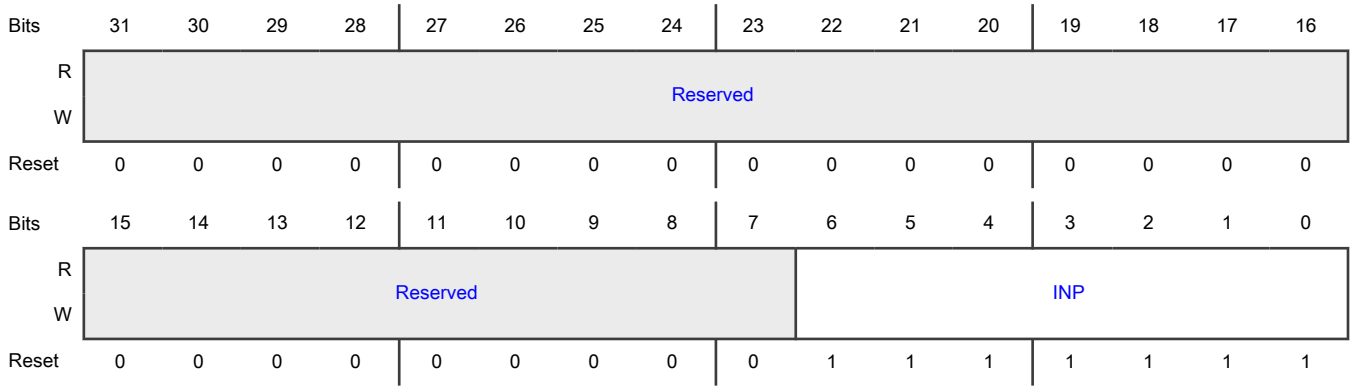
Offset

Register	Offset
QDC1_ICAP2	398h

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

13.5.1.34 QDC1 Trigger Input Connections (QDC1_ICAP3)

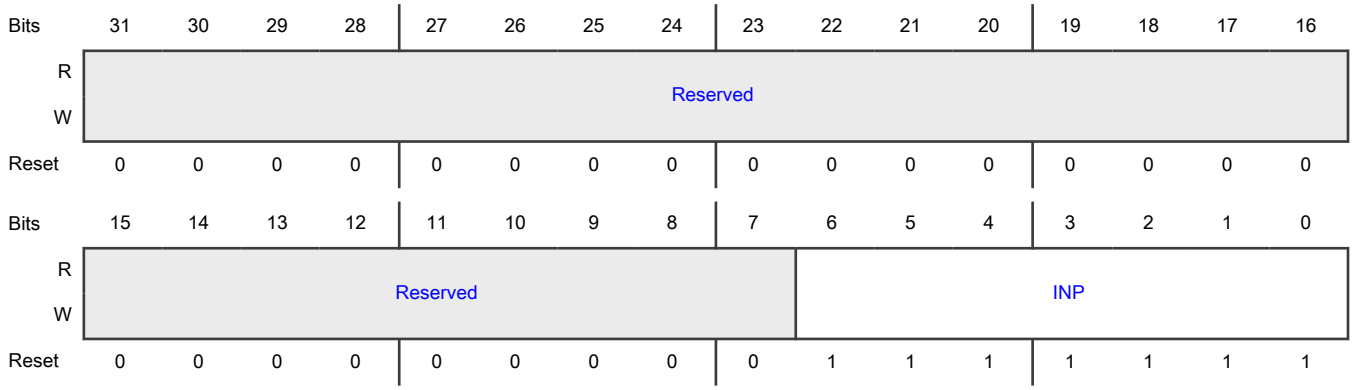
Offset

Register	Offset
QDC1_ICAP3	39Ch

Function

This register selects the QDC1 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	QDC1 input connections 000_0000b - Reserved 000_0001b - ARM_TXEV input is selected 000_0010b - AOI0_OUT0 input is selected 000_0011b - AOI0_OUT1 input is selected 000_0100b - AOI0_OUT2 input is selected 000_0101b - AOI0_OUT3 input is selected 000_0110b - CMP0_OUT input is selected 000_0111b - CMP1_OUT input is selected 000_1000b - Reserved 000_1001b - CTimer0_MAT2 input is selected 000_1010b - CTimer0_MAT3 000_1011b - CTimer1_MAT2 input is selected 000_1100b - CTimer1_MAT3 input is selected 000_1101b - CTimer2_MAT2 input is selected 000_1110b - CTimer2_MAT3 input is selected 000_1111b - Reserved 001_0000b - PWM0_SM0_MUX_TRIG0 input is selected 001_0001b - PWM0_SM0_MUX_TRIG1 input is selected 001_0010b - PWM0_SM1_MUX_TRIG0 input is selected 001_0011b - PWM0_SM1_MUX_TRIG1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_0100b - PWM0_SM2_MUX_TRIG0 input is selected 001_0101b - PWM0_SM2_MUX_TRIG1 input is selected 001_0110b - Reserved 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - TRIG_IN8 input is selected 010_0001b - TRIG_IN9 input is selected 010_0010b - TRIG_IN10 input is selected 010_0011b - TRIG_IN11 input is selected 010_0100b - GPIO0 Pin Event Trig 0 is selected 010_0101b - GPIO1 Pin Event Trig 0 input is selected 010_0110b - GPIO2 Pin Event Trig 0 input is selected 010_0111b - GPIO3 Pin Event Trig 0 input is selected 010_1000b - GPIO4 Pin Event Trig 0 input is selected 010_1001b - AOI1_OUT0 input is selected 010_1010b - AOI1_OUT1 input is selected 010_1011b - AOI1_OUT2 input is selected 010_1100b - AOI1_OUT3 input is selected 010_1101b - Reserved 010_1110b - Reserved 010_1111b - Reserved 011_0000b - Reserved 011_0001b - CTimer3_MAT2 input is selected 011_0010b - CTimer3_MAT3 input is selected 011_0011b - CTimer4_MAT2 input is selected 011_0100b - CTimer4_MAT3 End of Frame input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_0101b - Reserved 011_0110b - Reserved 011_0111b - Reserved 011_1000b - Reserved 011_1001b - Reserved 011_1010b - Reserved 011_1011b - Reserved 011_1100b - Reserved 011_1101b - Reserved 011_1110b - PWM1_SM0_OUT_TRIG0 input is selected 011_1111b - PWM1_SM0_OUT_TRIG1 input is selected 100_0000b - PWM1_SM1_OUT_TRIG0 input is selected 100_0001b - PWM1_SM1_OUT_TRIG1 input is selected 100_0010b - PWM1_SM2_OUT_TRIG0 input is selected 100_0011b - PWM1_SM2_OUT_TRIG1 input is selected 100_0100b - Reserved 100_0101b - Reserved All other values are reserved.

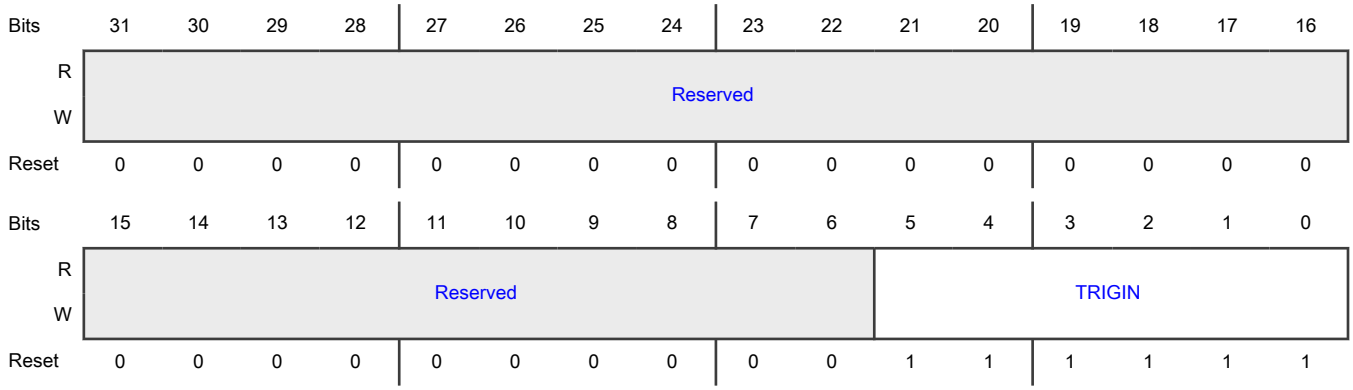
13.5.1.35 PWM0 input trigger connections (FlexPWM0_SM0_EXTA0)

Offset

Register	Offset
FlexPWM0_SM0_EXTA0	3A0h

Function
This register selects the PWM0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTA input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.36 PWM0 input trigger connections (FlexPWM0_SM0_EXTSYNC)

Offset

Register	Offset
FlexPWM0_SM0_EXTSYNC	3A4h

Function
This register selects the PWM0 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTSYNC input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.37 PWM0 input trigger connections (FlexPWM0_SM1_EXTA)

Offset

Register	Offset
FlexPWM0_SM1_EXTA	3A8h

Function
This register selects the PWM0 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTA input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

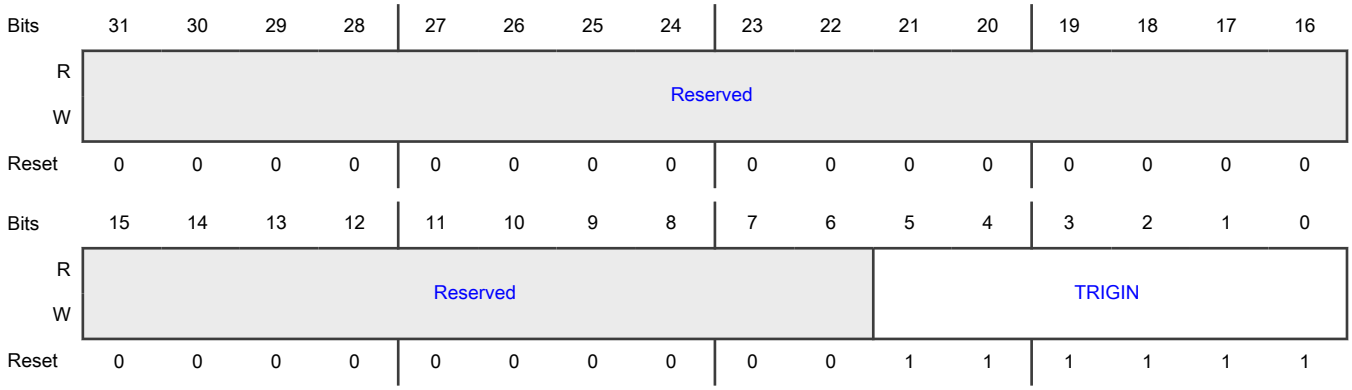
13.5.1.38 PWM0 input trigger connections (FlexPWM0_SM1_EXTSYNC)

Offset

Register	Offset
FlexPWM0_SM1_EXTSYNC	3ACh

Function
This register selects the PWM0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTSYNC input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.39 PWM0 input trigger connections (FlexPWM0_SM2_EXTA)

Offset

Register	Offset
FlexPWM0_SM2_EXTA	3B0h

Function

This register selects the PWM0 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 TRIGIN	EXTA input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.40 PWM0 input trigger connections (FlexPWM0_SM2_EXTSYNC)

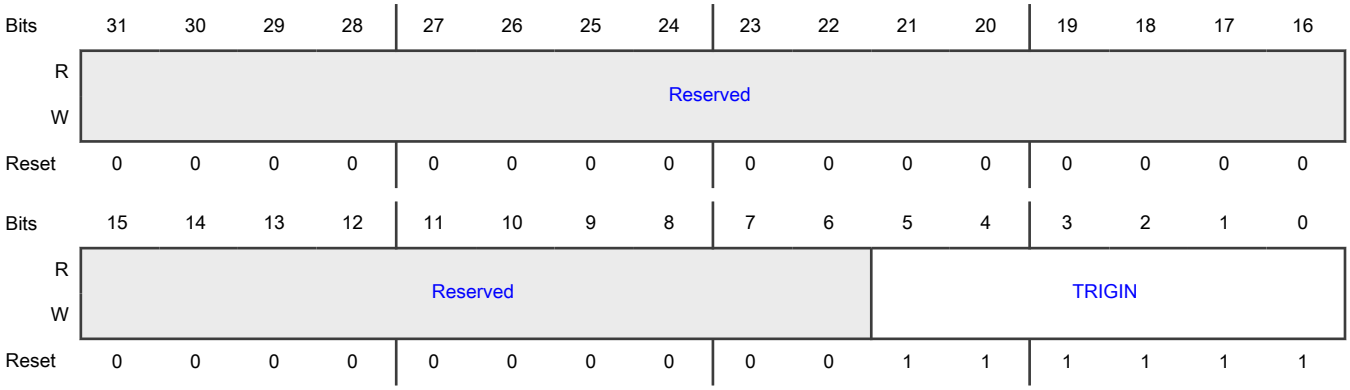
Offset

Register	Offset
FlexPWM0_SM2_EXTSYNC	3B4h

Function

This register selects the PWM0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTSYNC input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.41 PWM0 Fault Input Trigger Connections (FlexPWM0_FAULT0 - FlexPWM0_FAULT3)

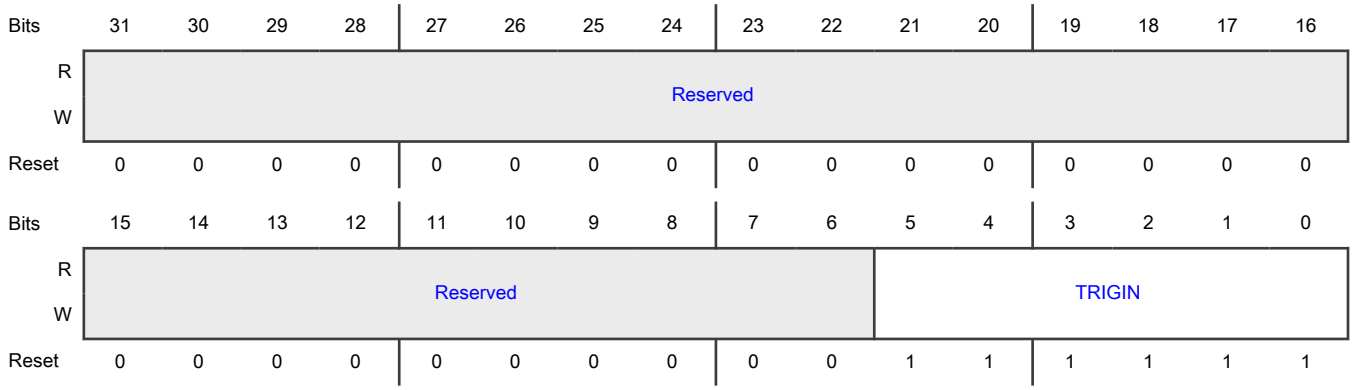
Offset

Register	Offset
FlexPWM0_FAULT0	3C0h
FlexPWM0_FAULT1	3C4h
FlexPWM0_FAULT2	3C8h
FlexPWM0_FAULT3	3CCh

Function

This register selects the PWM0 FAULT inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	FAULT input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.42 PWM0 input trigger connections (FlexPWM0_FORCE)

Offset

Register	Offset
FlexPWM0_FORCE	3D0h

Function

This register selects the PWM0 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 TRIGIN	Trigger input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM1_SM0_MUX_TRIG0 input is selected 11_0111b - PWM1_SM0_MUX_TRIG1 input is selected 11_1000b - PWM1_SM1_MUX_TRIG0 input is selected 11_1001b - PWM1_SM1_MUX_TRIG1 input is selected 11_1010b - PWM1_SM2_MUX_TRIG0 input is selected 11_1011b - PWM1_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.43 PWM1 input trigger connections (FlexPWM1_SM0_EXTA0)

Offset

Register	Offset
FlexPWM1_SM0_EXTA0	3E0h

Function
This register selects the PWM1 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTA input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

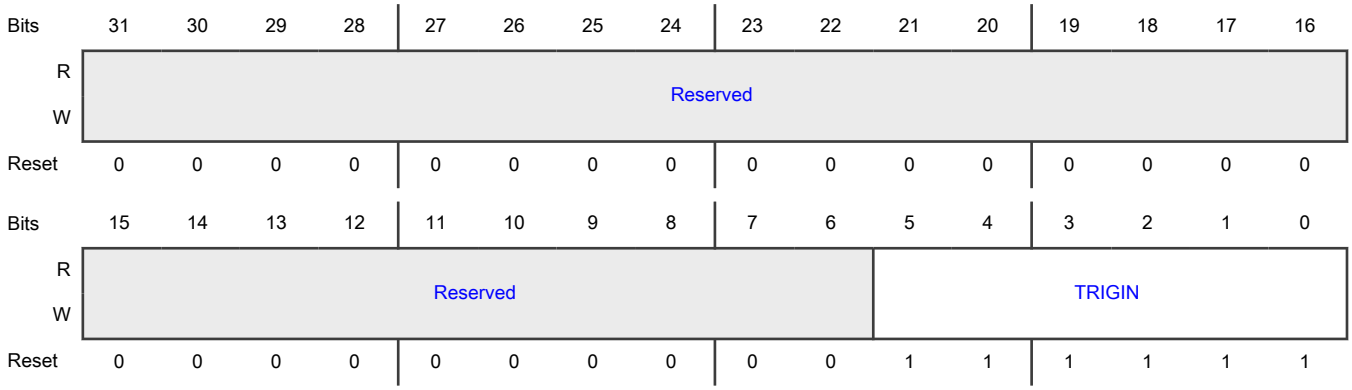
13.5.1.44 PWM1 input trigger connections (FlexPWM1_SM0_EXTSYNC)

Offset

Register	Offset
FlexPWM1_SM0_EXTSYNC	3E4h

Function
This register selects the PWM1 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTSYNC input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.45 PWM1 input trigger connections (FlexPWM1_SM1_EXTA)

Offset

Register	Offset
FlexPWM1_SM1_EXTA	3E8h

Function

This register selects the PWM1 trigger inputs.

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										TRIGIN					
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 TRIGIN	EXTA input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.46 PWM1 input trigger connections (FlexPWM1_SM1_EXTSYNC)

Offset

Register	Offset
FlexPWM1_SM1_EXTSYNC	3ECh

Function
This register selects the PWM1 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTSYNC input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

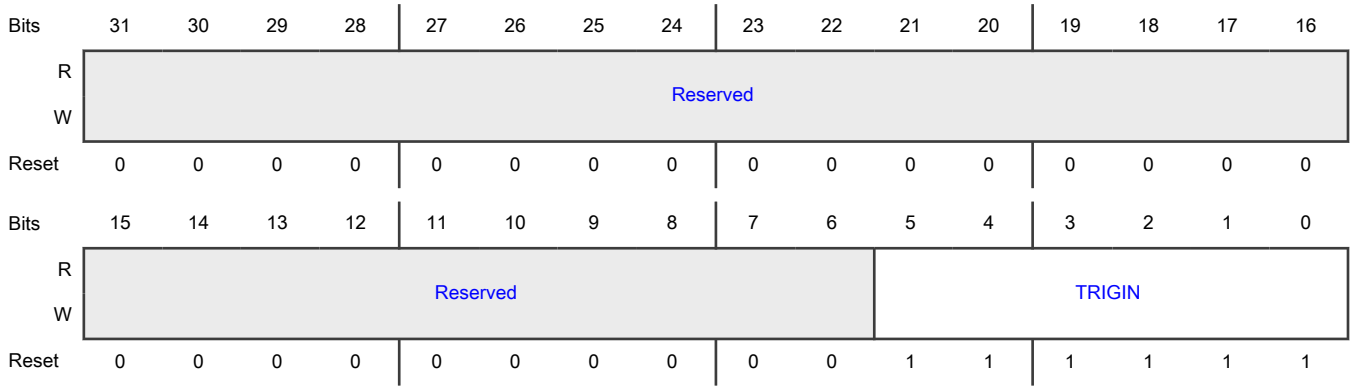
13.5.1.47 PWM1 input trigger connections (FlexPWM1_SM2_EXTA)

Offset

Register	Offset
FlexPWM1_SM2_EXTA	3F0h

Function
This register selects the PWM1 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTA input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.48 PWM1 input trigger connections (FlexPWM1_SM2_EXTSYNC)

Offset

Register	Offset
FlexPWM1_SM2_EXTSYNC	3F4h

Function
This register selects the PWM1 trigger inputs.

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										TRIGIN					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	EXTSYNC input connections for PWM0 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.49 PWM1 Fault Input Trigger Connections (FlexPWM1_FAULT0 - FlexPWM1_FAULT3)

Offset

Register	Offset
FlexPWM1_FAULT0	400h
FlexPWM1_FAULT1	404h
FlexPWM1_FAULT2	408h
FlexPWM1_FAULT3	40Ch

Function
This register selects the PWM1 FAULT inputs.

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										TRIGIN					
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	FAULT input connections for PWM1 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected 01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected 11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

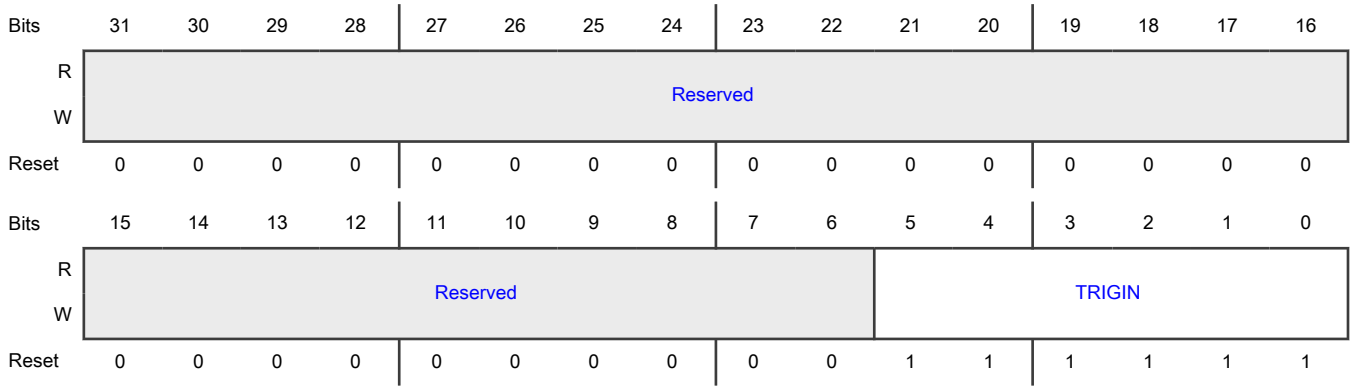
13.5.1.50 PWM1 input trigger connections (FlexPWM1_FORCE)

Offset

Register	Offset
FlexPWM1_FORCE	410h

Function
This register selects the PWM1 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	Trigger input connections for PWM1 00_0000b - Reserved 00_0001b - ARM_TXEV input is selected 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - QDC0_CMP_FLAG0 input is selected 01_0000b - QDC0_CMP_FLAG1 input is selected 01_0001b - QDC0_CMP_FLAG2 input is selected 01_0010b - QDC0_CMP_FLAG3 input is selected 01_0011b - QDC0_POS_MATCH0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN0 input is selected 01_0101b - TRIG_IN1 input is selected 01_0110b - TRIG_IN2 input is selected 01_0111b - TRIG_IN3 input is selected 01_1000b - TRIG_IN4 input is selected 01_1001b - TRIG_IN5 input is selected 01_1010b - TRIG_IN6 input is selected 01_1011b - TRIG_IN7 input is selected 01_1100b - TRIG_IN8 input is selected 01_1101b - TRIG_IN9 input is selected 01_1110b - TRIG_IN10 input is selected 01_1111b - TRIG_IN11 input is selected 10_0000b - GPIO0 Pin Event Trig 0 input is selected 10_0001b - GPIO1 Pin Event Trig 0 input is selected 10_0010b - GPIO2 Pin Event Trig 0 input is selected 10_0011b - GPIO3 Pin Event Trig 0 input is selected 10_0100b - GPIO4 Pin Event Trig 0 input is selected 10_0101b - AOI1_OUT0 input is selected 10_0110b - AOI1_OUT1 input is selected 10_0111b - AOI1_OUT2 input is selected 10_1000b - AOI1_OUT3 input is selected 10_1001b - Reserved 10_1010b - Reserved 10_1011b - Reserved 10_1100b - Reserved 10_1101b - CTimer3_MAT2 input is selected 10_1110b - CTimer3_MAT3 input is selected 10_1111b - CTimer4_MAT2 input is selected 11_0000b - CTimer4_MAT3 input is selected 11_0001b - QDC1_CMP_FLAG0 input is selected 11_0010b - QDC1_CMP_FLAG1 input is selected 11_0011b - QDC1_CMP_FLAG2 input is selected 11_0100b - QDC1_CMP_FLAG3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11_0101b - QDC1_POS_MATCH0 input is selected 11_0110b - PWM0_SM0_MUX_TRIG0 input is selected 11_0111b - PWM0_SM0_MUX_TRIG1 input is selected 11_1000b - PWM0_SM1_MUX_TRIG0 input is selected 11_1001b - PWM0_SM1_MUX_TRIG1 input is selected 11_1010b - PWM0_SM2_MUX_TRIG0 input is selected 11_1011b - PWM0_SM2_MUX_TRIG1 input is selected 11_1100b - Reserved 11_1101b - Reserved All other values are reserved.

13.5.1.51 PWM0 external clock trigger (PWM0_EXT_CLK)

Offset

Register	Offset
PWM0_EXT_CLK	420h

Function

PWM0 external clock trigger connections

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												TRIGIN			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Fields

Field	Function
31-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 TRIGIN	Trigger input connections for PWM 0000b - Reserved 0001b - clk_16k[1] input is selected 0010b - clk_in input is selected 0011b - AOI0_OUT0 input is selected 0100b - AOI0_OUT1 input is selected 0101b - EXTTRIG_IN0 input is selected 0110b - EXTTRIG_IN7 input is selected 0111b - AOI1_OUT0 input is selected 1000b - AOI1_OUT1 input is selected All other values are reserved.

13.5.1.52 PWM1 external clock trigger (PWM1_EXT_CLK)

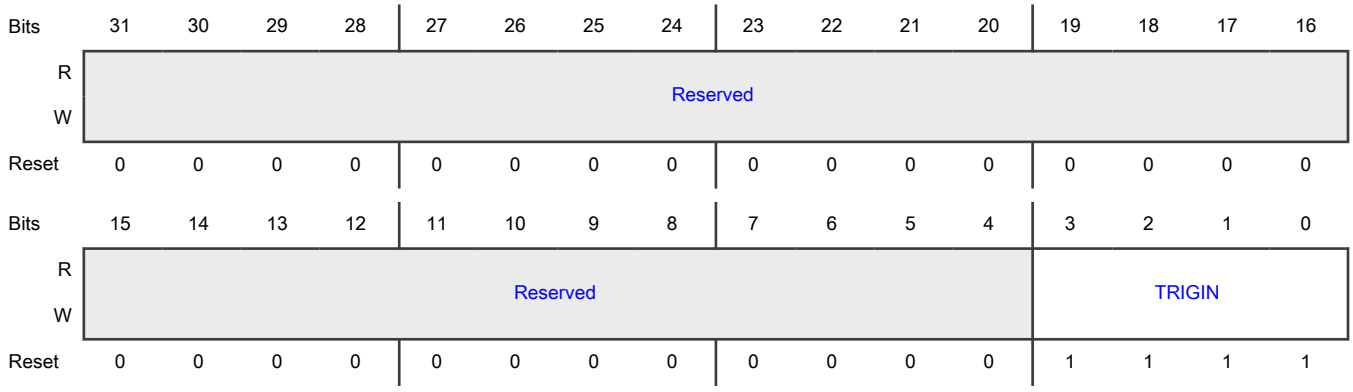
Offset

Register	Offset
PWM1_EXT_CLK	424h

Function

PWM1 external clock trigger connections

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TRIGIN	Trigger input connections for PWM 0000b - Reserved 0001b - clk_16k[1] input is selected 0010b - clk_in input is selected 0011b - AOI0_OUT0 input is selected 0100b - AOI0_OUT1 input is selected 0101b - EXTTRIG_IN0 input is selected 0110b - EXTTRIG_IN7 input is selected 0111b - AOI1_OUT0 input is selected 1000b - AOI1_OUT1 input is selected All other values are reserved.

13.5.1.53 AOI0 trigger input connections 0 (AOI0_INPUT0 - AOI0_INPUT15)

Offset

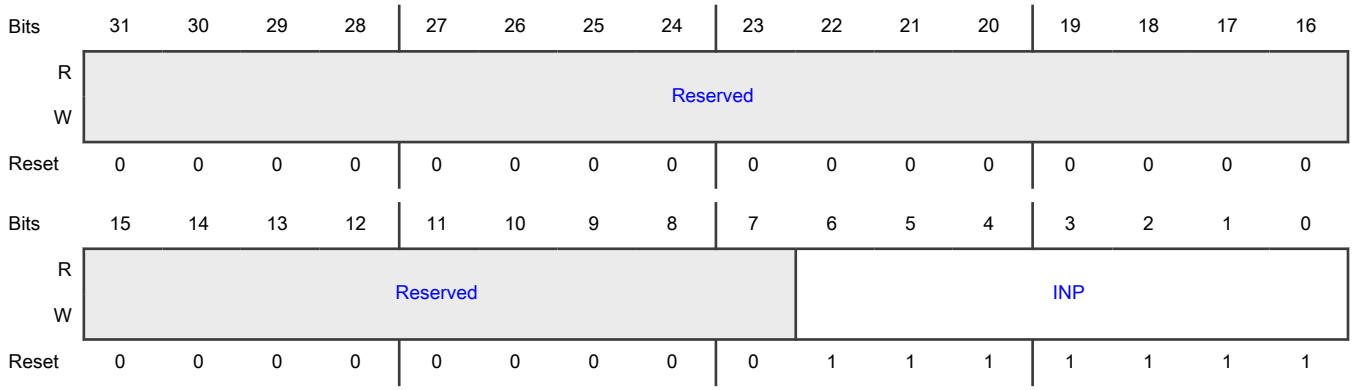
For a = 0 to 15:

Register	Offset
AOI0_INPUTa	440h + (a × 4h)

Function

This register is used to select the AOIO0 trigger inputs.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 INP	AOI0 trigger input connections 000_0000b - Reserved 000_0001b - ADC0_tcomp[0] input is selected 000_0010b - ADC0_tcomp[1] input is selected 000_0011b - ADC0_tcomp[2] input is selected 000_0100b - ADC0_tcomp[3] input is selected 000_0101b - CMP0_OUT input is selected 000_0110b - CMP1_OUT input is selected 000_0111b - Reserved 000_1000b - CTimer0_MAT0 input is selected 000_1001b - CTimer0_MAT1 input is selected 000_1010b - CTimer0_MAT2 input is selected 000_1011b - CTimer0_MAT3 input is selected 000_1100b - CTimer1_MAT0 000_1101b - CTimer1_MAT1 input is selected 000_1110b - CTimer1_MAT2 input is selected 000_1111b - CTimer1_MAT3 input is selected 001_0000b - CTimer2_MAT0 input is selected 001_0001b - CTimer2_MAT1 input is selected 001_0010b - CTimer2_MAT2 input is selected 001_0011b - CTimer2_MAT3 input is selected 001_0100b - LPTMR0 input is selected 001_0101b - Reserved 001_0110b - QDC0_CMP_FLAG0 input is selected 001_0111b - QDC0_CMP_FLAG1 input is selected 001_1000b - QDC0_CMP_FLAG2 input is selected 001_1001b - QDC0_CMP_FLAG3 input is selected 001_1010b - QDC0_POS_MATCH input is selected 001_1011b - PWM0_SM0_MUX_TRIG0 0 input is selected 001_1100b - PWM0_SM0_MUX_TRIG1 input is selected 001_1101b - PWM0_SM1_MUX_TRIG0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001_1110b - PWM0_SM1_MUX_TRIG1 input is selected 001_1111b - PWM0_SM2_MUX_TRIG0 input is selected 010_0000b - PWM0_SM2_MUX_TRIG1 input is selected 010_0001b - Reserved 010_0010b - Reserved 010_0011b - TRIG_IN0 input is selected 010_0100b - TRIG_IN1 input is selected 010_0101b - TRIG_IN2 input is selected 010_0110b - TRIG_IN3 input is selected 010_0111b - TRIG_IN4 input is selected 010_1000b - TRIG_IN5 input is selected 010_1001b - TRIG_IN6 input is selected 010_1010b - TRIG_IN7 input is selected 010_1011b - TRIG_IN8 input is selected 010_1100b - TRIG_IN9 input is selected 010_1101b - TRIG_IN10 input is selected 010_1110b - TRIG_IN11 input is selected 010_1111b - GPIO0 Pin Event Trig 0 input is selected 011_0000b - GPIO1 Pin Event Trig 0 input is selected 011_0001b - GPIO2 Pin Event Trig 0 input is selected 011_0010b - GPIO3 Pin Event Trig 0 input is selected 011_0011b - GPIO4 Pin Event Trig 0 input is selected 011_0100b - ADC1_tcomp[0] input is selected 011_0101b - ADC1_tcomp[1] input is selected 011_0110b - ADC1_tcomp[2] input is selected 011_0111b - ADC1_tcomp[3] input is selected 011_1000b - CTimer3_MAT0 input is selected 011_1001b - CTimer3_MAT1 input is selected 011_1010b - CTimer3_MAT2 input is selected 011_1011b - CTimer3_MAT3 input is selected 011_1100b - CTimer4_MAT0 input is selected 011_1101b - CTimer4_MAT1 input is selected 011_1110b - CTimer4_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011_1111b - CTimer4_MAT3 input is selected 100_0000b - FlexIO CH0 input is selected 100_0001b - FlexIO CH1 input is selected 100_0010b - FlexIO CH2 input is selected 100_0011b - FlexIO CH3 input is selected 100_0100b - QDC1_CMP_FLAG0 input is selected 100_0101b - QDC1_CMP_FLAG1 input is selected 100_0110b - QDC1_CMP_FLAG2 input is selected 100_0111b - QDC1_CMP_FLAG3 input is selected 100_1000b - QDC1_POS_MATCH0 input is selected 100_1001b - PWM1_SM0_MUX_TRIG0 input is selected 100_1010b - PWM1_SM0_MUX_TRIG1 input is selected 100_1011b - PWM1_SM1_MUX_TRIG0 input is selected 100_1100b - PWM1_SM1_MUX_TRIG1 input is selected 100_1101b - PWM1_SM2_MUX_TRIG0 input is selected 100_1110b - PWM1_SM2_MUX_TRIG1 input is selected 100_1111b - Reserved 101_0000b - Reserved All other values are reserved.

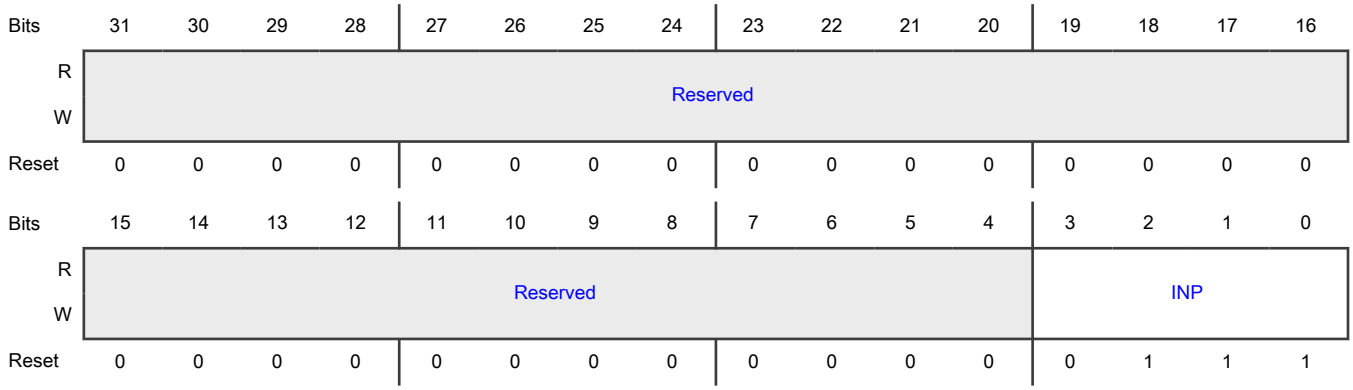
13.5.1.54 USB-FS trigger input connections (USBFS_TRIG)

Offset

Register	Offset
USBFS_TRIG	480h

Function
This register is used to select the USB-FS trigger inputs.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 INP	USB-FS trigger input connections. 0000b - Reserved 0001b - LPUART0 lpuart_trg_txdata input is selected 0010b - LPUART1 lpuart_trg_txdata input is selected 0011b - LPUART2 lpuart_trg_txdata input is selected 0100b - LPUART3 lpuart_trg_txdata input is selected 0101b - LPUART4 lpuart_trg_txda input is selected All other values are reserved.

13.5.1.55 EXT trigger connections (EXT_TRIG0 - EXT_TRIG7)

Offset

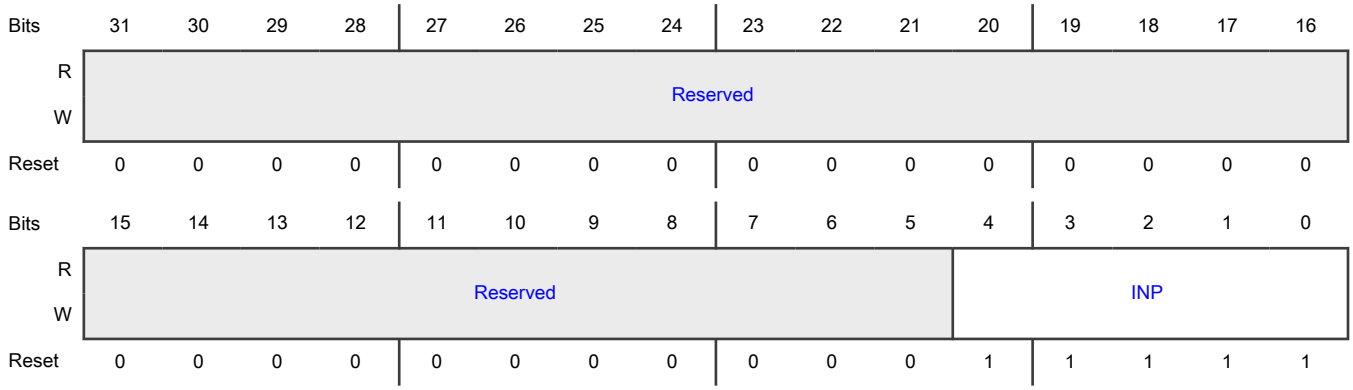
For a = 0 to 7:

Register	Offset
EXT_TRIGa	4C0h + (a × 4h)

Function

This register is used to select the EXT trigger inputs.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 INP	EXT trigger input connections 0_0000b - Reserved 0_0001b - Reserved 0_0010b - AOI0_OUT0 input is selected 0_0011b - AOI0_OUT1 input is selected 0_0100b - AOI0_OUT2 input is selected 0_0101b - AOI0_OUT3 input is selected 0_0110b - CMP0_OUT input is selected 0_0111b - CMP1_OUT input is selected 0_1000b - Reserved 0_1001b - LPUART0 input is selected 0_1010b - LPUART1 input is selected 0_1011b - LPUART2 input is selected 0_1100b - LPUART3 input is selected 0_1101b - LPUART4 input is selected 0_1110b - AOI1_OUT0 input is selected 0_1111b - AOI1_OUT1 input is selected 1_0000b - AOI1_OUT2 input is selected 1_0001b - AOI1_OUT3 input is selected All other values are reserved.

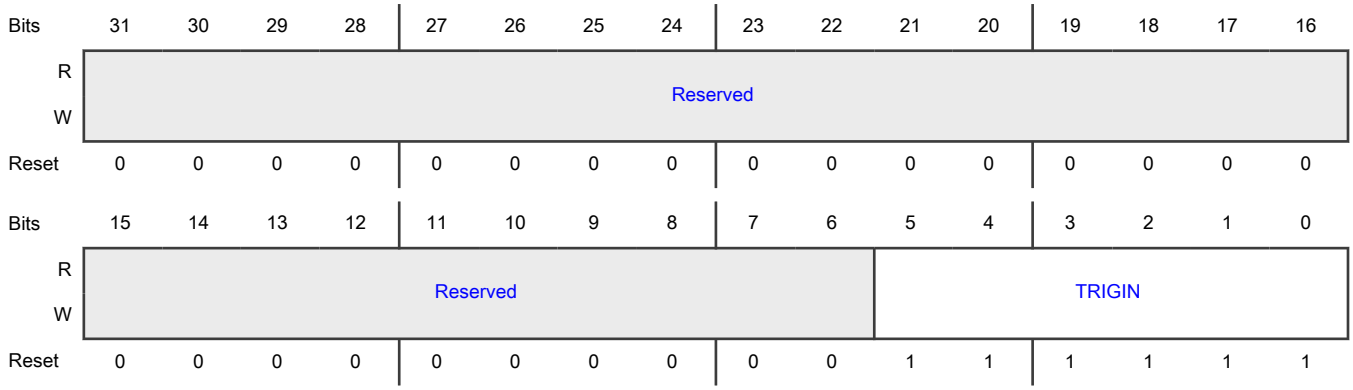
13.5.1.56 CMP1 input connections (CMP1_TRIG)

Offset

Register	Offset
CMP1_TRIG	4E0h

Function
This register selects the CMP1 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 TRIGIN	CMP0 input trigger 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - Reserved 00_1000b - CTimer0_MAT0 input is selected 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer1_MAT0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer2_MAT0 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - LPTMR0 input is selected 00_1111b - Reserved 01_0000b - QDC0_POS_MATCH0 01_0001b - PWM0_SM0_MUX_TRIG0 input is selected 01_0010b - PWM0_SM0_MUX_TRIG1 input is selected 01_0011b - PWM0_SM1_MUX_TRIG0 input is selected 01_0100b - PWM0_SM1_MUX_TRIG1 input is selected 01_0101b - PWM0_SM2_MUX_TRIG0 input is selected 01_0110b - PWM0_SM2_MUX_TRIG1 input is selected 01_0111b - Reserved 01_1000b - Reserved 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected 01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - Reserved 10_0100b - Reserved 10_0101b - Reserved 10_0110b - Reserved 10_0111b - CTimer3_MAT0 10_1000b - CTimer3_MAT1 10_1001b - CTimer4_MAT0 input is selected 10_1010b - CTimer4_MAT1 input is selected 10_1011b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1100b - Reserved 10_1101b - Reserved 10_1110b - Reserved 10_1111b - QDC1_POS_MATCH0 input is selected 11_0000b - PWM1_SM0_MUX_TRIG0 input is selected 11_0001b - PWM1_SM0_MUX_TRIG1 input is selected 11_0010b - PWM1_SM1_MUX_TRIG0 input is selected 11_0011b - PWM1_SM1_MUX_TRIG1 input is selected 11_0100b - PWM1_SM2_MUX_TRIG0 input is selected 11_0101b - PWM1_SM2_MUX_TRIG1 input is selected 11_0110b - Reserved 11_0111b - Reserved All other values are reserved.

13.5.1.57 LPI2C2 trigger input connections (LPI2C2_TRIG)

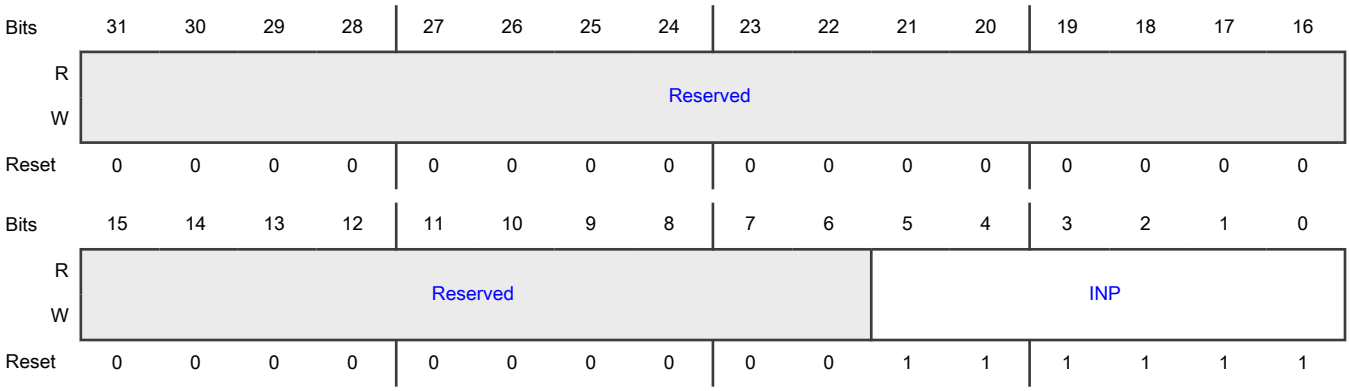
Offset

Register	Offset
LPI2C2_TRIG	540h

Function

This register is used to select the LPI2C2 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPI2C2 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected 00_1011b - CTimer1_MAT0 input is selected 00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - CTimer3_MAT2 input is selected 10_0100b - CTimer3_MAT3 input is selected 10_0101b - CTimer4_MAT2 input is selected 10_0110b - CTimer4_MAT3 input is selected 10_0111b - FlexIO CH0 input is selected 10_1000b - FlexIO CH1 input is selected 10_1001b - FlexIO CH2 input is selected 10_1010b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.58 OPAMP0 Trigger Input Connections (OPAMP0_TRIG)

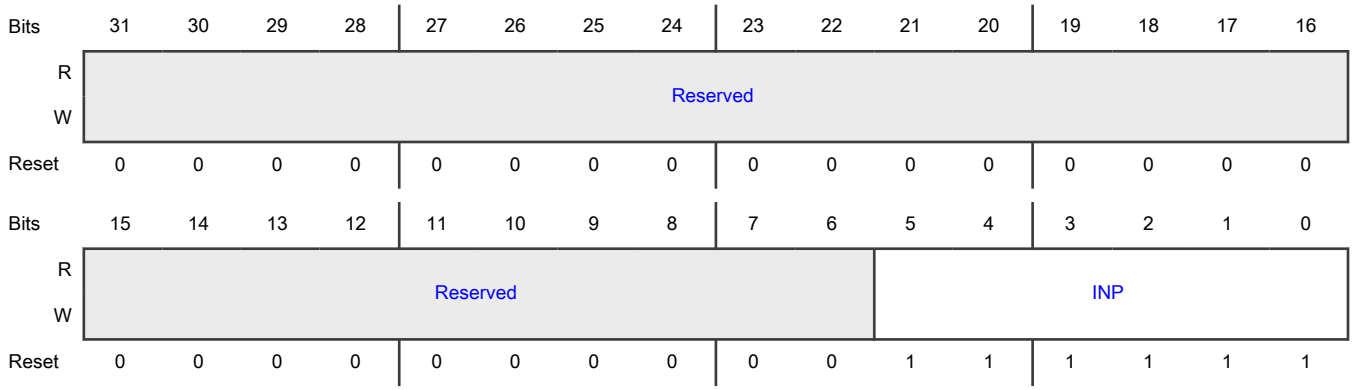
Offset

Register	Offset
OPAMP0_TRIG	580h

Function

This register is used to select the OPAMP0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	DAC0 trigger input 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected 00_1011b - CTimer1_MAT0 input is selected 00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - Reserved 01_0010b - PWM0_SM0_MUX_TRIG0 input is selected 01_0011b - PWM0_SM0_MUX_TRIG1 input is selected 01_0100b - PWM0_SM1_MUX_TRIG0 input is selected 01_0101b - PWM0_SM1_MUX_TRIG1 input is selected 01_0110b - PWM0_SM2_MUX_TRIG0 input is selected 01_0111b - PWM0_SM2_MUX_TRIG1 input is selected 01_1000b - Reserved 01_1001b - Reserved 01_1010b - GPIO0 Pin Event Trig 0 input is selected 01_1011b - GPIO1 Pin Event Trig 0 input is selected 01_1100b - GPIO2 Pin Event Trig 0 input is selected 01_1101b - GPIO3 Pin Event Trig 0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1110b - GPIO4 Pin Event Trig 0 input is selected 01_1111b - WUU input is selected 10_0000b - Reserved 10_0001b - AOI1_OUT0 input is selected 10_0010b - AOI1_OUT1 input is selected 10_0011b - AOI1_OUT2 input is selected 10_0100b - AOI1_OUT3 input is selected 10_0101b - ADC0_tcomp[0] input is selected 10_0110b - ADC0_tcomp[1] input is selected 10_0111b - ADC1_tcomp[0] input is selected 10_1000b - ADC1_tcomp[1] input is selected 10_1001b - CTimer3_MAT0 input is selected 10_1010b - CTimer3_MAT1 input is selected 10_1011b - CTimer4_MAT0 input is selected 10_1100b - CTimer4_MAT1 input is selected 10_1101b - FlexIO CH0 input is selected 10_1110b - FlexIO CH1 input is selected 10_1111b - FlexIO CH2 input is selected 11_0000b - FlexIO CH3 input is selected 11_0001b - Reserved 11_0010b - PWM1_SM0_MUX_TRIG0 input is selected 11_0011b - PWM1_SM0_MUX_TRIG1 input is selected 11_0100b - PWM1_SM1_MUX_TRIG0 input is selected 11_0101b - PWM1_SM1_MUX_TRIG1 input is selected 11_0110b - PWM1_SM2_MUX_TRIG0 input is selected 11_0111b - PWM1_SM2_MUX_TRIG1 input is selected 11_1000b - Reserved 11_1001b - Reserved All other values are reserved.

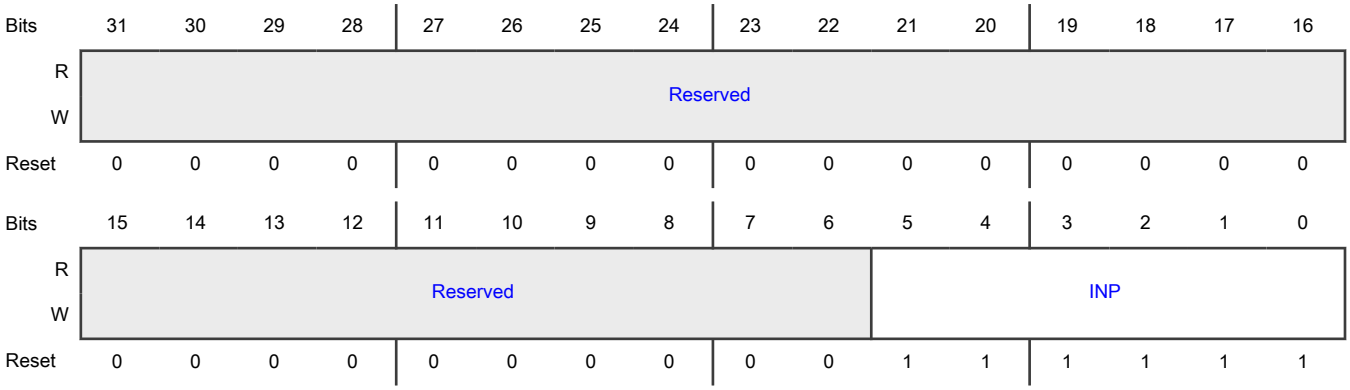
13.5.1.59 LPI2C0 trigger input connections (LPI2C0_TRIG)

Offset

Register	Offset
LPI2C0_TRIG	5A0h

Function
This register is used to select the LPI2C0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPI2C0 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT0 input is selected 00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected 01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - CTimer3_MAT2 input is selected 10_0100b - CTimer3_MAT3 input is selected 10_0101b - CTimer4_MAT2 input is selected 10_0110b - CTimer4_MAT3 input is selected 10_0111b - FlexIO CH0 input is selected 10_1000b - FlexIO CH1 input is selected 10_1001b - FlexIO CH2 input is selected 10_1010b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.60 LPI2C1 trigger input connections (LPI2C1_TRIG)

Offset

Register	Offset
LPI2C1_TRIG	5C0h

Function
This register is used to select the LPI2C1 trigger inputs.

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										INP					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPI2C1 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT0 input is selected 00_1010b - CTimer0_MAT1 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT0 input is selected 00_1100b - CTimer1_MAT1 input is selected 00_1101b - CTimer2_MAT0 input is selected 00_1110b - CTimer2_MAT1 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected 01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - CTimer3_MAT2 input is selected 10_0100b - CTimer3_MAT3 input is selected 10_0101b - CTimer4_MAT2 input is selected 10_0110b - CTimer4_MAT3 input is selected 10_0111b - FlexIO CH0 input is selected 10_1000b - FlexIO CH1 input is selected 10_1001b - FlexIO CH2 input is selected 10_1010b - FlexIO CH3 input is selected All other values are reserved.

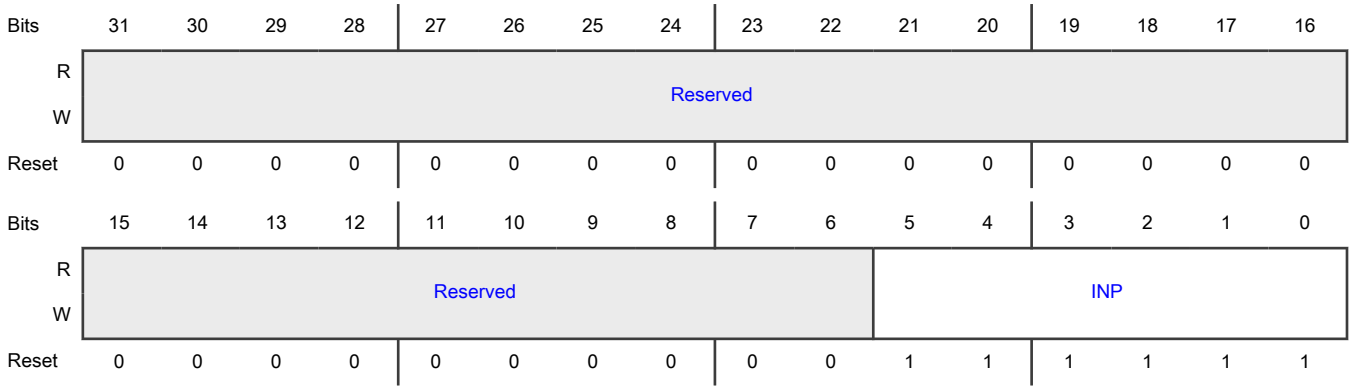
13.5.1.61 LPSPI0 trigger input connections (LPSPI0_TRIG)

Offset

Register	Offset
LPSPI0_TRIG	5E0h

Function
This register is used to select the LPSPI0 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPSPI0 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT1 input is selected 00_1010b - CTimer0_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT1 input is selected 00_1100b - CTimer1_MAT2 input is selected 00_1101b - CTimer2_MAT1 input is selected 00_1110b - CTimer2_MAT2 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected 01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - CTimer3_MAT2 input is selected 10_0100b - CTimer3_MAT3 input is selected 10_0101b - CTimer4_MAT2 input is selected 10_0110b - CTimer4_MAT3 input is selected 10_0111b - FlexIO CH0 input is selected 10_1000b - FlexIO CH1 input is selected 10_1001b - FlexIO CH2 input is selected 10_1010b - FlexIO CH3 input is selected All other values are reserved.

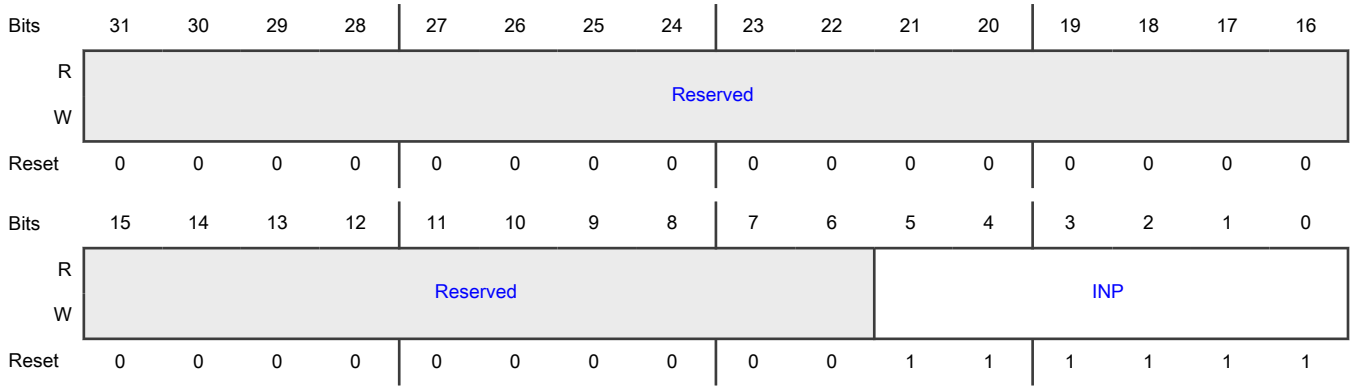
13.5.1.62 LPSPI1 trigger input connections (LPSPI1_TRIG)

Offset

Register	Offset
LPSPI1_TRIG	600h

Function
This register is used to select the LPSPI1 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPSPI1 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT1 input is selected 00_1010b - CTimer0_MAT2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT1 input is selected 00_1100b - CTimer1_MAT2 input is selected 00_1101b - CTimer2_MAT1 input is selected 00_1110b - CTimer2_MAT2 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - GPIO0 Pin Event Trig 0 input is selected 01_1010b - GPIO1 Pin Event Trig 0 input is selected 01_1011b - GPIO2 Pin Event Trig 0 input is selected 01_1100b - GPIO3 Pin Event Trig 0 input is selected 01_1101b - GPIO4 Pin Event Trig 0 input is selected 01_1110b - WUU input is selected 01_1111b - AOI1_OUT0 input is selected 10_0000b - AOI1_OUT1 input is selected 10_0001b - AOI1_OUT2 input is selected 10_0010b - AOI1_OUT3 input is selected 10_0011b - CTimer3_MAT2 input is selected 10_0100b - CTimer3_MAT3 input is selected 10_0101b - CTimer4_MAT2 input is selected 10_0110b - CTimer4_MAT3 input is selected 10_0111b - FlexIO CH0 input is selected 10_1000b - FlexIO CH1 input is selected 10_1001b - FlexIO CH2 input is selected 10_1010b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.63 LPUART0 trigger input connections (LPUART0)

Offset

Register	Offset
LPUART0	620h

Function
This register is used to select the LPUART0 trigger inputs.

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										INP					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPUART0 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - TRIG_IN8 input is selected 01_1010b - TRIG_IN9 input is selected 01_1011b - TRIG_IN10 input is selected 01_1100b - TRIG_IN11 input is selected 01_1101b - GPIO0 Pin Event Trig 0 input is selected 01_1110b - GPIO1 Pin Event Trig 0 input is selected 01_1111b - GPIO2 Pin Event Trig 0 input is selected 10_0000b - GPIO3 Pin Event Trig 0 input is selected 10_0001b - GPIO4 Pin Event Trig 0 input is selected 10_0010b - WUU selected 10_0011b - USB0 ipp_ind_uart_rxd_usbmux input is selected 10_0100b - AOI1_OUT0 input is selected 10_0101b - AOI1_OUT1 input is selected 10_0110b - AOI1_OUT2 input is selected 10_0111b - AOI1_OUT3 input is selected 10_1000b - CTimer3_MAT2 input is selected 10_1001b - CTimer3_MAT3 input is selected 10_1010b - CTimer4_MAT2 input is selected 10_1011b - CTimer4_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1100b - FlexIO CH0 input is selected 10_1101b - FlexIO CH1 input is selected 10_1110b - FlexIO CH2 input is selected 10_1111b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.64 LPUART1 trigger input connections (LPUART1)

Offset

Register	Offset
LPUART1	640h

Function

This register is used to select the LPUART1 trigger inputs.

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										INP					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPUART1 trigger input connections 00_0000b - Reserved 00_0001b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - TRIG_IN8 input is selected 01_1010b - TRIG_IN9 input is selected 01_1011b - TRIG_IN10 input is selected 01_1100b - TRIG_IN11 input is selected 01_1101b - GPIO0 Pin Event Trig 0 input is selected 01_1110b - GPIO1 Pin Event Trig 0 input is selected 01_1111b - GPIO2 Pin Event Trig 0 input is selected 10_0000b - GPIO3 Pin Event Trig 0 input is selected 10_0001b - GPIO4 Pin Event Trig 0 input is selected 10_0010b - WUU selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_0011b - USB0 ipp_ind_uart_rxd_usbmux input is selected 10_0100b - AOI1_OUT0 input is selected 10_0101b - AOI1_OUT1 input is selected 10_0110b - AOI1_OUT2 input is selected 10_0111b - AOI1_OUT3 input is selected 10_1000b - CTimer3_MAT2 input is selected 10_1001b - CTimer3_MAT3 input is selected 10_1010b - CTimer4_MAT2 input is selected 10_1011b - CTimer4_MAT3 input is selected 10_1100b - FlexIO CH0 input is selected 10_1101b - FlexIO CH1 input is selected 10_1110b - FlexIO CH2 input is selected 10_1111b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.65 LPUART2 trigger input connections (LPUART2)

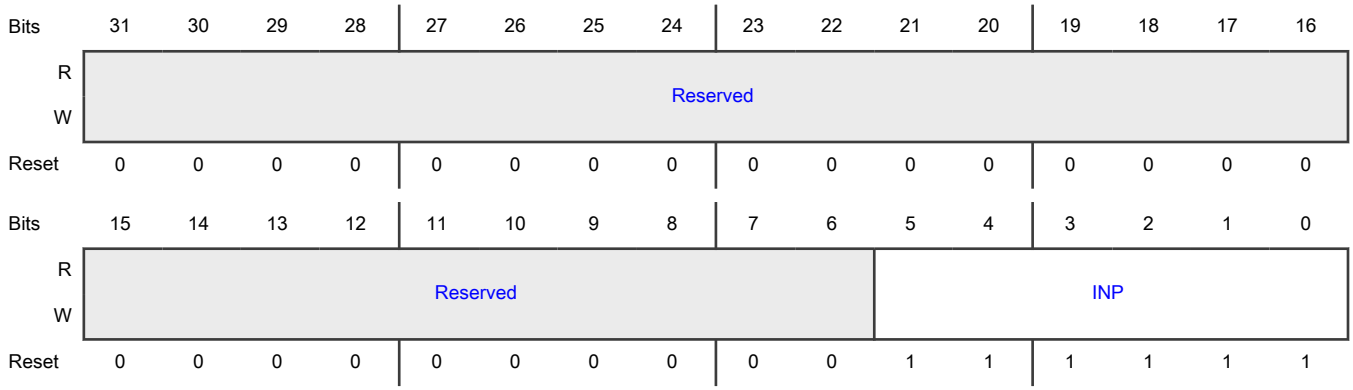
Offset

Register	Offset
LPUART2	660h

Function

This register is used to select the LPUART2 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPUART2 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - TRIG_IN8 input is selected 01_1010b - TRIG_IN9 input is selected 01_1011b - TRIG_IN10 input is selected 01_1100b - TRIG_IN11 input is selected 01_1101b - GPIO0 Pin Event Trig 0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_1110b - GPIO1 Pin Event Trig 0 input is selected 01_1111b - GPIO2 Pin Event Trig 0 input is selected 10_0000b - GPIO3 Pin Event Trig 0 input is selected 10_0001b - GPIO4 Pin Event Trig 0 input is selected 10_0010b - WUU selected 10_0011b - USB0 ipp_ind_uart_rxd_usbmux input is selected 10_0100b - AOI1_OUT0 input is selected 10_0101b - AOI1_OUT1 input is selected 10_0110b - AOI1_OUT2 input is selected 10_0111b - AOI1_OUT3 input is selected 10_1000b - CTimer3_MAT2 input is selected 10_1001b - CTimer3_MAT3 input is selected 10_1010b - CTimer4_MAT2 input is selected 10_1011b - CTimer4_MAT3 input is selected 10_1100b - FlexIO CH0 input is selected 10_1101b - FlexIO CH1 input is selected 10_1110b - FlexIO CH2 input is selected 10_1111b - FlexIO CH3 input is selected All other values are reserved.

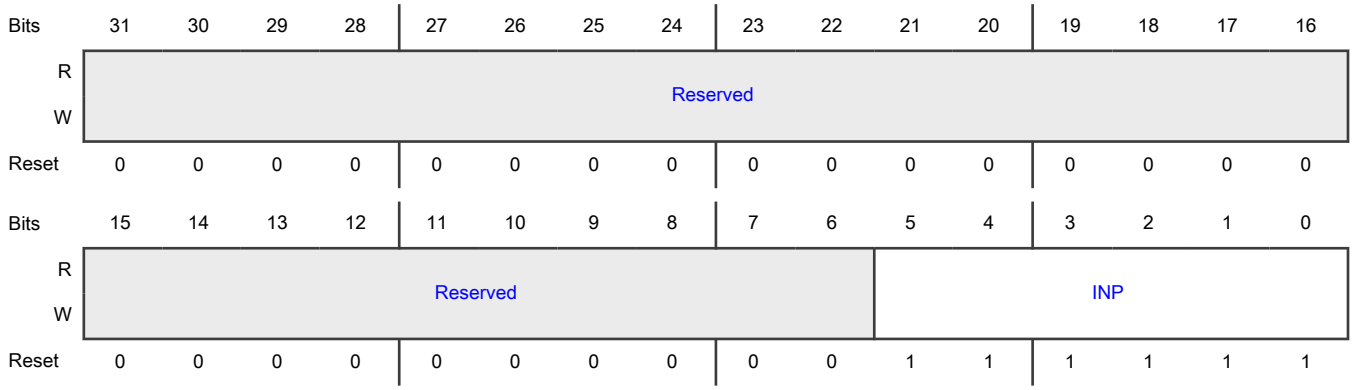
13.5.1.66 LPUART3 trigger input connections (LPUART3)

Offset

Register	Offset
LPUART3	680h

Function
This register is used to select the LPUART3 trigger inputs.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPUART3 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected 00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - TRIG_IN8 input is selected 01_1010b - TRIG_IN9 input is selected 01_1011b - TRIG_IN10 input is selected 01_1100b - TRIG_IN11 input is selected 01_1101b - GPIO0 Pin Event Trig 0 input is selected 01_1110b - GPIO1 Pin Event Trig 0 input is selected 01_1111b - GPIO2 Pin Event Trig 0 input is selected 10_0000b - GPIO3 Pin Event Trig 0 input is selected 10_0001b - GPIO4 Pin Event Trig 0 input is selected 10_0010b - WUU selected 10_0011b - USB0 ipp_ind_uart_rxd_usbmux input is selected 10_0100b - AOI1_OUT0 input is selected 10_0101b - AOI1_OUT1 input is selected 10_0110b - AOI1_OUT2 input is selected 10_0111b - AOI1_OUT3 input is selected 10_1000b - CTimer3_MAT2 input is selected 10_1001b - CTimer3_MAT3 input is selected 10_1010b - CTimer4_MAT2 input is selected 10_1011b - CTimer4_MAT3 input is selected 10_1100b - FlexIO CH0 input is selected 10_1101b - FlexIO CH1 input is selected 10_1110b - FlexIO CH2 input is selected 10_1111b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.67 LPUART4 trigger input connections (LPUART4)

Offset

Register	Offset
LPUART4	6A0h

Function
This register is used to select the LPUART4 trigger inputs.

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										INP					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Fields

Field	Function
31-6 —	Reserved
5-0 INP	LPUART4 trigger input connections 00_0000b - Reserved 00_0001b - Reserved 00_0010b - AOI0_OUT0 input is selected 00_0011b - AOI0_OUT1 input is selected 00_0100b - AOI0_OUT2 input is selected 00_0101b - AOI0_OUT3 input is selected 00_0110b - CMP0_OUT input is selected 00_0111b - CMP1_OUT input is selected 00_1000b - Reserved 00_1001b - CTimer0_MAT2 input is selected 00_1010b - CTimer0_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00_1011b - CTimer1_MAT2 input is selected 00_1100b - CTimer1_MAT3 input is selected 00_1101b - CTimer2_MAT2 input is selected 00_1110b - CTimer2_MAT3 input is selected 00_1111b - LPTMR0 input is selected 01_0000b - Reserved 01_0001b - TRIG_IN0 input is selected 01_0010b - TRIG_IN1 input is selected 01_0011b - TRIG_IN2 input is selected 01_0100b - TRIG_IN3 input is selected 01_0101b - TRIG_IN4 input is selected 01_0110b - TRIG_IN5 input is selected 01_0111b - TRIG_IN6 input is selected 01_1000b - TRIG_IN7 input is selected 01_1001b - TRIG_IN8 input is selected 01_1010b - TRIG_IN9 input is selected 01_1011b - TRIG_IN10 input is selected 01_1100b - TRIG_IN11 input is selected 01_1101b - GPIO0 Pin Event Trig 0 input is selected 01_1110b - GPIO1 Pin Event Trig 0 input is selected 01_1111b - GPIO2 Pin Event Trig 0 input is selected 10_0000b - GPIO3 Pin Event Trig 0 input is selected 10_0001b - GPIO4 Pin Event Trig 0 input is selected 10_0010b - WUU selected 10_0011b - USB0 ipp_ind_uart_rxd_usbmux input is selected 10_0100b - AOI1_OUT0 input is selected 10_0101b - AOI1_OUT1 input is selected 10_0110b - AOI1_OUT2 input is selected 10_0111b - AOI1_OUT3 input is selected 10_1000b - CTimer3_MAT2 input is selected 10_1001b - CTimer3_MAT3 input is selected 10_1010b - CTimer4_MAT2 input is selected 10_1011b - CTimer4_MAT3 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10_1100b - FlexIO CH0 input is selected 10_1101b - FlexIO CH1 input is selected 10_1110b - FlexIO CH2 input is selected 10_1111b - FlexIO CH3 input is selected All other values are reserved.

13.5.1.68 FlexIO Trigger Input Connections (FLEXIO_TRIG0 - FLEXIO_TRIG3)

Offset

Register	Offset
FLEXIO_TRIG0	6E0h
FLEXIO_TRIG1	6E4h
FLEXIO_TRIG2	6E8h
FLEXIO_TRIG3	6ECh

Function
This register is used to select the FlexIO trigger inputs.

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								INP							
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	Input number for FlexIO0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
INP	000_0000b - Reserved 000_0001b - AOI0_OUT0 input is selected 000_0010b - AOI0_OUT1 input is selected 000_0011b - AOI0_OUT2 input is selected 000_0100b - AOI0_OUT3 input is selected 000_0101b - ADC0_tcomp[0] input is selected 000_0110b - ADC0_tcomp[1] input is selected 000_0111b - ADC0_tcomp[2] input is selected 000_1000b - ADC0_tcomp[3] input is selected 000_1001b - CMP0_OUT input is selected 000_1010b - CMP1_OUT input is selected 000_1011b - Reserved 000_1100b - CTimer0_MAT1 input is selected 000_1101b - CTimer0_MAT2 input is selected 000_1110b - CTimer1_MAT1 input is selected 000_1111b - CTimer1_MAT2 input is selected 001_0000b - CTimer2_MAT1 input is selected 001_0001b - CTimer2_MAT2 input is selected 001_0010b - LPTMR0 input is selected 001_0011b - Reserved (LPTMR1) input is selected 001_0100b - PWM0_SM0_MUX_TRIG0 input is selected 001_0101b - PWM0_SM1_MUX_TRIG0 input is selected 001_0110b - PWM0_SM2_MUX_TRIG0 input is selected 001_0111b - Reserved 001_1000b - TRIG_IN0 input is selected 001_1001b - TRIG_IN1 input is selected 001_1010b - TRIG_IN2 input is selected 001_1011b - TRIG_IN3 input is selected 001_1100b - TRIG_IN4 input is selected 001_1101b - TRIG_IN5 input is selected 001_1110b - TRIG_IN6 input is selected 001_1111b - TRIG_IN7 input is selected 010_0000b - GPIO0 Pin Event Trig 0 input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010_0001b - GPIO2 Pin Event Trig 0 input is selected 010_0010b - GPIO3 Pin Event Trig 0 input is selected 010_0011b - GPIO4 Pin Event Trig 0 input is selected 010_0100b - WUU input is selected 010_0101b - PWM1_A0_TRIG0 input is selected 010_0110b - LPI2C0 Master End of Packet 010_0111b - LPI2C0 Slave End of Packet 010_1000b - LPI2C1 Master End of Packet 010_1001b - LPI2C1 Slave End of Packet 010_1010b - LPSPI0 End of Frame 010_1011b - LPSPI0 Received Data Word 010_1100b - LPSPI1 End of Frame 010_1101b - LPSPI1 Received Data Word 010_1110b - LPUART0 Received Data Word 010_1111b - LPUART0 Transmitted Data Word 011_0000b - LPUART0 Receive Line Idle 011_0001b - LPUART1 Received Data Word 011_0010b - LPUART1 Transmitted Data Word 011_0011b - LPUART1 Receive Line Idle 011_0100b - LPUART2 Received Data Word 011_0101b - LPUART2 Transmitted Data Word 011_0110b - LPUART2 Receive Line Idle 011_0111b - LPUART3 Received Data Word 011_1000b - LPUART3 Transmitted Data Word 011_1001b - LPUART3 Receive Line Idle 011_1010b - LPUART4 Received Data Word 011_1011b - LPUART4 Transmitted Data Word 011_1100b - LPUART4 Receive Line Idle 011_1101b - AOI1_OUT0 input is selected 011_1110b - AOI1_OUT1 input is selected 011_1111b - AOI1_OUT2 input is selected 100_0000b - AOI1_OUT3 input is selected 100_0001b - ADC1_tcomp[0] input is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100_0010b - ADC1_tcomp[1] input is selected 100_0011b - ADC1_tcomp[2] input is selected 100_0100b - ADC1_tcomp[3] input is selected 100_0101b - CTimer3_MAT2 input is selected 100_0110b - CTimer3_MAT3 input is selected 100_0111b - CTimer4_MAT2 input is selected 100_1000b - CTimer4_MAT3 input is selected 100_1001b - PWM1_SM0_MUX_TRIG0 input is selected 100_1010b - PWM1_SM1_MUX_TRIG0 input is selected 100_1011b - PWM1_SM2_MUX_TRIG0 input is selected 100_1100b - Reserved 100_1101b - LPI2C2 Master End of Packet 100_1110b - LPI2C2 Slave End of Packet 100_1111b - LPI2C3 Master End of Packet 101_0000b - LPI2C3 Slave End of Packet All other values are reserved.

Chapter 14
System Controller (SYSCON)

14.1 Chip-specific SYSCON information

Table 72. Reference links to related information

Topic	Related module	Reference
Full description	SYSCON	SYSCON
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

14.1.1 Module instances

This device has one instance of SYSCON module.

14.1.2 Configuration

Configure the SYSCON block as follows:

- No clock configuration is needed. The clock to the SYSCON block is always enabled.
- The SYSCON block controls use of the CLKOUT pin, which must also be configured through Port control module.

14.2 Overview

The SYSCON module provides controls and configurations on the system and peripherals for the multiple functions.

14.2.1 Features

The SYSCON module supports the following features and configurations:

- System and bus configuration
  - AHB matrix priority
  - CPUs control and status
  - CPU debug access
  - CPU LPCAC control
  - NMI source select
  - Calibrate system tick timer
  - Boot control
- System/Peripherals - clock select and control
  - Allows enabling and selection of clocks to individual peripherals and memories
  - Allows configuration of clock dividers
- Memory configuration

  - ROM access
  - RAM ECC
  - FLASH cache, bus error
- Gray-2-Binary converter
  - Allows decoding gray value coming from OS Event Timer
- Peripherals configuration
  - PWM control
  - CTIMER global start
- Security configuration
  - Security control
  - Security attestation
- Reset control
  - Monitors and release resets to individual peripherals
- Device ID register

14.3 Functional description

The MRCC submodule provides on-chip modules their own dedicated MRCC bits for clock gating, reset control and configuration options. Generally, MRCC register contains a clock configuration (CC = {MRCC_GLB_ACC*[peripherals name]*, MRCC_GLB_CC*[peripherals name]* }) field for the module's clocks.

NOTE

Before a module can be used, its clocks must be enabled (CC != 00) and it must be released from reset (MRCC_GLB_RST*[peripherals name]*= 1). If a module is *not* released from reset (MRCC_GLB_RST*[peripherals name]*= 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.

NOTE

Before configuring a functional clock, the module's clocks must be disabled (CC = 0b00).

14.4 Signals

Table 73. SYSCON pin description

Function	Type	Pin	Description
CLKOUT	O	P0_6, P3_6, P3_8, P4_2	CLKOUT clock output.

14.5 Memory map and register definition

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

14.5.1 SYSCON register descriptions

14.5.1.1 SYSCON memory map

SYSCON.syscon base address: 4009_1000h

Offset	Register	Width (In bits)	Access	Reset value
200h	AHB Matrix Remap Control (REMAP)	32	RW	0000_0000h
210h	AHB Matrix Priority Control (AHBMATPRIO)	32	RW	0000_0000h
23Ch	Non-Secure CPU0 System Tick Calibration (CPU0NSTCKCAL)	32	RW	0000_0000h
248h	NMI Source Select (NMISRC)	32	RW	0000_0000h
378h	SLOW_CLK Clock Divider (SLOWCLKDIV)	32	RW	0000_0000h
380h	System Clock Divider (AHBCLKDIV)	32	RW	0000_0000h
3FCh	Clock Configuration Unlock (CLKUNLOCK)	32	RW	0000_0000h
400h	NVM Control (NVM_CTRL)	32	RW	0002_0400h
404h	ROM Wait State (ROMCR)	32	R	0000_0000h
80Ch	CPU Status (CPUSTAT)	32	R	0000_0000h
824h	LPCAC Control (LPCAC_CTRL)	32	RW	0000_0031h
938h	PWM0 Submodule Control (PWM0SUBCTL)	32	RW	0000_0000h
93Ch	PWM1 Submodule Control (PWM1SUBCTL)	32	RW	0000_0000h
940h	CTIMER Global Start Enable (CTIMERGLOBALSTARTEN)	32	RW	0000_0000h
944h	RAM Control (RAM_CTRL)	32	RW	0000_0001h
B60h	Gray to Binary Converter Gray Code [31:0] (GRAY_CODE_LSB)	32	RW	0000_0000h
B64h	Gray to Binary Converter Gray Code [41:32] (GRAY_CODE_MSB)	32	RW	0000_0000h
B68h	Gray to Binary Converter Binary Code [31:0] (BINARY_CODE_LSB)	32	R	0000_0000h
B6Ch	Gray to Binary Converter Binary Code [41:32] (BINARY_CODE_MSB)	32	R	0000_0000h
E3Ch	ROP State Register (ROP_STATE)	32	R	See section
E58h	RAM XEN Control (SRAM_XEN)	32	RW	0000_0000h
E5Ch	RAM XEN Control (Duplicate) (SRAM_XEN_DP)	32	RW	0000_0000h
E80h	Life Cycle State Register (ELS_OTP_LC_STATE)	32	R	See section
E84h	Life Cycle State Register (Duplicate) (ELS_OTP_LC_STATE_DP)	32	R	See section
FA0h	Control Write Access to Security (DEBUG_LOCK_EN)	32	RW	0000_000Ah
FA4h	Cortex Debug Features Control (DEBUG_FEATURES)	32	RW	See section
FA8h	Cortex Debug Features Control (Duplicate) (DEBUG_FEATURES_DP)	32	RW	See section
FB4h	CPU0 Software Debug Access (SWD_ACCESS_CPU0)	32	RW	0000_0000h
FC0h	Debug Authentication BEACON (DEBUG_AUTH_BEACON)	32	RW	0000_0000h
FF0h	JTAG Chip ID (JTAG_ID)	32	R	0726_702Bh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
FF4h	Device Type (DEVICE_TYPE)	32	R	See section
FF8h	Device ID (DEVICE_ID0)	32	R	See section
FFCh	Chip Revision ID and Number (DIEID)	32	R	0059_DAA0h

14.5.1.2 AHB Matrix Remap Control (REMAP)

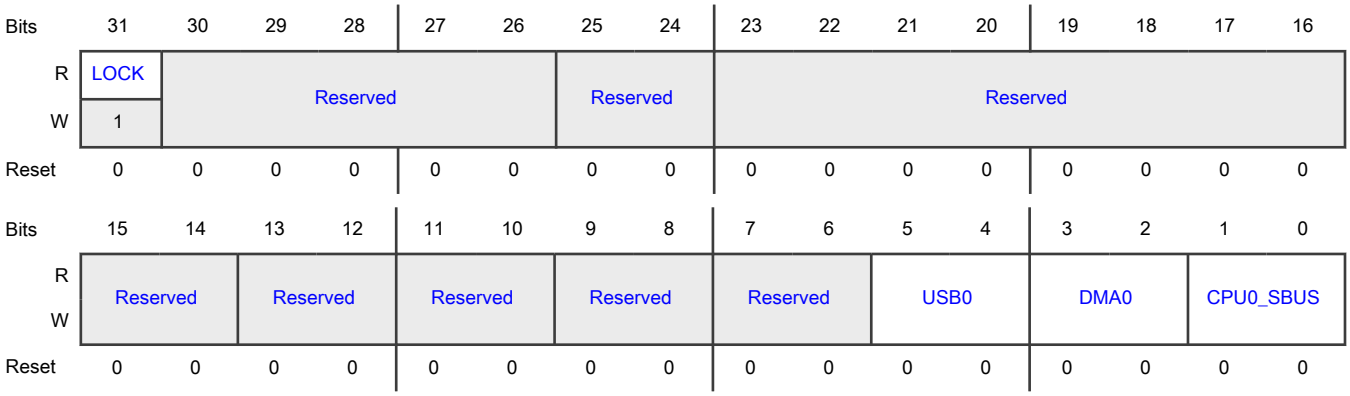
Offset

Register	Offset
REMAP	200h

Function

The Multilayer AHB Matrix remap for all masters, when they attempt to access the matrix slave port.

Diagram



Fields

Field	Function
31 LOCK	This 1-bit field provides a mechanism to limit writes to this register to protect its contents. Once set, this bit remains asserted until a system reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered until a system reset.
30-26 —	Reserved Read value is undefined, only zero should be written.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved Read value is undefined, only zero should be written.
23-14 —	Reserved Read value is undefined, only zero should be written.
13-12 —	Reserved Read value is undefined, only zero should be written.
11-10 —	Reserved Read value is undefined, only zero should be written.
9-8 —	Reserved Read value is undefined, only zero should be written.
7-6 —	Reserved Read value is undefined, only zero should be written.
5-4 USB0	RAMX0 address remap for USB0 00b - RAMX0: alias space is disabled. 01b - RAMX0: same alias space as CPU0_SBUS
3-2 DMA0	RAMX0 address remap for DMA0 00b - RAMX0: alias space is disabled. 01b - RAMX0: same alias space as CPU0_SBUS
1-0 CPU0_SBUS	RAMX0 address remap for CPU System bus 00b - RAMX0: alias space is disabled. 01b - RAMX0: alias space is enabled. It's linear address space from bottom of system ram. The start address is 0x20000000 + (system ram size - RAMX size)*1024.

14.5.1.3 AHB Matrix Priority Control (AHBMATPRIO)

Offset

Register	Offset
AHBMATPRIO	210h

Function

The Multilayer AHB Matrix arbitrates between masters, when they attempt to access the same matrix slave port at the same time. The priority values are 3 = highest, 0 = lowest. When the priority is the same, the master with the lower master number is given priority.

NOTE

Be careful when modifying this register as improper settings can seriously degrade the performance.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		Reserved		Reserved		USB_FS_ENET		Reserved		Reserved		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		Reserved		DMA0		Reserved		Reserved		CPU0_SBUS		CPU0_CBUS	
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 —	Reserved
27-26 —	Reserved
25-24 USB_FS_ENET	USB-FS bus master priority level 00b - level 0 01b - level 1 10b - level 2 11b - level 3
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 DMA0	DMA0 controller bus master priority level 00b - level 0 01b - level 1 10b - level 2 11b - level 3
7-6 —	Reserved Read value is undefined, only zero should be written.
5-4 —	Reserved Read value is undefined, only zero should be written.
3-2 CPU0_SBUS	CPU0 S-AHB bus master priority level 00b - level 0 01b - level 1 10b - level 2 11b - level 3
1-0 CPU0_CBUS	CPU0 C-AHB bus master priority level 00b - level 0 01b - level 1 10b - level 2 11b - level 3

14.5.1.4 Non-Secure CPU0 System Tick Calibration (CPU0NSTCKCAL)

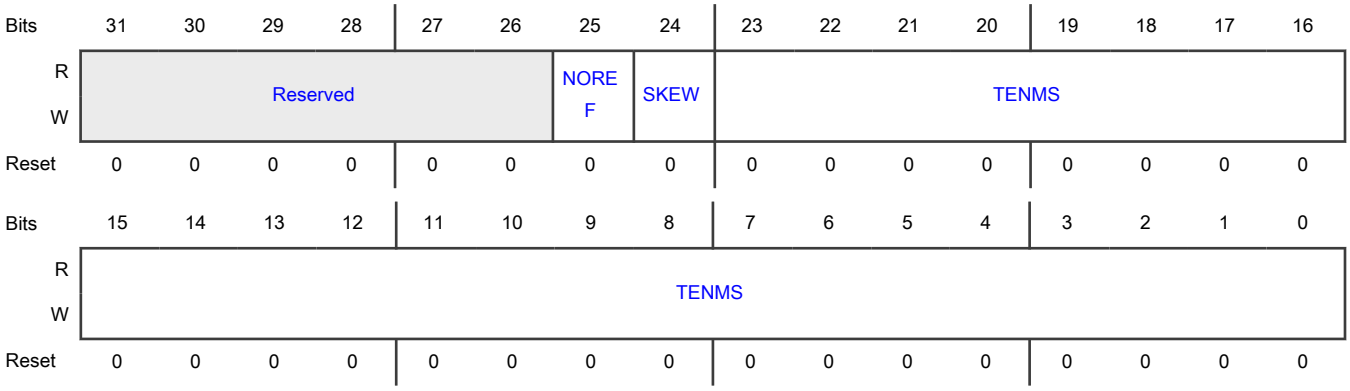
Offset

Register	Offset
CPU0NSTCKCAL	23Ch

Function

The CPU0NSTCKCAL register allows software to set up a default value for the SYST_CALIB register in the System Tick Timer of non-secure part of the CPU0.

Diagram



Fields

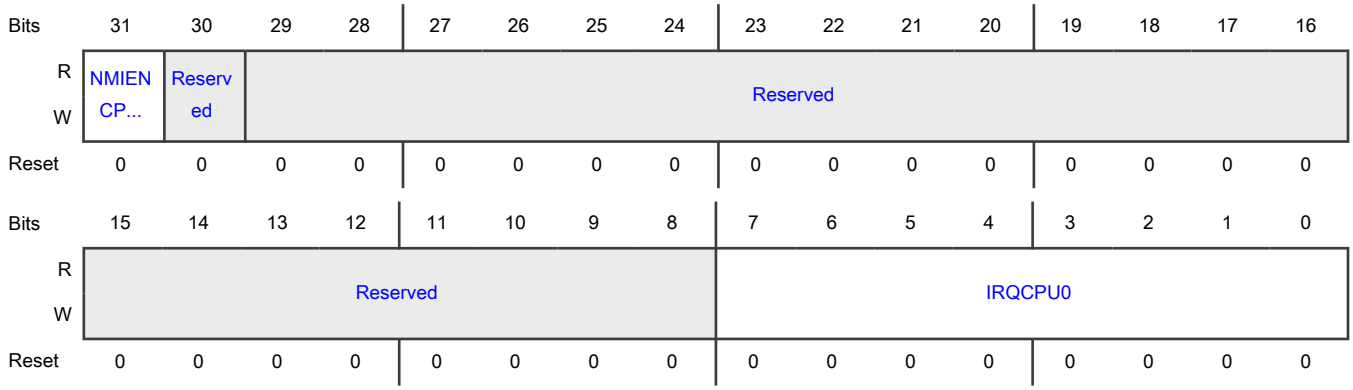
Field	Function
31-26 —	Reserved
25 NOREF	Indicates whether the device provides a reference clock to the processor. 0b - Reference clock is provided 1b - No reference clock is provided
24 SKEW	Indicates whether the TENMS value is exact. 0b - TENMS value is exact 1b - TENMS value is not exact or not given
23-0 TENMS	Reload value for 10 ms (100 Hz) timing, subject to system clock skew errors. If the value reads as zero, the calibration value is not known.

14.5.1.5 NMI Source Select (NMISRC)

Offset

Register	Offset
NMISRC	248h

Diagram



Fields

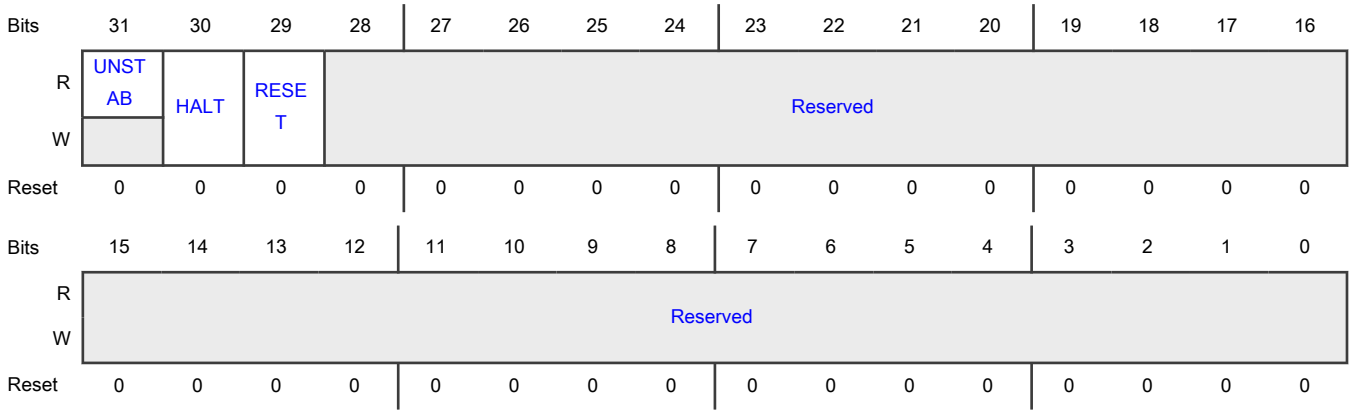
Field	Function
31 NMIENCPU0	Enables the Non-Maskable Interrupt (NMI) source selected by IRQCPU0. 0b - Disable. 1b - Enable.
30 —	Reserved
29-16 —	Reserved
15-8 —	Reserved
7-0 IRQCPU0	The IRQ number of the interrupt that acts as the Non-Maskable Interrupt (NMI) for CPU0, if enabled by NMIENCPU0.

14.5.1.6 SLOW_CLK Clock Divider (SLOWCLKDIV)

Offset

Register	Offset
SLOWCLKDIV	378h

Diagram



Fields

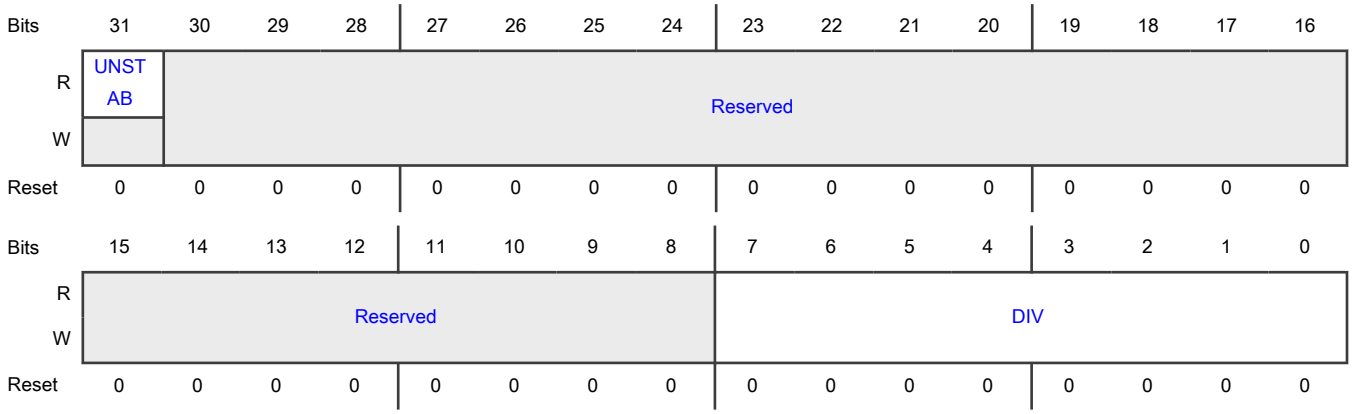
Field	Function
31 UNSTAB	Divider status flag 0b - Divider clock is stable 1b - Clock frequency is not stable
30 HALT	Halts the divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Resets the divider counter **NOTE** Limited to configure RESET bit when HALT bit is set to 1 0b - Divider is not reset 1b - Divider is reset
28-0 —	Reserved

14.5.1.7 System Clock Divider (AHBCLKDIV)

Offset

Register	Offset
AHBCLKDIV	380h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag 0b - Divider clock is stable 1b - Clock frequency is not stable
30-8 —	Reserved
7-0 DIV	Clock divider value The divider value = (DIV + 1)

14.5.1.8 Clock Configuration Unlock (CLKUNLOCK)

Offset

Register	Offset
CLKUNLOCK	3FCh

Function

This register controls access to the clock select and divider configuration registers.

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															UNLOCK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 UNLOCK	Controls clock configuration registers access (for example, MRCC_xxx_CLKDIV, MRCC_xxx_CLKSEL, MRCC_GLB_xxx) 0b - Updates are allowed to all clock configuration registers 1b - Freezes all clock configuration registers update.

14.5.1.9 NVM Control (NVM_CTRL)

Offset

Register	Offset
NVM_CTRL	400h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														DIS_MBE...	DIS_MBE...
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 W	Reserved					FLASH_S...	Reserved	Reserved					Reserved		DIS_DAT...	DIS_FLA...
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved
17 DIS_MBECC_ERR_DATA	Bus error on data multi-bit ECC error control **NOTE** Set this field to 0 if you want to enable flash speculative 0b - Enables bus error on multi-bit ECC error for data 1b - Disables bus error on multi-bit ECC error for data
16 DIS_MBECC_ERR_INST	Bus error on data multi-bit ECC error control **NOTE** Set this field to 0 if you want to enable flash speculative 0b - Enables bus error on multi-bit ECC error for instruction 1b - Disables bus error on multi-bit ECC error for instruction
15-11 —	Reserved
10 FLASH_STALL_EN	FLASH stall on busy control 0b - No stall on FLASH busy 1b - Stall on FLASH busy
9 —	Reserved Keep the default value.
8-6 —	Reserved
5-2 —	Reserved
1 DIS_DATA_SPEC	Flash data speculation control **NOTE** If DIS_MBECC_ERR_DATA and/or DIS_MBECC_ERR_INST are set, then speculation will not be enabled, even if this bit is cleared. 0b - Enables data speculation 1b - Disables data speculation
0	Flash speculation control

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIS_FLASH_SPEC	**NOTE** If DIS_MBECC_ERR_DATA and/or DIS_MBECC_ERR_INST are set, then speculation will not be enabled, even if this bit is cleared. 0b - Enables flash speculation 1b - Disables flash speculation

14.5.1.10 ROM Wait State (ROMCR)

Offset

Register	Offset
ROMCR	404h

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															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

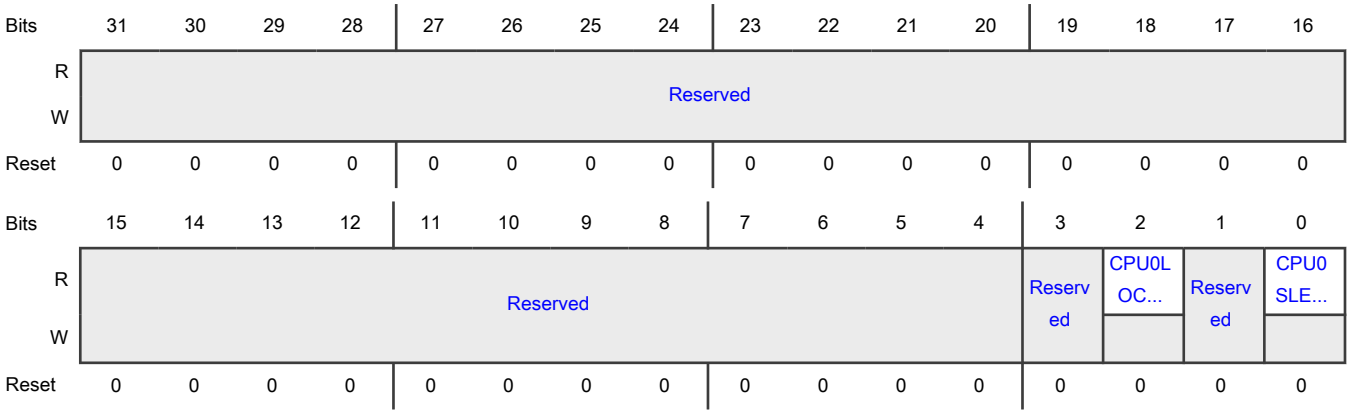
Field	Function
31-28 —	Reserved
27-0 —	Reserved

14.5.1.11 CPU Status (CPUSTAT)

Offset

Register	Offset
CPUSTAT	80Ch

Diagram



Fields

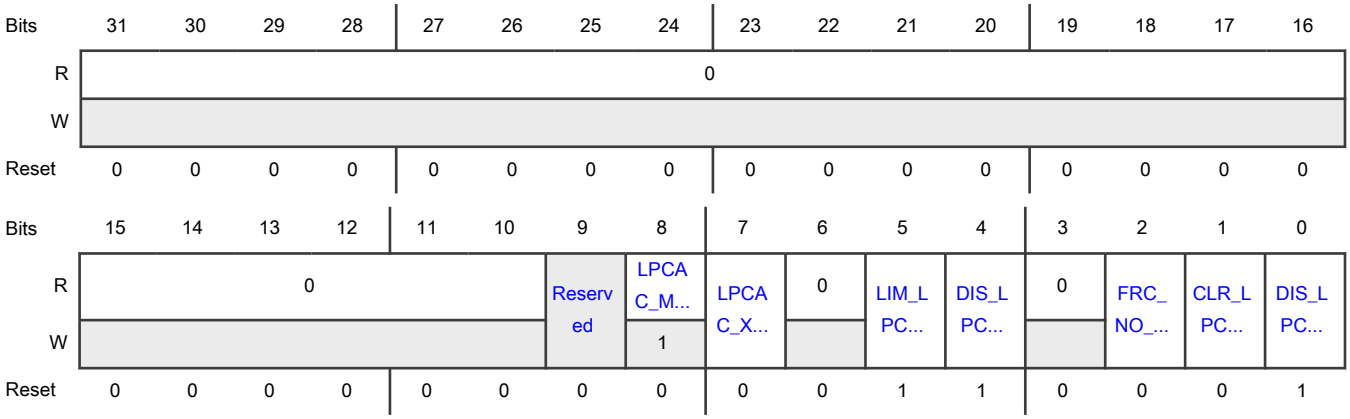
Field	Function
31-4 —	Reserved
3 —	Reserved
2 CPU0LOCKUP	CPU0 lockup state 0b - CPU is not in lockup 1b - CPU is in lockup
1 —	Reserved
0 CPU0SLEEPING	CPU0 sleeping state 0b - CPU is not sleeping 1b - CPU is sleeping

14.5.1.12 LPCAC Control (LPCAC_CTRL)

Offset

Register	Offset
LPCAC_CTRL	824h

Diagram



Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 LPCAC_MEM_REQ	Request LPCAC memories. 0b - Configure shared memories RAMX1 as general memories. 1b - Configure shared memories RAMX1 as LPCAC memories, write one lock until a system reset.
7 LPCAC_XOM	LPCAC XOM(eXecute-Only-Memory) attribute control Controls if the instruction fetch attribute is used as part of the address input to the LPCAC. When XOM regions in the internal flash are not configured at the MBC, then this option should be disabled so that instructions and data can be stored within the same cache line. This provides the best cache efficiency for non-XOM applications. When XOM areas in the internal flash are configured at the MBC, then this bit must be set so that instructions and data are cached using separate lines within the LPCAC. 0b - Disabled. 1b - Enabled.
6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5 LIM_LPCAC_WTBF	Limit LPCAC Write Through Buffer. 0b - Write buffer enabled when transaction is bufferable. 1b - Write buffer enabled when transaction is cacheable and bufferable
4 DIS_LPCAC_WTBF	Disable LPCAC Write Through Buffer. 0b - Enables write through buffer 1b - Disables write through buffer
3 —	Reserved
2 FRC_NO_ALLOC	Forces no allocation. 0b - Forces allocation 1b - Forces no allocation
1 CLR_LPCAC	Clears the cache function. 0b - Unclears the cache 1b - Clears the cache
0 DIS_LPCAC	Disables/enables the cache function. 0b - Enabled 1b - Disabled

14.5.1.13 PWM0 Submodule Control (PWM0SUBCTL)

Offset

Register	Offset
PWM0SUBCTL	938h

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													CLK3_EN	CLK2_EN	CLK1_EN	CLK0_EN
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-4 —	Reserved
3-0 CLKn_EN	Enables PWM0 SUB Clockn 0b - Disable 1b - Enable

14.5.1.14 PWM1 Submodule Control (PWM1SUBCTL)

Offset

Register	Offset
PWM1SUBCTL	93Ch

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													CLK3_EN	CLK2_EN	CLK1_EN	CLK0_EN
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-4 —	Reserved
3-0 CLKn_EN	Enables PWM1 SUB Clockn 0b - Disable 1b - Enable

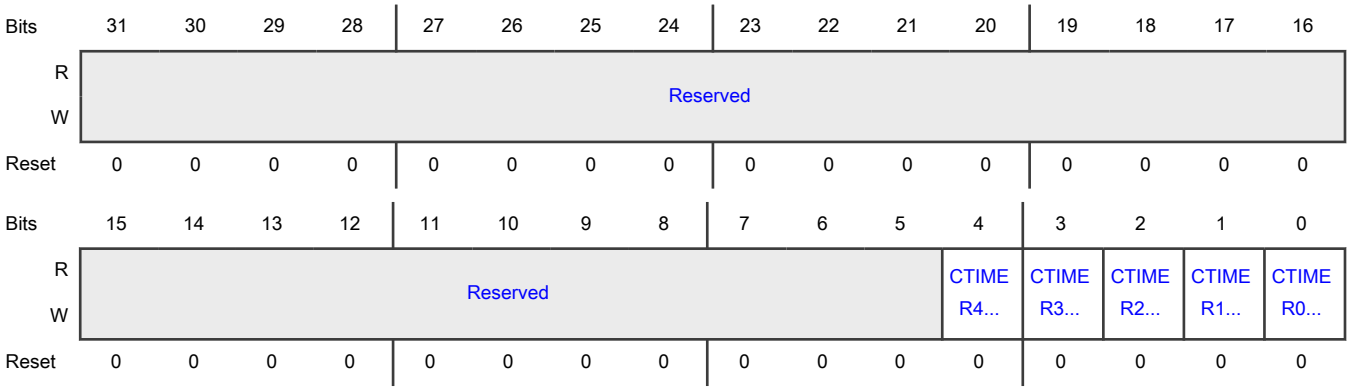
14.5.1.15 CTIMER Global Start Enable (CTIMERGLOBALSTARTEN)

Offset

Register	Offset
CTIMERGLOBALSTARTEN	940h

Function
Refer to CTIMERn TCR.AGCEN

Diagram



Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 CTIMER4_CLK_EN	Enables the CTIMER4 function clock 0b - Disable 1b - Enable
3 CTIMER3_CLK_EN	Enables the CTIMER3 function clock 0b - Disable 1b - Enable
2 CTIMER2_CLK_EN	Enables the CTIMER2 function clock 0b - Disable 1b - Enable
1 CTIMER1_CLK_EN	Enables the CTIMER1 function clock 0b - Disable 1b - Enable
0 CTIMER0_CLK_EN	Enables the CTIMER0 function clock 0b - Disable 1b - Enable

14.5.1.16 RAM Control (RAM_CTRL)

Offset

Register	Offset
RAM_CTRL	944h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								Reserved	Reserved	Reserved	Reserved	Reserved	RAMB_CG...	RAMX_CG...	RAMA_CG...
Reset	0	0	0	0	0	0	0	0	0	0	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	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	RAMA_EC...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 RAMB_CG_OVERRIDE	RAMB bank clock gating control 0b - Memory bank clock is gated automatically if no access more than 16 clock cycles 1b - Auto clock gating feature is disabled
17 RAMX_CG_OVERRIDE	RAMX bank clock gating control 0b - Memory bank clock is gated automatically if no access more than 16 clock cycles 1b - Auto clock gating feature is disabled
16 RAMA_CG_OVERRIDE	RAMA bank clock gating control, only avaiable when RAMA_ECC_ENABLE = 0. 0b - Memory bank clock is gated automatically if no access more than 16 clock cycles. Auto-clkgating using bank_adapter gives a timeout counter for each bank. Once it expires upon no access per specified width (16 cycles), then clock to the bank is stopped high and one cycle penalty will be incurred if a read is the first access after the break). A write will incur no penalty. 1b - Auto clock gating feature is disabled
15-8 —	Reserved
7 —	Reserved
6 —	Reserved
5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 RAMA_ECC_ENABLE	RAMA0 ECC enable 0b - ECC is disabled 1b - ECC is enabled

14.5.1.17 Gray to Binary Converter Gray Code [31:0] (GRAY_CODE_LSB)

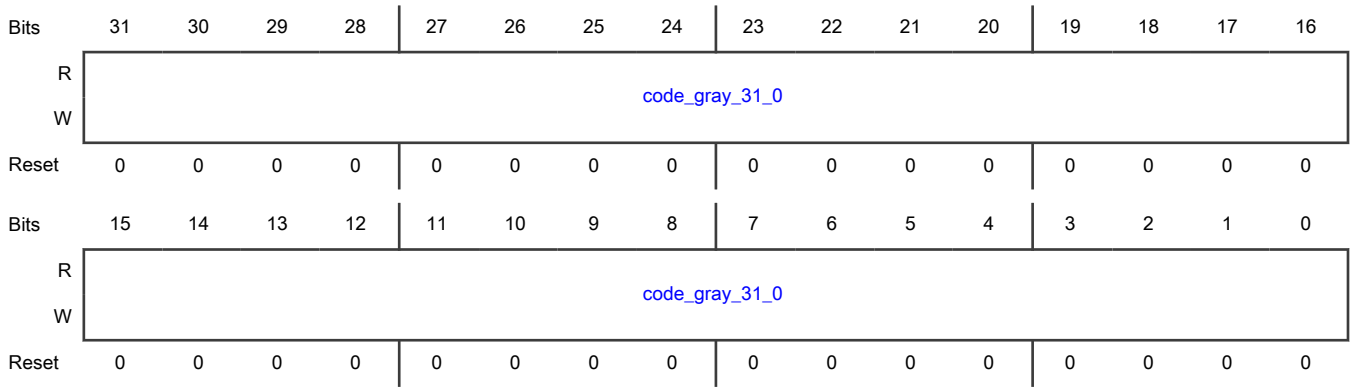
Offset

Register	Offset
GRAY_CODE_LSB	B60h

Function

The Gray Code LSB Input register (CODE_GRAY_LSB) contains the least-significant portion of the Gray code to be converted back to binary.

Diagram



Fields

Field	Function
31-0 code_gray_31_0	Gray code [31:0] CODE_GRAY_LSB is the least-significant 32 bits of the 42-bit Gray code to be converted.

14.5.1.18 Gray to Binary Converter Gray Code [41:32] (GRAY_CODE_MSB)

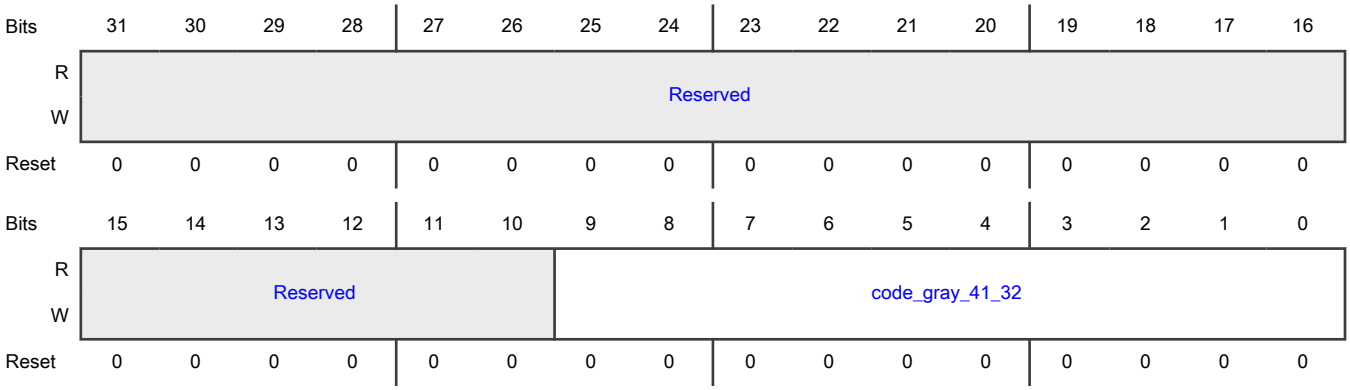
Offset

Register	Offset
GRAY_CODE_MSB	B64h

Function

The Gray Code MSB Input register (CODE_GRAY_MSB) contains the most-significant portion of the Gray code to be converted back to binary.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 code_gray_41_32	Gray code [41:32] CODE_GRAY_MSB is the most-significant 10 bits of the 42-bit Gray code to be converted.

14.5.1.19 Gray to Binary Converter Binary Code [31:0] (BINARY_CODE_LSB)

Offset

Register	Offset
BINARY_CODE_LSB	B68h

Function

The Binary Code LSB register (BINARY_CODE_LSB) contains the least-significant portion of the code converted from Gray to binary coding.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	code_bin_31_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	code_bin_31_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 code_bin_31_0	Binary code [31:0] code_bin_31_0 is the least-significant 32 bits of the 42-bit converted code.

14.5.1.20 Gray to Binary Converter Binary Code [41:32] (BINARY_CODE_MSB)

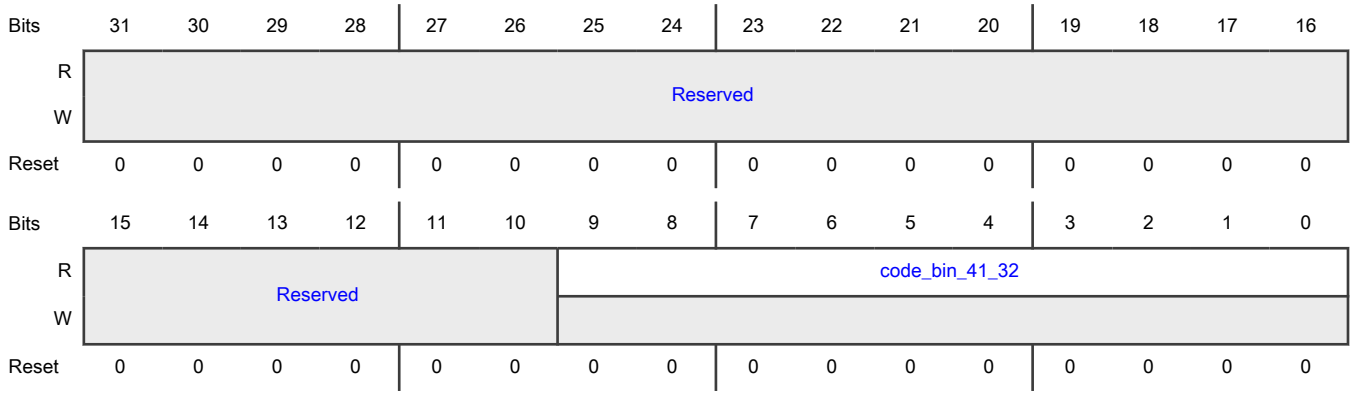
Offset

Register	Offset
BINARY_CODE_MSB	B6Ch

Function

The Binary Code MSB register (BINARY_CODE_MSB) contains the most-significant portion of the code converted from Gray to binary coding.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 code_bin_41_32	Binary code [41:32] code_bin_41_32 is the most-significant 10 bits of the 42-bit converted code.

14.5.1.21 ROP State Register (ROP_STATE)

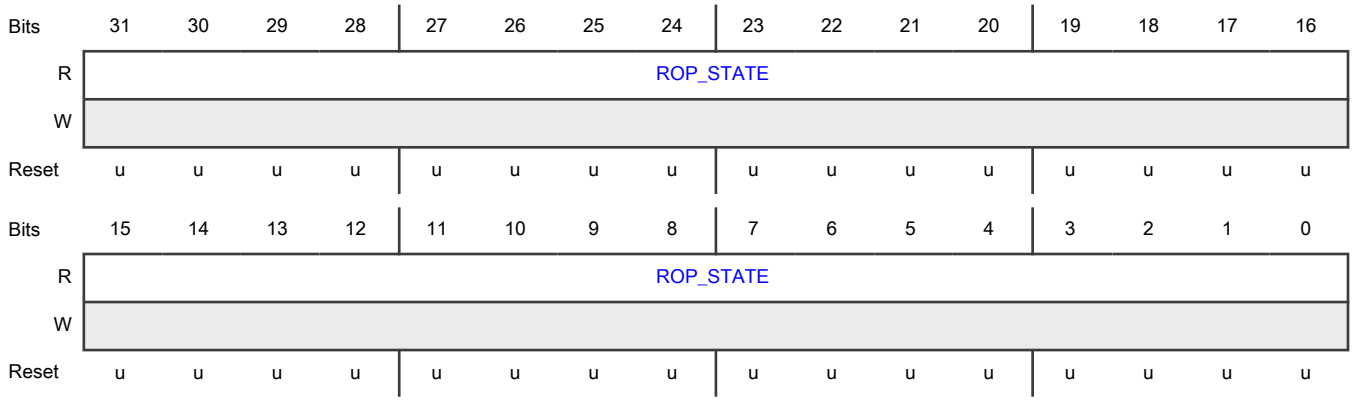
Offset

Register	Offset
ROP_STATE	E3Ch

Function

This register is written by Boot-ROM code and is only one-time write and locked down until a system reset.

Diagram



Fields

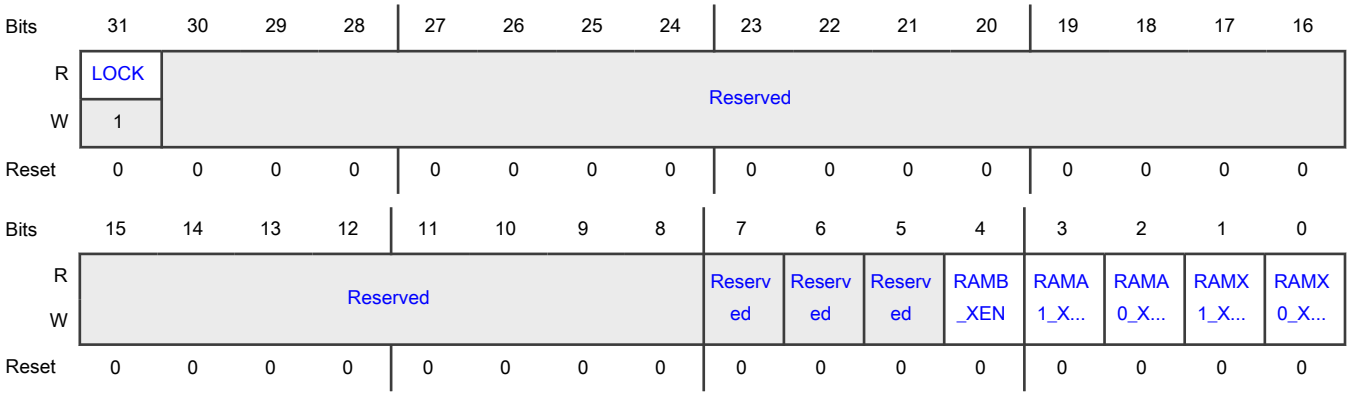
Field	Function
31-0 ROP_STATE	ROP state

14.5.1.22 RAM XEN Control (SRAM_XEN)

Offset

Register	Offset
SRAM_XEN	E58h

Diagram



Fields

Field	Function
31 LOCK	This 1-bit field provides a mechanism to limit writes to this register (and SRAM_XEN_DP) to protect its contents. Once set, this bit remains asserted until a system reset. 0b - This register is not locked and can be altered. 1b - This register is locked and cannot be altered.
30-8 —	Reserved
7 —	Reserved
6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 RAMB_XEN	RAMBx Execute permission control. 0b - Execute permission is disabled, R/W are enabled. 1b - Execute permission is enabled, R/W/X are enabled.
3 RAMA1_XEN	RAMAx (excepts RAMA0) Execute permission control. 0b - Execute permission is disabled, R/W are enabled. 1b - Execute permission is enabled, R/W/X are enabled.
2 RAMA0_XEN	RAMA0 Execute permission control. 0b - Execute permission is disabled, R/W are enabled. 1b - Execute permission is enabled, R/W/X are enabled.
1 RAMX1_XEN	RAMX1 Execute permission control. 0b - Execute permission is disabled, R/W are enabled. 1b - Execute permission is enabled, R/W/X are enabled.
0 RAMX0_XEN	RAMX0 Execute permission control. 0b - Execute permission is disabled, R/W are enabled. 1b - Execute permission is enabled, R/W/X are enabled.

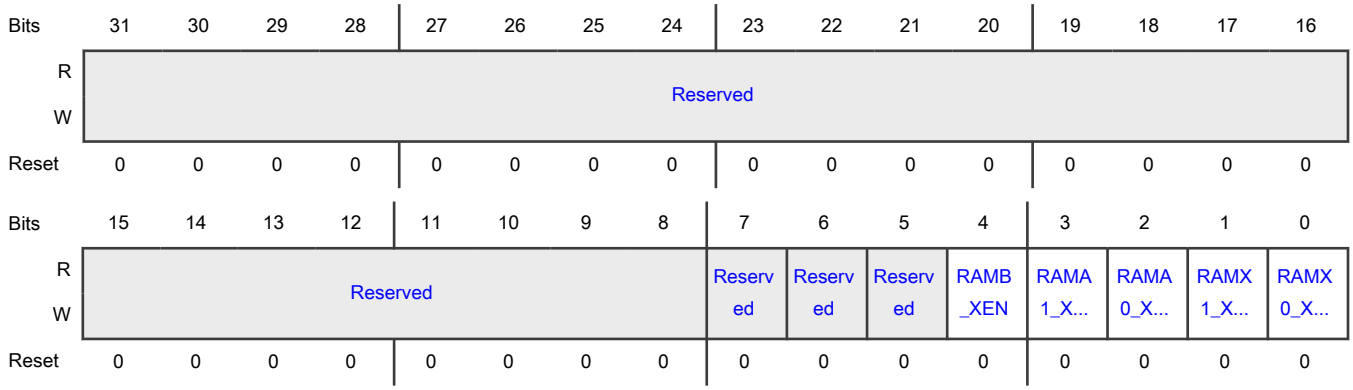
14.5.1.23 RAM XEN Control (Duplicate) (SRAM_XEN_DP)

Offset

Register	Offset
SRAM_XEN_DP	E5Ch

Function
Protected by GLIKEY CRTL_0.WRITE_INDEX[7:0]=2

Diagram



Fields

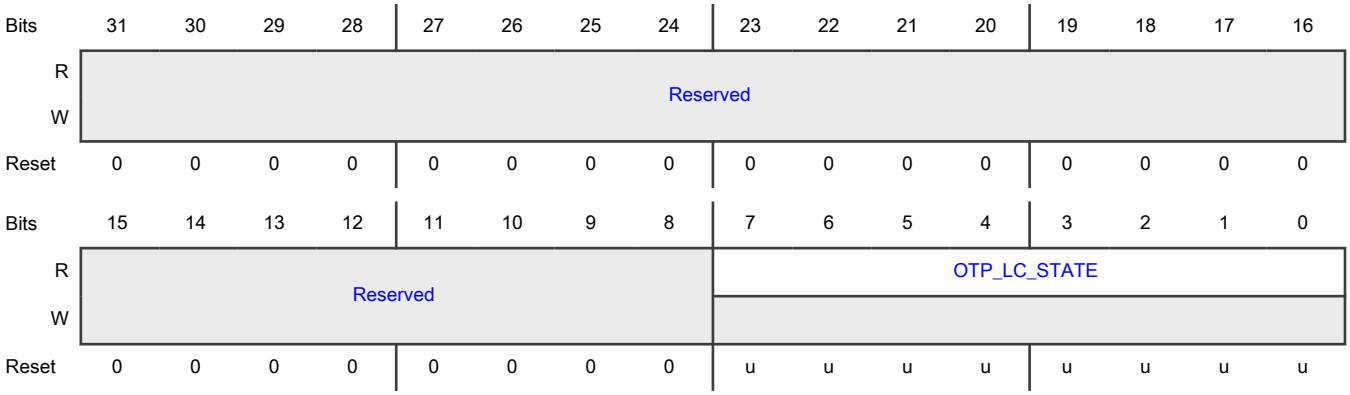
Field	Function
31-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 RAMB_XEN	Refer to SRAM_XEN for more details.
3 RAMA1_XEN	Refer to SRAM_XEN for more details.
2 RAMA0_XEN	Refer to SRAM_XEN for more details.
1 RAMX1_XEN	Refer to SRAM_XEN for more details.
0 RAMX0_XEN	Refer to SRAM_XEN for more details.

14.5.1.24 Life Cycle State Register (ELS_OTP_LC_STATE)

Offset

Register	Offset
ELS_OTP_LC_STATE	E80h

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 OTP_LC_STATE	OTP life cycle state

14.5.1.25 Life Cycle State Register (Duplicate) (ELS_OTP_LC_STATE_DP)

Offset

Register	Offset
ELS_OTP_LC_STATE_DP	E84h

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	Reserved								OTP_LC_STATE_DP							
W																
Reset	0	0	0	0	0	0	0	0	u	u	u	u	u	u	u	u

Fields

Field	Function
31-8 —	Reserved
7-0 OTP_LC_STATE_DP	OTP life cycle state

14.5.1.26 Control Write Access to Security (DEBUG_LOCK_EN)

Offset

Register	Offset
DEBUG_LOCK_EN	FA0h

Function

Controls write access to the SWD_ACCESS_CPU0, DEBUG_FEATURES, DEBUG_FEATURES_DP and DEBUG_AUTH_BEACON registers

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	Reserved												LOCK_ALL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

Fields

Field	Function
31-4 —	Reserved
3-0 LOCK_ALL	Controls write access to the security registers Controls write access to the SWD_ACCESS_CPU0, DEBUG_FEATURES, DEBUG_FEATURES_DP and DEBUG_AUTH_BEACON registers 0000b - Any other value than b1010: disables write access to all registers 1010b - Enables write access to all registers

14.5.1.27 Cortex Debug Features Control (DEBUG_FEATURES)

Offset

Register	Offset
DEBUG_FEATURES	FA4h

Function

Cortex M33 (CPU0) debug features control.

NOTE

Reset on PoR only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
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	Reserved		Reserved		Reserved				Reserved		Reserved		CPU0_NIDEN		CPU0_DBGEN	
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-14 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-12 —	Reserved
11-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 CPU0_NIDEN	CPU0 non-invasive debug control Enables non-invasive debug for CPU0. Values not listed are reserved. 01b - Disables debug 10b - Enables debug
1-0 CPU0_DBGEN	CPU0 invasive debug control Enables invasive debug for CPU0. Values not listed are reserved. 01b - Disables debug 10b - Enables debug

14.5.1.28 Cortex Debug Features Control (Duplicate) (DEBUG_FEATURES_DP)

Offset

Register	Offset
DEBUG_FEATURES_DP	FA8h

Function
Cortex M33 (CPU0) debug features control.

Diagram

Fields

Field	Function
31-14 —	Reserved
13-12 —	Reserved
11-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 CPU0_NIDEN	CPU0 non-invasive debug control Enables non-invasive debug for CPU0. Values not listed are reserved. 01b - Disables debug 10b - Enables debug
1-0 CPU0_DBGEN	CPU0 invasive debug control Enables invasive debug for CPU0. Values not listed are reserved. 01b - Disables debug 10b - Enables debug

14.5.1.29 CPU0 Software Debug Access (SWD_ACCESS_CPU0)

Offset

Register	Offset
SWD_ACCESS_CPU0	FB4h

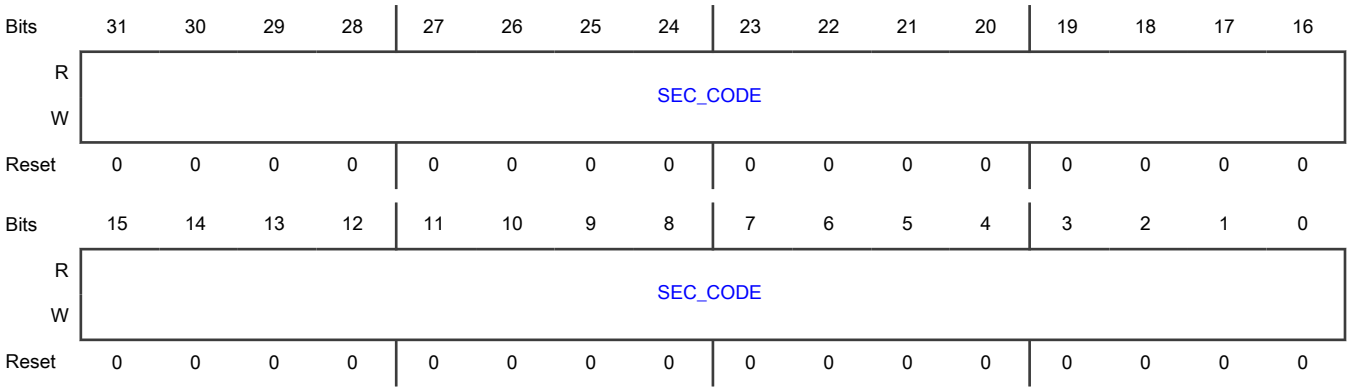
Function

This register is used by ROM during DEBUG authentication mechanism to enable debug access port for CPU0. write once only.

NOTE

Reset on PoR only

Diagram



Fields

Field	Function
31-0 SEC_CODE	CPU0 SWD-AP: 0x12345678 0000_0000_0000_0000_0000_0000_0000_0000b - CPU0 DAP is not allowed. Reading back register is read as 0x5. 0001_0010_0011_0100_0101_0110_0111_1000b - Value to write to enable CPU0 SWD access. Reading back register is read as 0xA.

14.5.1.30 Debug Authentication BEACON (DEBUG_AUTH_BEACON)

Offset

Register	Offset
DEBUG_AUTH_BEACON	FC0h

Function

This register protected by security. ROM sets register (read-only) with value received in debug credentials before passing control to the user code. This can be used to extend debug authentication control for the customer application.

NOTE

Reset on PoR only

Diagram

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

Fields

Field	Function
31-0 BEACON	Sets by the debug authentication code in ROM to pass the debug beacons (Credential Beacon and Authentication Beacon) to the application code.

14.5.1.31 JTAG Chip ID (JTAG_ID)

Offset

Register	Offset
JTAG_ID	FF0h

Diagram

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

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

Fields

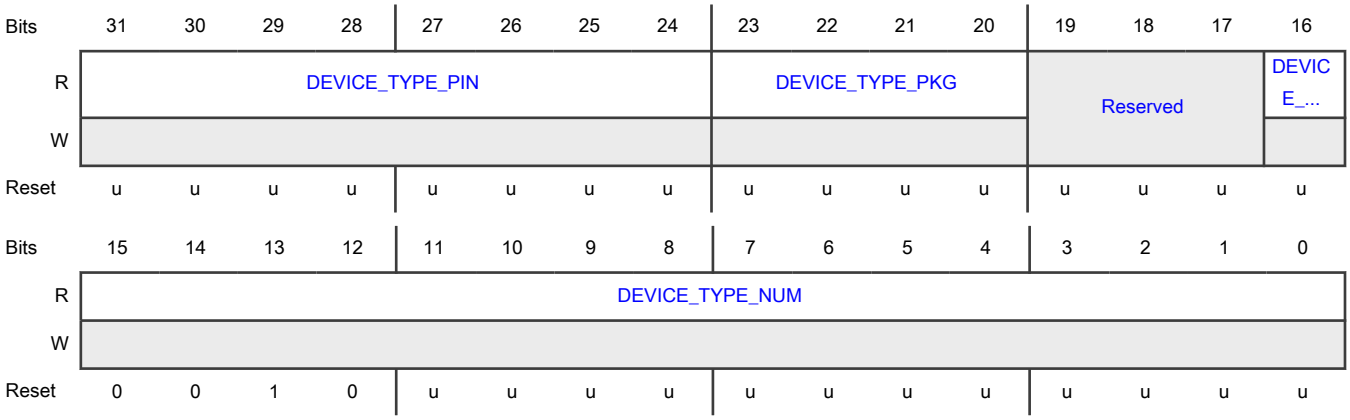
Field	Function
31-0 JTAG_ID	Indicates the device ID

14.5.1.32 Device Type (DEVICE_TYPE)

Offset

Register	Offset
DEVICE_TYPE	FF4h

Diagram



Fields

Field	Function
31-24 DEVICE_TYPE_PIN	Indicates the device's pin number
23-20 DEVICE_TYPE_PKG	Indicates the device's package type 0000b - HLQFP 0001b - HTQFP 0010b - BGA 0011b - HDQFP 0100b - QFN 0101b - CSP 0110b - LQFP

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-17 —	Reserved
16 DEVICE_TYPE_SEC	Indicates the device type 0b - Non Secure 1b - Secure
15-0 DEVICE_TYPE_NUM	Indicates the device part number [15:12]: ... 0010b: Axxx ... [11:0]: 3 digital of the part unmber. For example: A153: 0x2153

14.5.1.33 Device ID (DEVICE_ID0)

Offset

Register	Offset
DEVICE_ID0	FF8h

Function

This register contains the device ID.

Diagram

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

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

Fields

Field	Function
31-28 —	Reserved
27-24 SECURITY	0101b - Secure version. 1010b - Non secure version.
23-20 ROM_REV_MINOR	Indicates the device's ROM revision
19-8 —	Reserved
7-4 FLASH_SIZE	Indicates the device's flash size 0000b - 32KB. 0001b - 64KB. 0010b - 128KB. 0011b - 256KB. 0100b - 512KB. 0101b - 768KB. 0110b - 1MB. 0111b - 1.5MB. 1000b - 2MB.
3-0 RAM_SIZE	Indicates the device's ram size 0000b - 8KB. 0001b - 16KB. 0010b - 32KB. 0011b - 64KB. 0100b - 96KB. 0101b - 128KB. 0110b - 160KB. 0111b - 192KB. 1000b - 256KB. 1001b - 288KB. 1010b - 352KB. 1011b - 512KB.

14.5.1.34 Chip Revision ID and Number (DIEID)

Offset

Register	Offset
DIEID	FFCh

Function

This register contains the chip number and revision.

Diagram

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

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

Fields

Field	Function
31-28 —	Reserved
27-8 MCO_NUM_IN_DIE_ID	Chip number
7-4 MAJOR_REVISION	Chip major revision
3-0 MINOR_REVISION	Chip minor revision

14.5.2 MRCC register descriptions

14.5.2.1 MRCC memory map

SYSCON.MRCC base address: 4009_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Peripheral Reset Control 0 (MRCC_GLB_RST0)	32	RW	0000_0000h
4h	Peripheral Reset Control Set 0 (MRCC_GLB_RST0_SET)	32	W	0000_0000h
8h	Peripheral Reset Control Clear 0 (MRCC_GLB_RST0_CLR)	32	W	0000_0000h
10h	Peripheral Reset Control 1 (MRCC_GLB_RST1)	32	RW	0000_0000h
14h	Peripheral Reset Control Set 1 (MRCC_GLB_RST1_SET)	32	W	0000_0000h
18h	Peripheral Reset Control Clear 1 (MRCC_GLB_RST1_CLR)	32	W	0000_0000h
40h	AHB Clock Control 0 (MRCC_GLB_CC0)	32	RW	0000_8000h
44h	AHB Clock Control Set 0 (MRCC_GLB_CC0_SET)	32	W	0000_8000h
48h	AHB Clock Control Clear 0 (MRCC_GLB_CC0_CLR)	32	W	0000_8000h
50h	AHB Clock Control 1 (MRCC_GLB_CC1)	32	RW	0000_0000h
54h	AHB Clock Control Set 1 (MRCC_GLB_CC1_SET)	32	W	0000_0000h
58h	AHB Clock Control Clear 1 (MRCC_GLB_CC1_CLR)	32	W	0000_0000h
80h	Control Automatic Clock Gating 0 (MRCC_GLB_ACC0)	32	RW	0000_8200h
84h	Control Automatic Clock Gating 1 (MRCC_GLB_ACC1)	32	RW	020C_0000h
A0h	I3C0_FCLK clock selection control (MRCC_I3C0_FCLK_CLKSEL)	32	RW	0000_0007h
A4h	I3C0_FCLK clock divider control (MRCC_I3C0_FCLK_CLKDIV)	32	RW	4000_0000h
A8h	CTIMER0 clock selection control (MRCC_CTIMER0_CLKSEL)	32	RW	0000_0007h
ACh	CTIMER0 clock divider control (MRCC_CTIMER0_CLKDIV)	32	RW	4000_0000h
B0h	CTIMER1 clock selection control (MRCC_CTIMER1_CLKSEL)	32	RW	0000_0007h
B4h	CTIMER1 clock divider control (MRCC_CTIMER1_CLKDIV)	32	RW	4000_0000h
B8h	CTIMER2 clock selection control (MRCC_CTIMER2_CLKSEL)	32	RW	0000_0007h
BCh	CTIMER2 clock divider control (MRCC_CTIMER2_CLKDIV)	32	RW	4000_0000h
C0h	CTIMER3 clock selection control (MRCC_CTIMER3_CLKSEL)	32	RW	0000_0007h
C4h	CTIMER3 clock divider control (MRCC_CTIMER3_CLKDIV)	32	RW	4000_0000h
C8h	CTIMER4 clock selection control (MRCC_CTIMER4_CLKSEL)	32	RW	0000_0007h
CCh	CTIMER4 clock divider control (MRCC_CTIMER4_CLKDIV)	32	RW	4000_0000h
D4h	WWDT0 clock divider control (MRCC_WWDT0_CLKDIV)	32	RW	0000_0000h
D8h	FLEXIO0 clock selection control (MRCC_FLEXIO0_CLKSEL)	32	RW	0000_0007h
DCh	FLEXIO0 clock divider control (MRCC_FLEXIO0_CLKDIV)	32	RW	4000_0000h
E0h	LPI2C0 clock selection control (MRCC_LPI2C0_CLKSEL)	32	RW	0000_0007h
E4h	LPI2C0 clock divider control (MRCC_LPI2C0_CLKDIV)	32	RW	4000_0000h
E8h	LPI2C1 clock selection control (MRCC_LPI2C1_CLKSEL)	32	RW	0000_0007h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
ECh	LPI2C1 clock divider control (MRCC_LPI2C1_CLKDIV)	32	RW	4000_0000h
F0h	LPSPI0 clock selection control (MRCC_LPSPI0_CLKSEL)	32	RW	0000_0007h
F4h	LPSPI0 clock divider control (MRCC_LPSPI0_CLKDIV)	32	RW	4000_0000h
F8h	LPSPI1 clock selection control (MRCC_LPSPI1_CLKSEL)	32	RW	0000_0007h
FCh	LPSPI1 clock divider control (MRCC_LPSPI1_CLKDIV)	32	RW	4000_0000h
100h	LPUART0 clock selection control (MRCC_LPUART0_CLKSEL)	32	RW	0000_0007h
104h	LPUART0 clock divider control (MRCC_LPUART0_CLKDIV)	32	RW	4000_0000h
108h	LPUART1 clock selection control (MRCC_LPUART1_CLKSEL)	32	RW	0000_0007h
10Ch	LPUART1 clock divider control (MRCC_LPUART1_CLKDIV)	32	RW	4000_0000h
110h	LPUART2 clock selection control (MRCC_LPUART2_CLKSEL)	32	RW	0000_0007h
114h	LPUART2 clock divider control (MRCC_LPUART2_CLKDIV)	32	RW	4000_0000h
118h	LPUART3 clock selection control (MRCC_LPUART3_CLKSEL)	32	RW	0000_0007h
11Ch	LPUART3 clock divider control (MRCC_LPUART3_CLKDIV)	32	RW	4000_0000h
120h	LPUART4 clock selection control (MRCC_LPUART4_CLKSEL)	32	RW	0000_0007h
124h	LPUART4 clock divider control (MRCC_LPUART4_CLKDIV)	32	RW	4000_0000h
128h	USB0 clock selection control (MRCC_USB0_CLKSEL)	32	RW	0000_0003h
130h	LPTMR0 clock selection control (MRCC_LPTMR0_CLKSEL)	32	RW	0000_0007h
134h	LPTMR0 clock divider control (MRCC_LPTMR0_CLKDIV)	32	RW	4000_0000h
138h	OSTIMER0 clock selection control (MRCC_OSTIMER0_CLKSEL)	32	RW	0000_0003h
140h	ADC0 clock selection control (MRCC_ADC0_CLKSEL)	32	RW	0000_0007h
144h	ADC0 clock divider control (MRCC_ADC0_CLKDIV)	32	RW	4000_0000h
148h	ADC1 clock selection control (MRCC_ADC1_CLKSEL)	32	RW	0000_0007h
14Ch	ADC1 clock divider control (MRCC_ADC1_CLKDIV)	32	RW	4000_0000h
154h	CMP0_FUNC clock divider control (MRCC_CMP0_FUNC_CLKDIV)	32	RW	4000_0000h
158h	CMP0_RR clock selection control (MRCC_CMP0_RR_CLKSEL)	32	RW	0000_0007h
15Ch	CMP0_RR clock divider control (MRCC_CMP0_RR_CLKDIV)	32	RW	4000_0000h
164h	CMP1_FUNC clock divider control (MRCC_CMP1_FUNC_CLKDIV)	32	RW	4000_0000h
168h	CMP1_RR clock selection control (MRCC_CMP1_RR_CLKSEL)	32	RW	0000_0007h
16Ch	CMP1_RR clock divider control (MRCC_CMP1_RR_CLKDIV)	32	RW	4000_0000h
170h	DAC0 clock selection control (MRCC_DAC0_CLKSEL)	32	RW	0000_0007h
174h	DAC0 clock divider control (MRCC_DAC0_CLKDIV)	32	RW	4000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
178h	FLEXCAN0 clock selection control (MRCC_FLEXCAN0_CLKSEL)	32	RW	0000_0007h
17Ch	FLEXCAN0 clock divider control (MRCC_FLEXCAN0_CLKDIV)	32	RW	4000_0000h
180h	LPI2C2 clock selection control (MRCC_LPI2C2_CLKSEL)	32	RW	0000_0007h
184h	LPI2C2 clock divider control (MRCC_LPI2C2_CLKDIV)	32	RW	4000_0000h
188h	LPI2C3 clock selection control (MRCC_LPI2C3_CLKSEL)	32	RW	0000_0007h
18Ch	LPI2C3 clock divider control (MRCC_LPI2C3_CLKDIV)	32	RW	4000_0000h
190h	DBG_TRACE clock selection control (MRCC_DBG_TRACE_CLKSEL)	32	RW	0000_0000h
194h	DBG_TRACE clock divider control (MRCC_DBG_TRACE_CLKDIV)	32	RW	0000_0000h
198h	CLKOUT clock selection control (MRCC_CLKOUT_CLKSEL)	32	RW	0000_0007h
19Ch	CLKOUT clock divider control (MRCC_CLKOUT_CLKDIV)	32	RW	4000_0000h
1A0h	SYSTICK clock selection control (MRCC_SYSTICK_CLKSEL)	32	RW	0000_0003h
1A4h	SYSTICK clock divider control (MRCC_SYSTICK_CLKDIV)	32	RW	4000_0000h
1ACh	FRO_HF_DIV clock divider control (MRCC_FRO_HF_DIV_CLKDIV)	32	RW	0000_0000h

14.5.2.2 Peripheral Reset Control 0 (MRCC_GLB_RST0)

Offset

Register	Offset
MRCC_GLB_RST0	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FLEXPWM1	FLEXPWM0	QDC1	QDC0	USB0	LPUART4	LPUART3	LPUART2	LPUART1	LPUART0	LPSPI1	LPSPI0	LPI2C1	LPI2C0	FLEXIO0	AOI1
Reset	0	0	0	0	0	0	0	0	0	0	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	ERM0	EIM0	CRC0	AOI0	DMA	Reserved	UTICK0	FREQME	CTIMER4	CTIMER3	CTIMER2	CTIMER1	CTIMER0	I3C0	INPUTMU...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FLEXPWM1	FLEXPWM1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
30 FLEXPWM0	FLEXPWM0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
29 QDC1	QDC1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
28 QDC0	QDC0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
27 USB0	USB0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
26 LPUART4	LPUART4 0b - Peripheral is held in reset 1b - Peripheral is released from reset
25 LPUART3	LPUART3 0b - Peripheral is held in reset 1b - Peripheral is released from reset
24 LPUART2	LPUART2 0b - Peripheral is held in reset 1b - Peripheral is released from reset
23 LPUART1	LPUART1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
22 LPUART0	LPUART0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
21 LPSPI1	LPSPI1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Peripheral is held in reset 1b - Peripheral is released from reset
20 LPSPI0	LPSPI0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
19 LPI2C1	LPI2C1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
18 LPI2C0	LPI2C0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
17 FLEXIO0	FLEXIO0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
16 AOI1	AOI1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
15 —	reserved reserved
14 ERM0	ERM0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
13 EIM0	EIM0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
12 CRC0	CRC0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
11 AOI0	AOI0 0b - Peripheral is held in reset 1b - Peripheral is released from reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 DMA	DMA 0b - Peripheral is held in reset 1b - Peripheral is released from reset
9 —	reserved reserved
8 UTICK0	UTICK0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
7 FREQME	FREQME 0b - Peripheral is held in reset 1b - Peripheral is released from reset
6 CTIMER4	CTIMER4 0b - Peripheral is held in reset 1b - Peripheral is released from reset
5 CTIMER3	CTIMER3 0b - Peripheral is held in reset 1b - Peripheral is released from reset
4 CTIMER2	CTIMER2 0b - Peripheral is held in reset 1b - Peripheral is released from reset
3 CTIMER1	CTIMER1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
2 CTIMER0	CTIMER0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
1 I3C0	I3C0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
0 INPUTMUX0	INPUTMUX0 0b - Peripheral is held in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Peripheral is released from reset

14.5.2.3 Peripheral Reset Control Set n (MRCC_GLB_RST0_SET - MRCC_GLB_RST1_SET)

Offset

Register	Offset
MRCC_GLB_RST0_SET	4h
MRCC_GLB_RST1_SET	14h

Function

Writing a 1 to a bit position in a write-only MRCC_GLB_RSTn_SET register sets the corresponding position in MRCC_GLB_RSTn.

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 array value, refer to corresponding position in MRCC_GLB_RSTn.

14.5.2.4 Peripheral Reset Control Clear n (MRCC_GLB_RST0_CLR - MRCC_GLB_RST1_CLR)

Offset

Register	Offset
MRCC_GLB_RST0_CLR	8h
MRCC_GLB_RST1_CLR	18h

Function

Writing a 1 to a bit position in a write-only MRCC_GLB_RSTn_CLR register clears the corresponding position in MRCC_GLB_RSTn.

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 array value, refer to corresponding position in MRCC_GLB_RSTn.

14.5.2.5 Peripheral Reset Control 1 (MRCC_GLB_RST1)

Offset

Register	Offset
MRCC_GLB_RST1	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							GPIO4	GPIO3	GPIO2	GPIO1	GPIO0	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	LPI2C3	LPI2C2	FLEXCAN0	PORT4	PORT3	PORT2	PORT1	PORT0	OPAMP0	DAC0	CMP1	Reserved	ADC1	ADC0	OSTIMER0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	reserved reserved
24 GPIO4	GPIO4 0b - Peripheral is held in reset 1b - Peripheral is released from reset
23 GPIO3	GPIO3 0b - Peripheral is held in reset 1b - Peripheral is released from reset
22 GPIO2	GPIO2 0b - Peripheral is held in reset 1b - Peripheral is released from reset
21 GPIO1	GPIO1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
20 GPIO0	GPIO0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
19-16 —	reserved reserved
15 —	Reserved
14 LPI2C3	LPI2C3 0b - Peripheral is held in reset 1b - Peripheral is released from reset
13 LPI2C2	LPI2C2 0b - Peripheral is held in reset 1b - Peripheral is released from reset
12 FLEXCAN0	FLEXCAN0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
11	PORT4

Table continues on the next page...

Table continued from the previous page...

Field	Function
PORT4	0b - Peripheral is held in reset 1b - Peripheral is released from reset
10 PORT3	PORT3 0b - Peripheral is held in reset 1b - Peripheral is released from reset
9 PORT2	PORT2 0b - Peripheral is held in reset 1b - Peripheral is released from reset
8 PORT1	PORT1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
7 PORT0	PORT0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
6 OPAMP0	OPAMP0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
5 DAC0	DAC0 0b - Peripheral is held in reset 1b - Peripheral is released from reset
4 CMP1	CMP1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
3 —	reserved reserved
2 ADC1	ADC1 0b - Peripheral is held in reset 1b - Peripheral is released from reset
1 ADC0	ADC0 0b - Peripheral is held in reset 1b - Peripheral is released from reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 OSTIMER0	OSTIMER0 0b - Peripheral is held in reset 1b - Peripheral is released from reset

14.5.2.6 AHB Clock Control 0 (MRCC_GLB_CC0)

Offset

Register	Offset
MRCC_GLB_CC0	40h

Function

Both MRCC_GLB_CCx and MRCC_GLB_ACCx are used to control modules' clocks, see Application information for usage.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FLEXPWM1	FLEXPWM0	QDC1	QDC0	USB0	LPUART4	LPUART3	LPUART2	LPUART1	LPUART0	LPSPI1	LPSPI0	LPI2C1	LPI2C0	FLEXIO0	AOI1
Reset	0	0	0	0	0	0	0	0	0	0	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	FMC	ERM0	EIM0	CRC0	AOI0	DMA	WWDT0	UTICK0	FREQME	CTIMER4	CTIMER3	CTIMER2	CTIMER1	CTIMER0	I3C0	INPUTMU...
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FLEXPWM1	FLEXPWM1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
30 FLEXPWM0	FLEXPWM0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
29	QDC1

Table continues on the next page...

Table continued from the previous page...

Field	Function
QDC1	0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
28 QDC0	QDC0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
27 USB0	USB0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
26 LPUART4	LPUART4 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
25 LPUART3	LPUART3 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
24 LPUART2	LPUART2 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
23 LPUART1	LPUART1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
22 LPUART0	LPUART0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
21 LPSPI1	LPSPI1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
20 LPSPI0	LPSPI0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
19 LPI2C1	LPI2C1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 LPI2C0	LPI2C0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
17 FLEXIO0	FLEXIO0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
16 AOI1	AOI1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
15 FMC	FMC 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
14 ERM0	ERM0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
13 EIM0	EIM0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
12 CRC0	CRC0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
11 AOI0	AOI0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
10 DMA	DMA 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
9 WWDT0	WWDT0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
8 UTICK0	UTICK0

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
7 FREQME	FREQME 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
6 CTIMER4	CTIMER4 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
5 CTIMER3	CTIMER3 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
4 CTIMER2	CTIMER2 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
3 CTIMER1	CTIMER1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
2 CTIMER0	CTIMER0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
1 I3C0	I3C0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
0 INPUTMUX0	INPUTMUX0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled

14.5.2.7 AHB Clock Control Set 0 (MRCC_GLB_CC0_SET)

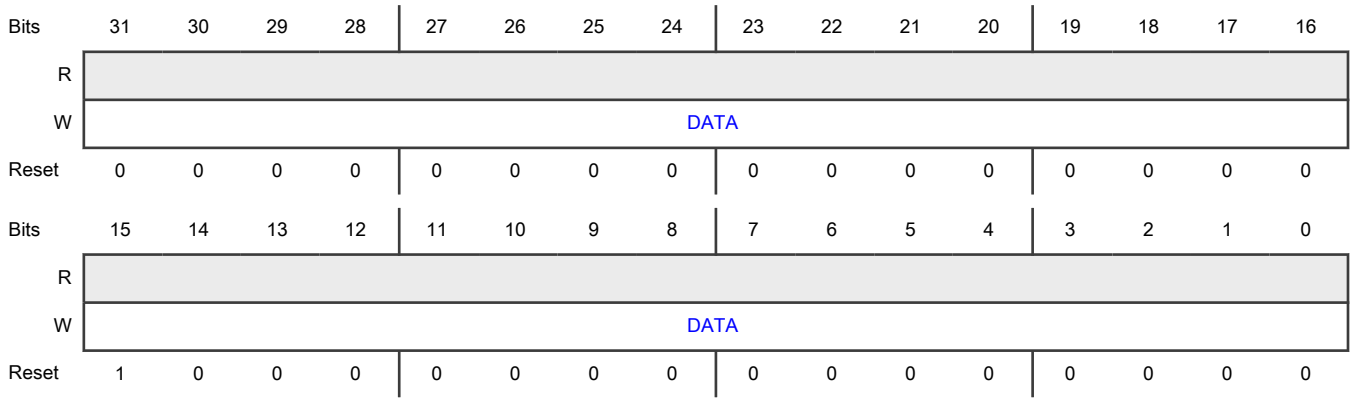
Offset

Register	Offset
MRCC_GLB_CC0_SET	44h

Function

Writing a 1 to a bit position in a write-only MRCC_GLB_CCn_SET register sets the corresponding position in MRCC_GLB_CCn.

Diagram



Fields

Field	Function
31-0 DATA	Data array value, refer to corresponding position in MRCC_GLB_CCn.

14.5.2.8 AHB Clock Control Clear 0 (MRCC_GLB_CC0_CLR)

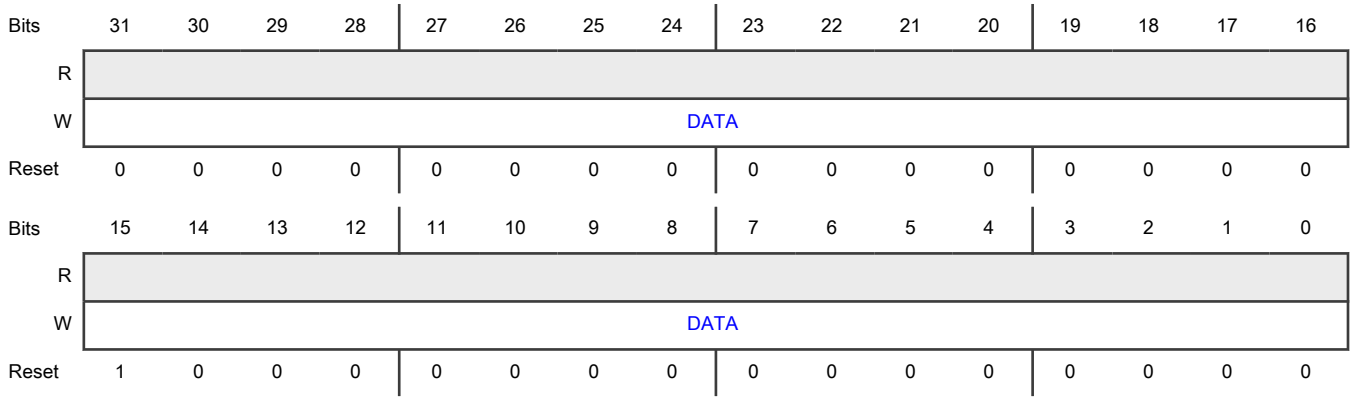
Offset

Register	Offset
MRCC_GLB_CC0_CLR	48h

Function

Writing a 1 to a bit position in a write-only MRCC_GLB_CCn_CLR register clears the corresponding position in MRCC_GLB_CCn.

Diagram



Fields

Field	Function
31-0 DATA	Data array value, refer to corresponding position in MRCC_GLB_CCn.

14.5.2.9 AHB Clock Control 1 (MRCC_GLB_CC1)

Offset

Register	Offset
MRCC_GLB_CC1	50h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved						ROMC	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0	RAMB	RAMA	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	LPI2C3	LPI2C2	FLEXCAN0	PORT4	PORT3	PORT2	PORT1	PORT0	OPAMP0	DAC0	CMP1	CMP0	ADC1	ADC0	OSTIMER0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	reserved reserved
25 ROMC	ROMC 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
24 GPIO4	GPIO4 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
23 GPIO3	GPIO3 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 GPIO2	GPIO2 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
21 GPIO1	GPIO1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
20 GPIO0	GPIO0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
19 RAMB	RAMB 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
18 RAMA	RAMA 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
17-15 —	Reserved
14 LPI2C3	LPI2C3 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
13 LPI2C2	LPI2C2 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
12 FLEXCAN0	FLEXCAN0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
11 PORT4	PORT4 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
10 PORT3	PORT3 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 PORT2	PORT2 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
8 PORT1	PORT1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
7 PORT0	PORT0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
6 OPAMP0	OPAMP0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
5 DAC0	DAC0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
4 CMP1	CMP1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
3 CMP0	CMP0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
2 ADC1	ADC1 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
1 ADC0	ADC0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled
0 OSTIMER0	OSTIMER0 0b - Peripheral clock is disabled 1b - Peripheral clock is enabled

14.5.2.10 AHB Clock Control Set 1 (MRCC_GLB_CC1_SET)

Offset

Register	Offset
MRCC_GLB_CC1_SET	54h

Function

Writing a 1 to a bit position in a write-only MRCC_GLB_CCn_SET register sets the corresponding position in MRCC_GLB_CCn.

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 array value, refer to corresponding position in MRCC_GLB_CCn.

14.5.2.11 AHB Clock Control Clear 1 (MRCC_GLB_CC1_CLR)

Offset

Register	Offset
MRCC_GLB_CC1_CLR	58h

Function

Writing a 1 to a bit position in a write-only MRCC_GLB_CCn_CLR register clears the corresponding position in MRCC_GLB_CCn.

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 array value, refer to corresponding position in MRCC_GLB_CCn.

14.5.2.12 Control Automatic Clock Gating 0 (MRCC_GLB_ACC0)

Offset

Register	Offset
MRCC_GLB_ACC0	80h

Function

Both MRCC_GLB_CCx and MRCC_GLB_ACCx are used to control modules' clocks, see Application information for usage.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FLEXPWM1	FLEXPWM0	QDC1	QDC0	USB0	LPUART4	LPUART3	LPUART2	LPUART1	LPUART0	LPSPI1	LPSPI0	LPI2C1	LPI2C0	FLEXIO0	AOI1
Reset	0	0	0	0	0	0	0	0	0	0	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	FMC	ERM0	EIM0	CRC0	AOI0	DMA	WWDT0	UTICK0	FREQME	CTIMER4	CTIMER3	CTIMER2	CTIMER1	CTIMER0	I3C0	INPUTMU...
Reset	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FLEXPWM1	FLEXPWM1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
30 FLEXPWM0	FLEXPWM0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
29 QDC1	QDC1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
28 QDC0	QDC0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
27 USB0	USB0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
26 LPUART4	LPUART4 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
25 LPUART3	LPUART3 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
24 LPUART2	LPUART2 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
23 LPUART1	LPUART1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
22 LPUART0	LPUART0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
21 LPSPI1	LPSPI1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
20 LPSPI0	LPSPI0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
19 LPI2C1	LPI2C1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
18 LPI2C0	LPI2C0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
17 FLEXIO0	FLEXIO0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
16 AOI1	AOI1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
15 FMC	FMC 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
14 ERM0	ERM0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
13 EIM0	EIM0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
12 CRC0	CRC0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
11 AOI0	AOI0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 DMA	DMA 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
9 WWDT0	WWDT0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
8 UTICK0	UTICK0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
7 FREQME	FREQME 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
6 CTIMER4	CTIMER4 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
5 CTIMER3	CTIMER3 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
4 CTIMER2	CTIMER2 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
3 CTIMER1	CTIMER1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
2 CTIMER0	CTIMER0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
1 I3C0	I3C0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
0	INPUTMUX0

Table continues on the next page...

Table continued from the previous page...

Field	Function
INPUTMUX0	0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled

14.5.2.13 Control Automatic Clock Gating 1 (MRCC_GLB_ACC1)

Offset

Register	Offset
MRCC_GLB_ACC1	84h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved						ROMC	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0	RAMB	RAMA	Reserved	
Reset	0	0	0	0	0	0	1	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 W	Reserved	LPI2C3	LPI2C2	FLEXCAN0	PORT4	PORT3	PORT2	PORT1	PORT0	OPAMP0	DAC0	CMP1	CMP0	ADC1	ADC0	OSTIMER0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	reserved reserved
25 ROMC	ROMC 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
24 GPIO4	GPIO4 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
23 GPIO3	GPIO3 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 GPIO2	GPIO2 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
21 GPIO1	GPIO1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
20 GPIO0	GPIO0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
19 RAMB	RAMB 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
18 RAMA	RAMA 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
17-16 —	reserved reserved
15 —	Reserved
14 LPI2C3	LPI2C3 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
13 LPI2C2	LPI2C2 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
12 FLEXCAN0	FLEXCAN0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
11 PORT4	PORT4 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 PORT3	PORT3 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
9 PORT2	PORT2 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
8 PORT1	PORT1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
7 PORT0	PORT0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
6 OPAMP0	OPAMP0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
5 DAC0	DAC0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
4 CMP1	CMP1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
3 CMP0	CMP0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
2 ADC1	ADC1 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
1 ADC0	ADC0 0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled
0	OSTIMER0

Table continues on the next page...

Table continued from the previous page...

Field	Function
OSTIMER0	0b - Automatic clock gating is disabled 1b - Automatic clock gating is enabled

14.5.2.14 I3C0_FCLK clock selection control (MRCC_I3C0_FCLK_CLKSEL)

Offset

Register	Offset
MRCC_I3C0_FCLK_CLKSEL	A0h

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												MUX			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

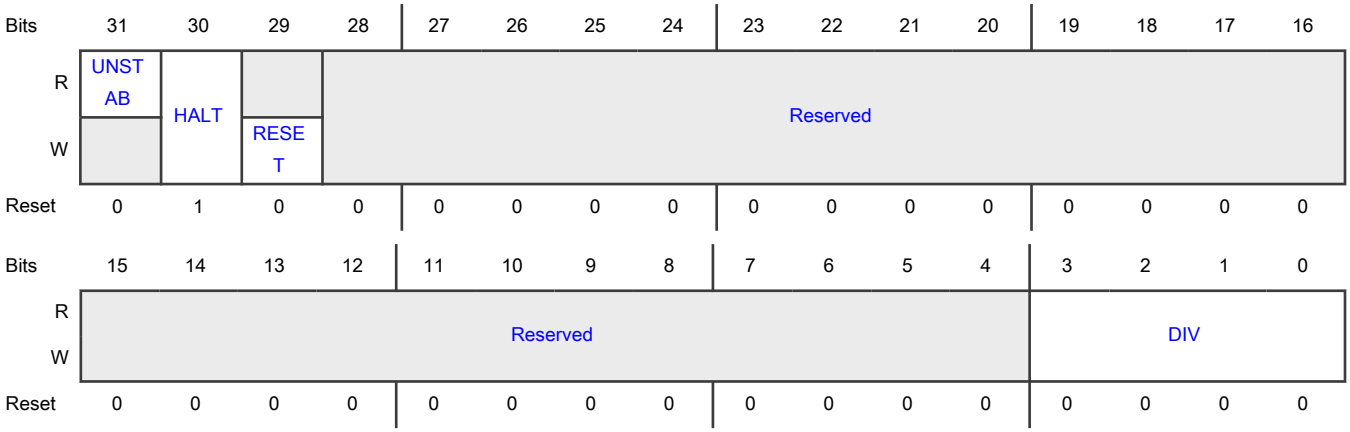
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.15 I3C0_FCLK clock divider control (MRCC_I3C0_FCLK_CLKDIV)

Offset

Register	Offset
MRCC_I3C0_FCLK_CLKDIV	A4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

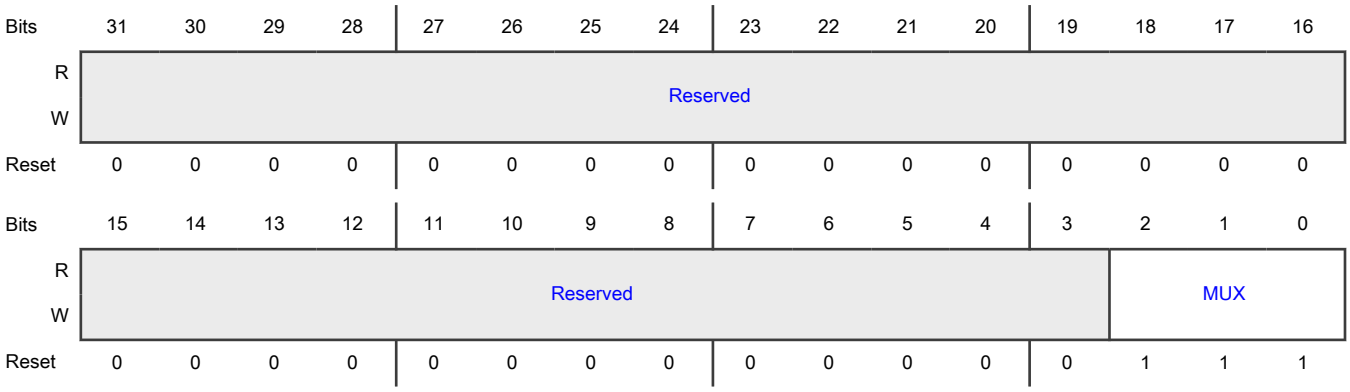
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.16 CTIMER0 clock selection control (MRCC_CTIMER0_CLKSEL)

Offset

Register	Offset
MRCC_CTIMER0_CLKSEL	A8h

Diagram



Fields

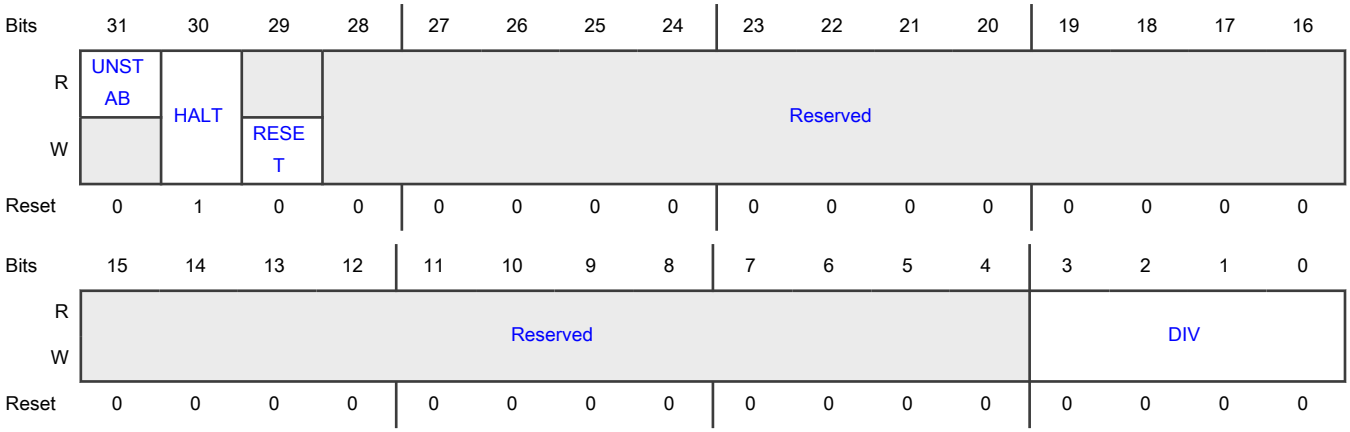
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.17 CTIMER0 clock divider control (MRCC_CTIMER0_CLKDIV)

Offset

Register	Offset
MRCC_CTIMER0_CLKDIV	ACh

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

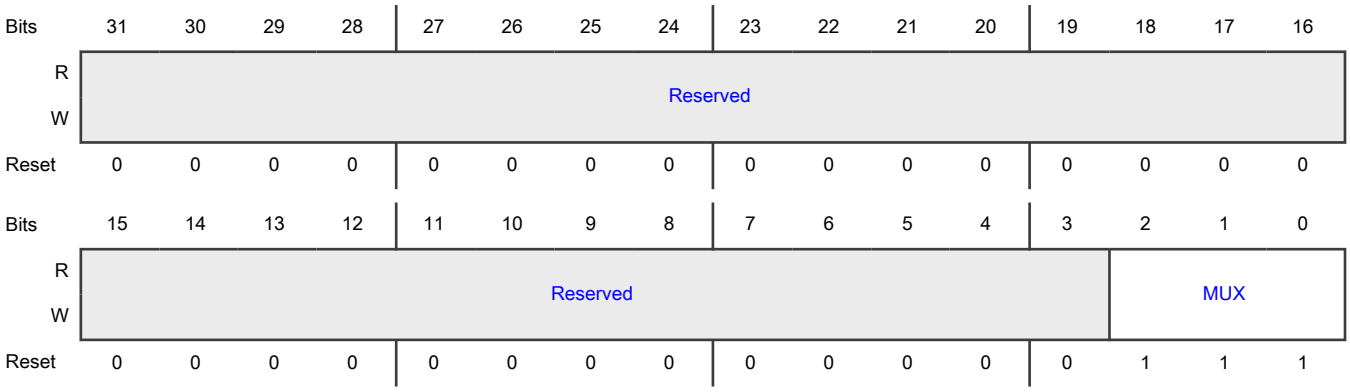
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.18 CTIMER1 clock selection control (MRCC_CTIMER1_CLKSEL)

Offset

Register	Offset
MRCC_CTIMER1_CLKSEL	B0h

Diagram



Fields

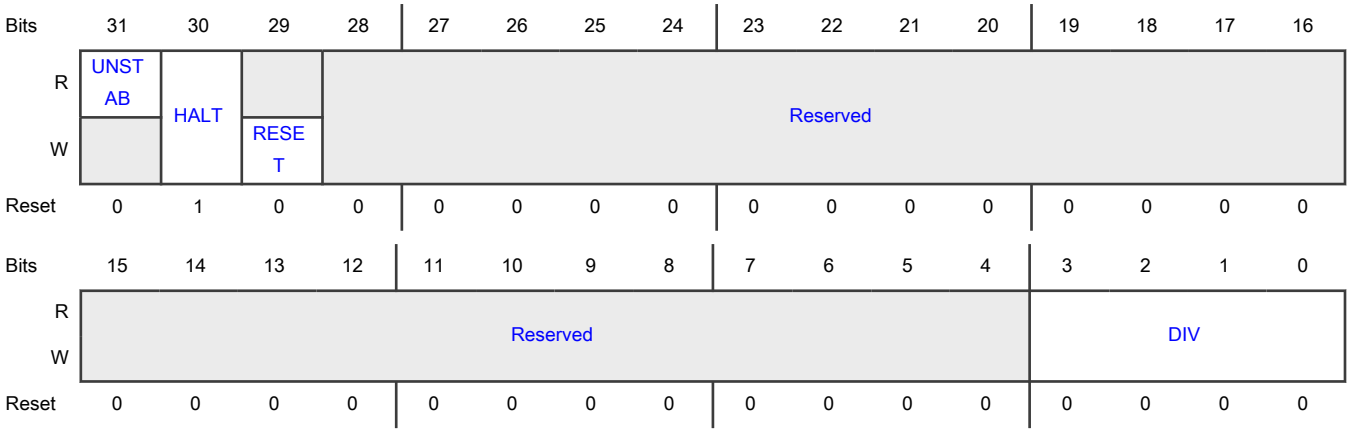
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.19 CTIMER1 clock divider control (MRCC_CTIMER1_CLKDIV)

Offset

Register	Offset
MRCC_CTIMER1_CLKDIV	B4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

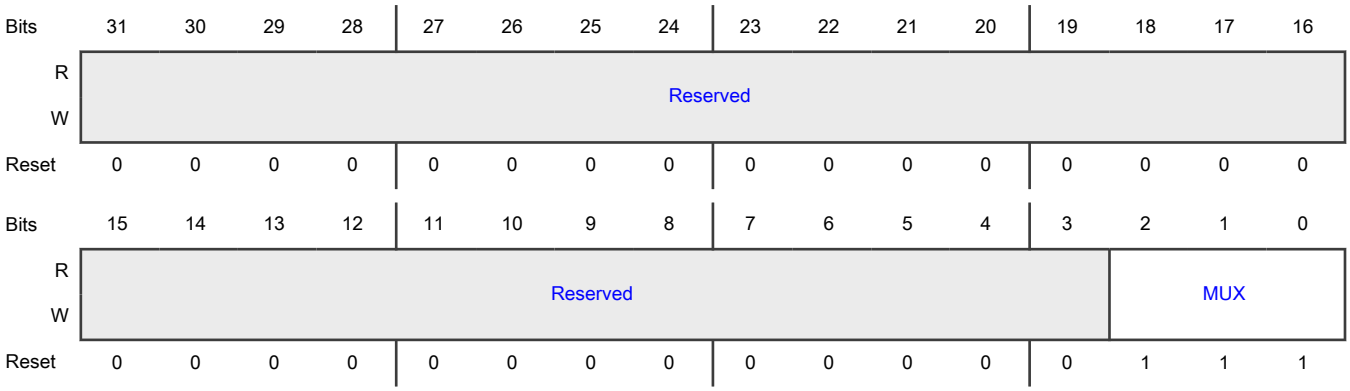
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.20 CTIMER2 clock selection control (MRCC_CTIMER2_CLKSEL)

Offset

Register	Offset
MRCC_CTIMER2_CLKSEL	B8h

Diagram



Fields

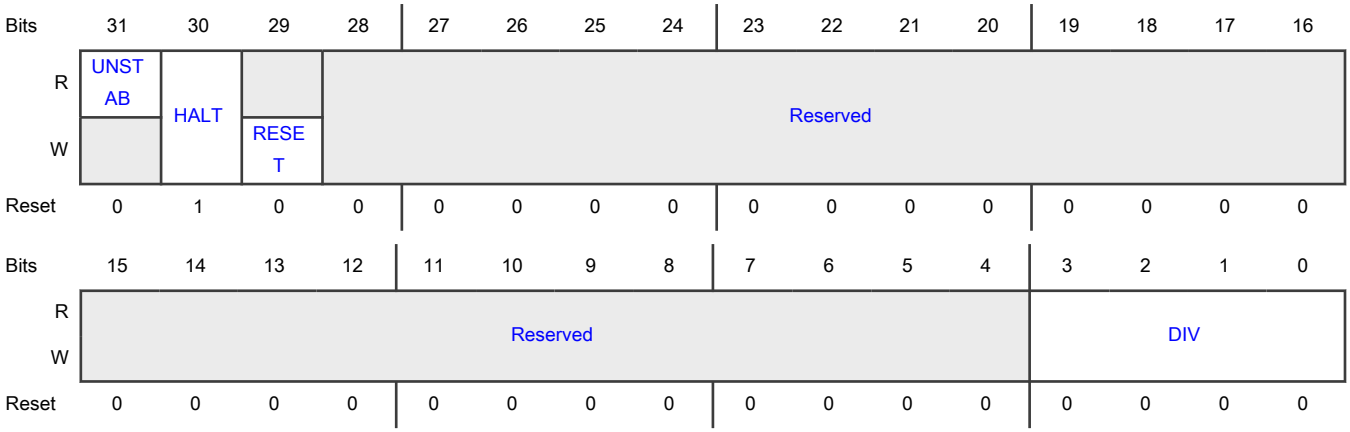
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.21 CTIMER2 clock divider control (MRCC_CTIMER2_CLKDIV)

Offset

Register	Offset
MRCC_CTIMER2_CLKDIV	BCh

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

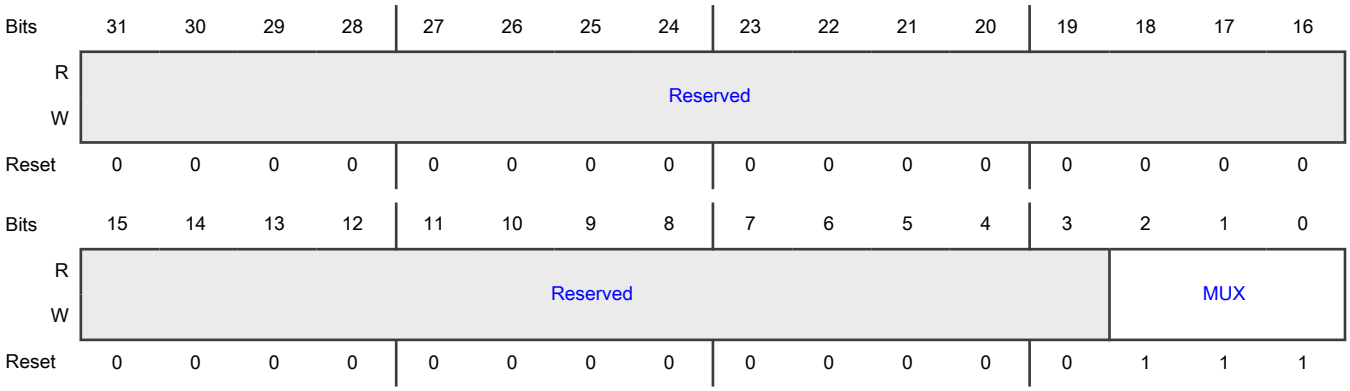
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.22 CTIMER3 clock selection control (MRCC_CTIMER3_CLKSEL)

Offset

Register	Offset
MRCC_CTIMER3_CLKSEL	C0h

Diagram



Fields

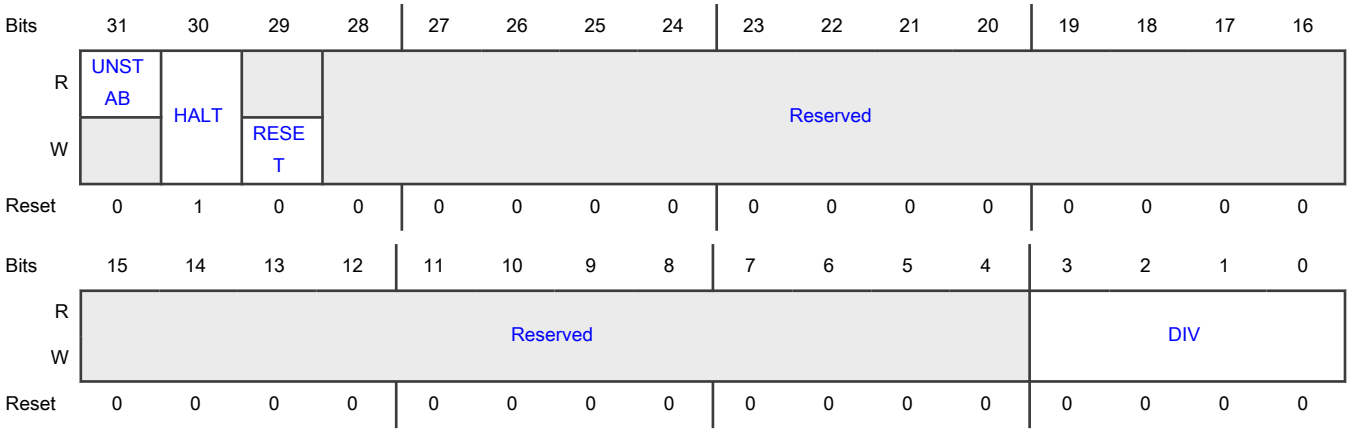
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.23 CTIMER3 clock divider control (MRCC_CTIMER3_CLKDIV)

Offset

Register	Offset
MRCC_CTIMER3_CLKDIV	C4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

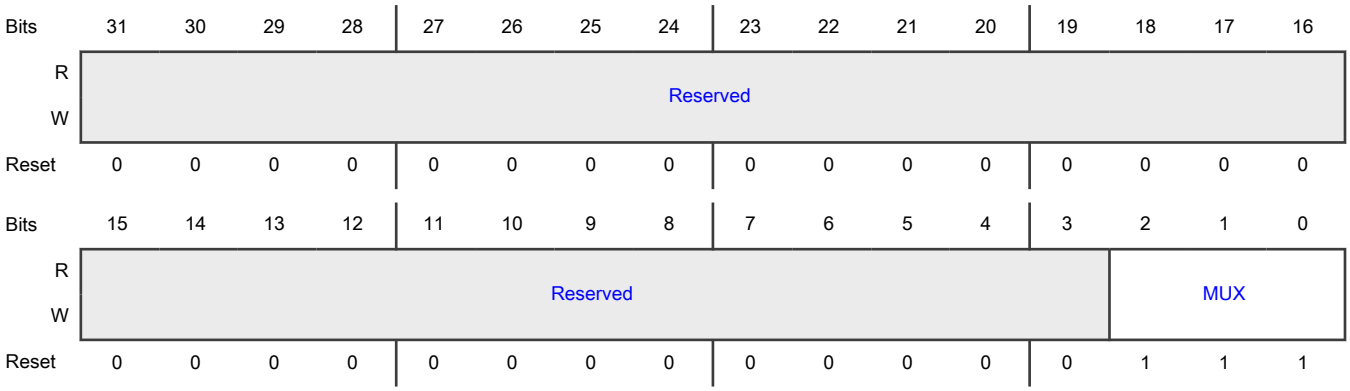
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.24 CTIMER4 clock selection control (MRCC_CTIMER4_CLKSEL)

Offset

Register	Offset
MRCC_CTIMER4_CLKSEL	C8h

Diagram



Fields

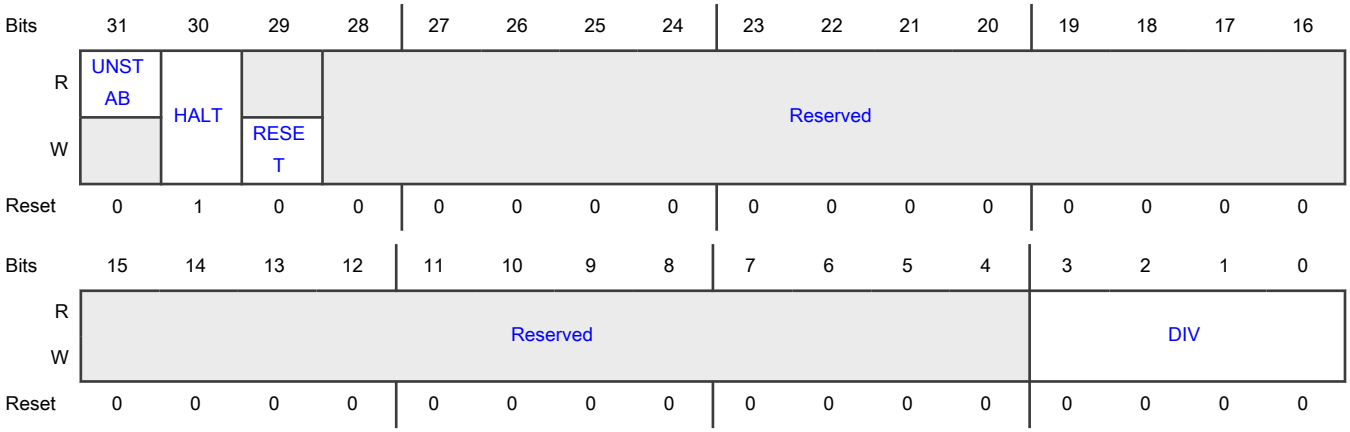
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.25 CTIMER4 clock divider control (MRCC_CTIMER4_CLKDIV)

Offset

Register	Offset
MRCC_CTIMER4_CLKDIV	CCh

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

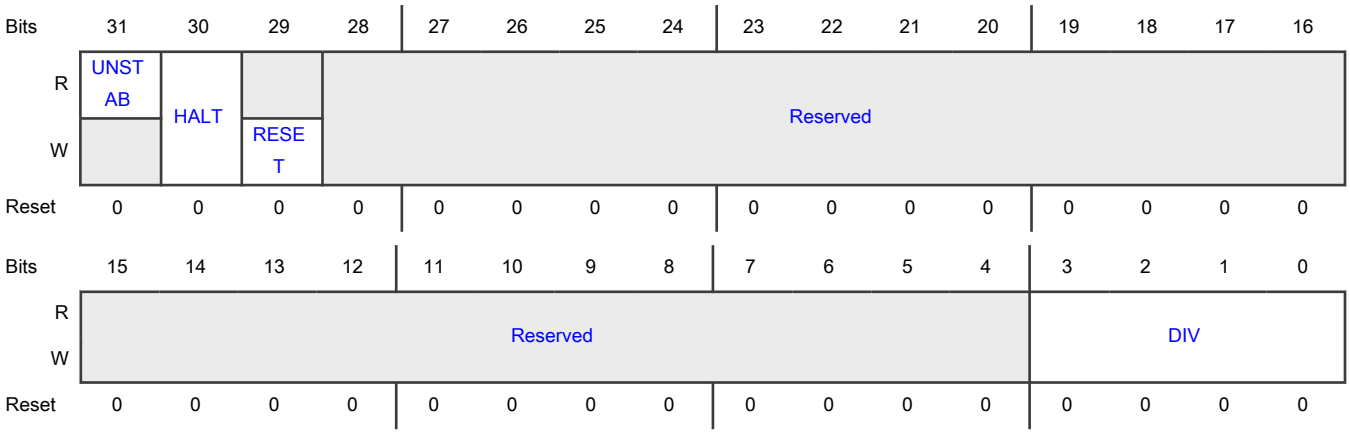
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.26 WWDT0 clock divider control (MRCC_WWDT0_CLKDIV)

Offset

Register	Offset
MRCC_WWDT0_CLKDIV	D4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped

Table continues on the next page...

Table continued from the previous page...

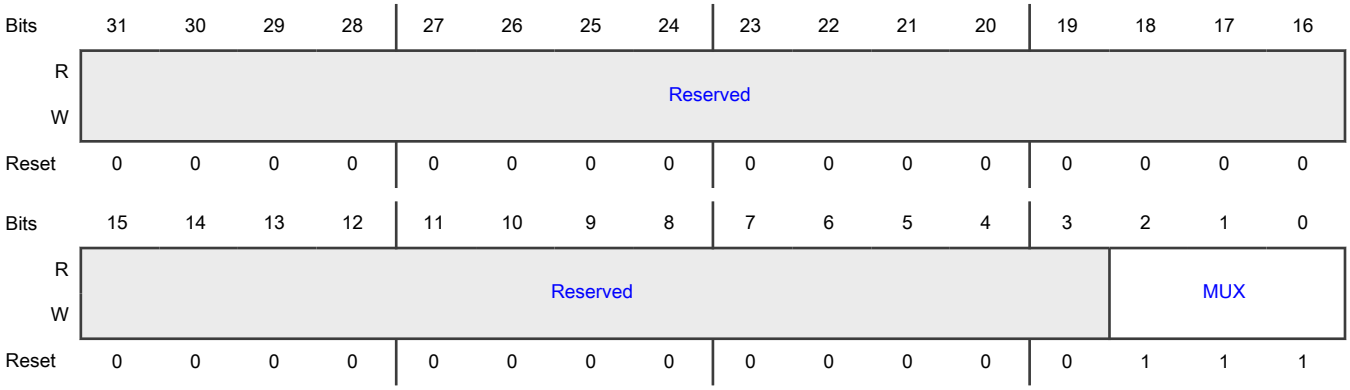
Field	Function
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.27 FLEXIO0 clock selection control (MRCC_FLEXIO0_CLKSEL)

Offset

Register	Offset
MRCC_FLEXIO0_CLKSEL	D8h

Diagram



Fields

Field	Function
31-3 —	reserved reserved
2-0	Functional Clock Mux Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
MUX	Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.28 FLEXIO0 clock divider control (MRCC_FLEXIO0_CLKDIV)

Offset

Register	Offset
MRCC_FLEXIO0_CLKDIV	DCh

Diagram

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

Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30	Halt divider counter

Table continues on the next page...

Table continued from the previous page...

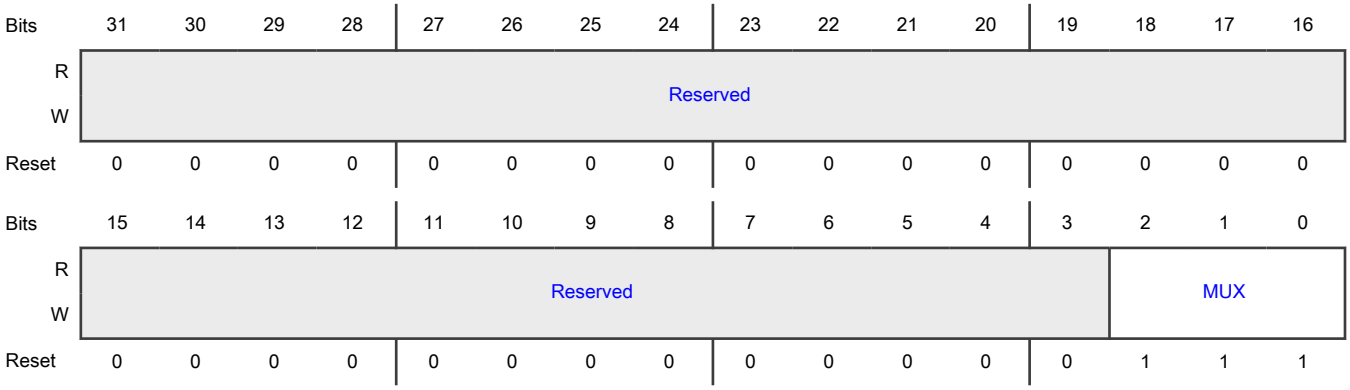
Field	Function
HALT	Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.29 LPI2C0 clock selection control (MRCC_LPI2C0_CLKSEL)

Offset

Register	Offset
MRCC_LPI2C0_CLKSEL	E0h

Diagram



Fields

Field	Function
31-3	reserved

Table continues on the next page...

Table continued from the previous page...

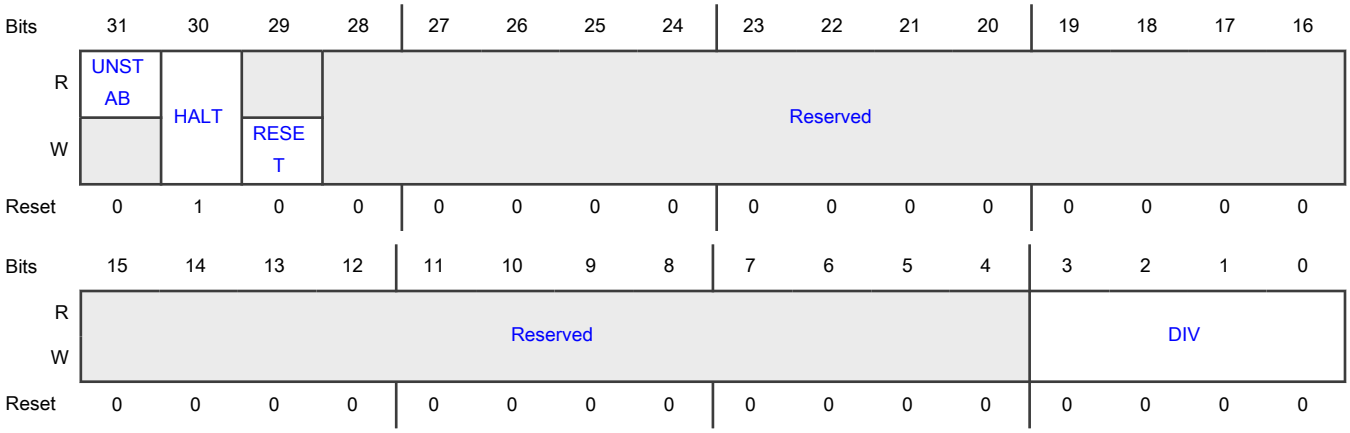
Field	Function
—	reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.30 LPI2C0 clock divider control (MRCC_LPI2C0_CLKDIV)

Offset

Register	Offset
MRCC_LPI2C0_CLKDIV	E4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable

Table continues on the next page...

Table continued from the previous page...

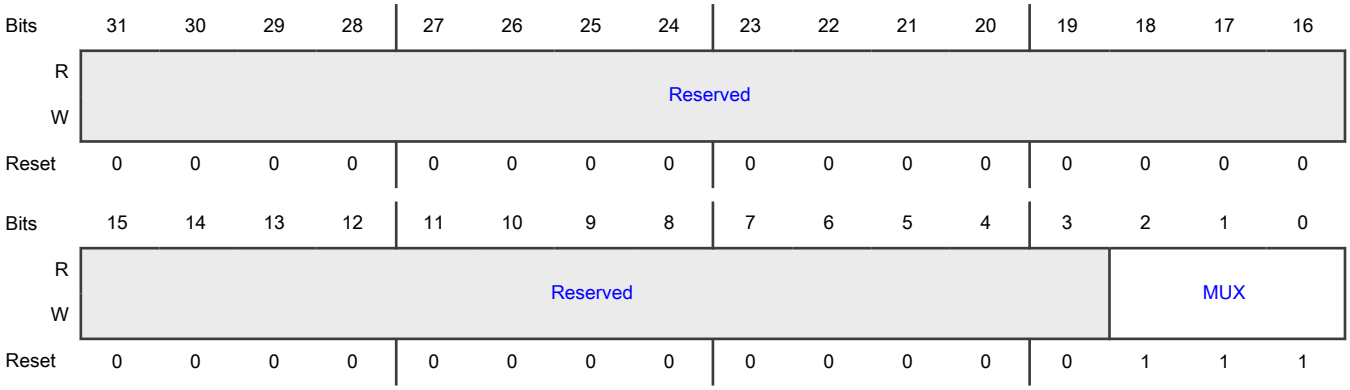
Field	Function
	1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.31 LPI2C1 clock selection control (MRCC_LPI2C1_CLKSEL)

Offset

Register	Offset
MRCC_LPI2C1_CLKSEL	E8h

Diagram



Fields

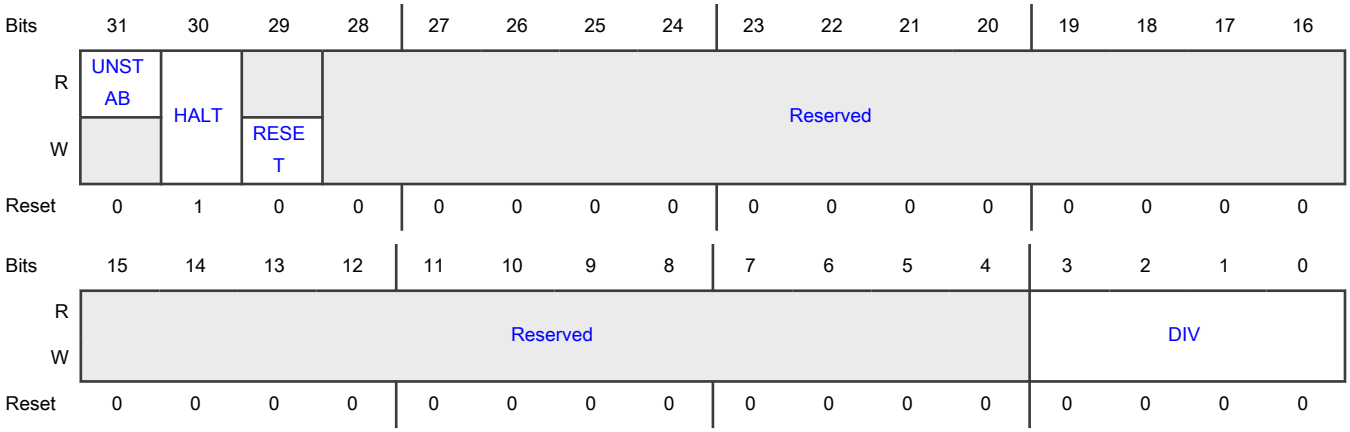
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.32 LPI2C1 clock divider control (MRCC_LPI2C1_CLKDIV)

Offset

Register	Offset
MRCC_LPI2C1_CLKDIV	ECh

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock.

Table continues on the next page...

Table continued from the previous page...

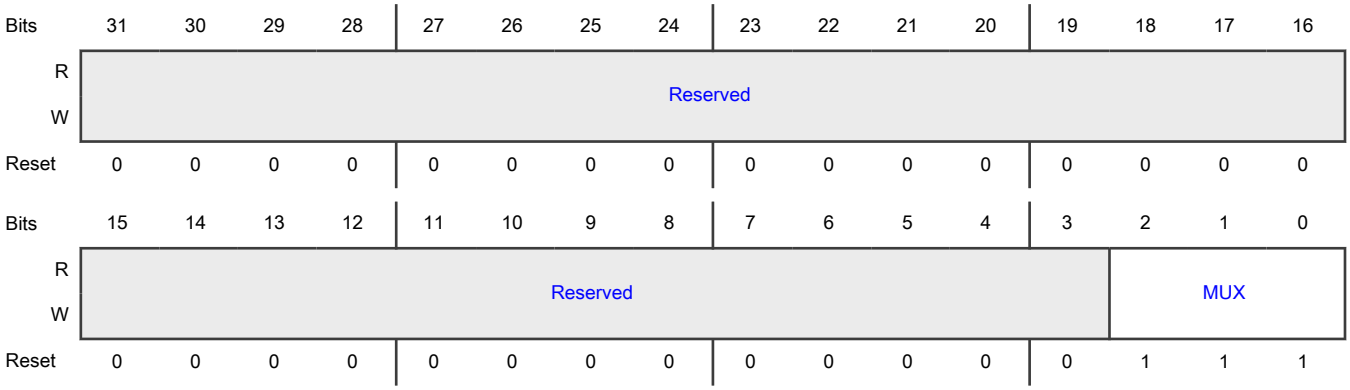
Field	Function
	0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.33 LPSPI0 clock selection control (MRCC_LPSPI0_CLKSEL)

Offset

Register	Offset
MRCC_LPSPI0_CLKSEL	F0h

Diagram



Fields

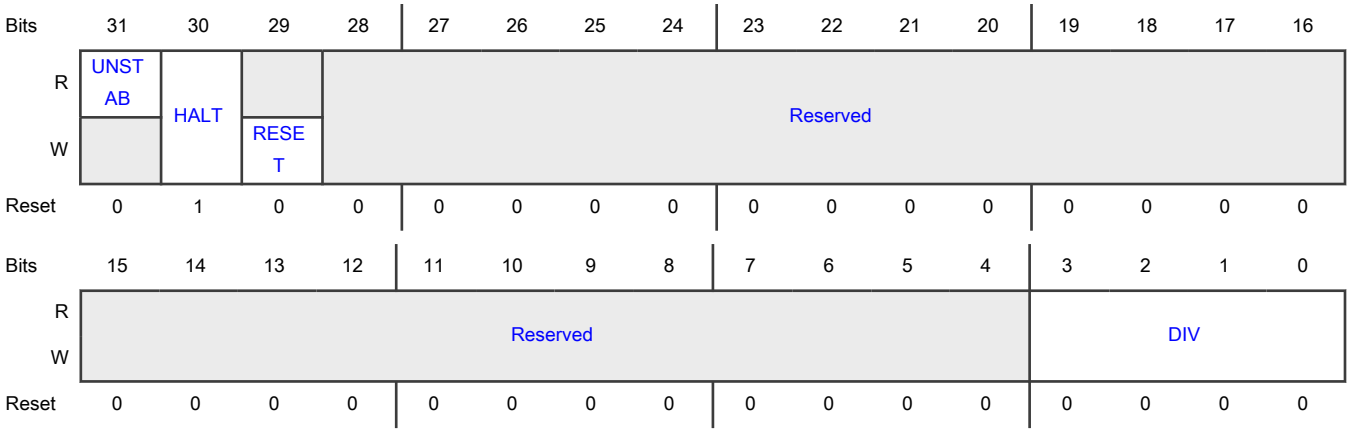
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.34 LPSPI0 clock divider control (MRCC_LPSPI0_CLKDIV)

Offset

Register	Offset
MRCC_LPSPI0_CLKDIV	F4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock.

Table continues on the next page...

Table continued from the previous page...

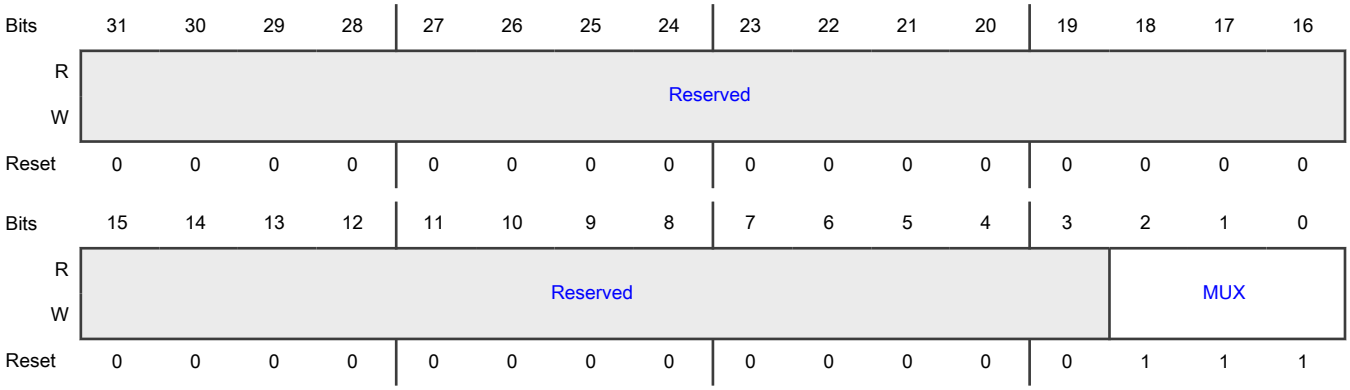
Field	Function
	0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.35 LPSPI1 clock selection control (MRCC_LPSPI1_CLKSEL)

Offset

Register	Offset
MRCC_LPSPI1_CLKSEL	F8h

Diagram



Fields

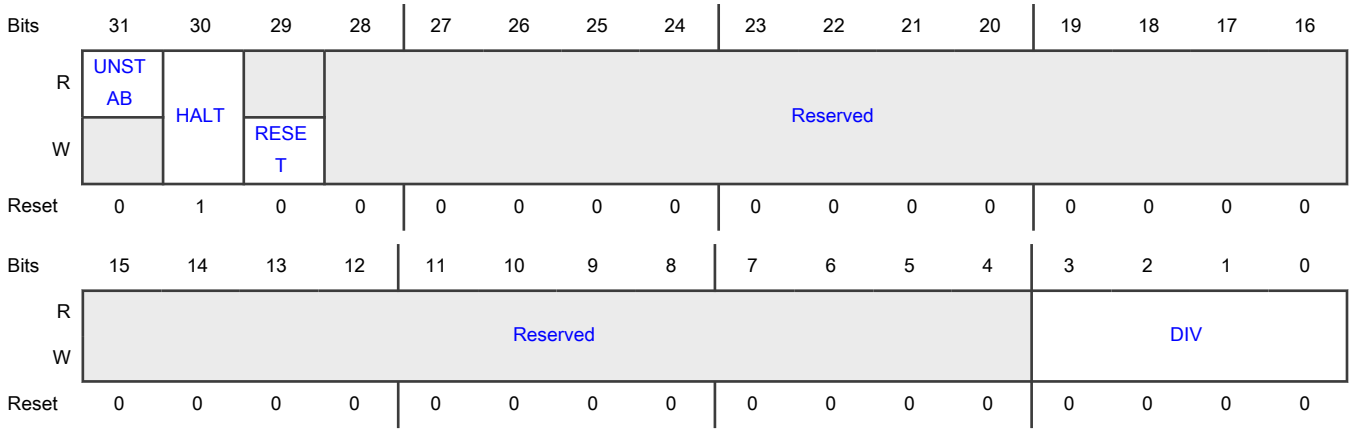
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.36 LPSPI1 clock divider control (MRCC_LPSPI1_CLKDIV)

Offset

Register	Offset
MRCC_LPSPI1_CLKDIV	FCh

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock.

Table continues on the next page...

Table continued from the previous page...

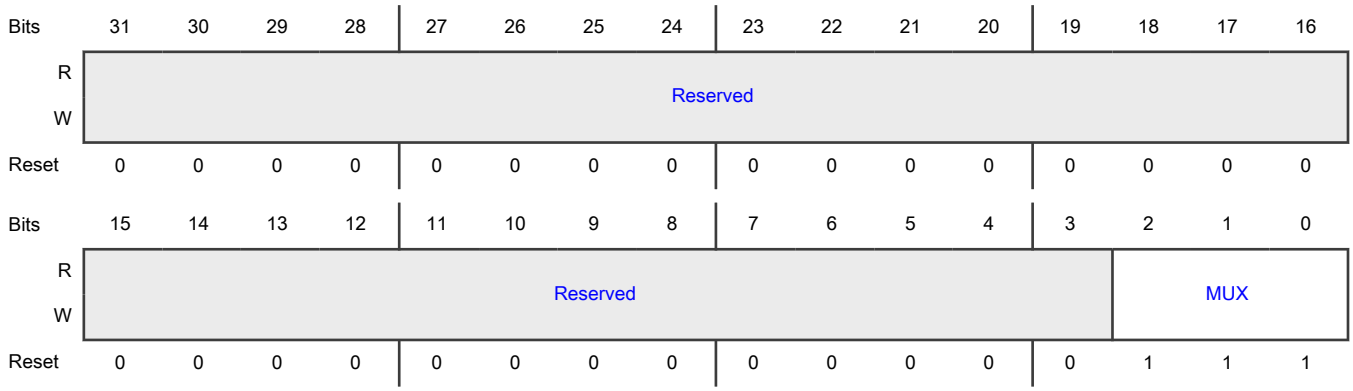
Field	Function
	0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.37 LPUART0 clock selection control (MRCC_LPUART0_CLKSEL)

Offset

Register	Offset
MRCC_LPUART0_CLKSEL	100h

Diagram



Fields

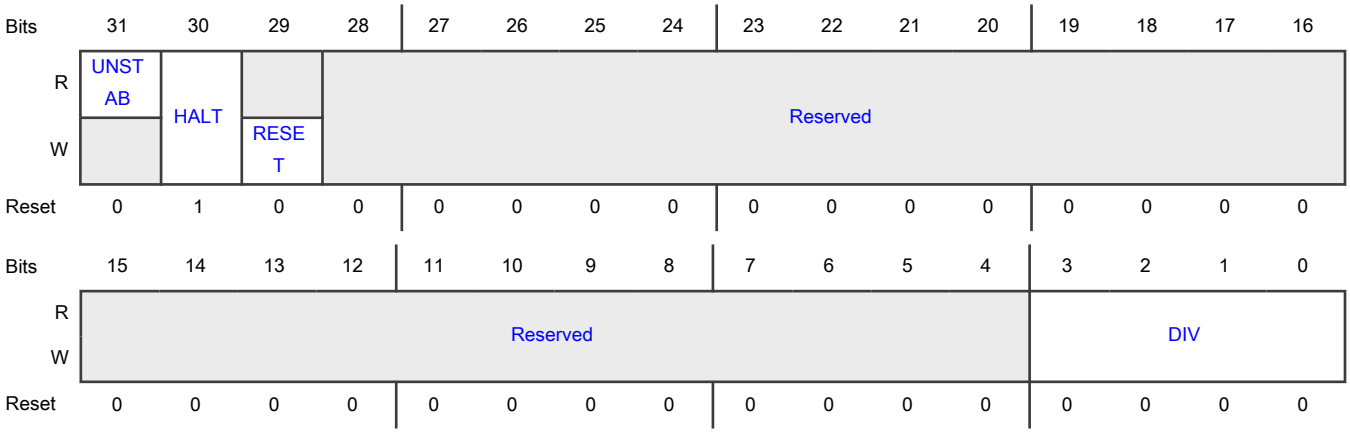
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.38 LPUART0 clock divider control (MRCC_LPUART0_CLKDIV)

Offset

Register	Offset
MRCC_LPUART0_CLKDIV	104h

Diagram



Fields

Field	Function
31	Divider status flag

Table continues on the next page...

Table continued from the previous page...

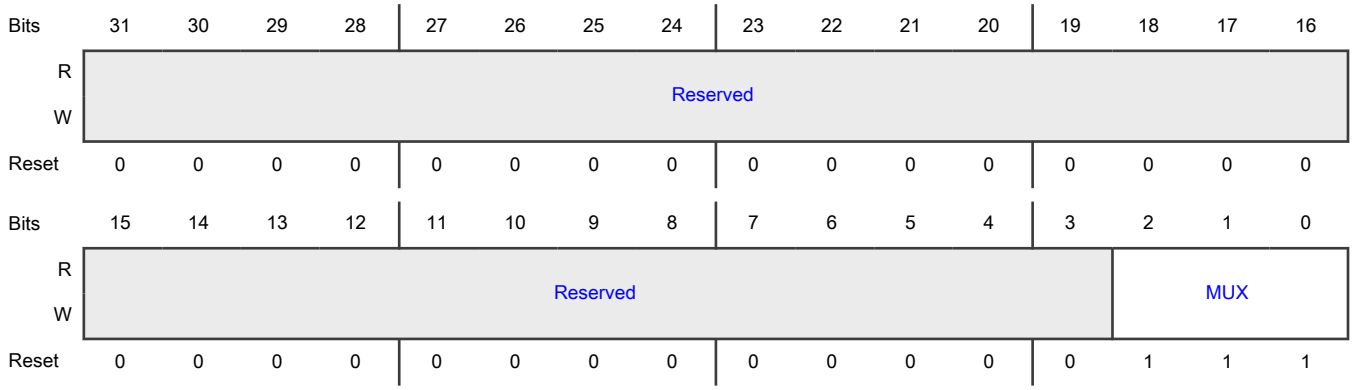
Field	Function
UNSTAB	Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.39 LPUART1 clock selection control (MRCC_LPUART1_CLKSEL)

Offset

Register	Offset
MRCC_LPUART1_CLKSEL	108h

Diagram



Fields

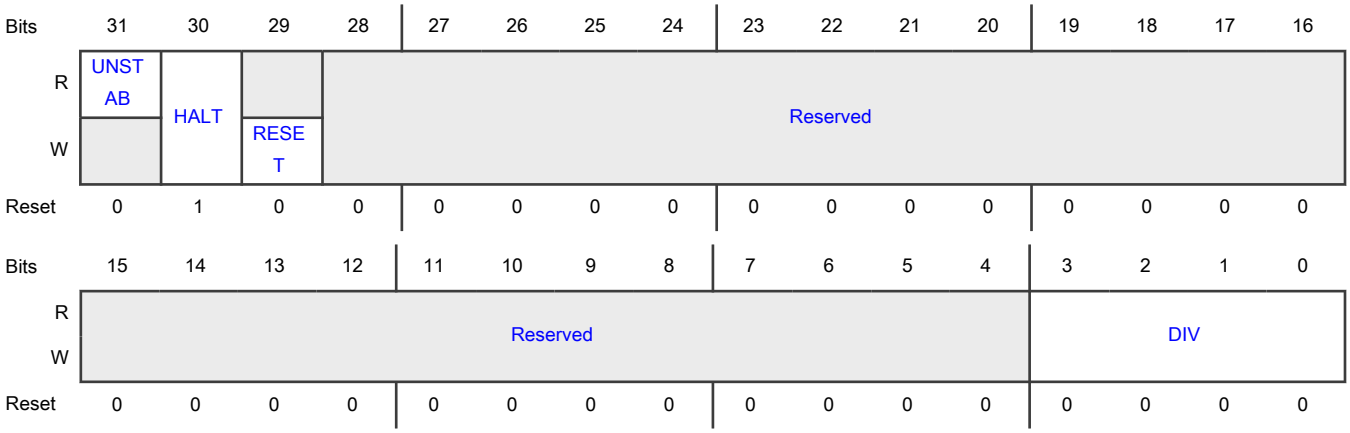
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.40 LPUART1 clock divider control (MRCC_LPUART1_CLKDIV)

Offset

Register	Offset
MRCC_LPUART1_CLKDIV	10Ch

Diagram



Fields

Field	Function
31	Divider status flag

Table continues on the next page...

Table continued from the previous page...

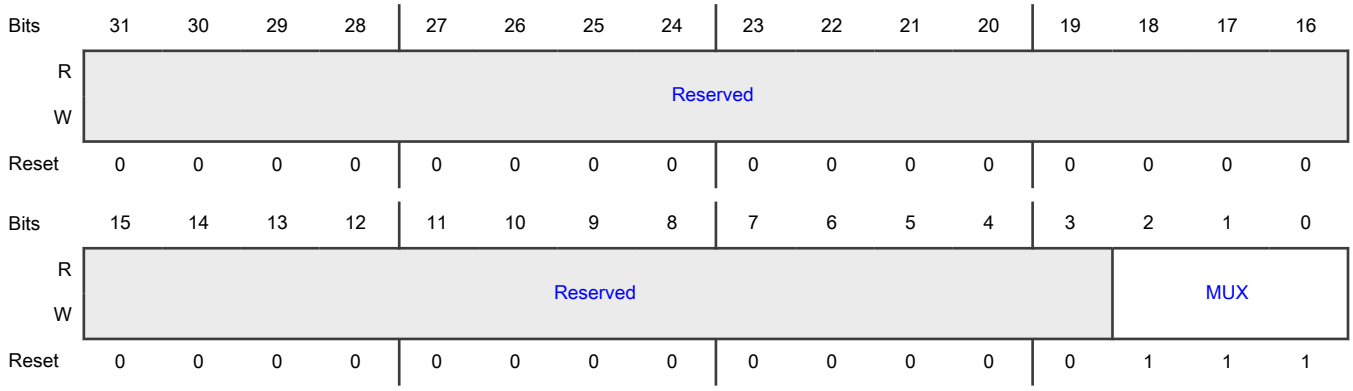
Field	Function
UNSTAB	Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.41 LPUART2 clock selection control (MRCC_LPUART2_CLKSEL)

Offset

Register	Offset
MRCC_LPUART2_CLKSEL	110h

Diagram



Fields

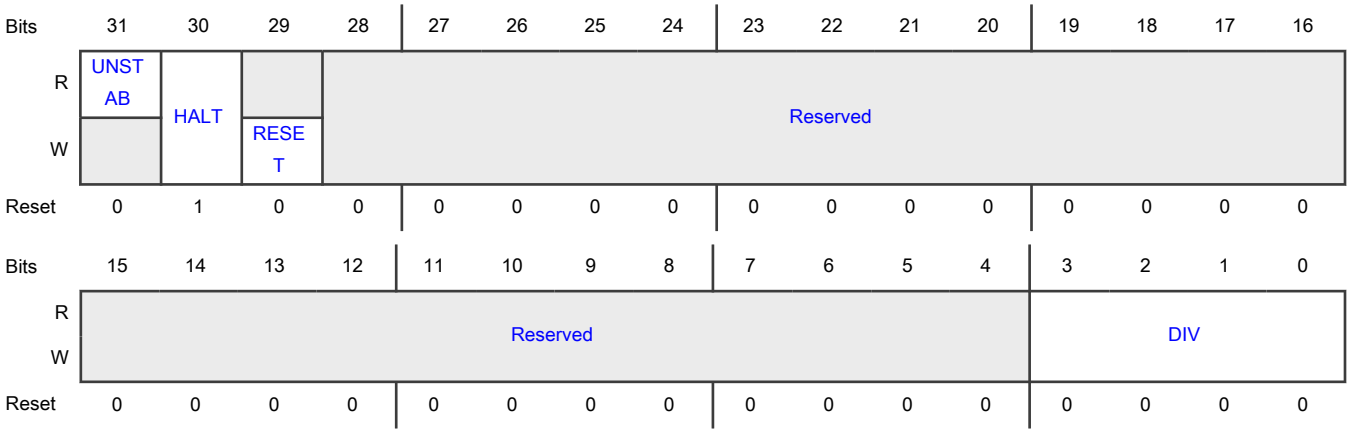
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.42 LPUART2 clock divider control (MRCC_LPUART2_CLKDIV)

Offset

Register	Offset
MRCC_LPUART2_CLKDIV	114h

Diagram



Fields

Field	Function
31	Divider status flag

Table continues on the next page...

Table continued from the previous page...

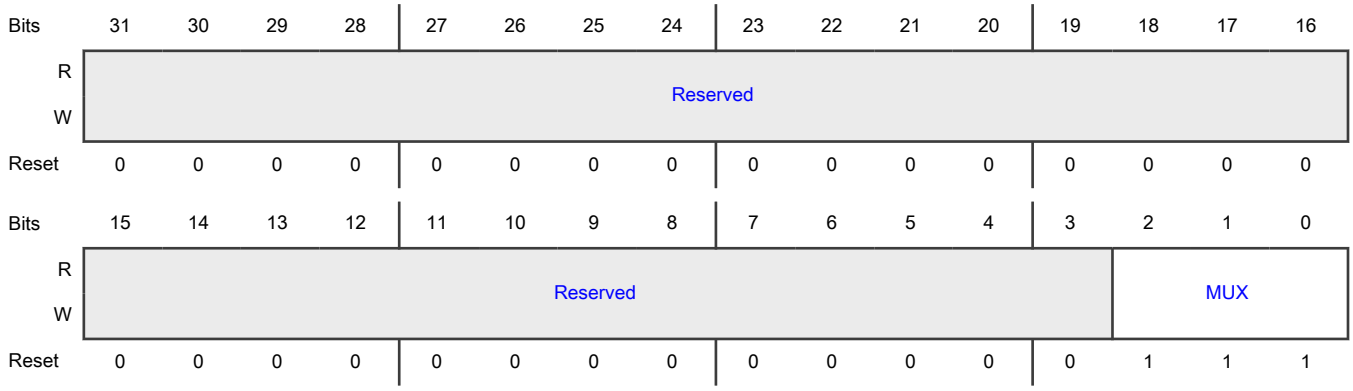
Field	Function
UNSTAB	Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.43 LPUART3 clock selection control (MRCC_LPUART3_CLKSEL)

Offset

Register	Offset
MRCC_LPUART3_CLKSEL	118h

Diagram



Fields

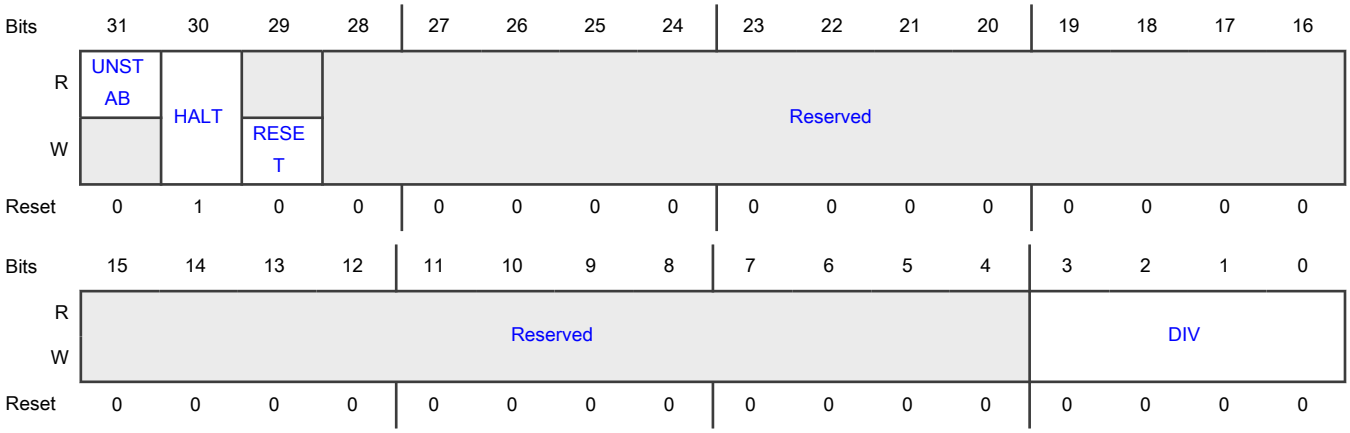
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.44 LPUART3 clock divider control (MRCC_LPUART3_CLKDIV)

Offset

Register	Offset
MRCC_LPUART3_CLKDIV	11Ch

Diagram



Fields

Field	Function
31	Divider status flag

Table continues on the next page...

Table continued from the previous page...

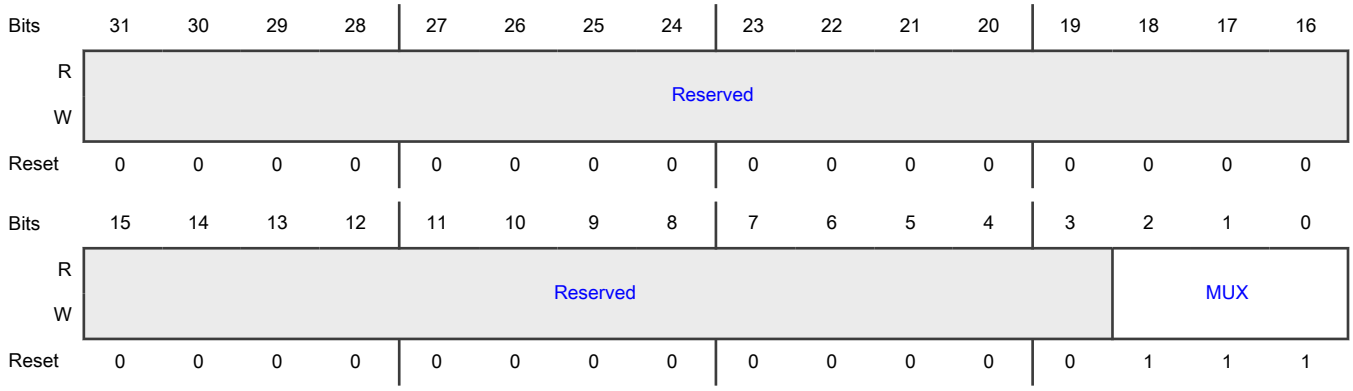
Field	Function
UNSTAB	Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.45 LPUART4 clock selection control (MRCC_LPUART4_CLKSEL)

Offset

Register	Offset
MRCC_LPUART4_CLKSEL	120h

Diagram



Fields

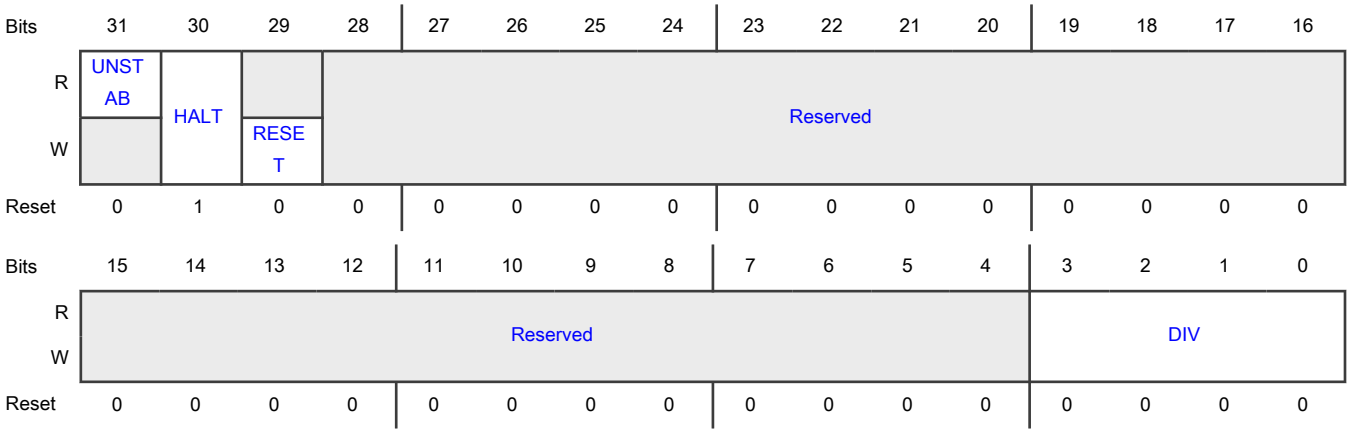
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 100b - CLK_16K 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.46 LPUART4 clock divider control (MRCC_LPUART4_CLKDIV)

Offset

Register	Offset
MRCC_LPUART4_CLKDIV	124h

Diagram



Fields

Field	Function
31	Divider status flag

Table continues on the next page...

Table continued from the previous page...

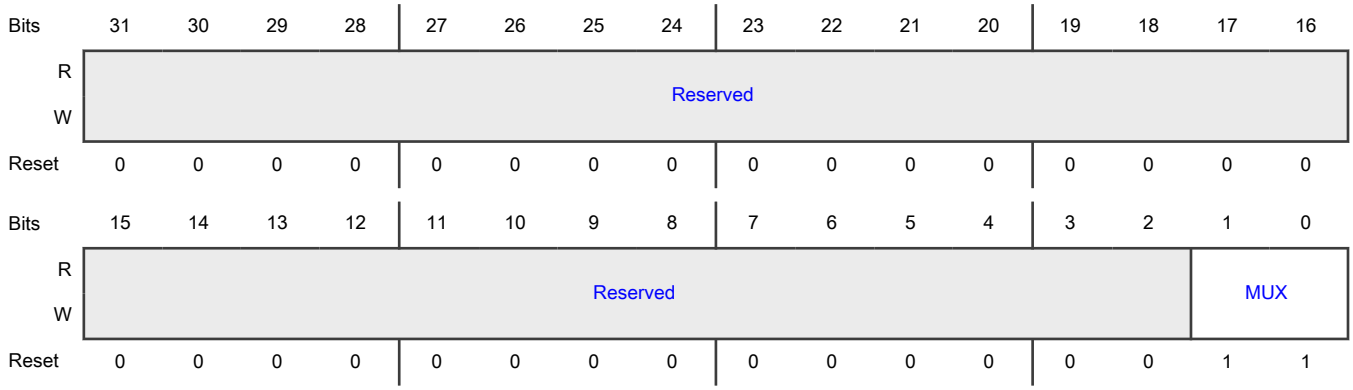
Field	Function
UNSTAB	Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.47 USB0 clock selection control (MRCC_USB0_CLKSEL)

Offset

Register	Offset
MRCC_USB0_CLKSEL	128h

Diagram



Fields

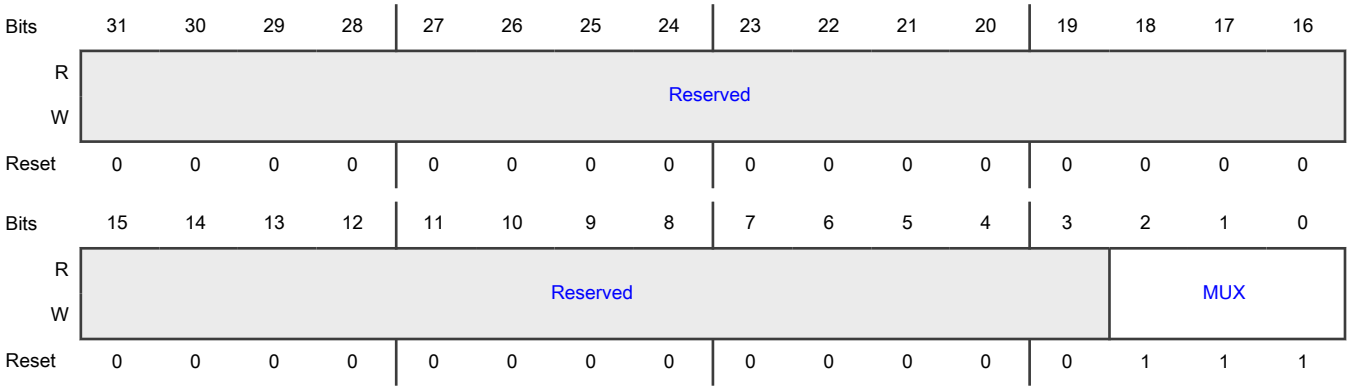
Field	Function
31-2 —	reserved reserved
1-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 01b - CLK_48M 10b - CLK_IN 11b - Reserved(NO Clock)

14.5.2.48 LPTMR0 clock selection control (MRCC_LPTMR0_CLKSEL)

Offset

Register	Offset
MRCC_LPTMR0_CLKSEL	130h

Diagram



Fields

Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0.

Table continues on the next page...

Table continued from the previous page...

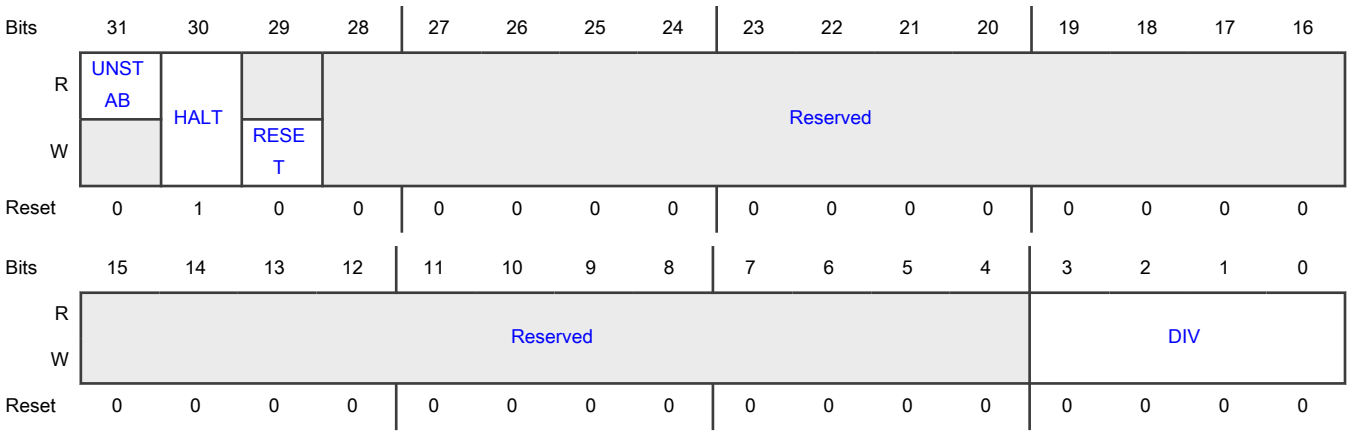
Field	Function
	000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.49 LPTMR0 clock divider control (MRCC_LPTMR0_CLKDIV)

Offset

Register	Offset
MRCC_LPTMR0_CLKDIV	134h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter

Table continues on the next page...

Table continued from the previous page...

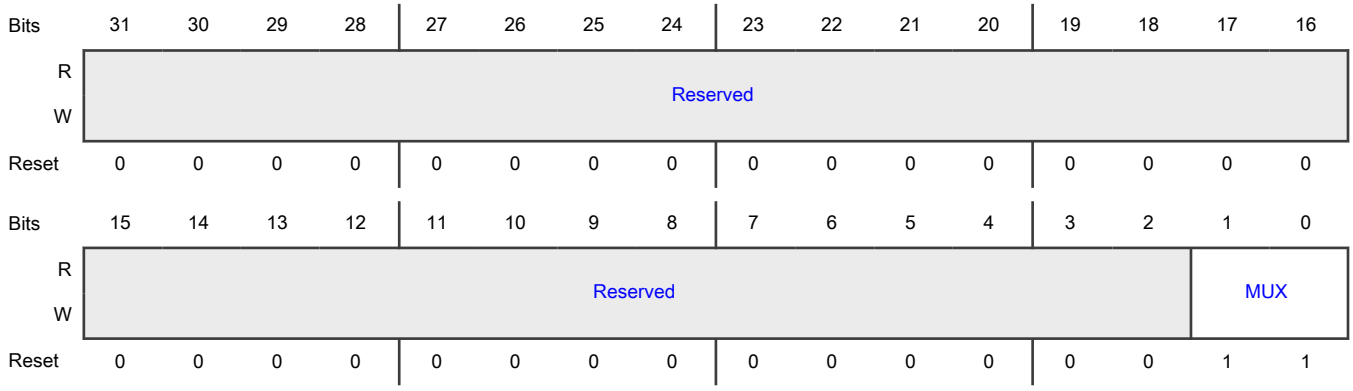
Field	Function
	0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.50 OSTIMER0 clock selection control (MRCC_OSTIMER0_CLKSEL)

Offset

Register	Offset
MRCC_OSTIMER0_CLKSEL	138h

Diagram



Fields

Field	Function
31-2	reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	reserved
1-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 00b - CLK_16K 10b - CLK_1M 11b - Reserved2(NO Clock)

14.5.2.51 ADC0 clock selection control (MRCC_ADC0_CLKSEL)

Offset

Register	Offset
MRCC_ADC0_CLKSEL	140h

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													MUX		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M

Table continues on the next page...

Table continued from the previous page...

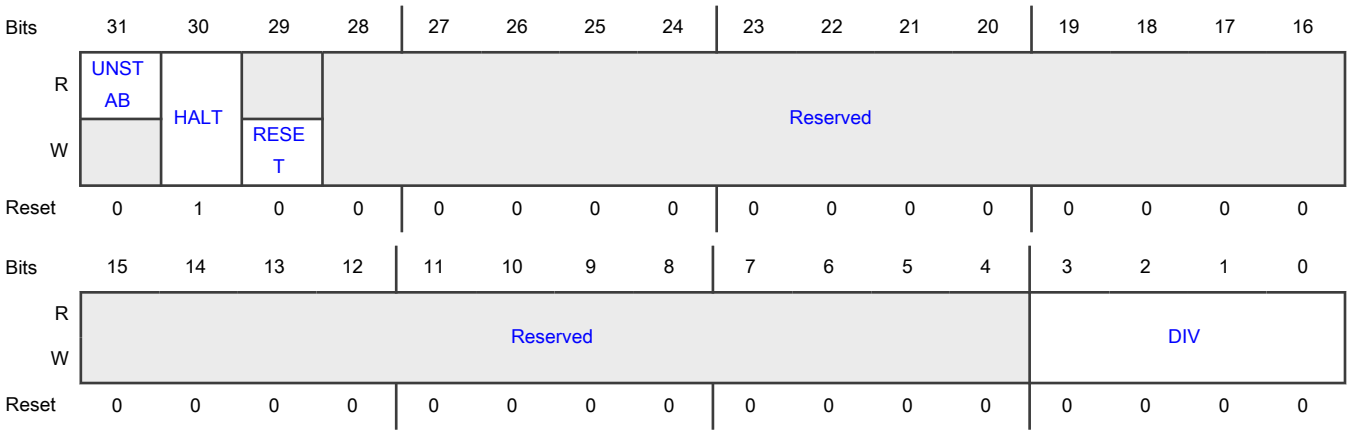
Field	Function
	001b - FRO_HF_GATED 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.52 ADC0 clock divider control (MRCC_ADC0_CLKDIV)

Offset

Register	Offset
MRCC_ADC0_CLKDIV	144h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.53 ADC1 clock selection control (MRCC_ADC1_CLKSEL)

Offset

Register	Offset
MRCC_ADC1_CLKSEL	148h

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												MUX			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select

Table continues on the next page...

Table continued from the previous page...

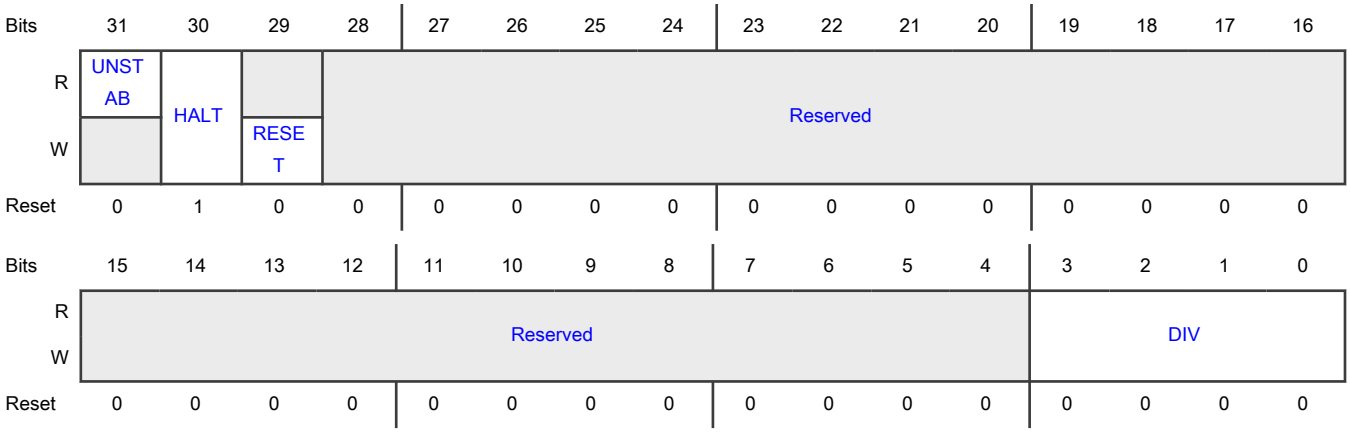
Field	Function
	Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_GATED 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.54 ADC1 clock divider control (MRCC_ADC1_CLKDIV)

Offset

Register	Offset
MRCC_ADC1_CLKDIV	14Ch

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30	Halt divider counter Halt divider counter

Table continues on the next page...

Table continued from the previous page...

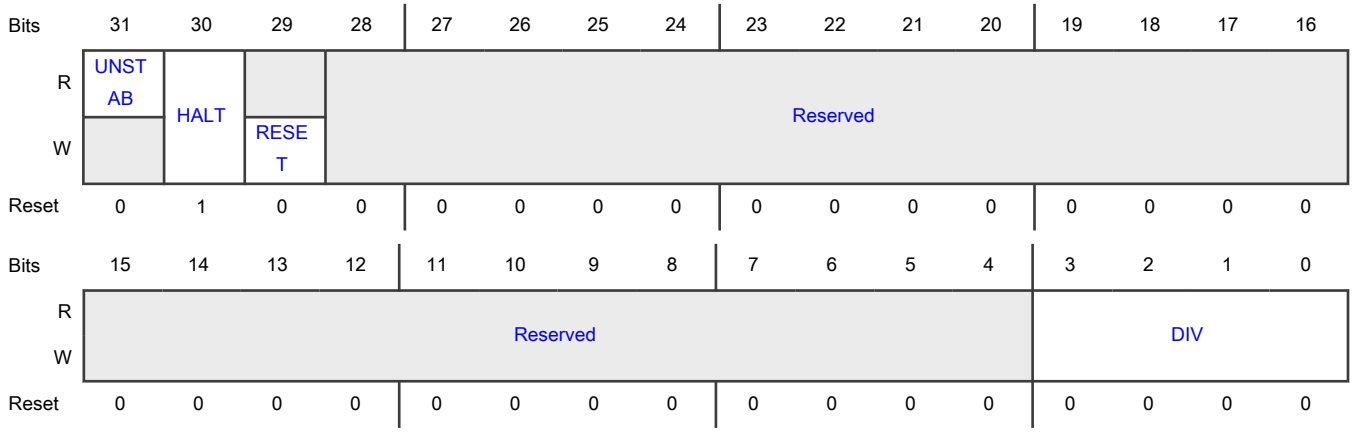
Field	Function
HALT	0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.55 CMP0_FUNC clock divider control (MRCC_CMP0_FUNC_CLKDIV)

Offset

Register	Offset
MRCC_CMP0_FUNC_CLKDIV	154h

Diagram



Fields

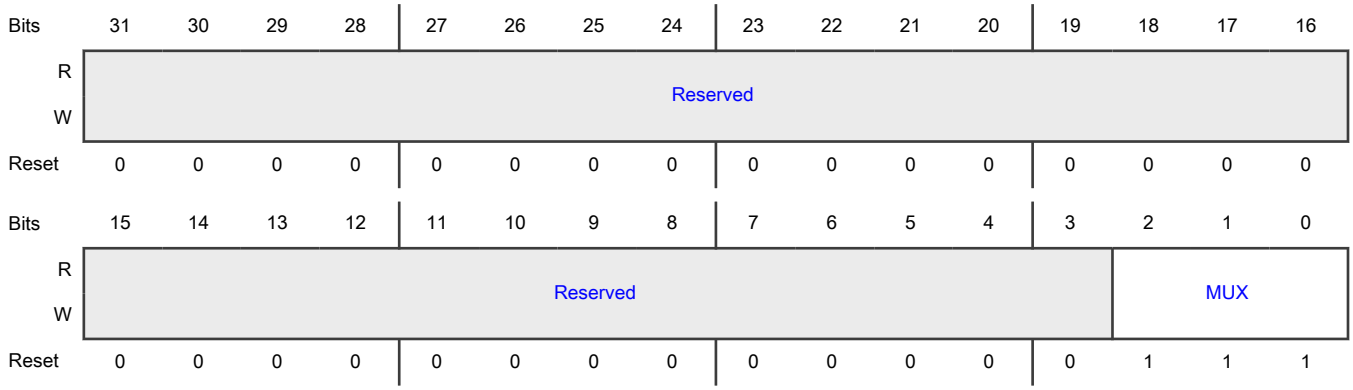
Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.56 CMP0_RR clock selection control (MRCC_CMP0_RR_CLKSEL)

Offset

Register	Offset
MRCC_CMP0_RR_CLKSEL	158h

Diagram



Fields

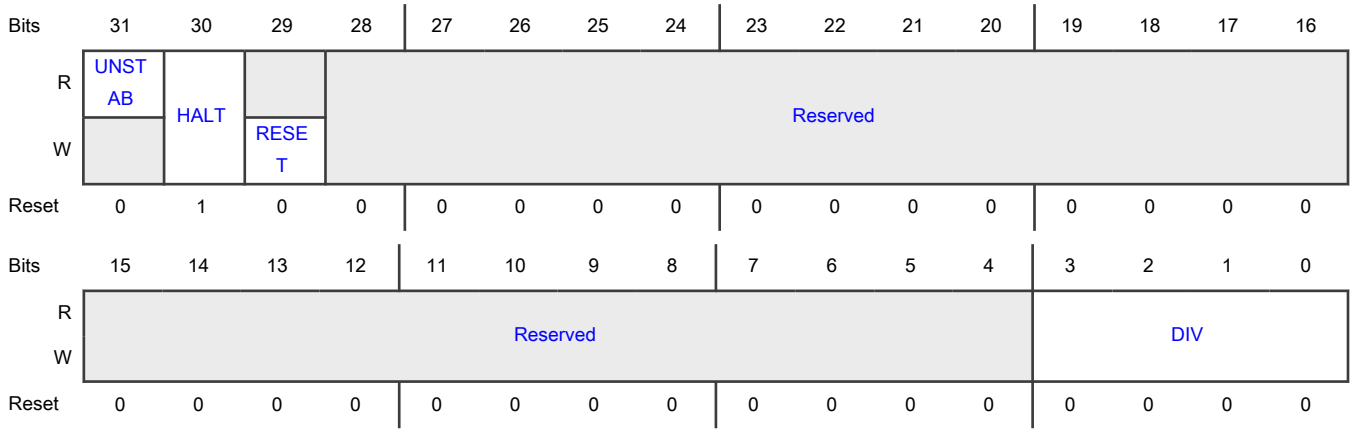
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.57 CMP0_RR clock divider control (MRCC_CMP0_RR_CLKDIV)

Offset

Register	Offset
MRCC_CMP0_RR_CLKDIV	15Ch

Diagram



Fields

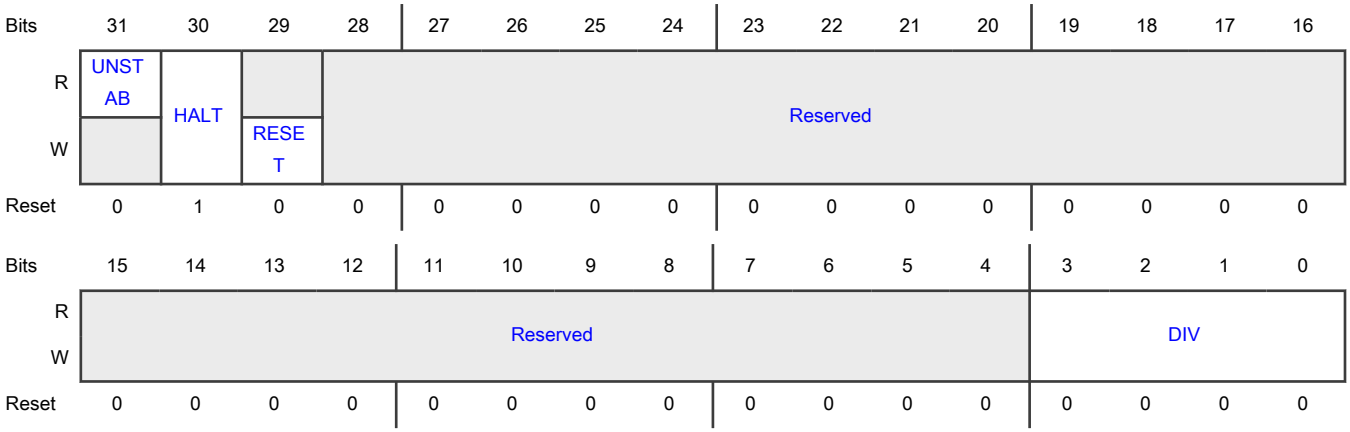
Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.58 CMP1_FUNC clock divider control (MRCC_CMP1_FUNC_CLKDIV)

Offset

Register	Offset
MRCC_CMP1_FUNC_CLKDIV	164h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.59 CMP1_RR clock selection control (MRCC_CMP1_RR_CLKSEL)

Offset

Register	Offset
MRCC_CMP1_RR_CLKSEL	168h

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													MUX		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

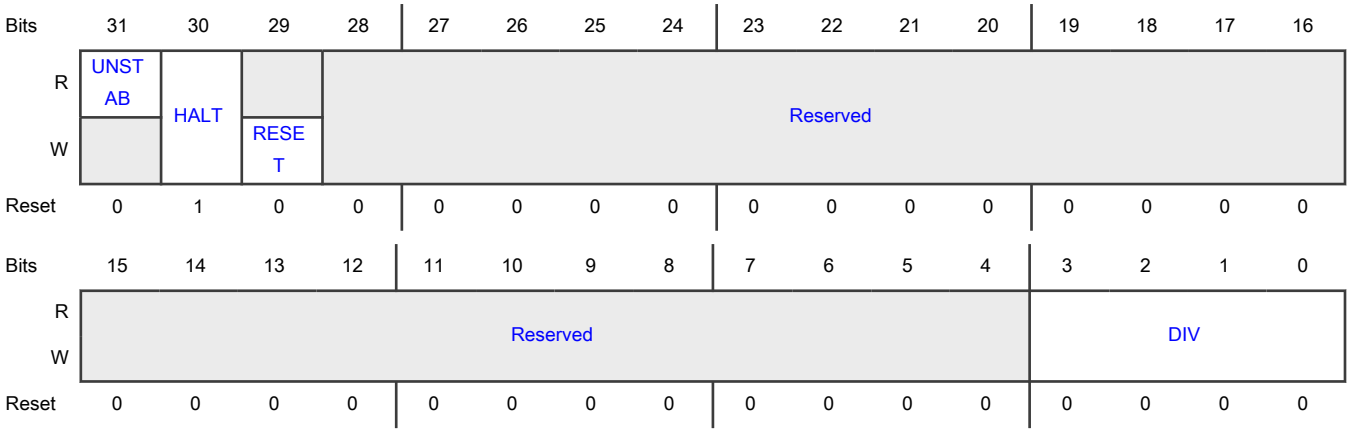
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.60 CMP1_RR clock divider control (MRCC_CMP1_RR_CLKDIV)

Offset

Register	Offset
MRCC_CMP1_RR_CLKDIV	16Ch

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.61 DAC0 clock selection control (MRCC_DAC0_CLKSEL)

Offset

Register	Offset
MRCC_DAC0_CLKSEL	170h

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												MUX			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

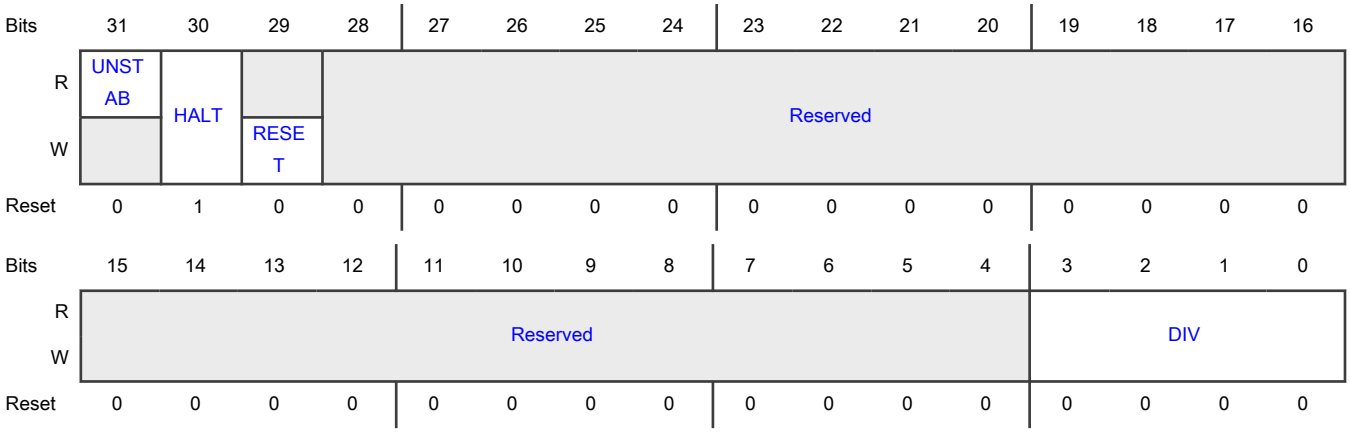
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.62 DAC0 clock divider control (MRCC_DAC0_CLKDIV)

Offset

Register	Offset
MRCC_DAC0_CLKDIV	174h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0	Functional Clock Divider

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIV	Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.63 FLEXCAN0 clock selection control (MRCC_FLEXCAN0_CLKSEL)

Offset

Register	Offset
MRCC_FLEXCAN0_CLKSEL	178h

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												MUX			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

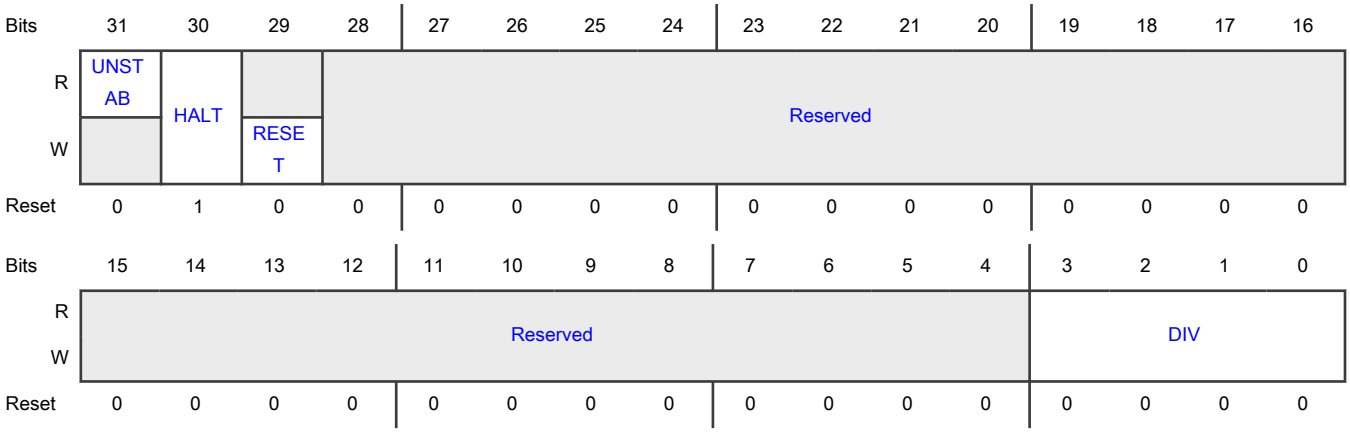
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 010b - FRO_HF_DIV 011b - CLK_IN 111b - Reserved(NO Clock)

14.5.2.64 FLEXCAN0 clock divider control (MRCC_FLEXCAN0_CLKDIV)

Offset

Register	Offset
MRCC_FLEXCAN0_CLKDIV	17Ch

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.65 LPI2C2 clock selection control (MRCC_LPI2C2_CLKSEL)

Offset

Register	Offset
MRCC_LPI2C2_CLKSEL	180h

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												MUX			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

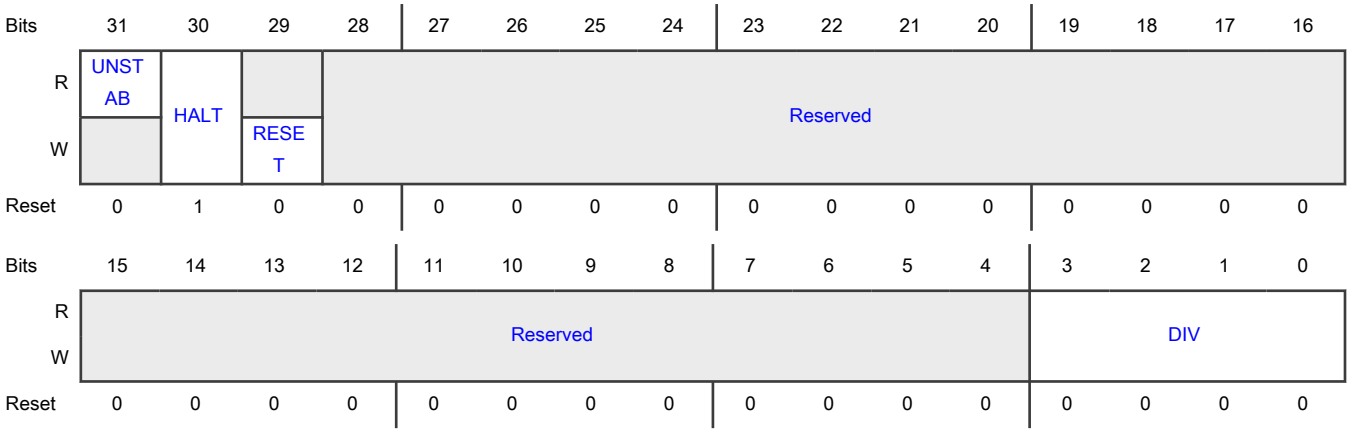
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.66 LPI2C2 clock divider control (MRCC_LPI2C2_CLKDIV)

Offset

Register	Offset
MRCC_LPI2C2_CLKDIV	184h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0	Functional Clock Divider

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIV	Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.67 LPI2C3 clock selection control (MRCC_LPI2C3_CLKSEL)

Offset

Register	Offset
MRCC_LPI2C3_CLKSEL	188h

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												MUX			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

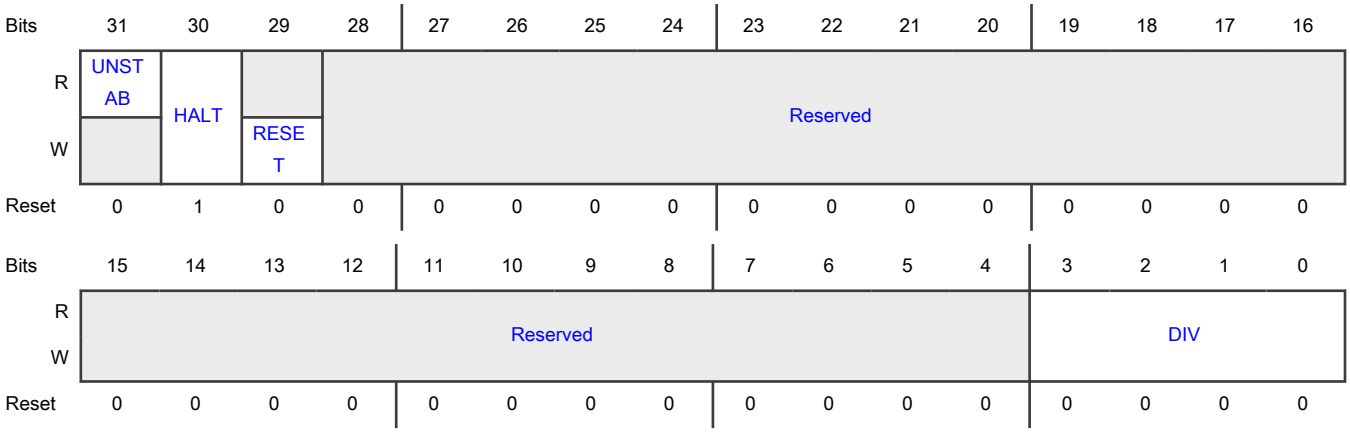
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 010b - FRO_HF_DIV 011b - CLK_IN 101b - CLK_1M 111b - Reserved(NO Clock)

14.5.2.68 LPI2C3 clock divider control (MRCC_LPI2C3_CLKDIV)

Offset

Register	Offset
MRCC_LPI2C3_CLKDIV	18Ch

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved
3-0	Functional Clock Divider

Table continues on the next page...

Table continued from the previous page...

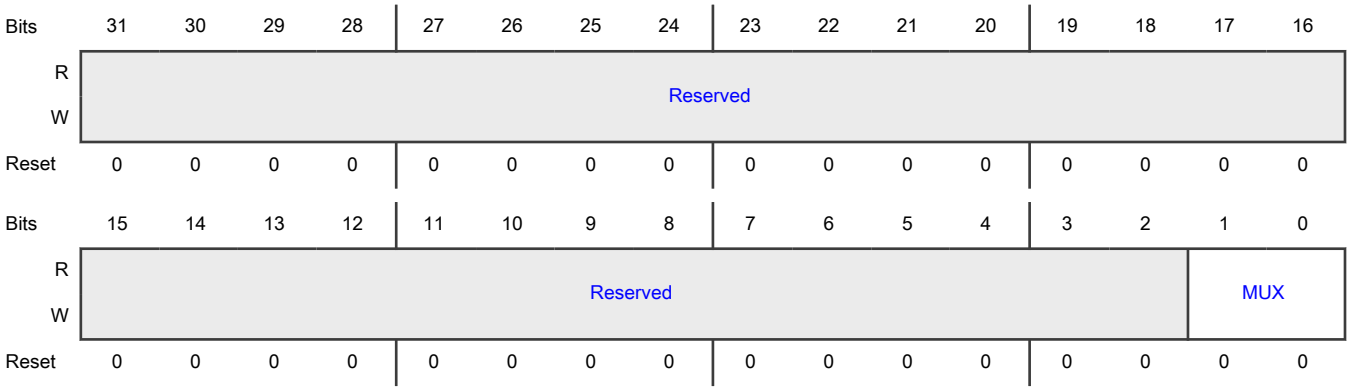
Field	Function
DIV	Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.69 DBG_TRACE clock selection control (MRCC_DBG_TRACE_CLKSEL)

Offset

Register	Offset
MRCC_DBG_TRACE_CLKSEL	190h

Diagram



Fields

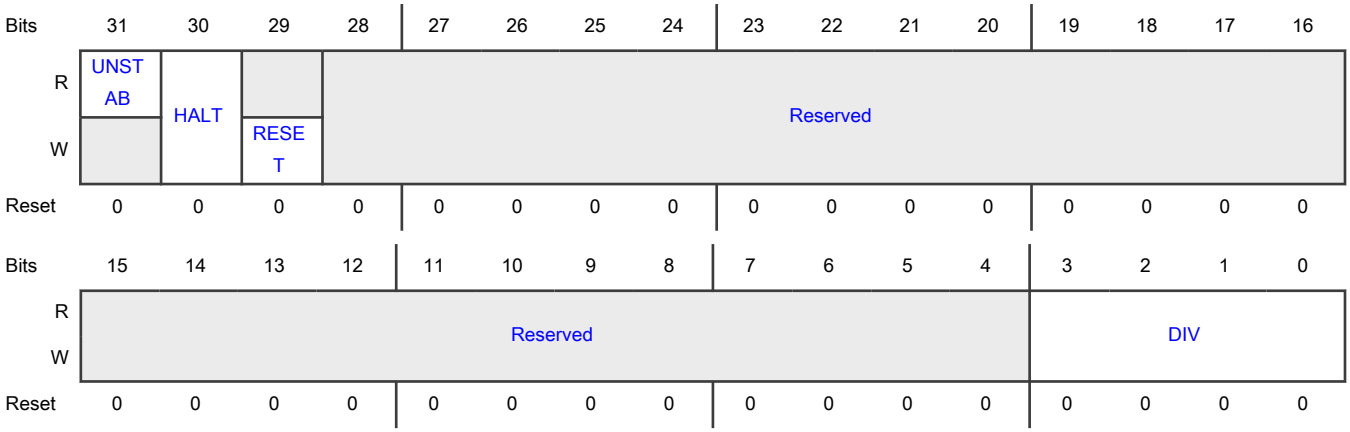
Field	Function
31-2 —	reserved reserved
1-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 00b - CPU_CLK 01b - CLK_1M 10b - CLK_16K 11b - Reserved1(NO Clock)

14.5.2.70 DBG_TRACE clock divider control (MRCC_DBG_TRACE_CLKDIV)

Offset

Register	Offset
MRCC_DBG_TRACE_CLKDIV	194h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.71 CLKOUT clock selection control (MRCC_CLKOUT_CLKSEL)

Offset

Register	Offset
MRCC_CLKOUT_CLKSEL	198h

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													MUX		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Fields

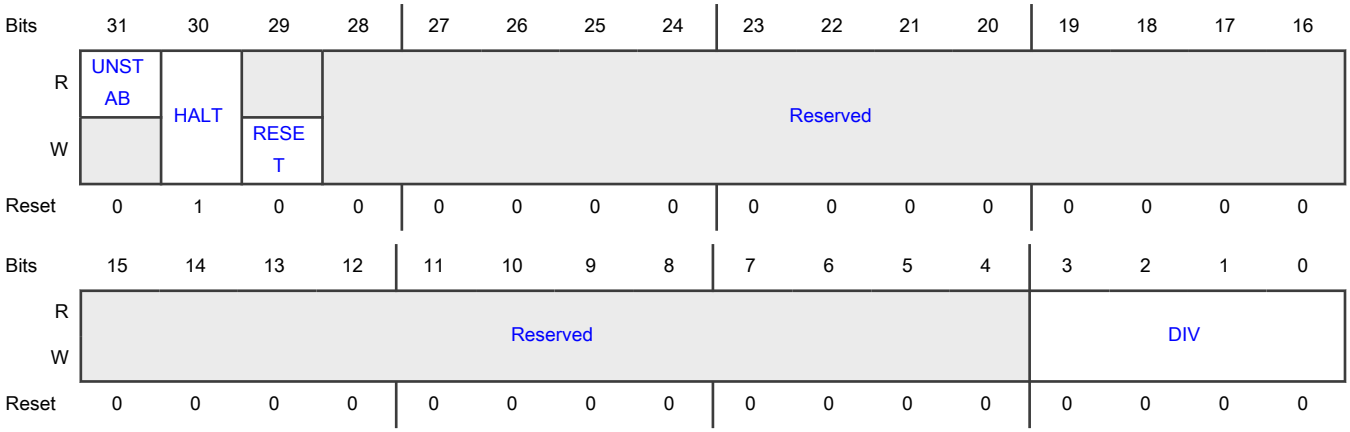
Field	Function
31-3 —	reserved reserved
2-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 000b - FRO_12M 001b - FRO_HF_DIV 010b - CLK_IN 011b - CLK_16K 110b - SLOW_CLK 111b - Reserved(NO Clock)

14.5.2.72 CLKOUT clock divider control (MRCC_CLKOUT_CLKDIV)

Offset

Register	Offset
MRCC_CLKOUT_CLKDIV	19Ch

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.73 SYSTICK clock selection control (MRCC_SYSTICK_CLKSEL)

Offset

Register	Offset
MRCC_SYSTICK_CLKSEL	1A0h

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														MUX	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Fields

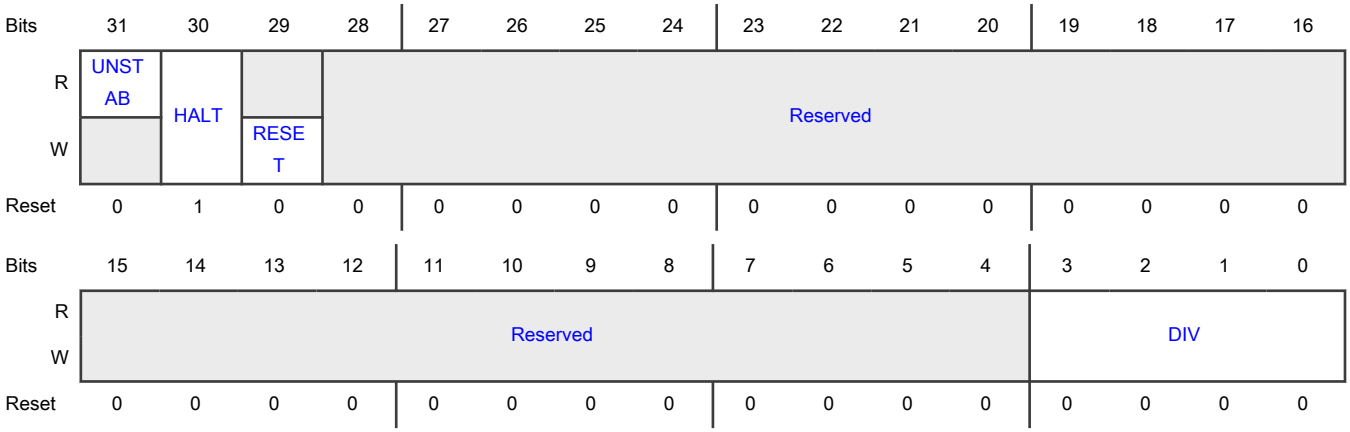
Field	Function
31-2 —	reserved reserved
1-0 MUX	Functional Clock Mux Select Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. 00b - CPU_CLK 01b - CLK_1M 10b - CLK_16K 11b - Reserved1(NO Clock)

14.5.2.74 SYSTICK clock divider control (MRCC_SYSTICK_CLKDIV)

Offset

Register	Offset
MRCC_SYSTICK_CLKDIV	1A4h

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30 HALT	Halt divider counter Halt divider counter 0b - Divider clock is running 1b - Divider clock is stopped
29 RESET	Reset divider counter Reset divider counter 0b - Divider isn't reset 1b - Divider is reset
28-4 —	reserved reserved

Table continues on the next page...

Table continued from the previous page...

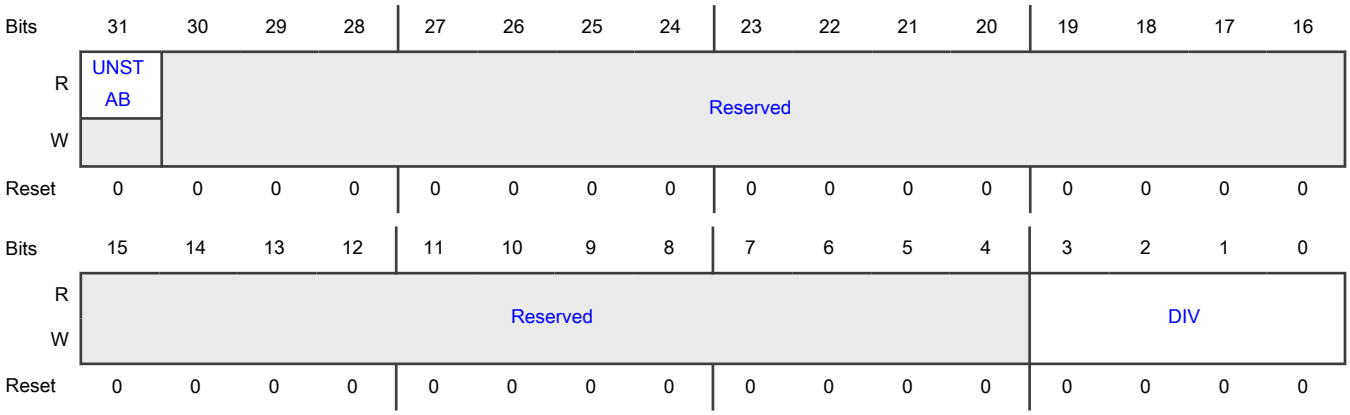
Field	Function
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.5.2.75 FRO_HF_DIV clock divider control (MRCC_FRO_HF_DIV_CLKDIV)

Offset

Register	Offset
MRCC_FRO_HF_DIV_CLKDIV	1ACh

Diagram



Fields

Field	Function
31 UNSTAB	Divider status flag Determines the stability of the divider clock. 0b - Divider clock is stable 1b - Clock frequency isn't stable
30-4 —	reserved reserved
3-0 DIV	Functional Clock Divider Before change peripheral's funcitional clock source, should configure the module's MRCC_GLB_CC and MRCC_GLB_ACC bits to 0. divider value = (DIV+1)

14.6 Application information

Peripheral clock is gated by using CC (CC = {MRCC_GLB_ACC.*[peripherals name]*, MRCC_GLB_CC.*[peripherals name]* })

1. CC = 0b00 : Peripheral clocks are disabled; module does not stall low power mode entry. Clock is always gated, peripheral access causes hard fault.
2. CC = 0b01 : Peripheral clocks are enabled; module does not stall low power mode entry. Clock is never gated unless root clock source is gated in DEEPSLEEP mode.

   set CC=0b10 or CC=0b11 when using async DMA/interrupt of LPUART, LPSPI, LPI2C, FLEXIO and GPIO in DEEPSLEEP0 mode.
3. CC = 0b10 : Peripheral clocks are enabled unless module is idle; low power mode entry stalls until module is idle. Only LPUART, LPSPI, LPI2C, FLEXIO, GPIO, FMU support CC=0b10. for other modules, it's same as CC=0b11.
4. CC = 0b11 : Peripheral clocks are enabled unless in DEEPSLEEP (or lower) mode; low power mode entry stalls until module is idle.

NOTE

FMC's CC bits are only available during low power mode, FMC is accessible and no hardfault occurs if CC=0b00 in RUN mode.

Chapter 15
Enhanced DMA Controller (eDMA3)

15.1 Chip-specific eDMA information

Table 74. Reference links to related information

Topic	Related module	Reference
Full description	eDMA	DMA3
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

15.1.1 Module instances

This device has one 8-channel eDMA controller.

15.1.2 Direct Memory Access Multiplexer (DMAMUX)

The peripheral request line for each channel of the Direct Memory Access Controller (DMA) 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 66 DMA request signals (30 unused signals are reserved) 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.

15.1.2.1 DMAMUX0 Request Assignments

See the attached DMA_Configuration spreadsheet to know the DMA channel and source assignments for DMAMUX0.

15.2 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 8 channels

15.2.1 Block diagram

Figure 18 illustrates the components of the eDMA system, including the eDMA module (engine).

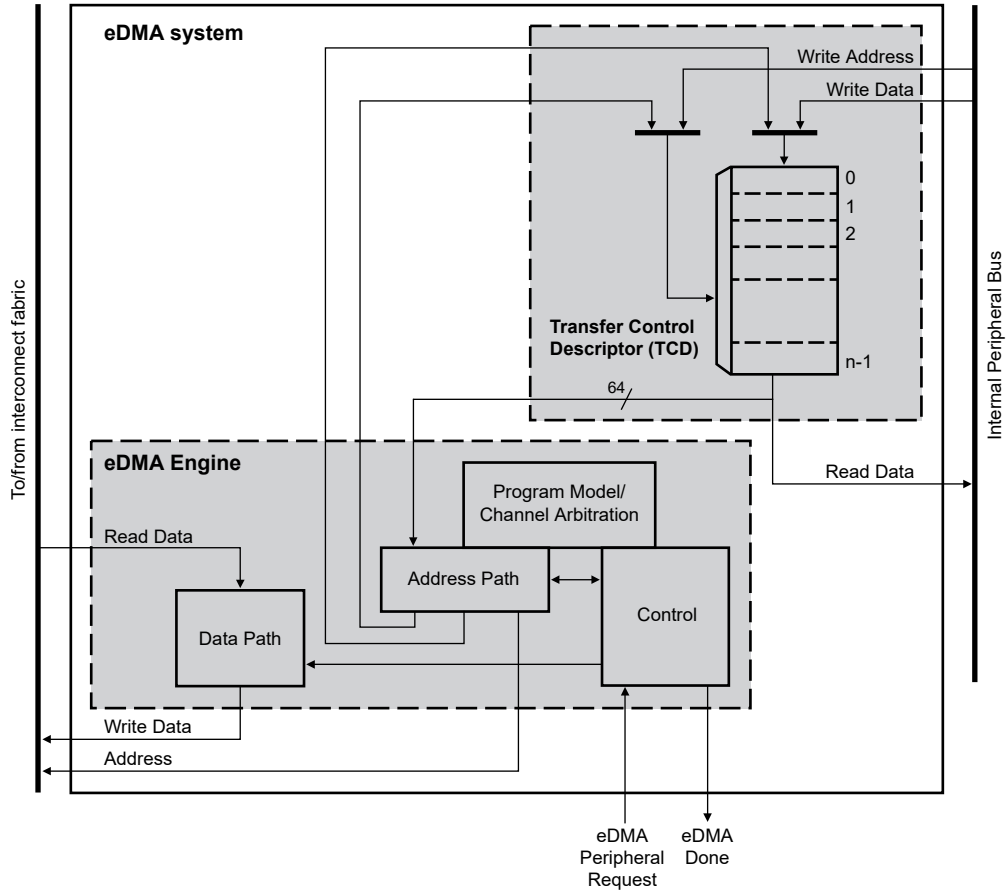


Figure 18. Block diagram

15.2.2 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 75. 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 75. 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 76. 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.

15.2.3 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
- 8-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.

15.3 Functional description

The operation of eDMA is described in the following subsections.

15.3.1 Modes of operation

The eDMA operates in the following modes:

Table 77. 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.

15.3.2 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:

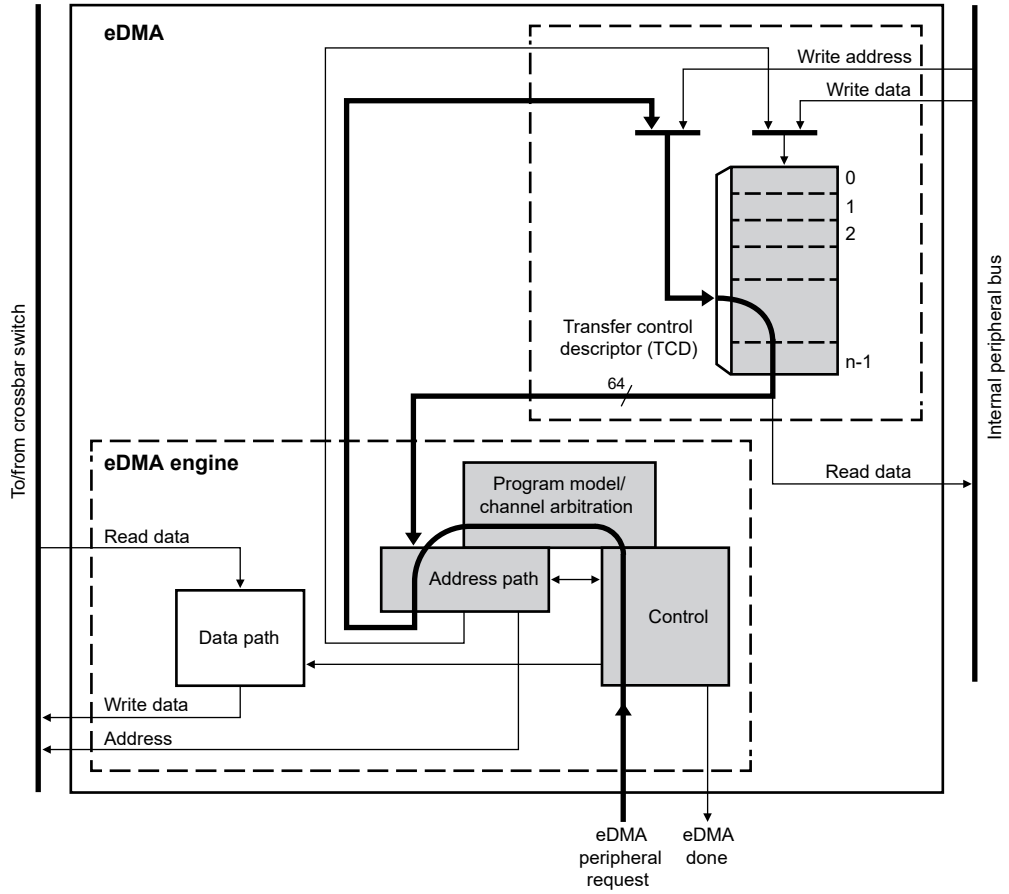


Figure 19. eDMA operation, part 1

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:

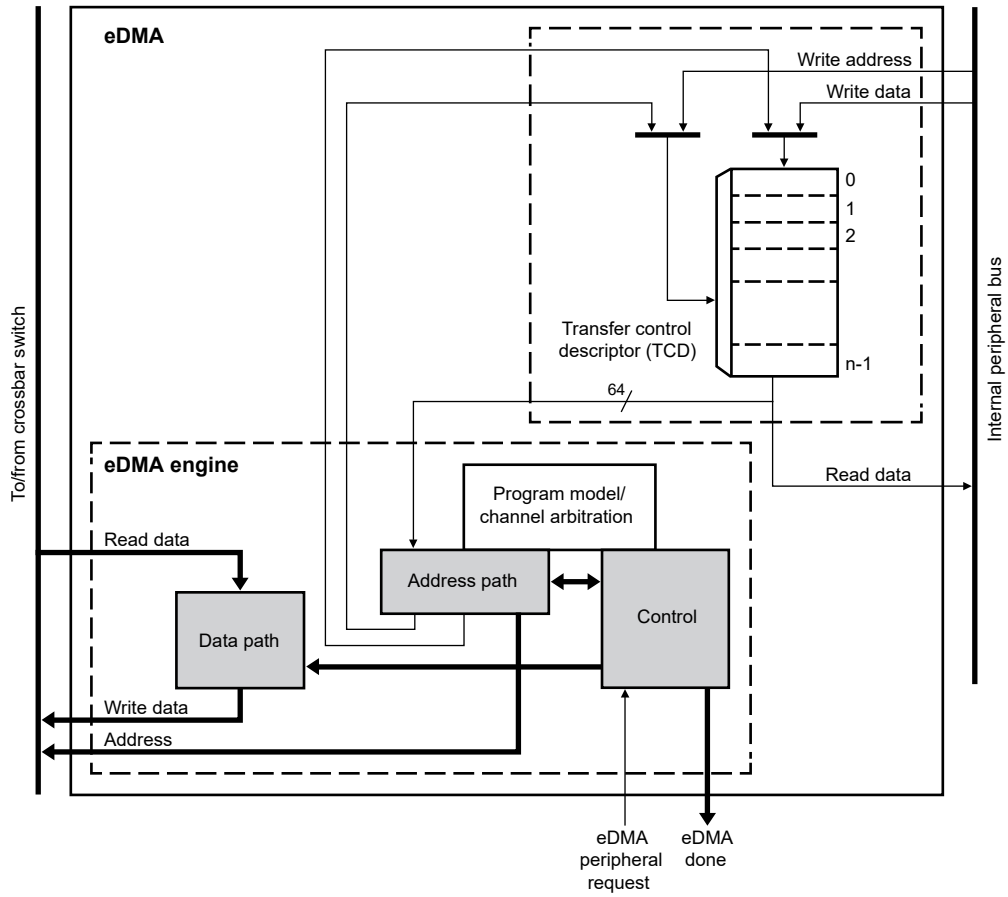


Figure 20. eDMA operation, part 2

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.

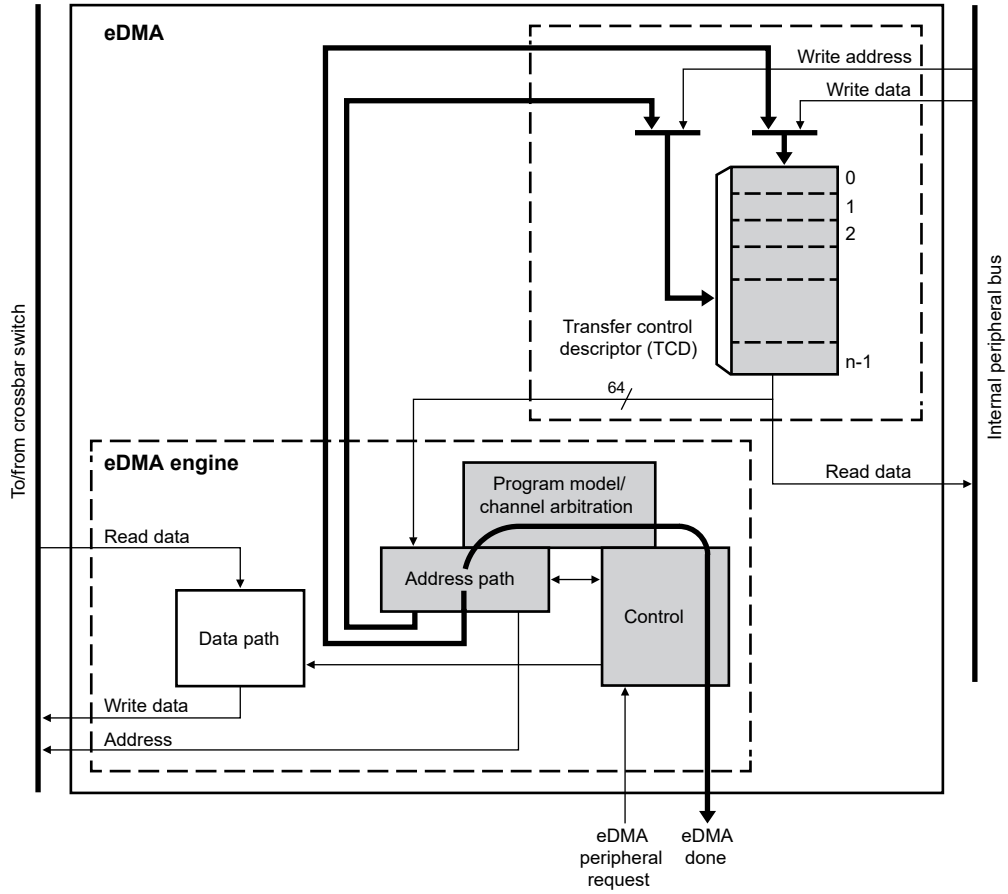


Figure 21. eDMA operation, part 3

15.3.3 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.

15.3.4 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]

It can be considered a number composed of these concatenated priority levels: CHn_GRPRI : CHn_PRI[APL]

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

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.

15.3.5 Clocking

The following table describes the clock sources:

Table 78. Clocks

Clock	Description
hclk	Clock used for the DMA engine
ipg_clk	Clock used for the DMA programming model
dma_ram_clk	Clock used for the TCD RAMs

15.3.6 Interrupts

Software can enable the interrupt for each channel for the following events:

1. The major loop is half complete (INTHALF)
2. The major loop is complete (INTMAJOR)
3. A configuration error occurs (EEI)

15.4 External signals

This module has no external signals.

15.5 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

15.5.1 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 79, 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 79. 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
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 80. 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).

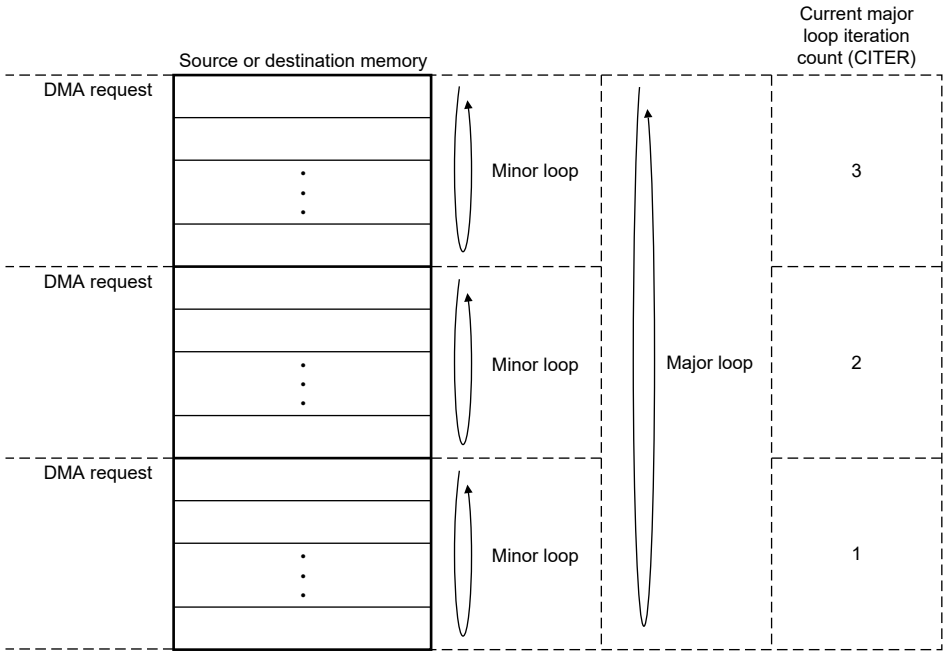


Figure 22. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings are related.

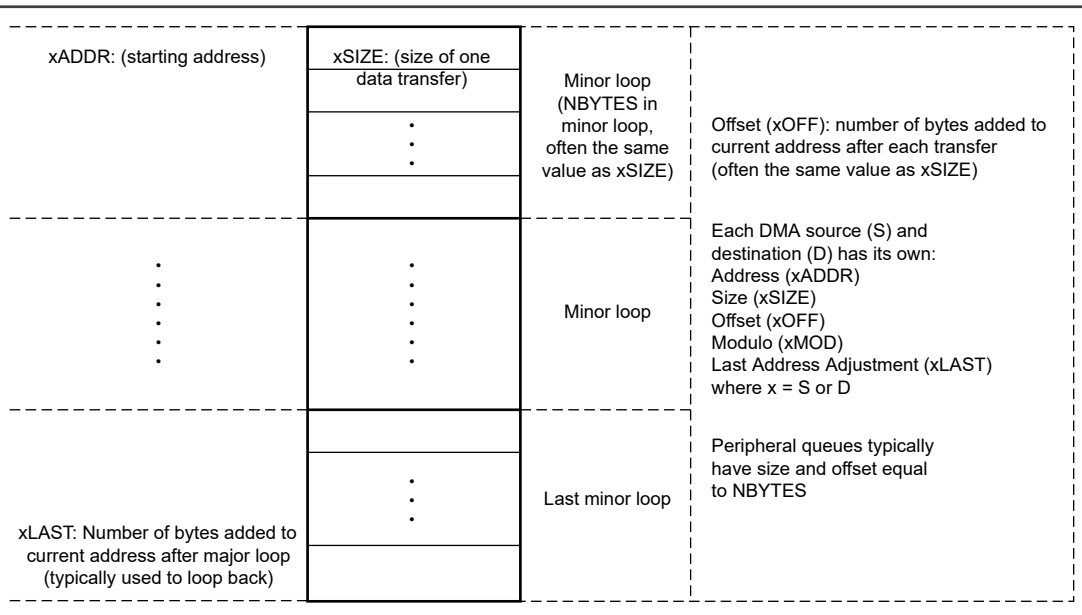


Figure 23. Memory array terms

15.5.2 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 81. 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 given 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.

15.5.3 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.

15.5.4 Arbitration mode considerations

This section discusses arbitration considerations for eDMA.

15.5.4.1 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.

15.5.4.2 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.

15.5.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

15.5.5.1 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.

15.5.5.2 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.

15.5.5.3 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 82. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

15.5.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

15.5.6.1 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.

15.5.6.2 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.

15.5.6.3 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.

15.5.7 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:

```
MP_CSR[GCLC] = 1
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 83. 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

15.5.8 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

Steps for channel linking:

1. Set GCLC = 1
2. For linking after major loop complete:
   a. Set MAJORLINKCH = target channel number
   b. Set MAJORELINK = 1, enable the major loop link
   c. Upon major loop completion (CITER decrements to 0), the link will be made.
3. For linking on minor loop completion:
   a. Set LINKCH = target channel number
   b. Set ELINK = 1, enable the minor loop link
   c. Set the beginning iteration count to the same values; LINKCH = target channel and ELINK = 1,
   d. Upon minor loop completion (that is, after each service request except when the major loop is complete) the link will be made.

15.5.8.1 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.

15.5.8.2 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.

15.5.8.3 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[ESG] 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.

15.5.8.3.1 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).

15.5.8.3.2 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).

15.5.9 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.

15.5.9.1 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 SPI 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/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the SPI.
2. Ensure there is no DMA service request from the SPI 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.

15.5.9.2 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.

15.6 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 DMA TCD register descriptions for details.

TCD initialization

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

TCD structure

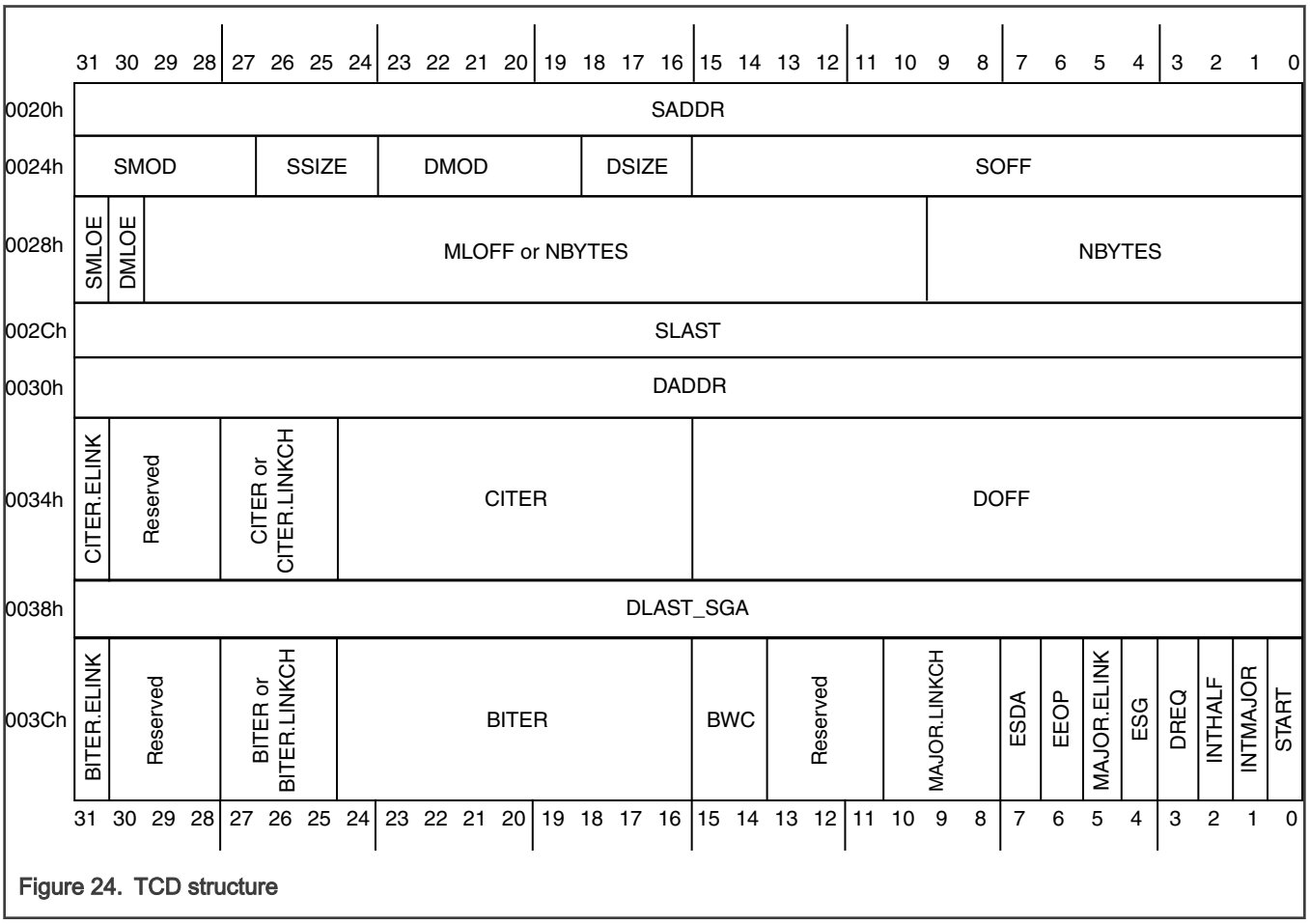


Figure 24. TCD structure

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.

15.6.1 DMA MP register descriptions

15.6.1.1 MP memory map

eDMA_0_MP base address: 4008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Management Page Control (MP_CSR)	32	RW	0031_0000h
4h	Management Page Error Status (MP_ES)	32	R	0000_0000h
8h	Management Page Interrupt Request Status (MP_INT)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	Management Page Hardware Request Status (MP_HRS)	32	R	0000_0000h
100h - 11Ch	Channel Arbitration Group (CH0_GRPRI - CH7_GRPRI)	32	RW	0000_0000h

15.6.1.2 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.

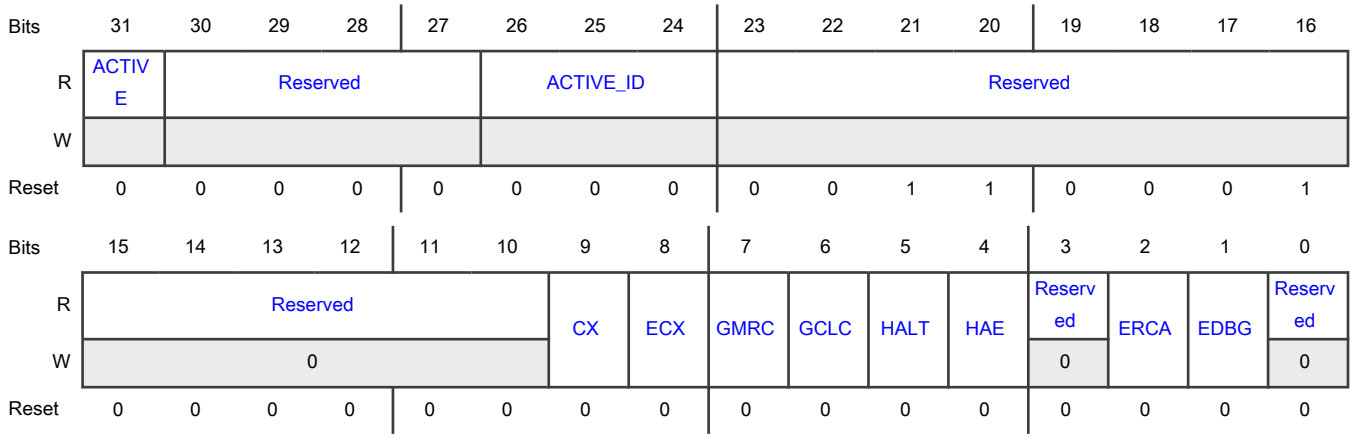
The channel priority registers assign the relative priorities within each arbitration group; see CHn_PRI. All channels with a non-zero CHn_PRI value use fixed-priority arbitration.

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 - CH7_GRPRI).

NOTE

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

Diagram



Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel
30-27 —	Reserved
26-24 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 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 privileged protection level (Supervisor mode) 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	Global Channel Linking Control

Table continues on the next page...

Table continued from the previous page...

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

15.6.1.3 Management Page Error Status (MP_ES)

Offset

Register	Offset
MP_ES	4h

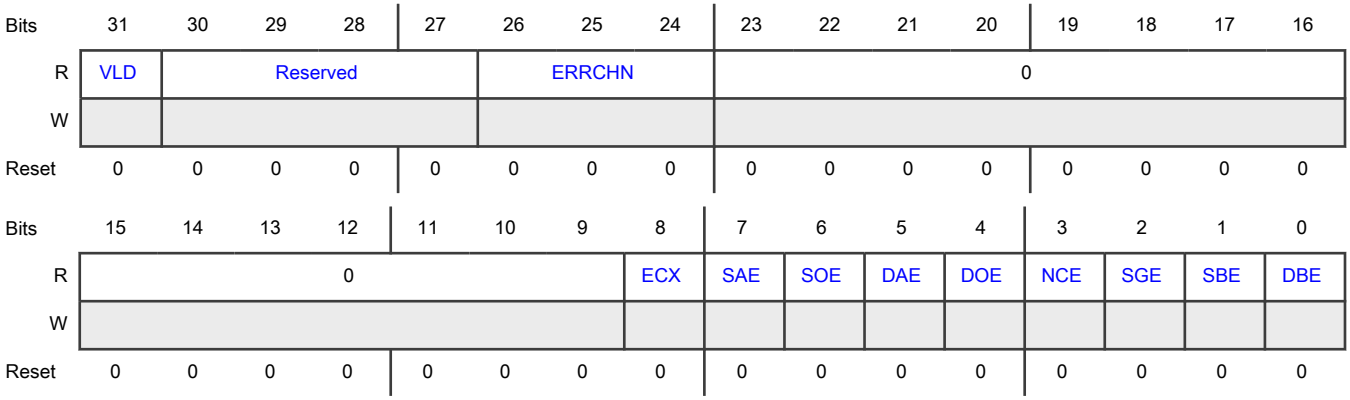
Function

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



Fields

Field	Function
31 VLD	Valid Logical OR of all CHn_ES[ERR] status fields. 0b - No CHn_ES[ERR] fields are set to 1 1b - At least one CHn_ES[ERR] field is set to 1, indicating a valid error exists that software has not cleared
30-27 —	Reserved
26-24 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 CHn_ES[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

Table continues on the next page...

Table continued from the previous page...

Field	Function
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 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 - 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

15.6.1.4 Management Page Interrupt Request Status (MP_INT)

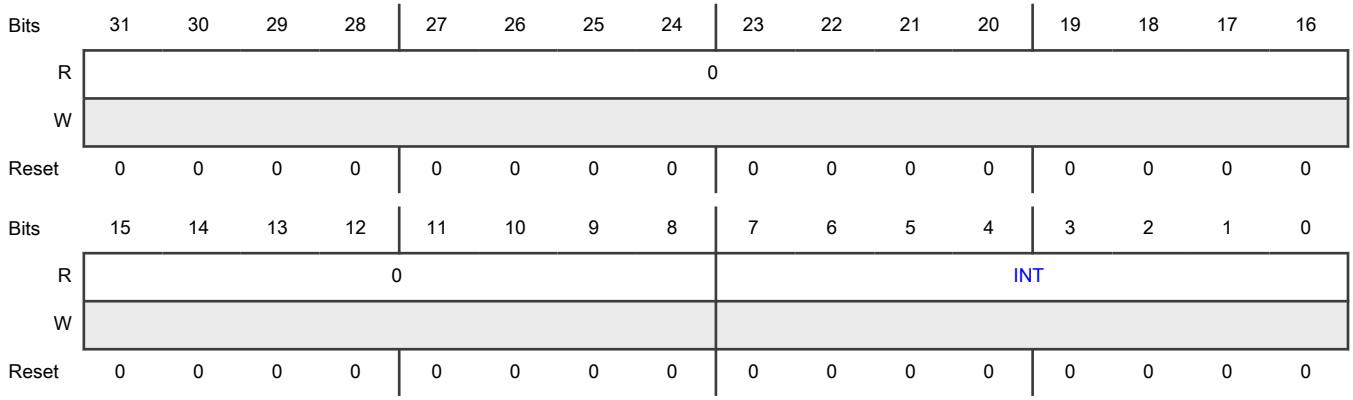
Offset

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
31-8 —	Reserved
7-0 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

15.6.1.5 Management Page Hardware Request Status (MP_HRS)

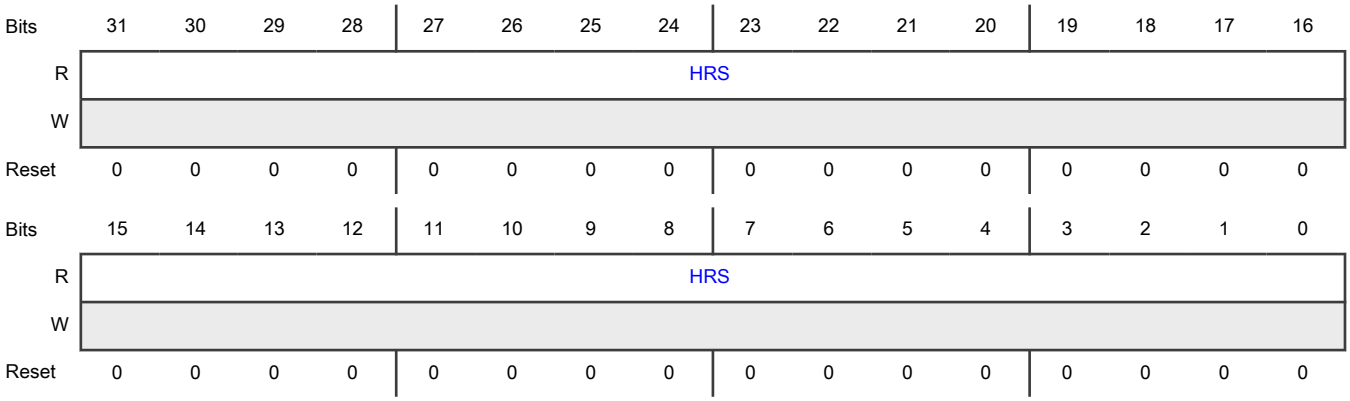
Offset

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



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

15.6.1.6 Channel Arbitration Group (CH0_GRPRI - CH7_GRPRI)

Offset

For n = 0 to 7:

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

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												GRPRI			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

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

15.6.2 DMA TCD register descriptions

15.6.2.1 TCD memory map

eDMA_0_TCD0 base address: 4008_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7000h	Channel Control and Status (CH0_CSR - CH7_CSR)	32	RW	0000_0000h
4h - 7004h	Channel Error Status (CH0_ES - CH7_ES)	32	RW	0000_0000h
8h - 7008h	Channel Interrupt Status (CH0_INT - CH7_INT)	32	RW	0000_0000h
Ch - 700Ch	Channel System Bus (CH0_SBR - CH7_SBR)	32	RW	0000_0005h
10h - 7010h	Channel Priority (CH0_PRI - CH7_PRI)	32	RW	0000_0000h
14h - 7014h	Channel Multiplexor Configuration (CH0_MUX - CH7_MUX)	32	RW	0000_0000h
20h - 7020h	TCD Source Address (TCD0_SADDR - TCD7_SADDR)	32	RW	See section
24h - 7024h	TCD Signed Source Address Offset (TCD0_SOFF - TCD7_SOFF)	16	RW	See section
26h - 7026h	TCD Transfer Attributes (TCD0_ATTR - TCD7_ATTR)	16	RW	See section
28h - 7028h	TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD7_NBYTES_MLOFFNO)	32	RW	See section
28h - 7028h	TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD7_NBYTES_MLOFFYES)	32	RW	See section
2Ch - 702Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD7_SLAST_SDA)	32	RW	See section
30h - 7030h	TCD Destination Address (TCD0_DADDR - TCD7_DADDR)	32	RW	See section
34h - 7034h	TCD Signed Destination Address Offset (TCD0_DOFF - TCD7_DOFF)	16	RW	See section
36h - 7036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD7_CITER_ELINKNO)	16	RW	See section
36h - 7036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD7_CITER_ELINKYES)	16	RW	See section
38h - 7038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD7_DLAST_SGA)	32	RW	See section
3Ch - 703Ch	TCD Control and Status (TCD0_CSR - TCD7_CSR)	16	RW	See section
3Eh - 703Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD7_BITER_ELINKNO)	16	RW	See section
3Eh - 703Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD7_BITER_ELINKYES)	16	RW	See section

15.6.2.2 Channel Control and Status (CH0_CSR - CH7_CSR)

Offset

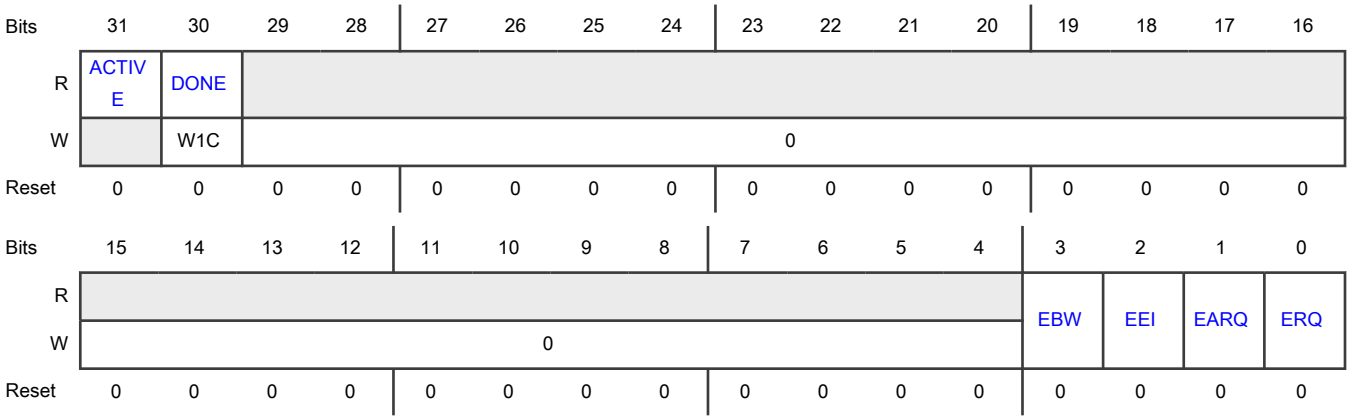
For n = 0 to 7:

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 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

15.6.2.3 Channel Error Status (CH0_ES - CH7_ES)

Offset

For n = 0 to 7:

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.

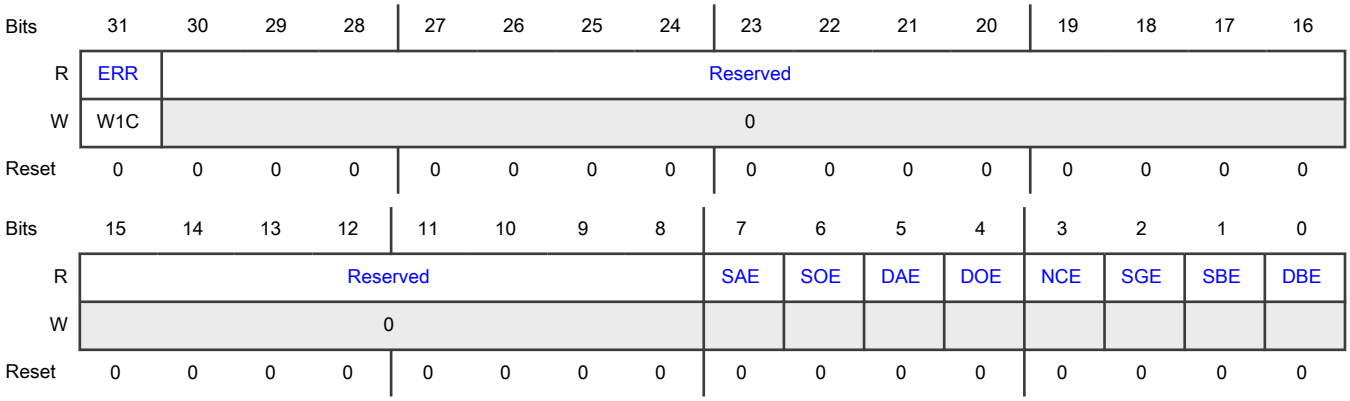
During the execution of the interrupt service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion 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]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field

Table continues on the next page...

Table continued from the previous page...

Field	Function
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

15.6.2.4 Channel Interrupt Status (CH0_INT - CH7_INT)

Offset

For n = 0 to 7:

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

15.6.2.5 Channel System Bus (CH0_SBR - CH7_SBR)

Offset

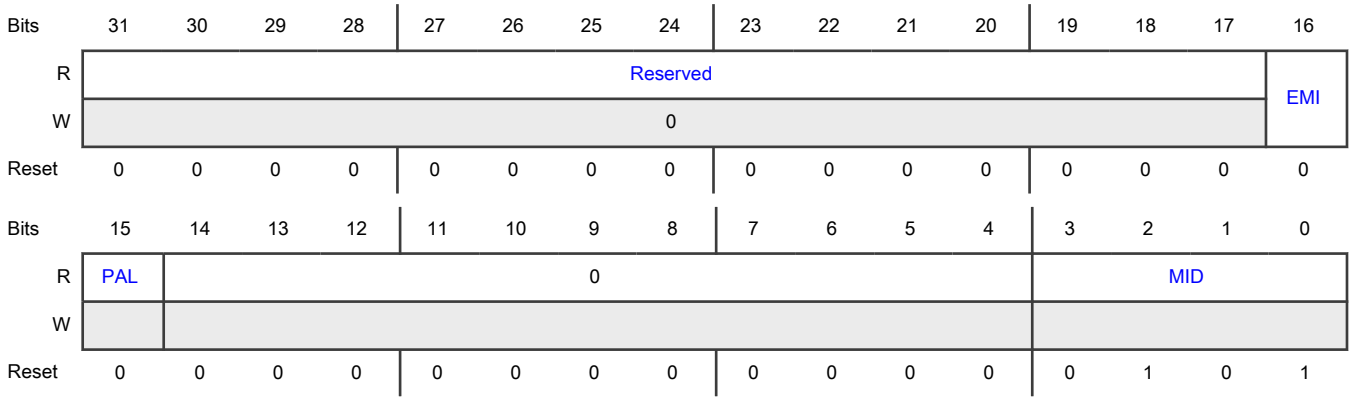
For n = 0 to 7:

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
31-17 —	Reserved
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 PAL	Privileged Access Level This field controls DMA's protection level on the system bus when the channel is active. **NOTE** When you enable master ID replication, the value captured in this register is the privilege level of the core or other master writing the channel's transfer control descriptor, which is the lower byte of TCD*n*_CSR. 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 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 control attributes, which are in the lower byte of TCD*n*_CSR.

15.6.2.6 Channel Priority (CH0_PRI - CH7_PRI)

Offset

For n = 0 to 7:

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 - CH7_GRPRI).

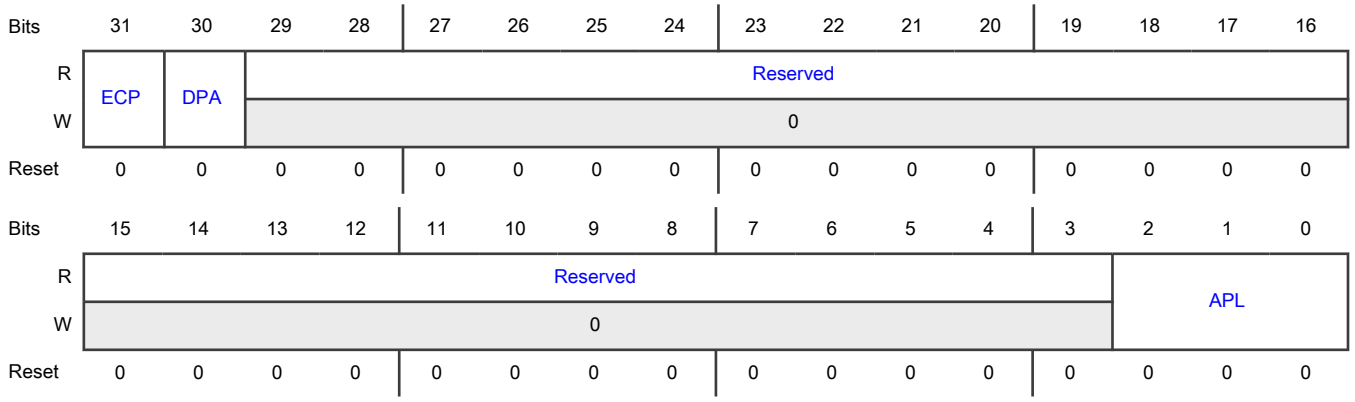
The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected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

15.6.2.7 Channel Multiplexor Configuration (CH0_MUX - CH7_MUX)

Offset

For n = 0 to 7:

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

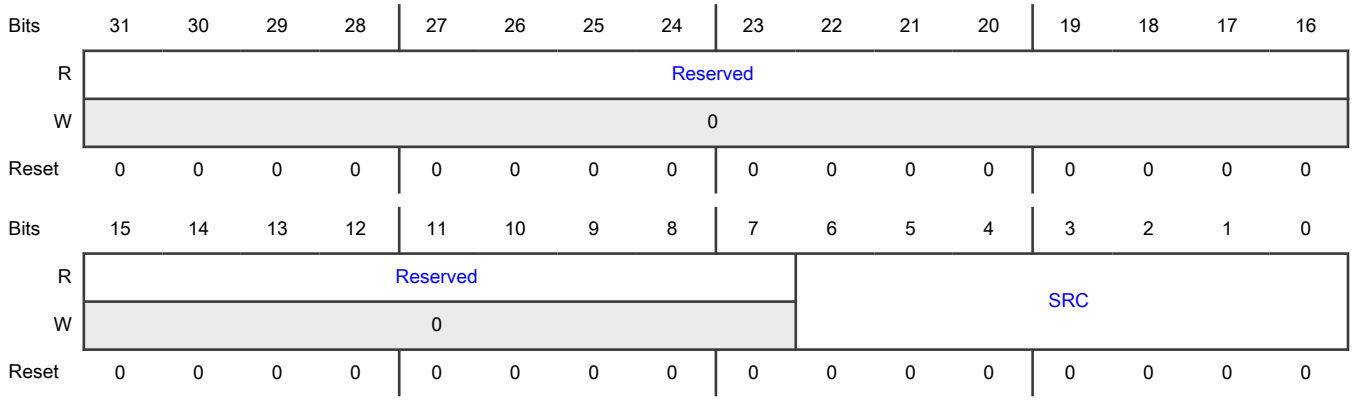
Function

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.

15.6.2.8 TCD Source Address (TCD0_SADDR - TCD7_SADDR)

Offset

For n = 0 to 7:

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

Function

This register contains the address for the read transactions.

Diagram

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

Fields

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

15.6.2.9 TCD Signed Source Address Offset (TCD0_SOFF - TCD7_SOFF)

Offset

For n = 0 to 7:

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

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

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.

15.6.2.10 TCD Transfer Attributes (TCD0_ATTR - TCD7_ATTR)

Offset

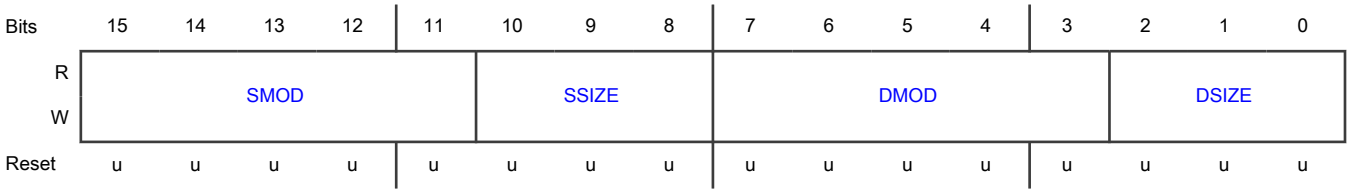
For n = 0 to 7:

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. 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
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-3 DMOD	Destination Address Modulo See the SMOD definition.
2-0 DSIZE	Destination Data Transfer Size See the SSIZE definition.

15.6.2.11 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD7_NBYTES_MLOFFNO)

Offset

For n = 0 to 7:

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)

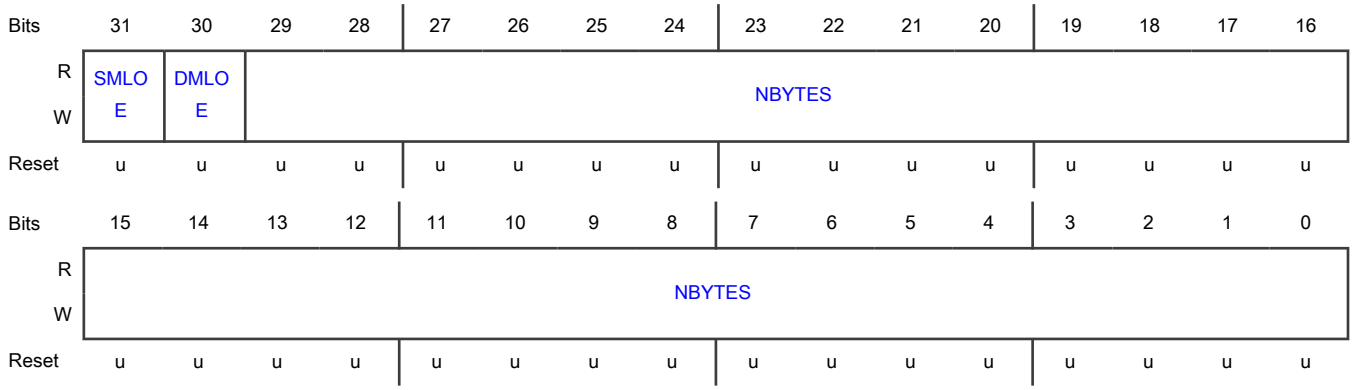
The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit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

15.6.2.12 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD7_NBYTES_MLOFFYES)

Offset

For n = 0 to 7:

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.

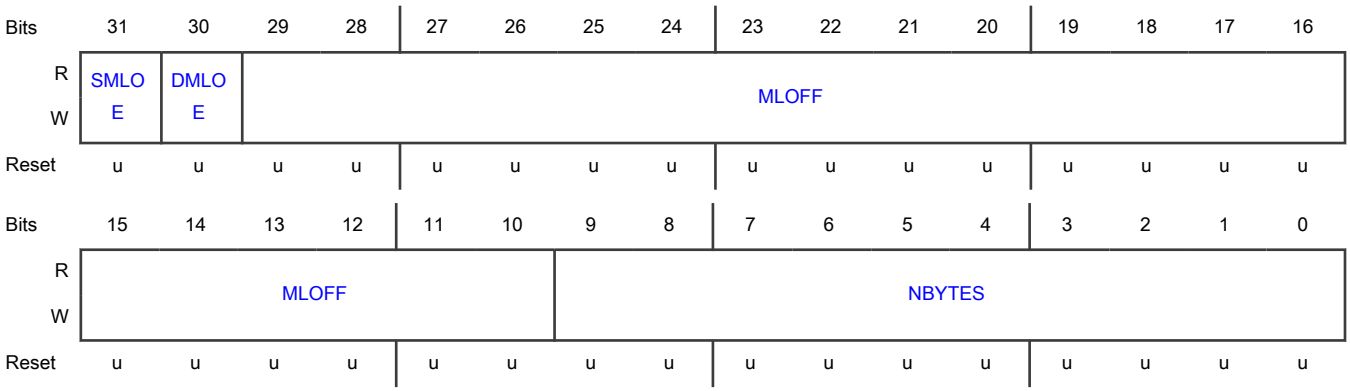
When you enable the minor loop offset overlay (either SMLOE or DMLOE is 1), eDMA redefines TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES specifies the sign-extended minor loop offset value (MLOFF). The same offset value (MLOFF) applies to both source 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.

15.6.2.13 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD7_SLAST_SDA)

Offset

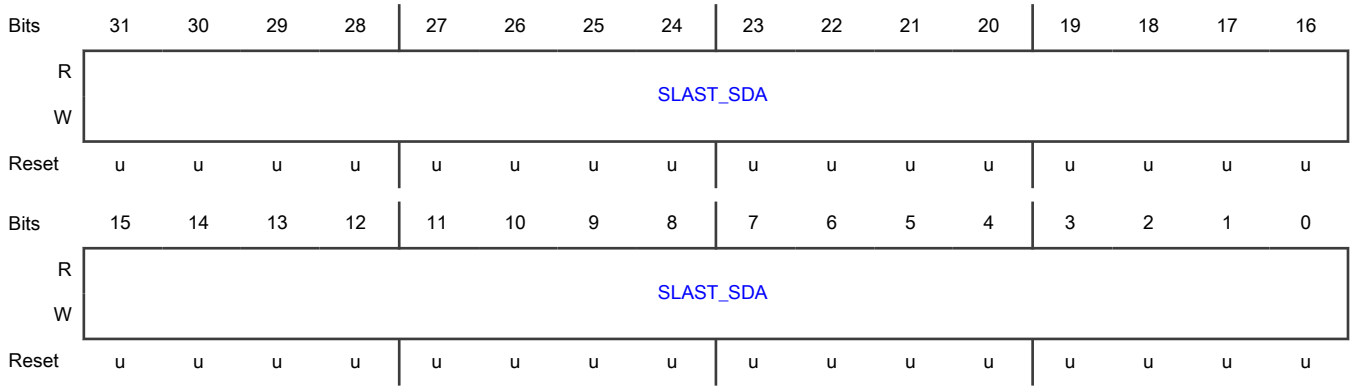
For n = 0 to 7:

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.

15.6.2.14 TCD Destination Address (TCD0_DADDR - TCD7_DADDR)

Offset

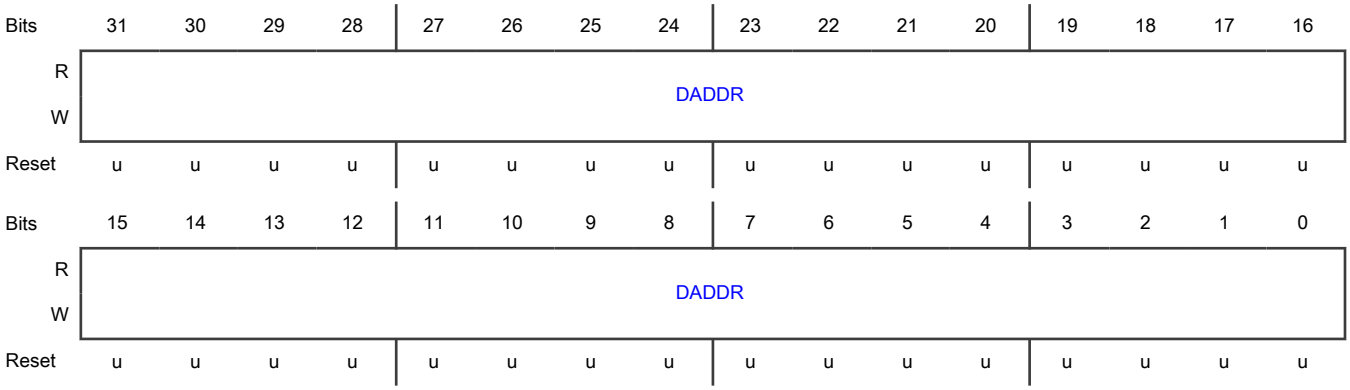
For n = 0 to 7:

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

Function

This register contains the address for the write transactions.

Diagram



Fields

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

15.6.2.15 TCD Signed Destination Address Offset (TCD0_DOFF - TCD7_DOFF)

Offset

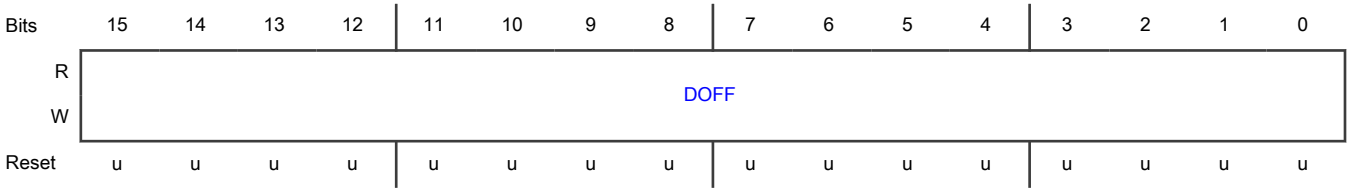
For n = 0 to 7:

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.

15.6.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD7_CITER_ELINKNO)

Offset

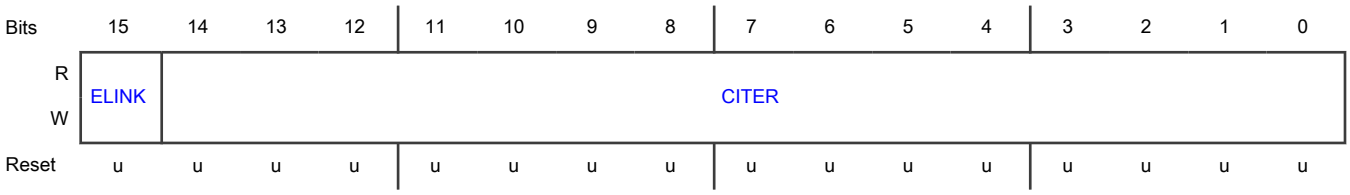
For n = 0 to 7:

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. **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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

15.6.2.17 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD7_CITER_ELINKYES)

Offset

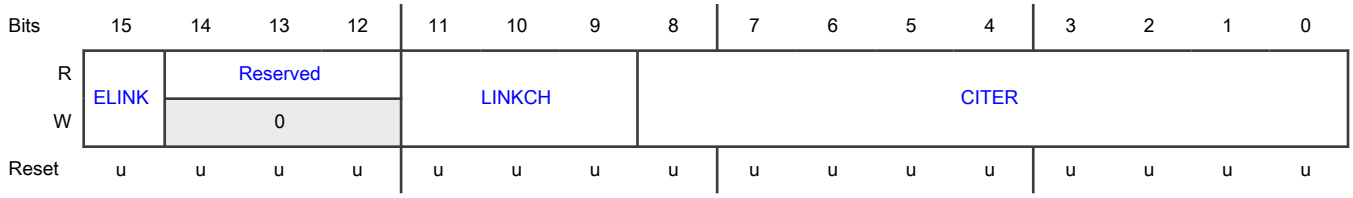
For n = 0 to 7:

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.

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
14-12 —	Reserved
11-9 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.

15.6.2.18 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD7_DLAST_SGA)

Offset

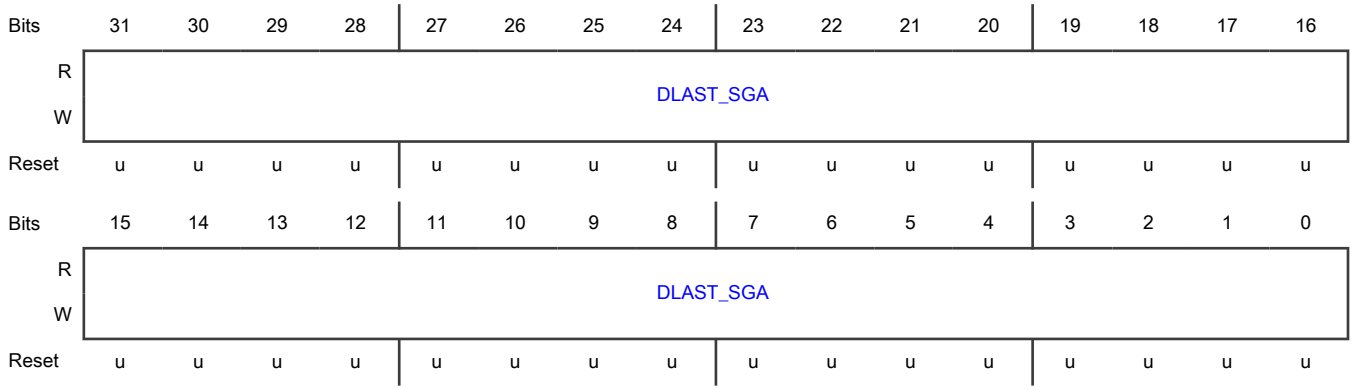
For n = 0 to 7:

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.

15.6.2.19 TCD Control and Status (TCD0_CSR - TCD7_CSR)

Offset

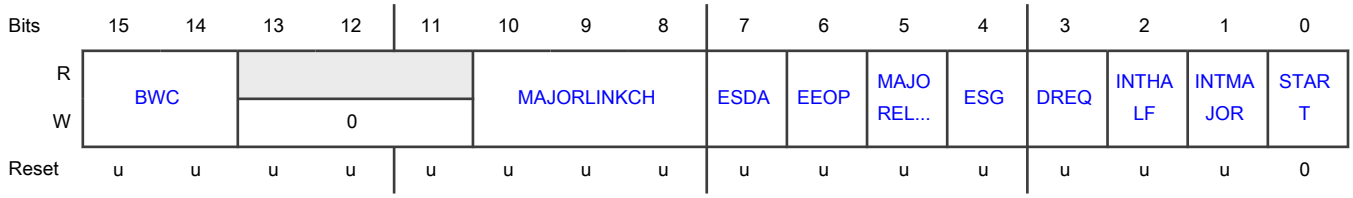
For n = 0 to 7:

Register	Offset
TCDn_CSR	3Ch + (n × 1000h)

Function

This register is used to enable optional features.

Diagram



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
13-11 —	Reserved
10-8 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: • 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 next DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation. When the ESDA 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. 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	Enable Interrupt If Major Counter Half-complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
INTHALF	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

15.6.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD7_BITER_ELINKNO)

Offset

For n = 0 to 7:

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.

15.6.2.21 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD7_BITER_ELINKYES)

Offset

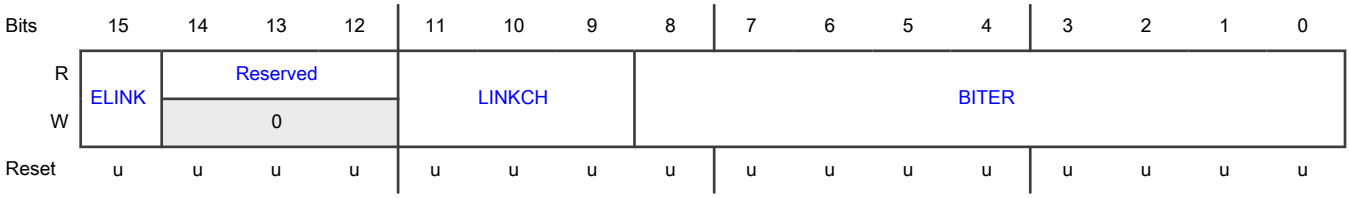
For n = 0 to 7:

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
14-12 —	Reserved
11-9 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 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.

Chapter 16
Wakeup Unit (WUU)

16.1 Chip-specific WUU information

Table 84. Reference links to related information

Topic	Related module	Reference
Full description	WUU	WUU
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

16.1.1 Module instances

This device contains one instance of the WUU module, WUU0.

16.1.2 Clock signal names

LPO clock is clk_16k[0].

16.1.3 WUU sources

The device uses the following inputs as wakeup sources to the WUU module. The WUU_Px inputs are connections to external physical pins. The WUU_MxIF inputs are connections to the internal peripheral interrupt flags from different modules that can operate in low-power modes. The WUU_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 WUU wakeup sources, the device also wakes from low power modes when NMI or RESET pins are enabled and the respective pin is asserted.

Table 85. Wakeup Sources for WUU inputs

WUU input	Source description
WUU_P0	Reserved
WUU_P1	Reserved
WUU_P2	P0_16
WUU_P3	Reserved
WUU_P4	Reserved
WUU_P5	Reserved
WUU_P6	P1_0 / LPTMR0_ALT3
WUU_P7	P1_3

Table continues on the next page...

Table 85. Wakeup Sources for WUU inputs (continued)

WUU input	Source description
WUU_P8	P1_4
WUU_P9	P1_7
WUU_P10	P1_8
WUU_P11	P1_11
WUU_P12	P1_12
WUU_P13	Reserved
WUU_P14	Reserved
WUU_P15	Reserved
WUU_P16	P4_2
WUU_P17	P4_6
WUU_P18	P2_0
WUU_P19	P2_3
WUU_P20	P2_12
WUU_P21	Reserved
WUU_P22	P3_0
WUU_P23	P3_8
WUU_P24	P3_11
WUU_P25	P3_14
WUU_P26	P3_28
WUU_P27	P3_29
WUU_P28	USB0_DM
WUU_P29	USB0_DP
WUU_P30	WUU_P30 P3_27
WUU_P31	CMP1_OUT
WUU_M0IF	SPC Interrupt
WUU_M1IF	Reserved
WUU_M2IF	Wake-up Timer
WUU_M3IF	Reserved
WUU_M4IF	Reserved
WUU_M5IF	Reserved
WUU_M6IF	LPTMR0 Interrupt
WUU_M7IF	Reserved

Table continues on the next page...

Table 85. Wakeup Sources for WUU inputs (continued)

WUU input	Source description
WUU_M8IF	CMP0 Interrupt
WUU_M9IF	Reserved
WUU_M0DR	Reserved
WUU_M1DR	Reserved
WUU_M2DR	Reserved
WUU_M3DR	Reserved
WUU_M4DR	LPTMR0 DMA Request
WUU_M5DR	Reserved
WUU_M6DR	LPTMR0 Trig
WUU_M7DR	Reserved
WUU_M8DR	CMP0 DMA Request
WUU_M9DR	Reserved

16.1.4 Access modes

If the privilege access is unavailable, the peripheral could be access in the user mode.

16.2 Overview

WUU facilitates in selecting external pins and on-chip modules as interrupt wake-up sources from Power Down/Deep Power Down power modes. You can also configure the external pins to act as interrupt wake-up sources in all power modes. WUU helps pins and modules generate a temporary wake-up from Power Down mode for servicing DMA or trigger events. Digital filtering is also available for the external wake-up pins.

See the chip-specific WUU information section for input sources to WUU.

16.2.1 Block diagram

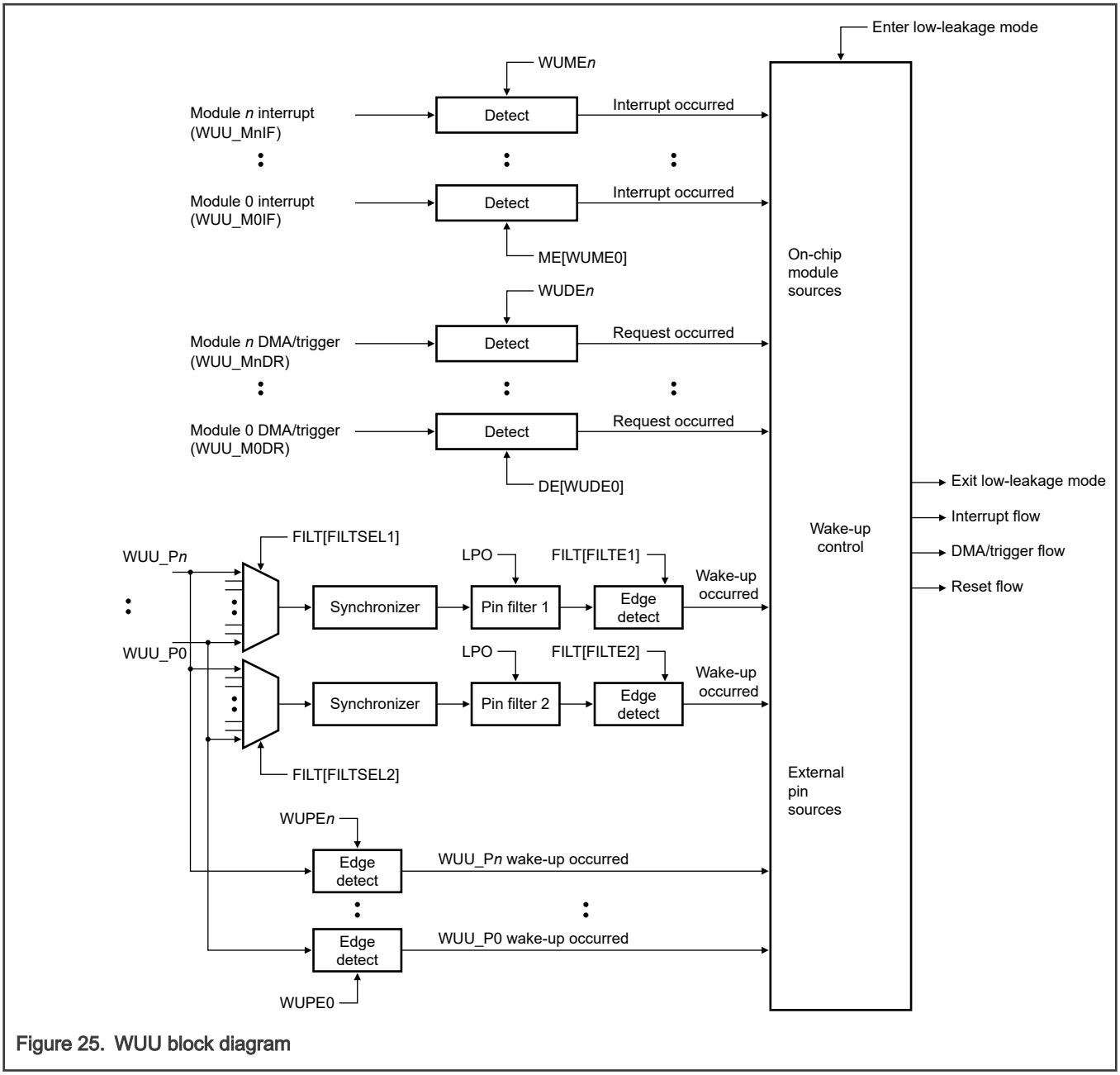


Figure 25. WUU block diagram

16.2.2 Features

- Supports external input pins and on-chip modules with individual enable bits to wake the chip from low-leakage modes.
- Contains input sources that may be external pins or from on-chip modules capable of running in Power Down or Deep Power Down power modes. See the chip-specific WUU information for the wake-up input sources for this chip.
- Supports configuring on-chip modules as DMA or trigger sources for temporary wake-up from Power Down mode.
- Supports configuring external pins as DMA request or trigger sources for temporary wake-up from Power Down mode.
- Provides external input pins programmable for falling-edge, rising-edge, or any edge detection.
- Provides optional digital filters to qualify an external pin detect. Disabling the LPO clock bypasses and disables filters.

- Filters all external input pins; but only 2 filters are available at a given time.
- Enables external input pin and filter detection in all power modes (not limited to Power Down/Deep Power Down power modes).

16.3 Functional description

16.3.1 Operation

WUU allows on-chip modules and external input pins as a source of wake-up from low-leakage modes.

WUU contains pin enables for each external pin and on-chip module. For each external pin, you can disable or select the edge type for the wake-up with the following options:

- Falling edge
- Rising edge
- Either edge

Exit due to a pin event triggers a WUU interrupt after wake-up is complete. When enabling an external pin as a wake-up source, configure the pin as an input pin. You can also configure a pin to generate an interrupt from a Power Down/Deep Power Down mode or a DMA/trigger request for temporary wake-up from Power Down mode. Detection logic for a pin can optionally remain enabled during all power modes using Pin Mode Configuration (PMC).

On the basis of the LPO clock, WUU implements optional 3-cycle glitch filters. Detected external pin must remain asserted until the enabled glitch filter times out. Additional latency of up to two cycles is due to synchronization, which results in a total of up to five cycles of delay before the detect circuit alerts the system to the wake-up event when the filter function is enabled. Two wake-up detect filters are available for selected external pins. Glitch filtering is not provided for the on-chip modules. You can configure a filter to generate an interrupt from Power Down/Deep Power Down mode or a DMA/trigger request for temporary wake-up from Power Down mode. Detection logic for a filter can optionally remain enabled during all power modes using Pin Filter Mode Configuration (FMC).

For on-chip module interrupts, ME[WUME*n*] enables the associated module interrupt as a wake-up source. Exit due to a module interrupt will not trigger a WUU interrupt.

For on-chip module DMA/trigger requests, DE[WUDE*n*] enables the associated module DMA/trigger request as a wake-up source. Exit due to a module DMA/trigger request does not trigger a WUU interrupt.

16.3.2 Modes of operation

WUU becomes functional on entry into a low-leakage power mode. After recovery from Power Down mode, WUU is immediately disabled. After recovery from Deep Power Down mode, WUU continues to detect wake-up events until you have reinitialized the system and deasserted isolation. Detection of external pin/filter events can also optionally remain enabled in all power modes using Pin Mode Configuration (PMC)/Pin Filter Mode Configuration (FMC).

16.3.2.1 Power Down mode

Wake-up 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 wake-up 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 interrupt controller must not mask the WUU interrupt to avoid a scenario where the system does not fully recover from Power Down mode after an external pin event.

Wake-up events due to an on-chip module request (WUU_MxDR) DMA/trigger request result in a temporary exit from Power Down mode to allow the request to be serviced when the CPU clock remains gated. The system reenters Power Down mode after servicing the request.

Wake-up events due to an external pin (WUU_Px) DMA/trigger request result in a temporary exit from Power Down mode to allow the request to be serviced when the CPU clock remains gated. The system reenters Power Down mode after servicing the request.

16.3.2.2 Non–low-leakage modes

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 Pin Mode Configuration (PMC)/Pin Filter Mode Configuration (FMC). WUU chips are accessible in non–low-leakage modes and are available for configuring and reading status when bus transactions are possible.

16.3.3 Clocking

During chip low-power operation, in order to provide the wake-up function for the chip, ensure that clocking is available for WUU by the LPO clock. See the chip-specific WUU information for the chip clock connection details.

16.3.4 Reset

WUU resets after VSYS warm reset. This is the global chip reset, but it does not include the reset that asserts due to wake-up from low-power mode.

16.3.5 Interrupts

You can enable on-chip module interrupts as wake-up sources in Module Interrupt Enable (ME).

16.3.6 DMA

You can enable on-chip DMA requests as wake-up sources in Module DMA/Trigger Enable (DE).

With external pin inputs, you can generate a DMA request using either rising edge, falling edge, or both edge detection in Pin DMA/Trigger Configuration 1 (PDC1) and Pin DMA/Trigger Configuration 2 (PDC2).

16.4 External signals

Table 86. External signals

Signal	Description	I/O
WUU_Pn	External pin wake-up 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

16.5 Initialization

For an enabled module wake-up input, clear the module's flag before entering Power Down or Deep Power Down mode to avoid an immediate exit from the mode.

Clear the flags associated with external input pins, filtered and unfiltered, prior to entry to Power Down or Deep Power Down mode.

When an external wake-up pin filter is first enabled in Pin Filter (FILT), filter operation begins immediately. After enabling an external pin filter or changing its source pin, wait for at least five LPO clock cycles to allow the filter to initialize before entering Power Down or Deep Power Down mode or before enabling filter detection for all power modes in Pin Filter Mode Configuration (FMC).

NOTE

After recovering from a Deep Power Down mode, you 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.

16.6 Application information

To enable an external pin as a wake-up source in low-power modes, follow these steps:

1. If the flag is not 0, write 1 to PF[WUF*n*].
2. Configure the PDC*m*[WUPDC*n*] as either an interrupt or DMA request.
3. Configure the PE*m*[WUPE*n*] for rising edge, falling edge, or both edge detection.

When you have configured an external pin to be an active wake-up source in all power modes (PMC[WUPMC*n*] = 1), you must not change the corresponding wake-up pin enable (PE*m*[WUPE*n*]) and wake-up pin configuration (PDC*m*[WUPDC*n*]) settings for that pin. To modify the PE*m*[WUPE*n*] or PDC*m*[WUPDC*n*] settings, follow these steps:

1. Write 0 to PMC[WUPMC*n*] for that pin.
2. Update the corresponding PE*m*[WUPE*n*] or PDC*m*[WUPDC*n*] settings as needed.
3. Write 1 to PMC[WUPMC*n*] for that pin.

16.7 Memory map and register definition

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

16.7.1 WUU register descriptions

NOTE

You can write to WUU registers only in Supervisor mode. A bus error results from write accesses in User mode.

VSYS warm reset resets all WUU registers.

16.7.1.1 WUU memory map

WUU0 base address: 4009_2000h

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

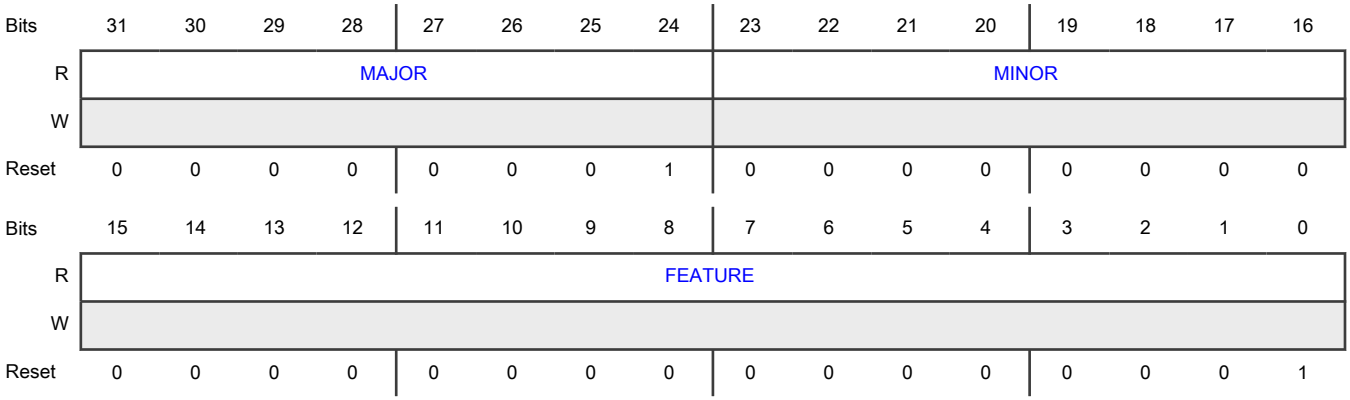
16.7.1.2 Version ID (VERID)

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 Specifies the major version number for the module specification.
23-16 MINOR	Minor Version Number Specifies the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number Specifies the feature set number. 0000_0000_0000_0000b - Standard features implemented 0000_0000_0000_0001b - Support for DMA/Trigger generation from wake-up pins and filters enabled. Support for external pin/filter detection during all power modes enabled. All other values are reserved.

16.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Contains parameter values 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 Indicates the number of pin wake-up sources supported.
23-16 MODULES	Module Number Indicates the number of module wake-up sources.
15-8 DMAS	DMA Number Indicates the number of DMA wake-up sources.
7-0 FILTERS	Filter Number Indicates the number of pin filters.

16.7.1.4 Pin Enable 1 (PE1)

Offset

Register	Offset
PE1	8h

Function

Enables and selects the edge detect type for the available external wake-up input pins in the range from WUU_P0 to WUU_P15.

NOTE

- Do not modify the value of PE1[WUPE*n*] when its corresponding PMC[WUPMC*n*] = 1.
- VSYS warm reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		WUPE12		WUPE11		WUPE10		WUPE9		WUPE8	
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	WUPE7		WUPE6		0		0		0		WUPE2		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 Reserved15	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
29-28 Reserved14	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Not supported 11b - Not supported
27-26 Reserved13	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
25-24 WUPE12	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
23-22 WUPE11	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-20 WUPE10	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
19-18 WUPE9	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
17-16 WUPE8	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
15-14	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows:

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUPE7	• For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
13-12 WUPE6	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
11-10 Reserved5	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
9-8 Reserved4	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, 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
	• For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
7-6 Reserved3	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
5-4 WUPE2	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
3-2 Reserved1	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Not supported 11b - Not supported
1-0 Reserved0	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported

16.7.1.5 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 wake-up input pins in the range from WUU_P16 to WUU_P31.

NOTE

- Do not modify the value of PE2[WUPE*n*] when its corresponding PMC[WUPMC*n*] = 1.
- VSYS warm reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WUPE31		WUPE30		WUPE29		WUPE28		WUPE27		WUPE26		WUPE25		WUPE24	
Reset	0	0	0	0	0	0	0	0	0	0	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	WUPE23		WUPE22		0		WUPE20		WUPE19		WUPE18		WUPE17		WUPE16	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 WUPE31	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
29-28 WUPE30	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
27-26 WUPE29	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
25-24	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUPE28	• For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
23-22 WUPE27	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
21-20 WUPE26	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
19-18 WUPE25	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
17-16 WUPE24	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
15-14 WUPE23	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
13-12 WUPE22	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
11-10 Reserved21	Reserved • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
9-8 WUPE20	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
7-6 WUPE19	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
5-4 WUPE18	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
3-2 WUPE17	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level. • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)
1-0 WUPE16	Wake-up Pin Enable for WUU_Pn Enables the external input pin for wake-up and configures the edge detection. • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level. • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge. 00b - Disable 01b - Enable (detect on rising edge or high level) 10b - Enable (detect on falling edge or low level) 11b - Enable (detect on any edge)

16.7.1.6 Module Interrupt Enable (ME)

Offset

Register	Offset
ME	18h

Function

Contains the bits to enable an on-chip module interrupt as a wake-up source.

NOTE

VSYS warm reset resets this register.

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	WUME8	0	WUME6	0	0	0	WUME2	0	WUME0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	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 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 WUME8	Module Interrupt Wake-up Enable for Module 8 Enables an on-chip module interrupt as a wake-up source input. 0b - Disable 1b - Enable
7 —	Reserved
6 WUME6	Module Interrupt Wake-up Enable for Module 6 Enables an on-chip module interrupt as a wake-up source input. 0b - Disable 1b - Enable
5 —	Reserved
4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 —	Reserved
2 WUME2	Module Interrupt Wake-up Enable for Module 2 Enables an on-chip module interrupt as a wake-up source input. 0b - Disable 1b - Enable
1 —	Reserved
0 WUME0	Module Interrupt Wake-up Enable for Module 0 Enables an on-chip module interrupt as a wake-up source input. 0b - Disable 1b - Enable

16.7.1.7 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 wake-up source.

NOTE

VSYS warm reset resets this register.

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	WUDE8	0	WUDE6	0	WUDE4	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 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 WUDE8	DMA/Trigger Wake-up Enable for Module 8 Enables an on-chip module DMA/trigger request as a wake-up source. 0b - Disable 1b - Enable
7 —	Reserved
6 WUDE6	DMA/Trigger Wake-up Enable for Module 6 Enables an on-chip module DMA/trigger request as a wake-up source. 0b - Disable 1b - Enable
5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 WUDE4	DMA/Trigger Wake-up Enable for Module 4 Enables an on-chip module DMA/trigger request as a wake-up source. 0b - Disable 1b - Enable
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

16.7.1.8 Pin Flag (PF)

Offset

Register	Offset
PF	20h

Function

Contains the wake-up flags indicating which wake-up source caused the chip to exit Power Down (if the corresponding PMC[WUPMC*n*] = 1) or Deep Power Down mode. For Power Down mode, this is the source causing the CPU interrupt flow. For Deep Power Down mode, this is the source causing the chip reset flow.

To clear a flag, write a 1 to the corresponding PF[WUF*n*]. If set, the wake-up flag (WUF*n*) remains set even if the associated pin wake-up is disabled in its PE*m*[WUPE*n*] field.

NOTE

VSYS warm reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WUF31	WUF30	WUF29	WUF28	WUF27	WUF26	WUF25	WUF24	WUF23	WUF22	0	WUF20	WUF19	WUF18	WUF17	WUF16
W	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	0	0	0	WUF12	WUF11	WUF10	WUF9	WUF8	WUF7	WUF6	0	0	0	WUF2	0	0
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	0

Fields

Field	Function
31 WUF31	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
30 WUF30	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
29 WUF29	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
28 WUF28	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
27 WUF27	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
26	Wake-up Flag for WUU_Pn

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUF26	Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
25 WUF25	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
24 WUF24	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
23 WUF23	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
22 WUF22	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
21 Reserved21	Reserved 0b - Not supported 1b - Not supported
20 WUF20	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
19 WUF19	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
18	Wake-up Flag for WUU_Pn

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUF18	Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
17 WUF17	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
16 WUF16	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
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 WUF12	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
11 WUF11	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
10 WUF10	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up 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
9 WUF9	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
8 WUF8	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
7 WUF7	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
6 WUF6	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
5 Reserved5	Reserved 0b - Not supported 1b - Not supported
4 Reserved4	Reserved 0b - Not supported 1b - Not supported
3 Reserved3	Reserved 0b - Not supported 1b - Not supported
2 WUF2	Wake-up Flag for WUU_Pn Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 Reserved1	Reserved 0b - Not supported 1b - Not supported
0 Reserved0	Reserved 0b - Not supported 1b - Not supported

16.7.1.9 Pin Filter (FILT)

Offset

Register	Offset
FILT	30h

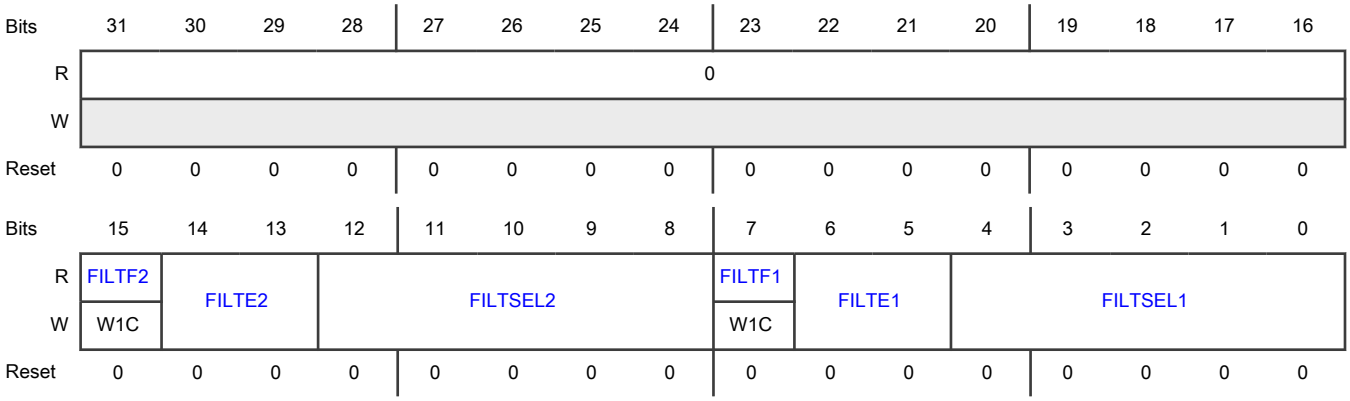
Function

Contains control and status fields to enable and configure the digital filter features for an external wake-up pin.

NOTE

VSYS warm reset resets this register.

Diagram



Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 FILTF2	Filter 2 Flag Indicates that the filtered external pin was a source of exiting a low-leakage power mode. 0b - No 1b - Yes
14-13 FILTE2	Filter 2 Enable Enables the filter for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge 00b - Disable 01b - Enable (Detect on rising edge or high level) 10b - Enable (Detect on falling edge or low level) 11b - Enable (Detect on any edge)
12-8 FILTSEL2	Filter 2 Pin Select Selects and filters the external pin WUU_Pn, 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. 0b - No 1b - Yes
6-5 FILTE1	Filter 1 Enable Enables the filter for wake-up. This field configures the edge detection as follows: • For field value = 01b, when configured as an interrupt/DMA request: Detect on rising edge. When configured as a trigger request: Detect on high level • For field value = 10b, when configured as an interrupt/DMA request: Detect on falling edge. When configured as a trigger request: Detect on low level • For field value = 11b, when configured as an interrupt/DMA request: Detect on any edge 00b - Disable 01b - Enable (Detect on rising edge or high level) 10b - Enable (Detect on falling edge or low level) 11b - Enable (Detect on any edge)
4-0	Filter 1 Pin Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
FILTSEL1	Selects and filters the external pin WUU_Pn, where *n* equals the value programmed into FILTSEL1.

16.7.1.10 Pin DMA/Trigger Configuration 1 (PDC1)

Offset

Register	Offset
PDC1	38h

Function

Configures the available external wake-up 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 Pin Enable 1 (PE1) is detected and remains asserted until the corresponding flag is cleared in Pin Flag (PF).
- When configured as a DMA request, the request is asserted when the edge programmed in Pin Enable 1 (PE1) is detected and remains asserted until either the corresponding flag is cleared in Pin Flag (PF) or until the requested DMA transfer is complete.
- When configured as a trigger request, the trigger is asserted when the level programmed in Pin Enable 1 (PE1) is detected and remains asserted when the input pin is asserted. The corresponding flag in Pin Flag (PF) will not be set.

NOTE

- Do not modify the value of PDC*m*[WUPDC*n*] field when its corresponding PMC[WUPMC*n*] = 1.
- VSYS warm reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		WUPDC12		WUPDC11		WUPDC10		WUPDC9		WUPDC8	
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	WUPDC7		WUPDC6		0		0		0		WUPDC2		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 Reserved15	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
29-28 Reserved14	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
27-26 Reserved13	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
25-24 WUPDC12	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
23-22 WUPDC11	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
21-20 WUPDC10	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up 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
19-18 WUPDC9	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
17-16 WUPDC8	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
15-14 WUPDC7	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
13-12 WUPDC6	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
11-10 Reserved5	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
9-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
Reserved4	00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
7-6 Reserved3	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
5-4 WUPDC2	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
3-2 Reserved1	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
1-0 Reserved0	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported

16.7.1.11 Pin DMA/Trigger Configuration 2 (PDC2)

Offset

Register	Offset
PDC2	3Ch

Function

Configures the available external wake-up 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 Pin Enable 2 (PE2) is detected and remains asserted until the corresponding flag is cleared in Pin Flag (PF).
- When configured as a DMA request, the request is asserted when the edge programmed in Pin Enable 2 (PE2) is detected and remains asserted until either the corresponding flag is cleared in Pin Flag (PF) or until the requested DMA transfer is complete.
- When configured as a trigger request, the trigger is asserted when the level programmed in Pin Enable 2 (PE2) is detected and remains asserted when the input pin is asserted. The corresponding flag in Pin Flag (PF) will not be set.

NOTE

- Do not modify the value of PDC*m*[WUPDC*n*] field when its corresponding PMC[WUPMC*n*] = 1.
- VSYS warm reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WUPDC31		WUPDC30		WUPDC29		WUPDC28		WUPDC27		WUPDC26		WUPDC25		WUPDC24	
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	WUPDC23		WUPDC22		0		WUPDC20		WUPDC19		WUPDC18		WUPDC17		WUPDC16	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 WUPDC31	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
29-28 WUPDC30	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up 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
27-26 WUPDC29	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
25-24 WUPDC28	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
23-22 WUPDC27	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
21-20 WUPDC26	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
19-18 WUPDC25	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up 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
17-16 WUPDC24	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
15-14 WUPDC23	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
13-12 WUPDC22	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
11-10 Reserved21	Reserved 00b - Not supported 01b - Not supported 10b - Not supported 11b - Not supported
9-8 WUPDC20	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
7-6 WUPDC19	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
5-4 WUPDC18	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
3-2 WUPDC17	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved
1-0 WUPDC16	Wake-up Pin Configuration for WUU_Pn Configures an external pin as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved

16.7.1.12 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 FILT*m*[FILTE*n*] is detected and remains asserted until the corresponding FILTF*n* flag is cleared.

- When configured as a DMA request, the request is asserted when the edge programmed in FILT*m*[FILTE*n*] is detected and remains asserted until either the corresponding FILTF*n* 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 Pin Filter (FILT) is detected and remains asserted when the input pin is asserted. The corresponding FILTF*n* flag will not be set.

NOTE

VSYS warm 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												FILTC2		FILTC1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-2: FILTC2 1-0: FILTC1	Filter Configuration for FILTn Configures a filter as an interrupt, DMA, or trigger wake-up source. 00b - Interrupt 01b - DMA request 10b - Trigger event 11b - Reserved

16.7.1.13 Pin Mode Configuration (PMC)

Offset

Register	Offset
PMC	50h

Function

Configures the detection logic for the available external wake-up input pins to remain enabled during all power modes, not just during Power Down/ mode.

NOTE

VSYS warm reset resets this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WUPMC31	WUPMC30	WUPMC29	WUPMC28	WUPMC27	WUPMC26	WUPMC25	WUPMC24	WUPMC23	WUPMC22	0	WUPMC20	WUPMC19	WUPMC18	WUPMC17	WUPMC16
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	WUPMC12	WUPMC11	WUPMC10	WUPMC9	WUPMC8	WUPMC7	WUPMC6	0	0	0	WUPMC2	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 WUPMC31	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
30 WUPMC30	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
29 WUPMC29	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
28	Wake-up Pin Mode Configuration for WUU_Pn

Table continues on the next page...

Table continued from the previous page...

Field	Function
WUPMC28	Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
27 WUPMC27	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
26 WUPMC26	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
25 WUPMC25	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
24 WUPMC24	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
23 WUPMC23	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 WUPMC22	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
21 Reserved21	Reserved 0b - Not supported 1b - Not supported
20 WUPMC20	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
19 WUPMC19	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
18 WUPMC18	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
17 WUPMC17	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
16 WUPMC16	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up 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. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
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 WUPMC12	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
11 WUPMC11	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
10 WUPMC10	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
9 WUPMC9	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up 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. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
8 WUPMC8	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
7 WUPMC7	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
6 WUPMC6	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up pin to provide active detection during all power modes. 0b - Active only during a low-leakage mode. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
5 Reserved5	Reserved 0b - Not supported 1b - Not supported
4 Reserved4	Reserved 0b - Not supported 1b - Not supported
3 Reserved3	Reserved 0b - Not supported 1b - Not supported
2 WUPMC2	Wake-up Pin Mode Configuration for WUU_Pn Configures an external wake-up 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. You can modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn). 1b - Active during all power modes. Do not modify the corresponding fields within Pin Enable (PEn) or Pin DMA/Trigger Configuration (PDCn).
1 Reserved1	Reserved 0b - Not supported 1b - Not supported
0 Reserved0	Reserved 0b - Not supported 1b - Not supported

16.7.1.14 Pin Filter Mode Configuration (FMC)

Offset

Register	Offset
FMC	58h

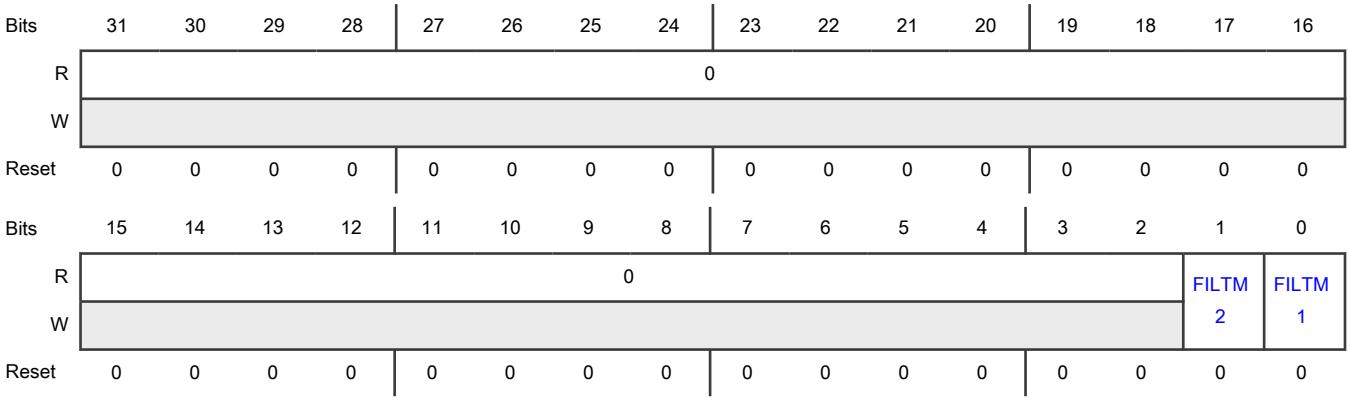
Function

Configures the detection logic for external pin filters to remain enabled during all power modes, not just during Power Down/Deep Power Down mode.

NOTE

VSYS warm reset resets this register.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 FILTMn	Filter Mode for FILTn Configures an external wake-up pin filter to provide active detection during all power modes. 0b - Active only during Power Down/Deep Power Down mode 1b - Active during all power modes

Chapter 17
AND/OR INVERT (AOI)

17.1 Chip-specific AOI information

Table 87. Reference links to related information

Topic	Related module	Reference
Full description	AOI	AOI
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

17.1.1 Module instances

This device has two instance of AOI (AOI0, AOI1).

17.2 Overview

AOI supports the generation of a configurable number of EVENT signals. Each output EVENT*n* is a configurable AOI function of four associated AOI inputs: A*n*, B*n*, C*n*, and D*n*, where *n* represents the channel number.

The AOI controller is a target peripheral module that connects event input indicators from a variety of modules and generates event output signals routed to other peripherals. You can access its programming model through the standard IPS target interface.

You can configure AOI to implement different Boolean functions.

17.2.1 Block diagram

AOI supports 4 event outputs, each representing a programmable, combinational Boolean function.

NOTE

The connections from AOI outputs to other functions are chip-specific.

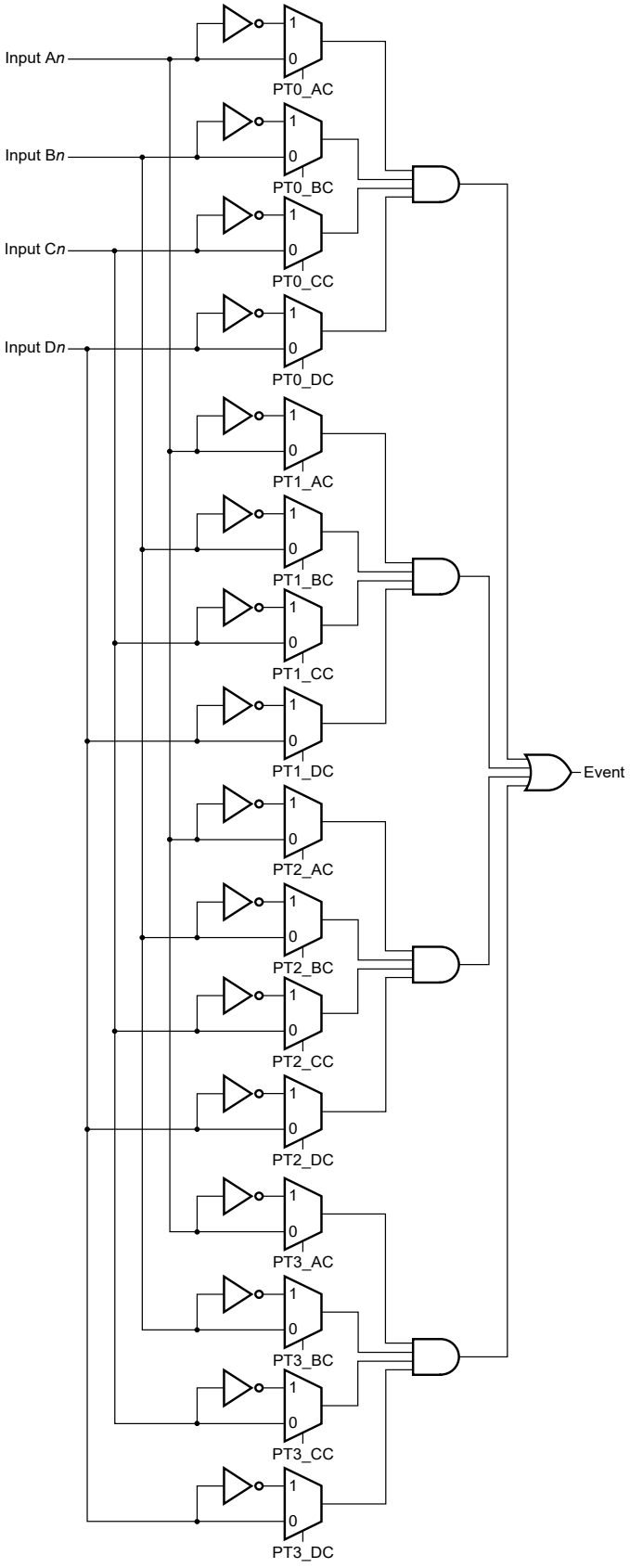


Figure 26. Block diagram

17.2.2 Features

- Enables you to create combinational Boolean events for use as hardware triggers:
  - Each channel includes four event inputs and one output
  - Evaluation of a combinational Boolean expression as the sum of four products, where each product term includes all four selected input sources available as true or complement values
  - Event output is formed as purely combinational logic and operates as a hardware trigger
- Includes a memory-mapped chip connected to the target peripheral (IPS) bus (programming model organized per channel for simplified software)

17.3 Functional description

AOI is highly programmable–you can use it for creating combinational Boolean outputs for use as hardware triggers. Each AOI output channel, as shown in Figure 26, has one logic function: Evaluation of combinational Boolean expression as a sum of four products, where each product term includes all four selected input sources available as true or complement values.

17.3.1 Modes of operation

AOI does not support any special modes of operation.

17.3.2 Clocks

The system uses the bus clock for register accesses, but AOI functions are asynchronous and do not use a clock.

17.3.3 Interrupts

This module has no interrupts.

17.4 External signals

This module has no external signals.

17.5 Initialization

This module does not require initialization.

17.6 Application information

The following sections describe configuration examples and Boolean expressions that you could use when programming AOI for your use.

17.6.1 Configuration examples for Boolean function evaluation

This section presents examples of the programming model configuration for simple Boolean expressions.

AOI provides a universal Boolean function generator using a four-term sum of product expression. Each product term contains true or complement values of the four selected event inputs (A*n*, B*n*, C*n*, and D*n*). See Code Listing 1 that presents a 4×4 Boolean expression defining the event output:

```
EVENTn
      = (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 0
      | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 1
```

```
      | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 2
      | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 3
```

Code Listing 1. Boolean expression defining event inputs (A*n*, B*n*, C*n*, and D*n*)

You can configure each selected input term in each product term to produce a logical 0 or 1, or pass the true or complement value of the selected event input. Each product term uses 8 bits of configuration information, 2 bits for each of the four selected event inputs. See Code Listing 2 that presents the actual Boolean expression implemented in each channel:

```
EVENTn
      = (PT0_AC[0] & An |  PT0_AC[1] & ~An)// product term 0
      & (PT0_BC[0] &  Bn | PT0_BC[1] & ~Bn)
      & (PT0_CC[0] &  Cn | PT0_CC[1] & ~Cn)
      & (PT0_DC[0] &  Dn | PT0_DC[1] & ~Dn)

      | (PT1_AC[0] & An |  PT1_AC[1] & ~An)// product term 1
      & (PT1_BC[0] &  Bn | PT1_BC[1] & ~Bn)
      & (PT1_CC[0] &  Cn | PT1_CC[1] & ~Cn)
      & (PT1_DC[0] &  Dn | PT1_DC[1] & ~Dn)

      | (PT2_AC[0] & An |  PT2_AC[1] & ~An)// product term 2
      & (PT2_BC[0] &  Bn | PT2_BC[1] & ~Bn)
      & (PT2_CC[0] &  Cn | PT2_CC[1] & ~Cn)
      & (PT2_DC[0] &  Dn | PT2_DC[1] & ~Dn)

      | (PT3_AC[0] & An |  PT3_AC[1] & ~An)// product term 3
      & (PT3_BC[0] &  Bn | PT3_BC[1] & ~Bn)
      & (PT3_CC[0] &  Cn | PT3_CC[1] & ~Cn)
      & (PT3_DC[0] &  Dn | PT3_DC[1] & ~Dn)
```

Code Listing 2. Boolean expressions for EVENT*n*

The combined fields of {Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013), Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)} correspond to the `PT{0-3}_{A,B,C,D}C[1:0]` terms in the aforementioned example.

Consider the settings of the combined 32-bit registers {Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013), Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)} for several simple Boolean expressions, as shown in Table 88.

Table 88. BFCRT*n* values for simple boolean expressions

Event output expression	PT0	PT1	PT2	PT3	{Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013), Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)}
A & B	A & B	0	0	0	01011111_00000000_00000000_00000000
A & B & C	A & B & C	0	0	0	01010111_00000000_00000000_00000000
(A & B & C) + D	A & B & C	D	0	0	01010111_11111101_00000000_00000000
A + B + C + D	A	B	C	D	01111111_11011111_11110111_11111101
(A & ~B) + (~A & B)	A & ~B	~A & B	0	0	01101111_10011111_00000000_00000000

As explained through the aforementioned examples, the resulting logic provides a simple yet powerful Boolean function evaluation for defining an event output.

17.7 AOI register descriptions

You can access the AOI programming model via a 16-bit peripheral bus connection. AOI is designed to support 16-bit accesses only. Other accesses are undefined.

AOI supports a specific number of event outputs. Each output EVENT*n* outputs a four-term AOI function of four binary inputs: A*n*, B*n*, C*n*, and D*n*. A pair of 16-bit registers configures this four-term AOI function: Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013) and Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233) define the configuration for the evaluation of the Boolean function defining EVENT*n*, where *n* is the event output channel number. Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013) defines the configuration of product terms 0 and 1, and Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233) defines the configuration of product terms 2 and 3.

AOI provides a universal Boolean function generator using a four-term sum of product expression with each product term containing true or complement values of the four selected event inputs (A*n*, B*n*, C*n*, and D*n*). Specifically, the following "4 x 4" Boolean expression defines the EVENT*n* output:

```
EVENTn
        term 0       = (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
        term 1     | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
        term 2     | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
        term 3     | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product
```

Code Listing 3. Boolean expression defining the EVENT*n* output

Using the aforementioned expression, you can configure each selected input in each product term to produce a logical 0 or 1 or pass the true or complement value of the selected event input. Each product term uses 8 bits of configuration information, 2 bits for each of the four selected event inputs. The resulting logic provides a simple yet powerful Boolean function evaluation for defining an event output.

These AOI functions are combinational in nature. You must sample and use them synchronously.

17.7.1 AOI memory map

AOI0 base address: 4008_9000h

AOI1 base address: 4009_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Boolean Function Term 0 and 1 Configuration for EVENT0 (BFCRT010)	16	RW	0000h
2h	Boolean Function Term 2 and 3 Configuration for EVENT0 (BFCRT230)	16	RW	0000h
4h	Boolean Function Term 0 and 1 Configuration for EVENT1 (BFCRT011)	16	RW	0000h
6h	Boolean Function Term 2 and 3 Configuration for EVENT1 (BFCRT231)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8h	Boolean Function Term 0 and 1 Configuration for EVENT2 (BFCRT012)	16	RW	0000h
Ah	Boolean Function Term 2 and 3 Configuration for EVENT2 (BFCRT232)	16	RW	0000h
Ch	Boolean Function Term 0 and 1 Configuration for EVENT3 (BFCRT013)	16	RW	0000h
Eh	Boolean Function Term 2 and 3 Configuration for EVENT3 (BFCRT233)	16	RW	0000h

17.7.2 Boolean Function Term 0 and 1 Configuration for EVENTn (BFCRT010 - BFCRT013)

Offset

Register	Offset
BFCRT010	0h
BFCRT011	4h
BFCRT012	8h
BFCRT013	Ch

Function

Configures the Boolean function for terms 0 and 1 of EVENT*n*.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PT0_AC		PT0_BC		PT0_CC		PT0_DC		PT1_AC		PT1_BC		PT1_CC		PT1_DC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 PT0_AC	Product Term 0, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 0. 00b - Force input A to become 0 01b - Pass input A

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Complement input A 11b - Force input A to become 1
13-12 PT0_BC	Product Term 0, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 0. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
11-10 PT0_CC	Product Term 0, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 0. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
9-8 PT0_DC	Product Term 0, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 0. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1
7-6 PT1_AC	Product Term 1, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 1. 00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input A to become 1
5-4 PT1_BC	Product Term 1, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 1. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 PT1_CC	Product Term 1, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 1. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
1-0 PT1_DC	Product Term 1, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 1. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1

17.7.3 Boolean Function Term 2 and 3 Configuration for EVENTn (BFCRT230 - BFCRT233)

Offset

Register	Offset
BFCRT230	2h
BFCRT231	6h
BFCRT232	Ah
BFCRT233	Eh

Function
Configures the Boolean function for terms 2 and 3 of EVENT*n*.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PT2_AC		PT2_BC		PT2_CC		PT2_DC		PT3_AC		PT3_BC		PT3_CC		PT3_DC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 PT2_AC	Product Term 2, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 2. 00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input A to become 1
13-12 PT2_BC	Product Term 2, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 2. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
11-10 PT2_CC	Product Term 2, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 2. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
9-8 PT2_DC	Product Term 2, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 2. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1
7-6 PT3_AC	Product Term 3, Input A Configuration Defines the Boolean evaluation associated with the selected input A in product term 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Force input A to become 0 01b - Pass input A 10b - Complement input A 11b - Force input to become 1
5-4 PT3_BC	Product Term 3, Input B Configuration Defines the Boolean evaluation associated with the selected input B in product term 3. 00b - Force input B to become 0 01b - Pass input B 10b - Complement input B 11b - Force input B to become 1
3-2 PT3_CC	Product Term 3, Input C Configuration Defines the Boolean evaluation associated with the selected input C in product term 3. 00b - Force input C to become 0 01b - Pass input C 10b - Complement input C 11b - Force input C to become 1
1-0 PT3_DC	Product Term 3, Input D Configuration Defines the Boolean evaluation associated with the selected input D in product term 3. 00b - Force input D to become 0 01b - Pass input D 10b - Complement input D 11b - Force input D to become 1

Chapter 18
Core Mode Controller (CMC)

18.1 Chip-specific CMC information

Table 89. Reference links to related information

Topic	Related module	Reference
Full description	CMC	CMC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Reset		Reset
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

18.1.1 Module instances

This device has one instance of CMC module.

18.2 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.

18.2.1 Block diagram

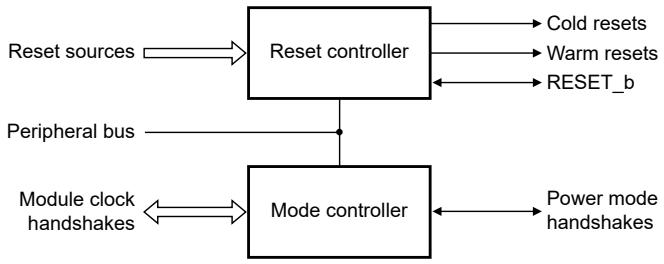


Figure 27. Block diagram

18.2.2 Features

- Configures the low-power mode options including SRAM retention
- Provides reset controller including reset status register and warm reset configuration
- Provides boot mode register and status bits

18.3 Functional description

This section provides the functional description of the block.

18.3.1 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 90. 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.
Sleep	WFI WFE SLEEPONEXIT	NVIC Reset	• Core waiting for interrupt or event, peripherals optionally active • Core and peripherals retain state. SRAM is active or retained.
Deep Sleep	WFI WFE SLEEPONEXIT	NVIC Reset	• Core waiting for interrupt or event, peripherals optionally active • Core and peripherals retain state, SRAM is retained
Power Down	WFI WFE SLEEPONEXIT	WUU via Active Reset	• Core and peripherals waiting for wakeup • Core and peripherals retain state • SRAM is optionally retained
Deep Power Down	WFI WFE SLEEPONEXIT	WUU via Reset Reset	Core, peripherals, and SRAM not retained.

18.3.2 Active mode

This section describes power options in Active mode.

18.3.2.1 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 read access to flash memory space (but not flash peripheral 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.

18.3.3 Low-power modes

This section describes the low-power modes.

18.3.3.1 Low-power entry

The following steps occur during the Low-Power mode entry sequence.

Table 91. 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 (Deep Sleep). • System clocks are gated.
8	Fh	3h	Power domain is placed in Low-Power retention state (Power Down).
9	Fh	7h	Power domain is switched off.
10	Fh	Fh	Regulators are powered off (Deep Power-Down).

18.3.3.2 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 in 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.

18.3.3.3 Power domains

CMC configures Low-Power mode of 1 power domain when the core enters a low-power mode with CKMODE = Fh. You can configure the different power domains to enter different low-power modes .

NOTE

The WAKE domain LPMODE register setting must always be less than or equal to the LPMODE register setting for all other power domains.

The power domain controlled by the CMC is MAIN.

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.

18.3.3.4 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.

18.3.4 Clocks

- The slow bus clock clocks CMC registers and logic.
- clk_1M clock clocks the timeout counters.

18.3.5 Reset

This section describes the sources of reset and the different resets that you can generate.

18.3.5.1 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[VD] become 1 on a POR condition.

Voltage detect (VD)

The voltage detect monitors enable by default and keeps the MCU in reset until the supply voltage rises above the LVD rising threshold. Whenever the voltage detect monitors are enabled, they trigger a reset condition if the supply voltage is outside the voltage detect trip point.

SRS[VD] becomes 1 on voltage detect condition. The voltage detect monitors assert a cold reset in all power domains.

Wakeup (WAKEUP)

On a wakeup from 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 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 65536 cycles of the clk_1M 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 65536 cycles of the clk_1M 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.

Windowed watchdog timer 0 (WWDT0)

The windowed watchdog timer monitors the software by expecting periodic refreshing of the watchdog counter. When this does not occur, it triggers the WWDT0 reset condition.

The WWDT0 reset condition asserts a warm reset in all power domains. SRS[WWDT0] and SRS[WARM] become 1 on a WWDT0 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.

Code watchdog (CDOG)

CDOG module helps protect the integrity of software by detecting unexpected changes (faults) in the code execution flow. CDOG module configures to reset or interrupt the processor core when the module detects a fault.

The CDOG0 reset condition asserts a warm reset in all power domains. SRS[CDOG0] and SRS[WARM] become 1 on a CDOG0 reset condition.

JTAG reset (JTAG)

When a JTAG instruction places the chip in reset, it triggers the JTAG reset condition.

The JTAG reset condition asserts a warm reset in all power domains. SRS[JTAG]SRS[JTAG], SRS[FATAL]SRS[FATAL], and SRS[WARM] become 1 on a JTAG reset condition.

18.3.5.2 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 Deep-Power Down (varies per domain)

A cold reset does not guarantee the contents of on-chip SRAM.

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 65536 cycles of the clk_1M 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.

18.3.5.3 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.

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.

18.3.6 Interrupts

CMC generates a single interrupt, which System Reset Interrupt Enable (SRIE) configures.

18.4 External signals

Table 92. 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
ISPMODE_n	The input pin is sampled at the end of the RESET_B assertion and stored in Mode (MR0).	Input

18.5 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].

18.6 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 2h to Power Mode Protection (PMPROT)
3. Write 3h to Global Power Mode Control (GPMCTRL)
4. Execute WFI instruction

To configure for Sleep mode entry:

1. Write 0h or 1h to Clock Control (CKCTRL)
2. Execute WFI instruction

To configure for Deep Sleep Low-Power mode entry:

1. Write Fh to Clock Control (CKCTRL)
2. Write 1h to Power Mode Protection (PMPROT)
3. Write 1h to Global Power Mode Control (GPMCTRL)
4. Execute WFI instruction

18.7 Memory map and register descriptions

This section describes the registers in the CMC module.

18.7.1 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.

18.7.1.1 CMC memory map

CMC base address: 4008_B000h

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
80h	System Reset Status (SRS)	32	R	See section
84h	Reset Pin Control (RPC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
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
E0h	Flash Control (FLASHCR)	32	RW	0000_0000h
110h	Core Control (CORECTL)	32	RW	0000_0000h
120h	Debug Control (DBGCTL)	32	RW	0000_0000h

18.7.1.2 Version ID (VERID)

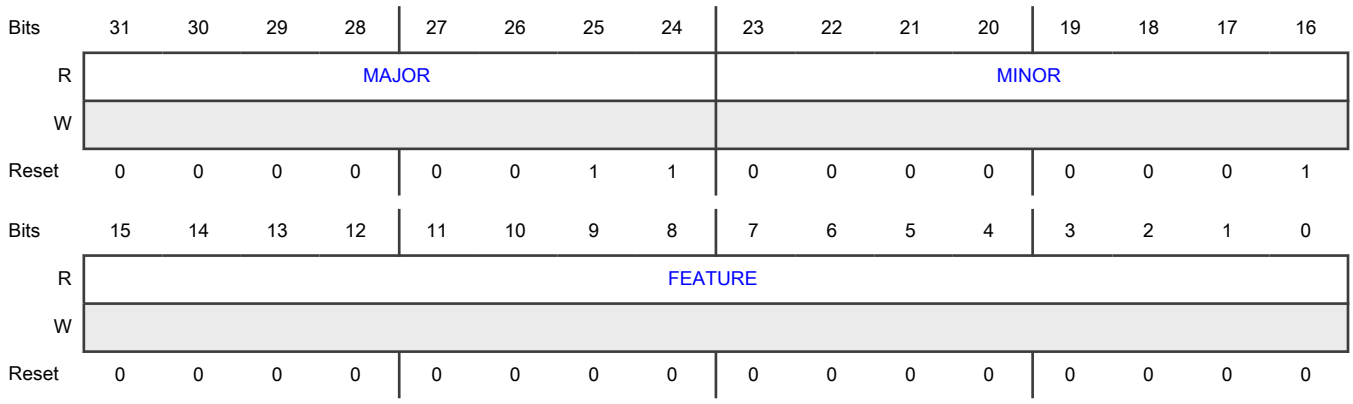
Offset

Register	Offset
VERID	0h

Function

Contains version numbers for 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.

18.7.1.3 Clock Control (CKCTRL)

Offset

Register	Offset
CKCTRL	10h

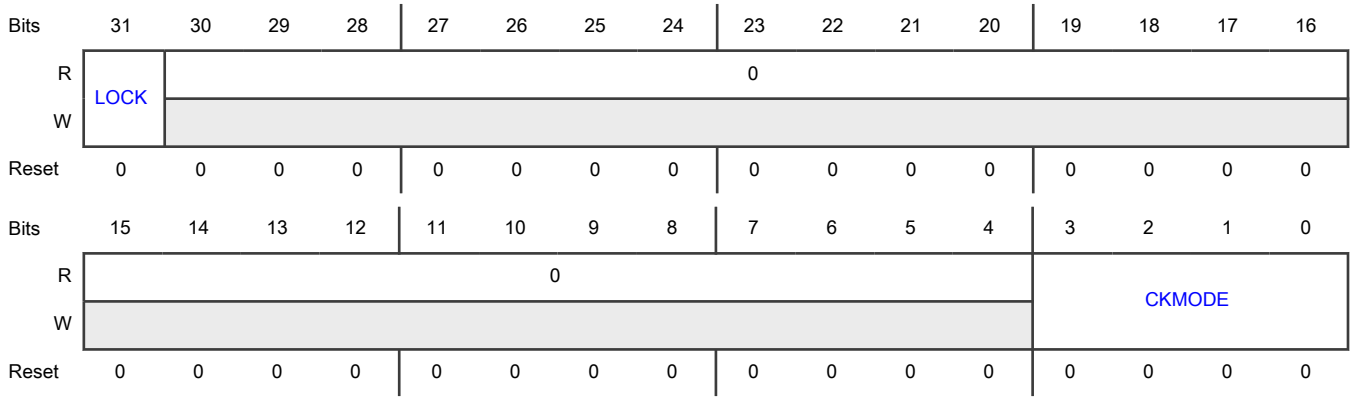
Function

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 1111b - Core, platform, and peripheral clocks are off, and core enters Low-Power mode

18.7.1.4 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	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
30-16 —	Reserved
15-8 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 reserved [6] - Wake-up source is reserved [7] - Wake-up source is reserved A wakeup from Deep Power-Down writes 1 to WAKEUP[0]. Depending on Low-Power mode, other wake-up bits may not become 1.
7-4 —	Reserved
3-0 CKMODE	Low Power Status Returns the result of the previous Low-Power mode entry. 0000b - Core clock is on 0001b - Core clock is off 1111b - Core, platform, and peripheral clocks are off, and core enters Low-Power mode All other values are reserved.

18.7.1.5 Power Mode Protection (PMPROT)

Offset

Register	Offset
PMPROT	18h

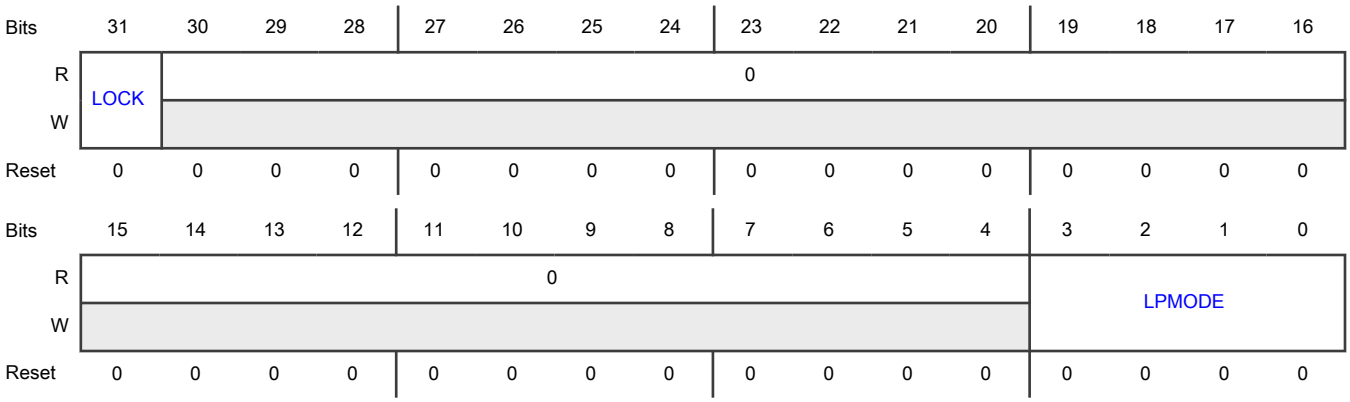
Function

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



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 Deep Sleep. Bit 1: 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Allowed 1000b - Allowed 1001b - Allowed 1010b - Allowed 1011b - Allowed 1100b - Allowed 1101b - Allowed 1110b - Allowed 1111b - Allowed

18.7.1.6 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.

18.7.1.7 Power Mode Control (PMCTRLMAIN)

Offset

Register	Offset
PMCTRLMAIN	20h

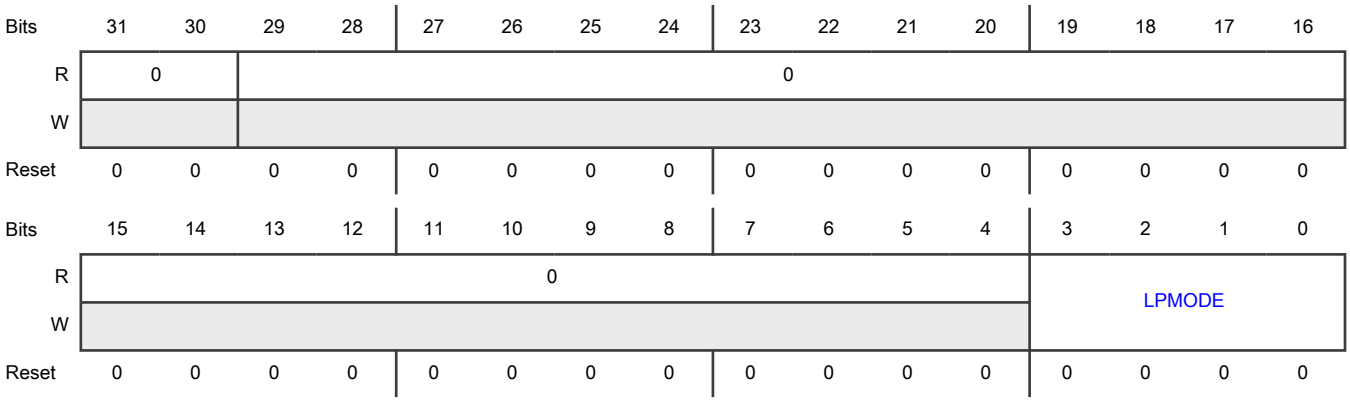
Function

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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/Sleep 0001b - Deep Sleep 0011b - Power Down 0111b - Reserved 1111b - Deep-Power Down

18.7.1.8 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	0	0	JTAG	0	CDOG 0	0	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	LOCK UP	SW	WWD T0	SCG	LPAC K	RSTA CK	DAP	PIN	0		FATAL	WARM	0	VD	POR	WAKE UP
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 JTAG	JTAG System Reset Indicates a JTAG system reset request that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated 1b - Reset generated
27 —	Reserved
26 CDOG0	Code Watchdog 0 Reset Indicates the CDOG0 fault that causes a reset. 0b - Reset is not generated 1b - Reset is generated
25 —	Reserved
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	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 WWDT0	Windowed Watchdog 0 Reset Indicates the WWDT 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 65536 cycles of clk_1M clock. 0b - Reset not generated 1b - Reset generated
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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 —	Reserved
2 VD	Voltage Detect Reset Indicates that an enabled voltage detect monitor causes a reset. 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 Deep-Power Down mode that causes a reset. 0b - Reset not generated 1b - Reset generated

18.7.1.9 Reset Pin Control (RPC)

Offset

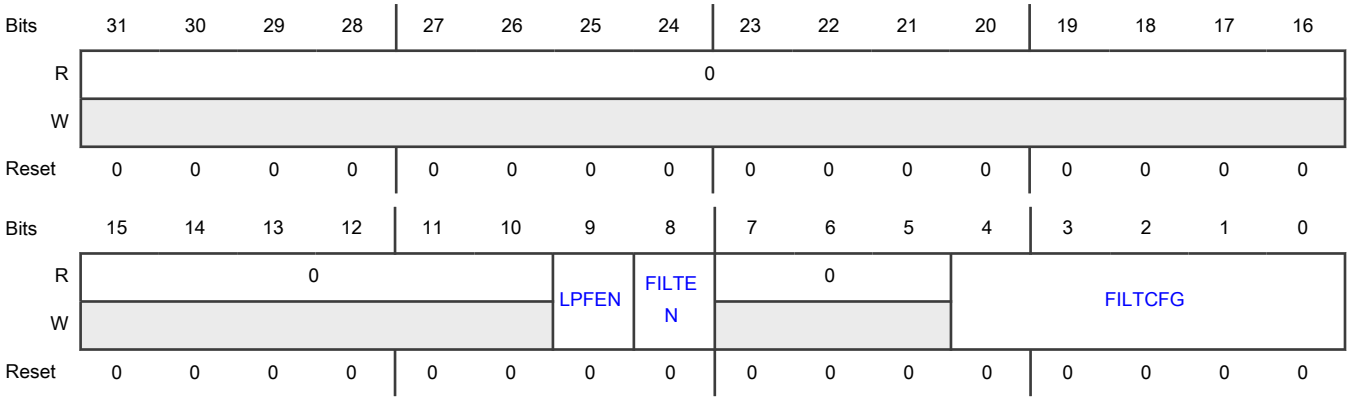
Register	Offset
RPC	84h

Function

NOTE

WAKE cold reset resets this register.

Diagram



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 clk_1M clock cycles to always propagate through the filter. A glitch less than two clk_1M 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

18.7.1.10 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	0	0	JTAG	0	CDOG0	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	LOCKUP	SW	WWDT0	SCG	LPACK	RSTACK	DAP	PIN	0		FATAL	WARM	0	VD	POR	WAKEUP
W	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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
30 —	Reserved
29 —	Reserved
28 JTAG	JTAG System Reset Indicates a JTAG system reset that causes a reset. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated 1b - Reset generated
27 —	Reserved
26 CDOG0	Code Watchdog 0 Reset Indicates a CDOG0 fault that causes a reset. 0b - Reset is not generated 1b - Reset is generated
25 —	Reserved
24 —	Reserved
23 —	Reserved
22-18 —	Reserved
17 —	Reserved
16 —	Reserved
15	Lockup Reset Indicates the core lockup that causes a reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
LOCKUP	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 WWDT0	Windowed Watchdog 0 Reset Indicates a windowed watchDog 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
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 —	Reserved
2 VD	Voltage Detect Reset Indicates that an enabled voltage detect monitor causes a reset. 0b - Reset not generated 1b - Reset generated
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 Deep-Power Down mode that causes a reset. 0b - Reset not generated 1b - Reset generated

18.7.1.11 System Reset Interrupt Enable (SRIE)

Offset

Register	Offset
SRIE	8Ch

Function

Delays the assertion of a system reset for 65538 cycles of the clk_1M 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 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	CDOG0	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	LOCKUP	SW	WWDT0	SCG	LPACK	0	DAP	PIN	0							
W																
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 CDOG0	Code Watchdog 0 Reset 0b - Interrupt disabled 1b - Interrupt enabled
25 —	Reserved
24 —	Reserved
23 —	Reserved
22-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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 WWDT0	Windowed Watchdog 0 Reset 0b - Interrupt disabled 1b - Interrupt enabled
12 SCG	System Clock Generation Reset 0b - Interrupt disabled 1b - Interrupt enabled
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

18.7.1.12 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

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0	CDOG 0	0	0	0	0					0	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	LOCK UP	SW	WWDT 0	0	LPAC K	0	DAP	PIN	0							
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
31-28 —	Reserved
27 —	Reserved
26	Code Watchdog 0 Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
CDOG0	0b - Reset source not pending 1b - Reset source pending
25 —	Reserved
24 —	Reserved
23 —	Reserved
22-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 LOCKUP	Lockup Reset 0b - Reset source not pending 1b - Reset source pending
14 SW	Software Reset 0b - Reset source not pending 1b - Reset source pending
13 WWDT0	Windowed 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
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

18.7.1.13 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.

18.7.1.14 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

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															ISPMODE...
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 ISPMODE_n	In System Programming Mode Returns the logic state of the ISPMODE_n pin on the last negation of the RESET_b pin.

18.7.1.15 Force Mode (FM0)

Offset

Register	Offset
FM0	B0h

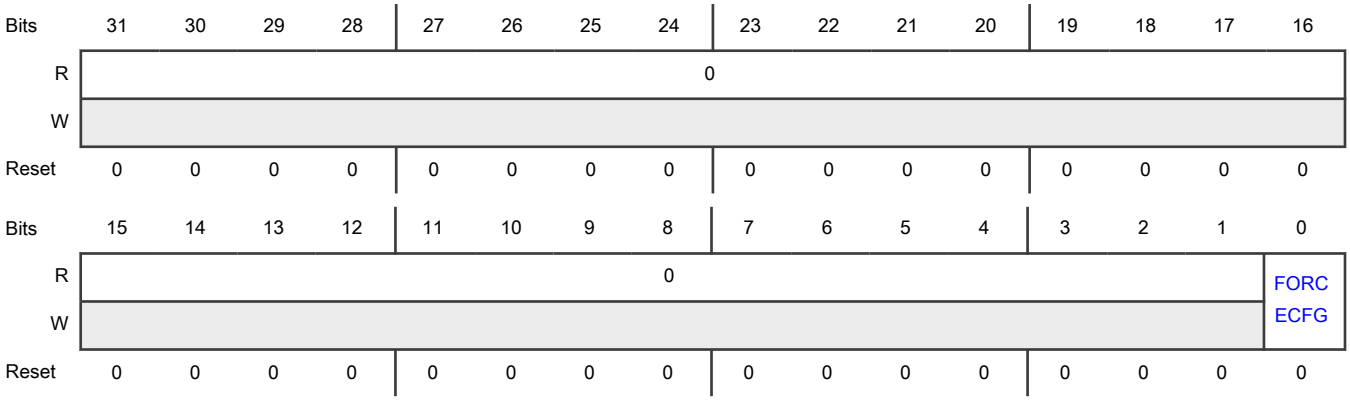
Function

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

18.7.1.16 Flash Control (FLASHCR)

Offset

Register	Offset
FLASHCR	E0h

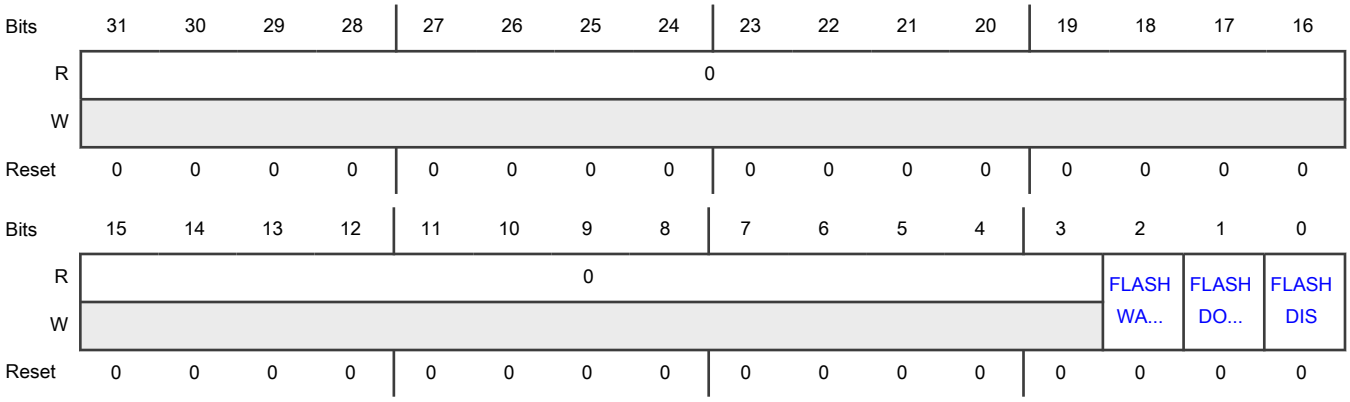
Function

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

18.7.1.17 Core Control (CORECTL)

Offset

Register	Offset
CORECTL	110h

Function

Configures options for the core.

NOTE

This register resets on MAIN warm 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															NPIE
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 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

18.7.1.18 Debug Control (DBGCTL)

Offset

Register	Offset
DBGCTL	120h

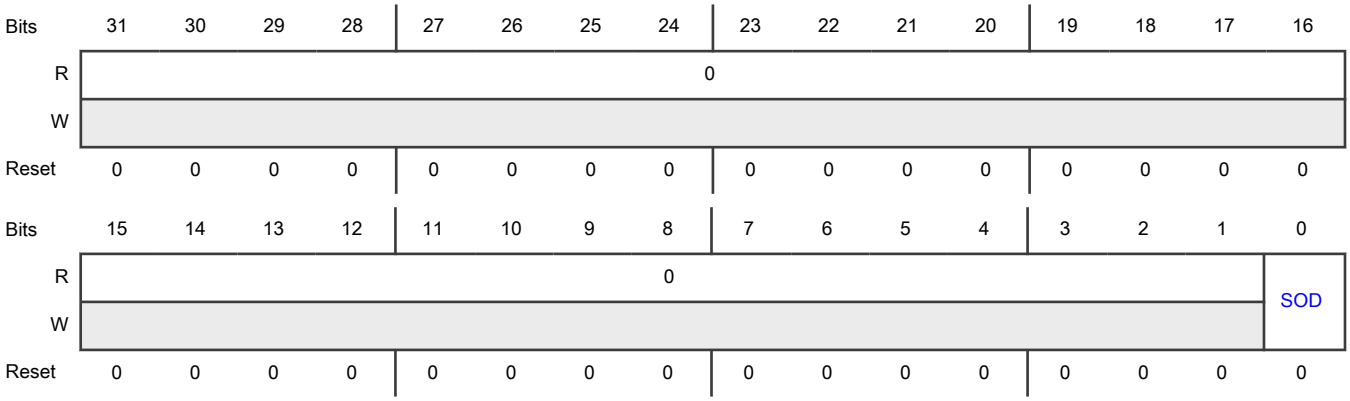
Function

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 19
Reset

19.1 Reset sources

The following reset sources exist in this chip:

Table 93. Reset sources

Reset source	Description
Power	System Power-On Reset (POR)
	System Low-Voltage Detect (LVD)
	System High-Voltage Detect (HVD)
External	Wake Up Unit (WUU)
	External pin reset (RESET_b)
Internal	• Code Watchdog reset • Watchdog (WDOG) reset • Software (SW) reset • System Clock Generator (SCG-lite) Loss of Clock • Reset Acknowledge timeout (RSTACK) • Low Power Acknowledge (LPACK) • Core Lockup (LOCKUP) • Debug Mailbox • Joint Test Action Group (JTAG)

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.

19.2 Power reset sources

To ensure predictable operation of the chip, the supply voltages must remain in their specified ranges during execution. Because this is not possible during power-up and power-down, the chip 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.

19.2.1 POR Reset

The system domain implements a “POR Only Reset (POR)” which only asserts on VDD_SYS analog POR. This reset source is only used for resetting the registers used by the LVD/HVD detectors. 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 chip or when the supply voltage drops below the POR falling threshold, the POR monitor triggers the POR reset condition to the chip.

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. See Block diagram.

19.2.2 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 using those power supplies 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 monitor is 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 chip 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, and Deep Sleep modes and are disabled in Deep Power Down modes. See Module operation in low power modes for the available modes.

The individual LVD monitor resets are combined into a single LVD reset trigger to the Core Mode Controller (CMC). The LVD reset condition asserts a reset to all power domains with only the POR/LVD/HVD detection logic remaining operational. A LVD reset will set the appropriate bits in the CMC registers so that software can determine that a LVD reset has occurred.

19.2.3 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 monitor is disabled by default, but when the HVD circuit is enabled, the HVD monitor will generate a reset when the voltage rises above the VHVDL threshold. The voltage threshold at which the HVD 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 HVD monitors are available in the Active, Sleep, and Deep Sleep modes. The HVD monitors are disabled in Deep Power Down modes.

The individual HVD monitor resets are combined into a single HVD reset trigger to the Core Mode Controller (CMC). The HVD reset condition asserts a reset to all power domains with only the POR/LVD/HVD detection logic remaining operational. A HVD reset is combined with LVD reset, and will set the appropriate bits in the CMC registers so that software can determine that a HVD reset has occurred.

19.3 External reset sources

External resets are provided so that the chip can be reset or wake up from very low power states. This allows the chip to start at the correct time from a known state.

19.3.1 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 chip into Pin reset condition.
- During reset, the RESET_b drives low until the chip has completed hardware initialization, at which point the RESET_b pin is released. If the RESET_b pin is asserted externally, then the chip 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 when the CMC clock is disabled.

The RESET_b is multiplexed on P1_29 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.

19.3.2 Wake-up (WAKEUP)

The Wake-Up Unit (WUU) provides for a number of external pins and on-chip peripherals to wake the chip from Deep Power Down mode. See the WUU chapter for a list of external and internal reset sources connected to the WUU.

On a wakeup from Deep Power Down mode, 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.

19.4 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.

19.4.1 Code Watchdog reset

The Code Watchdog Timer (CDOG) module helps protect the integrity of software by detecting unexpected changes (faults) in code execution flow. The CDOG module can be configured to reset or interrupt the processor core when the module detects a fault.

The CDOG reset condition triggers the Core Mode Controller (CMC) to assert a WARM reset to all power domains. A CDOG reset sets the appropriate bits in the CMC registers so that software can determine that a WDOG reset has occurred.

19.4.2 Windowed Watchdog timer (WDOG) reset

The WDOG reset is generated by the WWDT module on the chip.

The WWDT monitors the operation of the system by receiving periodic communication from the software. This communication is generally 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.

19.4.3 Software (SW) reset

During execution, if the software detects erroneous operation, the software can initiate a reset to restart the chip from a known state.

The SW reset condition, which is generated by SYSRESETREQ, 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. For details, refer System Reset Status Register (SRS) in the CMC chapter.

19.4.4 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 asserts a WARM reset in all power domains. The CMC_SRS[SCG] and CMC_SRS[WARM] status bits are set on a SCG reset condition.

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.

19.4.5 Reset Acknowledge timeout (RSTACK)

The reset state machine includes a timeout counter that is triggered by the reset state machine not progressing within 65536 cycles of the clk_1m clock, which triggers 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.

19.4.6 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 65536 cycles of the clk_1m clock; this triggers 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.

19.4.7 Lock up (LOCKUP)

The Cortex-M33 core enters the lockup state as a result of certain illegal operations, which triggers 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.

19.4.8 Debug Mailbox (DM)

Debug Mailbox (ISP-AP) can initiate a reset request from a connected debugger. This triggers the debug mailbox reset condition.

The DM reset condition triggers the CMC to assert a WARM reset to all power domains. A DM reset sets the appropriate fields in the CMC registers so that software can determine that a DM reset has occurred.

19.4.9 JTAG

Whenever a JTAG instruction places the chip in reset, it triggers the JTAG reset condition.

The JTAG reset condition triggers the CMC to assert a WARM reset to all power domains. A JTAG reset sets the appropriate bits in the CMC registers so that software can determine that a JTAG reset has occurred.

19.5 Reset type

This section describes the different resets that can be generated on this chip and their respective reset sequences. Reset conditions on this chip are categorized into two types, Cold and Warm resets.

19.5.1 Cold reset

A cold reset is a reset that only asserts when a power domain is powered up (Power On Reset, Power Down wakeup, etc) or one of the supply voltages is not valid (LVD, HVD, supply glitch detect). For simplicity, all LVD/HVD assert the Cold Reset for all power domains. Each power domain has its own "Cold Reset" which is used by any logic in that domain that needs a reset that does not assert on a warm reset source. The system domain also implements a "POR Only Reset" which only asserts on VDD_SYS analog POR. This reset source is only used for resetting the registers used by the LVD/HVD detectors. A POR does not guarantee the contents of on-chip SRAM, except for a Deep Power Down wakeup and then only for the SRAM that is configured to retain state.

19.5.1.1 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 32 CM33 clock cycles.
4. Internal reset to Flash and their initialization sequences commence.
5. Initialization sequences for Flash and IFR 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 chip wakes up from Deep Power Down, the reset sequence is slightly different from the POR reset sequence. They are described in the subsequent sections.

19.5.1.2 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 32 clock cycles.
4. Internal reset to Flash and their initialization sequences commence.
5. Initialization sequences for Flash and IFR 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.

19.5.2 Warm reset

A warm reset is a reset that asserts on either:

- Cold Reset for that power down
- Any of the warm reset sources

Any of the warm reset sources will trigger a warm reset for every power domain. An “early” warm reset is generated for some of the power domain, this reset asserts at the same time as the normal reset but negates early to allow processes that need to operate during reset to complete. This includes the MSF loading of IFR trim data.

A WARM reset re-initializes the chip 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 following table to know which CMC reset sources are fatal/non-fatal. Also see CMC chapter for details..

CMC reset source	Fatal Reset
Reset pin	N
Debug AP (debug mailbox)	N
Reset Monitor	Y
Low Power Monitor	N
Loss of Clock / Lock	N
WDOG0 reset	N
CM33 reset request	N
CM33 lockup	N
Code WDG0 reset	N
JTAG boundary scan reset	Y

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 64 ms 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 32 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.

19.5.3 Peripheral SW reset

Peripherals can be reset by SW. The chip implements Peripheral reset control registers in the System Controller (SYSCON) chapter.

Chapter 20
Boot ROM

20.1 System Boot ROM

20.2 Boot ROM

This architecture specification defines the structure and flow of the system boot ROM and its interaction with the extended bootloader on IFR0.

20.2.1 Overview

ROM on this chip only supports on-chip FLASH image boot, no external memory boot.

On this device, there are two software, one software is boot ROM, located on ROM memory, second software is called extended bootloader, which located on IFR0, sector 1 ~ 3. List on table 1.

Table 94. On chip software

Term	Definition
Boot ROM (software)	In the context of this document, “Boot ROM” is the piece of software present in ROM memory implementing features explained throughout this document.
Extended Bootloader (Software)	In the context of this document, “Extended Bootloader” is the piece of software present in IFR0 memory implementing features explained throughout this document. IFR0 sector 1 ~ 3 (24 KB)

ROM working with extended bootloader takes responsibility for boot flow.

Boot related parameters are in Customer Manufacturing/Factory Programming Area (CMPA), and ROM will use some settings on this field to update the booting options.

20.2.2 Features

The main features of the Boot ROM (16 KB) and extended bootloader (24 KB) include:

- Check lifecycle, set Read Out Protection (ROP).
- Enable/disable debug port per lifecycle.
- Set ROM hidden per lifecycle.
- Handling debug mailbox commands, but available commands sets are per lifecycle.
- Handling ISP (In-System Programming) commands, but available commands sets are per lifecycle.
- Provide Flash API
- Support Flash swap
- Only support ISP boot, not support master boot

20.2.3 Functional description

20.2.3.1 Software block diagram

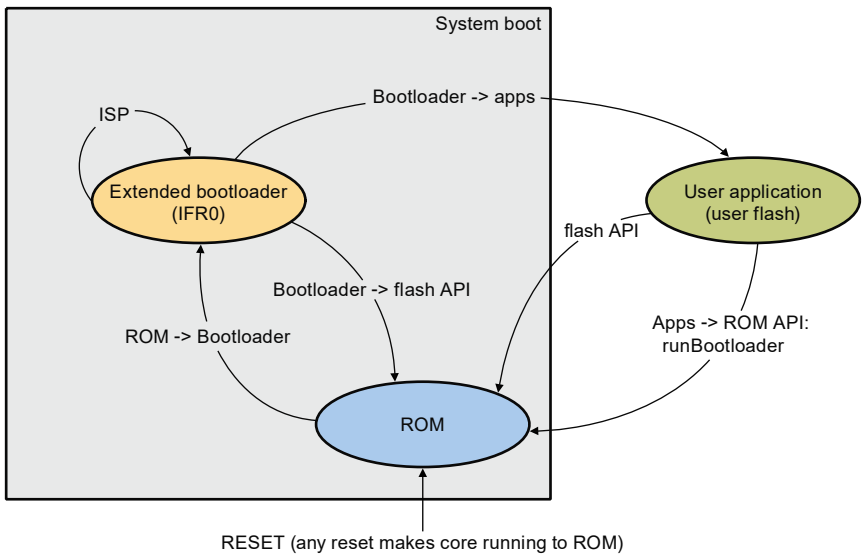


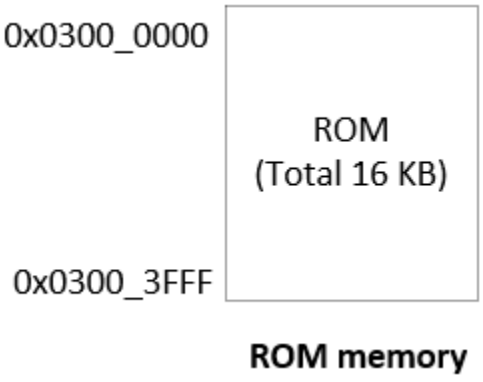
Figure 28. Software flow figure

20.2.3.2 Boot ROM

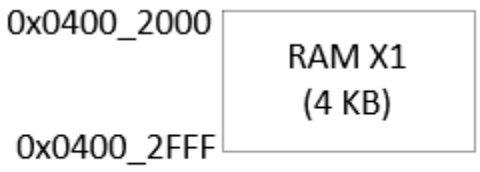
After any reset, including POR reset and warm reset sequence, CPU always runs to boot ROM. The main functions of boot ROM are:

- Check lifecycle,
- Process mailbox request through debug access port
- Configure the GLIKEY and MBC
- Run image integrity check based on wakeup source
- Before exit to extended bootloader (on IFR0), hidden critical sections
- ROM provide flash driver API to user

ROM memory starts at 0x0300_0000 with a size of 16KB.



ROM memory



RAM memory used by ROM and Extended bootlader

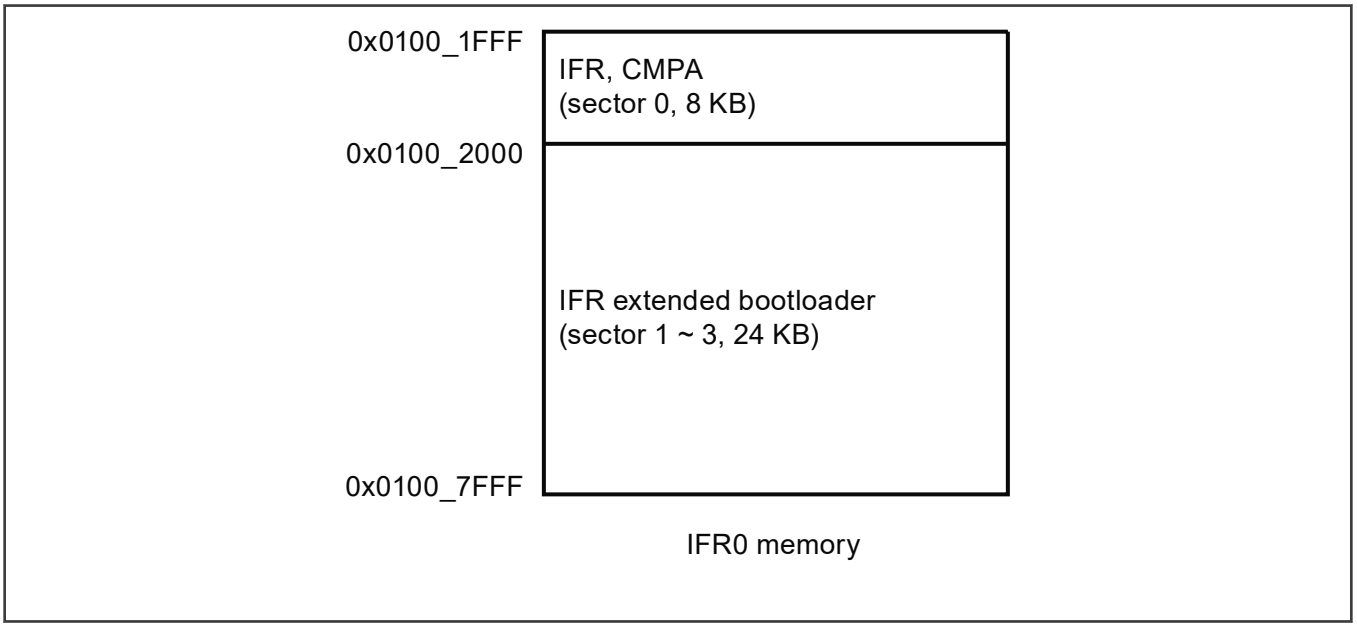
20.2.3.3 Extended bootloader

Extended bootloader is the secondary bootloader, which is located in IFR0, sector 1 ~ 3, total 24KB. It checks configuration, enabling the booting interfaces, perfumes ISP command following NXP bootloader protocol and others tasks. Please see “Extended Bootloader Specifications for more details

Due to limit IFR0 memory size, there are two extended bootloader releases to be programmed to IFR0.

- Supports USB and LPUART booting peripherals

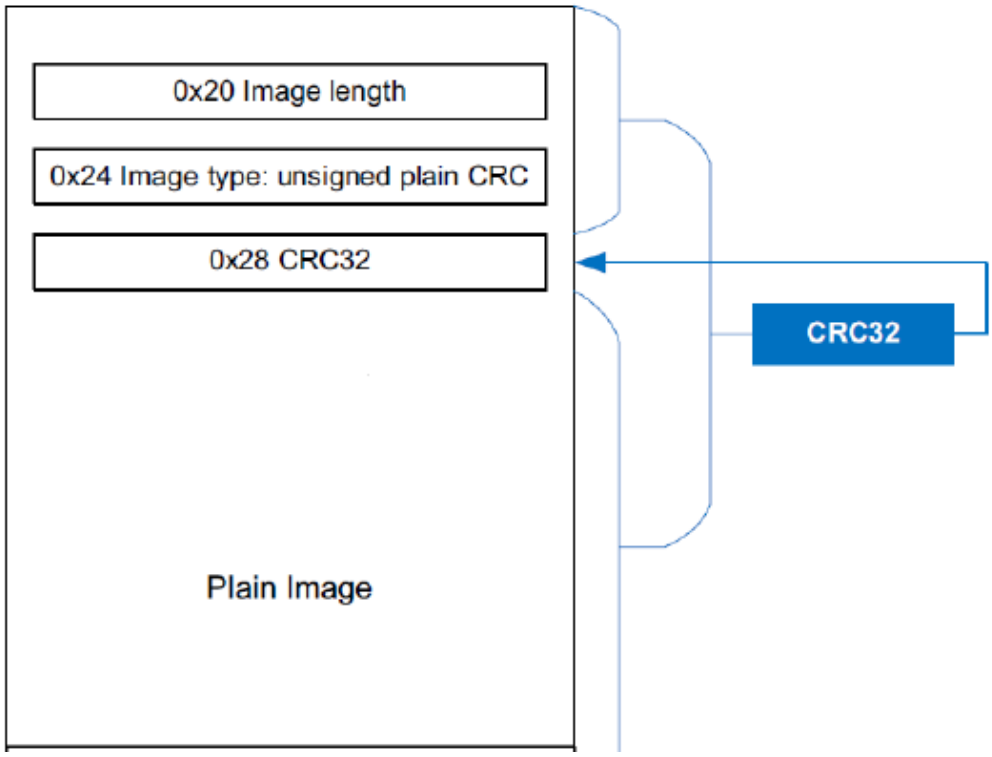
NXP programs one of them to IFR0, as per the required configuration, or customer request.



20.2.3.4 User Application

The user application is located in the main flash region. It can call the flash driver API provided by ROM. Secure boot is not supported by this device, so only a plain image or a CRC32 image is supported by the bootloader. Verification of the CRC32 checksum is performed only if the image indicates a valid image type at offset 0x24 (reserved words of the vector table).

User image structure:



20.2.3.5 User Image Header

Table 95.

Offset	Size in bytes	Symbol	Description
0x00	4	Initial SP	Stack pointer
0x04	4	Initial PC	The application first execution instruction
0x08	24	Vector table	Cortex-M33 Vector table entries
0x20	4	Image length	The length of the current image
0x24	4	Image type	Image Type: Bit [7:0]: - 0x0, plain image - 0x2, plain image with CRC - 0x4, signed image - 0x5, plain image with CRC - 0x6, SB3 mainfest Load_to_ram image: image has the non-zero image_load_address and image_length in the image header Other values are reserved

Table continues on the next page...

Table 95. (continued)

			Bit[10] – Image version included in Image Type [31:16]: 0 – Image version is not in the Image Type 1 – Image version is included in the Image Type Bit[31:16] – Image version (for on chip flash) when bit[10] is set
0x28	4	offsetToExtendedHeader	Offset to extended header For a signed image (Image Type = 0x04), this is offset to the certificate header block For a CRC image (Image Type = 0x2 or 0x5), this is the CRC checksum value
0x2C	8	Vector table	Cortex-M33 Vector table entries
0x34	4	ImageExecutionAddress	The execution address of the image Set 0 if image type is XIP Set to actual image execution address if the image type is load to RAM
0x38		Vector table	Cortex-M33 Vector table entries

20.2.4 Boot pins

Table below shows the ISP pin assignments and is the default assignment used by extend-bootloader. However they can be changed in the CMPA settings, more details, please refer to IFR map.

Table 96. boot ROM pin assignments

ISP pin	Port pin assignment
ISPMODE_N	P3_29
	P0_6
LPUART ISP pin	
LPUART, RX	P0_2
LPUART, TX	P0_3
USB FS ISP mode	
USB0_DM	
USB0_DP	
USB0_VBUS_DET	P2_12

NOTE

This is new for this device.

ISP_mode pin selection depends on IFR1 configuration field ispmode_pin_sel

- 0: P3_29 is used as ISPMODE_N
- 1: P0_6 is used as ISPMODE_N

For 100LQFP and 64BGA, it should be 1.

For 64LQFP, 48QFN/LQFP, 32QFN should be 0

20.2.5 Top-level Boot Flow

The below chart shows the top-level boot flow. The device always starts boot ROM after reset.

- By default, ROM runs at 48MHz
- Boot ROM checks if there is a valid extended bootloader image using CRC32. For power reset, boot ROM does a full image CRC32 check, and for Deep Power Down wake up, boot ROM only does a partial CRC32 check. If the extended bootloader image is valid, boot ROM jumps to the extended bootloader.
- The extended bootloader will determine whether to execute a normal boot or ISP boot based on the ISP_Mode pin or CMPA settings.
- The extended bootloader will run a user image validity check. It will run a CRC32 check if the value is present in the image header.

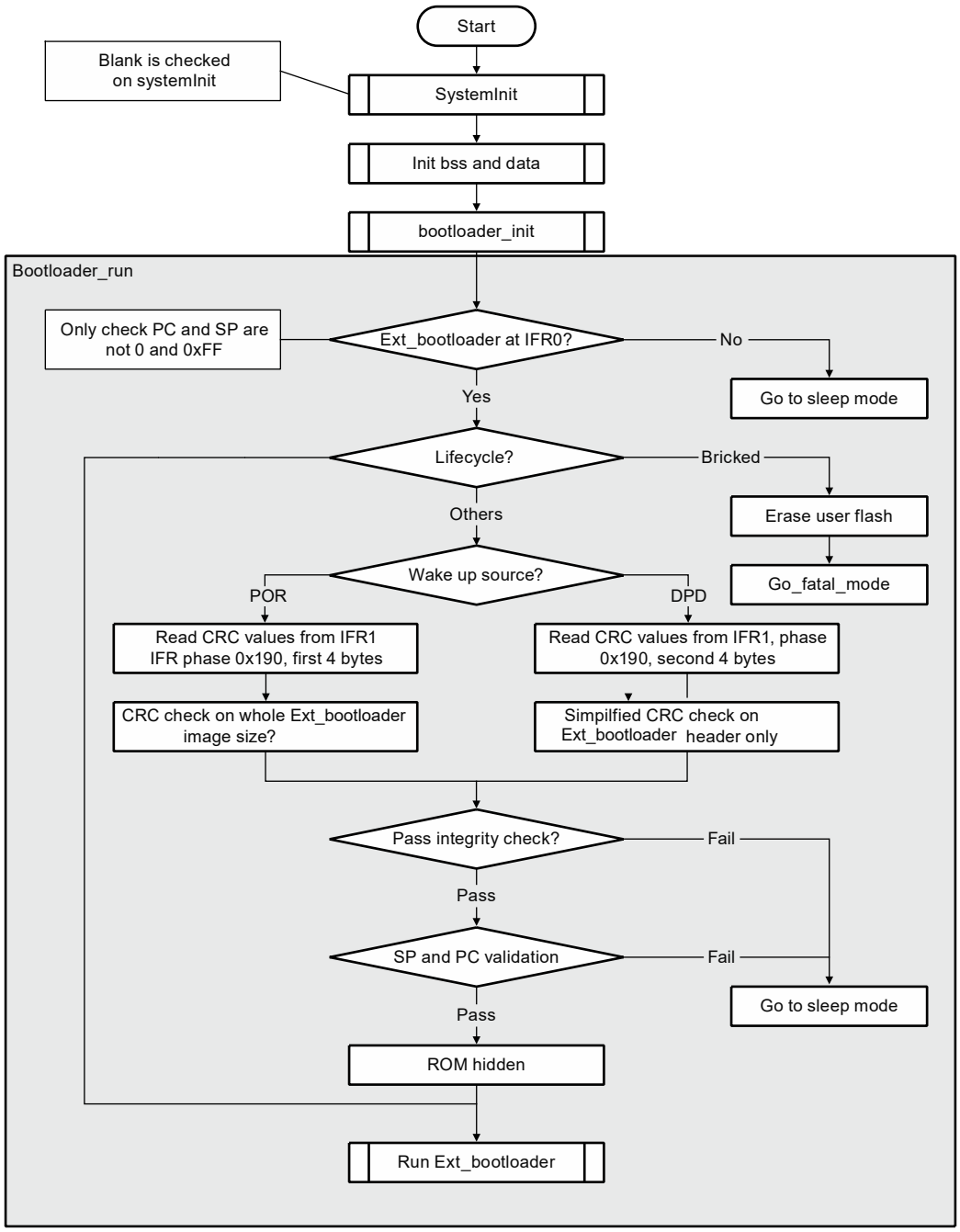
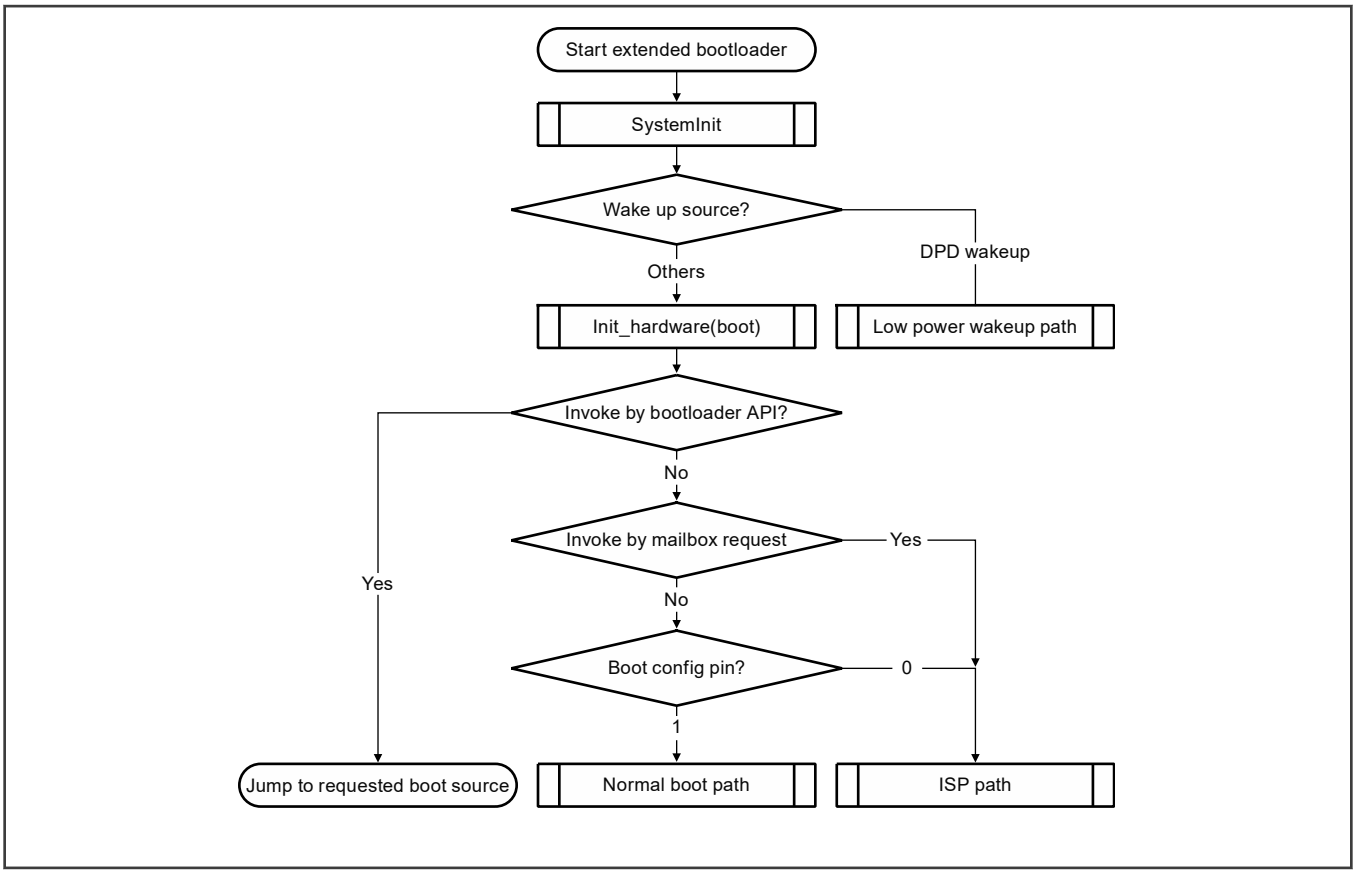
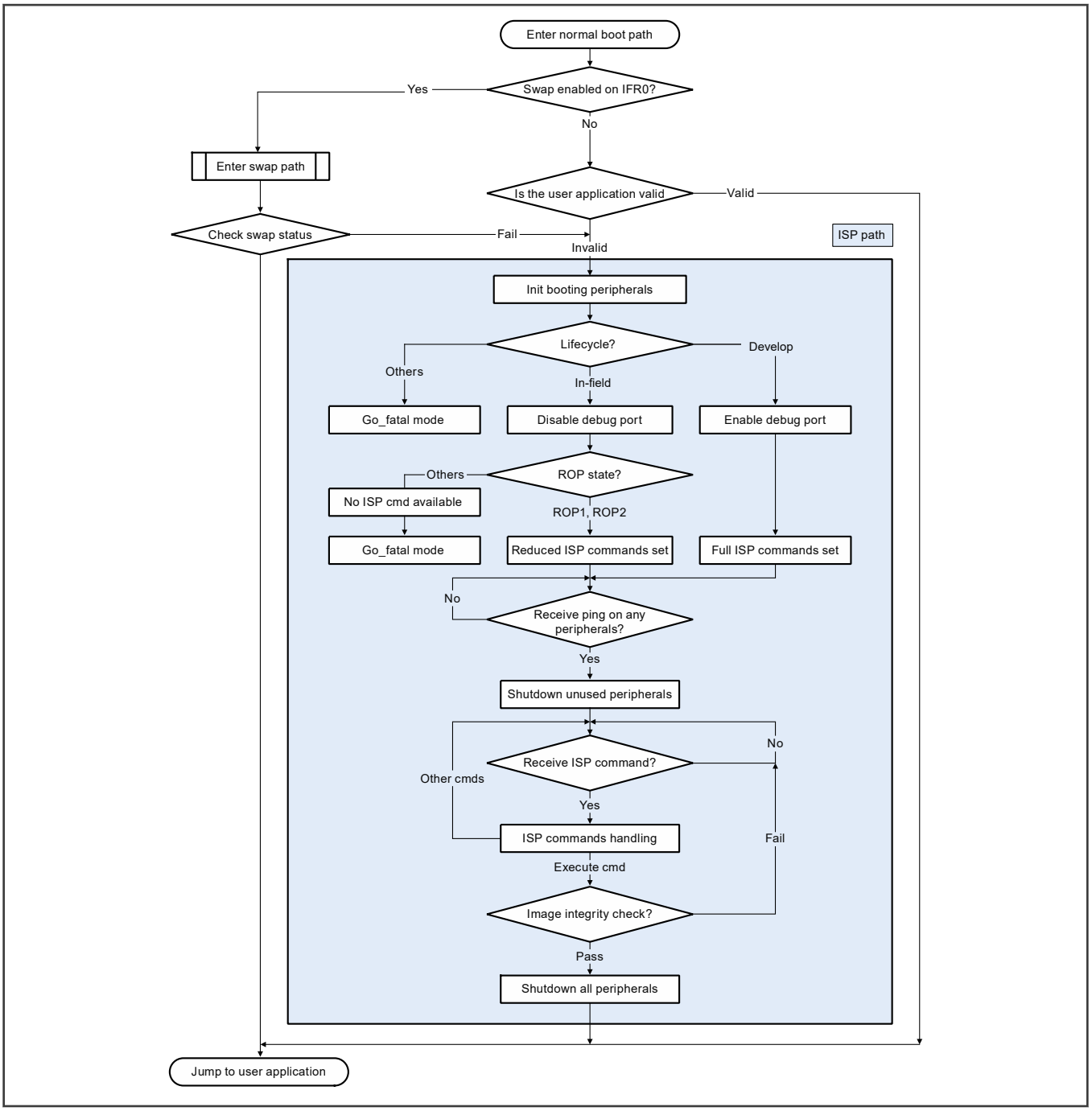


Figure 29. Top level boot flow


Start extended bootloader
SystemInit
Wake up source?
DPD wakeup
Others
Init_hardware(boot)
Low power wakeup path
Invoke by bootloader API?
No
Invoke by mailbox request
Yes
Yes
No
Boot config pin?
0
1
Jump to requested boot source
Normal boot path
ISP path



20.2.6 CMPA configuration options

Table 97. CMPA configuration supported by ROM and Ext_bootloader

CMPA Field	Offset	Bit	Description
Header	0	[31:15]	CMPA header marker, 16'h5963 If the CMPA header marker is cleared and the device is in the Develop lifecycle state, MBC Global Access

Table continues on the next page...

Table 97. CMPA configuration supported by ROM and Ext_bootloader (continued)

			Control 4 will be changed from R/X to R/W/X. Global Access Control 4 is the default permission of the main flash. When the CMPA header marker is written, the permissions of the sectors of the main flash assigned to Global Access Control 4 will be returned to R/W after reset.
BOOT_SPEED	0	[13:12]	Core clock, default 48MHz 00 - 48MHz FRO @1v0 01 - 96MHz FRO @1v1
ISP_BOOT_IF	0	[6:4]	ISP boot interface: 000 - Auto ISP 001 - UART ISP 100 - USB0_HID others – Reserved
ISP_DM_ENTRY	4	[13:12]	Disable ISP mode entry through debug mailbox command. 01 - ISP entry disabled. 00,10, 11 - ISP entry allowed.
ISP_PIN_ENTRY	4	[11:10]	Disable ISP mode entry through pin assertion. 01 - ISP entry disabled. 00,10, 11 - ISP entry allowed.
FLASH_REMAP_SIZE	4	[4:0]	Flash remap size. This field should be written to remap field in flash.
BOOT_FAIL_LED	8	[23:16]	Assert on fatal errors during boot. ROM toggles the GPIO pin identified by this field whenever primary boot fails due to fatal errors before locking-up/reset. Note, use QUICK_SET_/CLR_GPIOx field to set the default level of pin. [4:0] GPIO Pin number [7:5] GPIO port number
POWERDOWN_TIMEOUT_SECS	0x0c	[15:0]	Power down timeout: Timeout value in seconds. When a non-zero value is programmed in this field ROM uses it as idle timeout value to enter power-down state to conserve power
ISP_UART_CFG: UART_BAUD_RATE	0x10	[31:28]	Baud rate configured during UART ISP mode. 0000 - Auto baud detection.

Table continues on the next page...

Table 97. CMPA configuration supported by ROM and Ext_bootloader (continued)

			default: Auto baud detection.
ISP_UART_CFG: UART_TX_FUNC_SLOT	0x10	[27:24]	Identifies the pin mux function slot.
ISP_UART_CFG: UART_TX_PIN	0x10	[23:16]	Override default UART TX ISP pin. Identifies the pin to be used as UART_TX pin. [4:0] GPIO Pin number [7:5] GPIO port number
ISP_UART_CFG: UART_ISP_INSTANCE	0x10	[15:12]	Identifies the LPUART instance used for UART ISP mode.
ISP_UART_CFG: UART_RX_FUNC_SLOT	0x10	[11:8]	Identifies the pin mux function slot.
ISP_UART_CFG: UART_RX_PIN	0x10	[7:0]	Override default UART RX ISP pin. Identifies the pin to be used as UART_RX pin. [4:0] GPIO Pin number [7:5] GPIO port number
USB Product ID	0x24	[31:16]	USB Product ID
USB Vendor ID	0x24	[15:0]	USB Vendor ID
USBx_VBUS_FUNC_SLOT	0x28	[11:8]	Identifies the pin mux function slot.
USBx_VBUS_PIN	0x28	[7:0]	Override default USB0_VBUS_DETECT ISP pin. Identifies the pin to be used as USB0_VBUS detect pin. [4:0] GPIO Pin number [7:5] GPIO port number
ISP_MISC_CFG: ISP_UART_CUST	0x2C	[31:30]	Use customer defined UART ISP pins. 01: Customer defined. 00, 10, 11: Default ROM defined pins.
ISP_MISC_CFG: ISP_USB_CUST	0c2C	[23:22]	Use customer defined GPIO for USB VBUS detect function during ISP mode. 00: Use dedicated VBUS pins. 01: Customer defined GPIO for USB0_VBUS detect. 10: Reserved

Table continues on the next page...

Table 97. CMPA configuration supported by ROM and Ext_bootloader (continued)

			11: Use VDD_USB for VBUS presence. On board regulator should generate VDD_USB voltage using 5V input for VBUS pin on connector.
ACL_SEC_7	0x40 ~ 0x5C	[30:28]	ACL_SEC_7, 15, 23, 31, 39, 47, 55, 63
ACL_SEC_6		[26:24]	ACL_SEC_6, 14, 22, 30, 38, 46, 54, 62
ACL_SEC_5		[22:20]	ACL_SEC_5, 13, 21, 29, 37, 45, 53, 61
ACL_SEC_4		[18:16]	ACL_SEC_4, 12, 20, 28, 36, 44, 52, 60
ACL_SEC_3		[14:12]	ACL_SEC_3, 11, 19, 27, 35, 43, 51, 59
ACL_SEC_2		[10:8]	ACL_SEC_2, 10, 18, 26, 34, 42, 50, 58
ACL_SEC_1		[6:4]	ACL_SEC_1, 9, 17, 25, 33, 41, 49, 58
ACL_SEC_5		[3:0]	ACL_SEC_0, 8, 16, 24, 32, 40, 48, 56
QUICK_SET_GPIO_0	0x60	[31:0]	Drive GPIO 0 port pins high after reset. Each bit corresponds to the pin in GPIO port 0. When set ROM drives the corresponding pin high as soon as possible. By default most pins come-up as tri-stated inputs. This feature allows customer to specify active drive pins soon after reset instead of waiting till complete boot.
QUICK_CLR_GPIO_0	0x64	[31:0]	Drive GPIO 0 port pins low. See description of QUICK_SET_GPIO_0 field.
QUICK_SET_GPIO_1	0x68	[31:0]	
QUICK_CLR_GPIO_1	0x6C	[31:0]	
QUICK_SET_GPIO_2	0x70	[31:0]	
QUICK_CLR_GPIO_2	0x74	[31:0]	
QUICK_SET_GPIO_3	0x78	[31:0]	
QUICK_CLR_GPIO_3	0x7C	[31:0]	
QUICK_SET_GPIO_4	0x80	[31:0]	
QUICK_CLR_GPIO_4	0x84	[31:0]	
ROP_STATE	0x90	[31:0]	
ROP_STATE_DP (Duplicate)	0xA0	[31:0]	

NOTE

- Supports USB and LPUART booting peripherals

  CMPA ISP_BOOT_IF field setting choices are;

  000 - Auto ISP, USB & UART

  001 - UART ISP

  100 - USB0_HID

  Others - Reserved

20.2.6.1 ISP Boot mode

Depending on the values of ISP mode in the CMPA and ISP pin, bootloader decides whether to enter normal boot flow or entering ISP mode.

Table 98. ISP mode based on ISP_BOOT_IF CMPA bit field (word0[6:4])

ISP boot interface	CMPA field value	Description
Auto ISP	3'b000	Bootloader probes the active peripheral from the below serial interfaces to download image: LPUART and USB0. (however it depends on extended bootloader choice)
UART ISP	3'b001	LPUART is used to download the image
USB0 HID	3'100	USB0 is used to download the image
USB1 HID	3'101	NA
CAN ISP	3'110	NA
Reserved	3'111	

20.2.7 ROM API

ROM bootloader provides APIs for users. Disable the interrupts before making any ROM API calls is suggested, since API does not handle any interrupts.

The struct of bootloader_tree:

```
//! @brief Root of the bootloader API tree.
//!
//! An instance of this struct resides in read-only memory in the bootloader. It
//! provides a user application access to APIs exported by the bootloader.
//!
//! @note The order of existing fields must not be changed.
//!
//! @ingroup context
typedef struct BootloaderTree
{
    void (*runBootloader)(void *arg);           //!< Function to start the bootloader executing.
    const flash_driver_interface_t *flashDriver; //!< Internal Flash driver API.
} bootloader_tree_t;
```

The ROM API table is located at address (0x03003fe0u)

```
#define BOOTLOADER_TREE_LOCATION (0x03003fe0u)
#define g_bootloaderTree ((bootloader_tree_t *)BOOTLOADER_TREE_LOCATION)
```

20.2.7.1 runBootloader API

ROM provides an API for the user application to enter ISP mode based on the designated ISP interface mode.

Prototype

Void (*runBootloader) (void *arg)

Table 99. API Prototype Fields

Field	Offset	Description
TAG	[31:24]	Fixed value: 0xEB (Enter boot mode)
Boot mode	[23:20]	0: Enter passive mode 1: Enter ISP mode
ISP interface	[19:16]	0: Auto detection 1: USB HID 2: UART 5: Reserved
Reserved	[15:4]	
Image index	[03:00]	Used for Boot mode 0

20.2.7.2 Flash Driver API

Flash driver API provide the flash operation supported by CM33 based on flash technology.

```
//! @brief Interface for the flash driver.
typedef struct FlashDriverInterface
{
    // Flash driver
    status_t (*flash_init)(flash_config_t *config);
    status_t (*flash_erase_sector)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes, uint32_t key);
    status_t (*flash_program_phrase)(flash_config_t *config, uint32_t start, uint8_t *src, uint32_t lengthInBytes);
    status_t (*flash_program_page)(flash_config_t *config, uint32_t start, uint8_t *src, uint32_t lengthInBytes);
    status_t (*flash_verify_program)(flash_config_t *config,
                                     uint32_t start,
                                     uint32_t lengthInBytes,
                                     const uint8_t *expectedData,
                                     uint32_t *failedAddress,
                                     uint32_t *failedData);
    status_t (*flash_verify_erase_phrase)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
    status_t (*flash_verify_erase_page)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
    status_t (*flash_verify_erase_sector)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
    status_t (*flash_get_property)(flash_config_t *config, flash_property_tag_t whichProperty, uint32_t *value);
    // IFR driver
    status_t (*ifr_verify_erase_phrase)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
    status_t (*ifr_verify_erase_page)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
    status_t (*ifr_verify_erase_sector)(flash_config_t *config, uint32_t start, uint32_t lengthInBytes);
    status_t (*flash_read)(flash_config_t *config, uint32_t start, uint8_t *dest, uint32_t lengthInBytes);
    // version
    standard_version_t version; //!< flash driver API version number.
} flash_driver_interface_t;
```

And struct of flash_config_t is defined as:

```
/*! @brief Flash driver state information.
 *
 * An instance of this structure is allocated by the user of the flash driver and
 * passed into each of the driver APIs.
 */
typedef struct
{
    uint32_t PFlashBlockBase;  /*!< A base address of the first PFlash block */
    uint32_t PFlashTotalSize;  /*!< The size of the combined PFlash block. */
    uint32_t PFlashBlockCount; /*!< A number of PFlash blocks. */
    uint32_t PFlashPageSize;   /*!< The size in bytes of a page of PFlash. */
    uint32_t PFlashSectorSize; /*!< The size in bytes of a sector of PFlash. */
    flash_ffr_config_t ffrConfig;
    //    flash_mode_config_t modeConfig;
} flash_config_t;
```

Table 100. Flash driver API function details

Function name	Description
Flash_init	Initializes the global flash properties structure members
Flash_erase_sector	Erases the flash sectors encompassed by parameters passed into function
Flash_program_phrase	Programs flash phrases with data at locations passed in through parameters
Flash_program_page	Programs flash with data at locations passed in through parameters
Flash_verify_program	Verifies programming of the desired flash area at a specified margin level.
Flash_verify_erase_phrase	Verify that the flash phrases are erased
Flash_verify_erase_page	Verify that the flash pages are erased
Flash_verify_erase_sector	Verify that the flash sectors are erased
Flash_get_property	Returns the desired flash property
ifr_verify_erase_phrase	Verify that the IFR0 phrases are erased
ifr_verify_erase_page	Verify that the IFR0 pages are erased
ifr_verify_erase_sector	Verify that the IFR0 sectors are erased
flash_read	Reads flash at locations passed in through parameters

Table 101. Flash API return status code

Status	Code	Description
kStatus_FLASH_Success	0	The flash operation is successful
kStatus_FLASH_InvalidArgument	4	Invalid argument during executing API
kStatus_FLASH_SizeError	100	Invalid size during executing API

Table continues on the next page...

Table 101. Flash API return status code (continued)

kStatus_FLASH_AlignmentError	101	Alignment error during executing API
kStatus_FLASH_AddressError	102	Address error during executing API
kStatus_FLASH_AccessError	103	Invalid instruction codes during executing API
kStatus_FLASH_ProtectionViolation	104	Protection violation flag, indicate that program/erase operation is requested to execute on protected areas
kStatus_FLASH_CommandFailure	106	Command execution failure during executing API
kStatus_FLASH_UnknownProperty	106	Unknown property when to call flash_get_property
kStatus_FLASH_EraseKeyError	107	Invalid EraseKey during executing flash_erase API

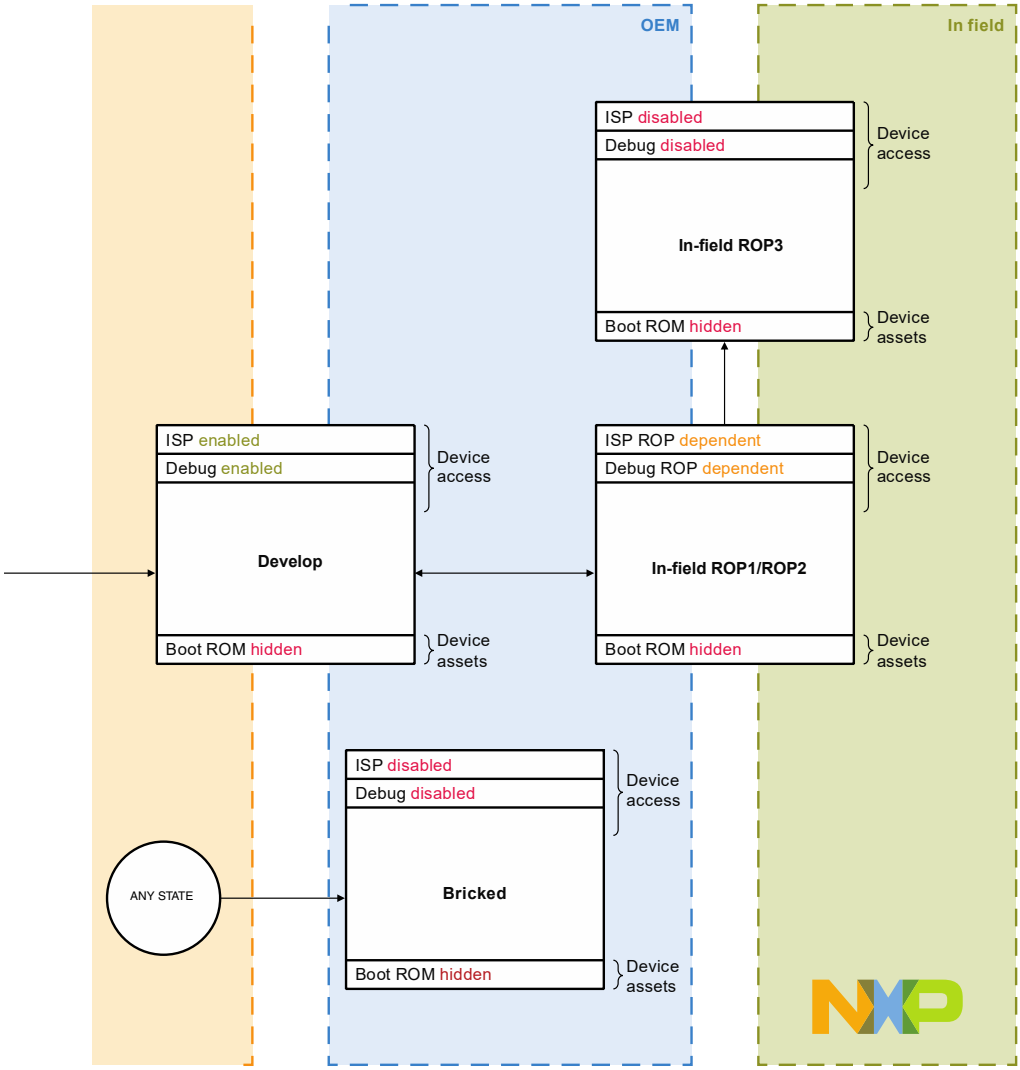
20.3 Lifecycle

20.3.1 overview

This device supports a lifecycle state model to protect code from reading from the device internal flash, which is called code read protection feature. There are different levels of protections in the system, so that access to the on-chip flash and use of ISP can be restricted. Also the lifecycle state of the device determines the debug access and ISP command availability. Please see other sections details. The lifecycle state is controlled by ROP_STATE(IFR0).

The boot ROM will check the lifecycle state, then determine what boot flow is to run, debug port to be enabled/locked and available debug mail access command sets. After ROM exits to extended bootloader, lifecycle will be checked again to determine available ISP command sets.

20.3.2 Lifecycle State transition



20.3.3 Lifecycle states

This device supports below lifecycle states,

Table 102. Lifecycle

Lifecycle	Lifecycle Type	Description
Develop	Customer	Initial customer development state after leaving NXP manufacturing
In-field ROP1/ROP2	Customer	In-field application state, with ROP protection
In-field ROP3	Customer	In-field application state, with ROP protection, prevent use of field return
Bricked	End-of-life	Bricked state to prevent device use

20.3.4 Read Out Protection (ROP)

This device does not support secure boot, but it does support four levels of Read Out Protection (ROP), also refer to ROP_STATE. This read out protection is a mechanism that allows user to enable different levels of protection in the system. It is a 32-bit field stored in IFR0. It can be programmed by customer.

- ROP_LEVEL0, ROP_STATE = 0xFFFF_FFFF (erased FLASH value), no ROP. Default for blank state.
- ROP_LEVEL1, ROP_STATE = 0xEEBA_04C3

  Debug is disabled and unlocked, however it can be modified by customer, only limited debug mailbox commands are available.
- ROP_LEVEL2, ROP_STATE = 0x4939_8D8B

  Debug is disabled and locked, it cannot be modified by customer, only limited debug mailbox commands are available.
- ROP_LEVEL3, ROP_STATE = 0xB0AB_B703

  Debug is disabled and locked, it cannot be modified by customer, no debug mailbox commands are available
- Anything else = ROP3-like behavior (Debug disabled/Locked, ISP disabled)
- ROP_STATE

  Value can be read out from SYSCON-> ROP_STATE (read/write register)

  ROM reads ROP_STATE from ROP_STATE and ROP_STATE_DP on IFR0, does the compare, and if it does not match, resets the device; if match, then programs it to SYSCON->ROP_STATE.

Table 103. ROP_STATE IFRR0 locations

IFR0	IFR0 Address
ROP_STATE	0x0100_0090
ROP_STATE_DP	0x0100_00A0

See Debugger Mailbox Access Port & Debug session protocol for details.

20.4 Debugger Mailbox Access Port & Debug session protocol

The Debugger Mailbox Access Port(DM-AP) offers a register-based mailbox accessible by CPU0 (on MXCA1xx, there is only core CM33) and the device debug port (DP) of the MCU. Debug Mail Box allows the debugger communicate with ROM. It is protected by lifecycle State. ROM will control the debug port based on LC State. For this device, no security hardware, so the Debug session protocol will be followed to interact with tools over 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
- Support data above 32-bits by using ACK_TOKEN that moderates the transfer in 32-bit value chunks
- Requests and responses sue the same basic structure

20.4.1 Debug session protocol

20.4.1.1 Request packet layout

The first word transmitted in a request is a header word containing the command ID and number of following data words. The command packet is set to the device by writing 32-bits at a time to the REQUEST register. When sending command packets greater than 32-bits, the debugger should read an ACK_TOKEN in the RETURN register before writing the next 32-bits.

Following the header are the number of 32-bit words specified in the header

Table 104. 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[];
};
```

20.4.1.2 Response packet layout

Response register byte description structure is the same as Request register byte description.

Table 105. Response Packet 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_reponse {
uint16_t commandID;
uint16_t dataWordCount;
uint32_t data[];
};
```

20.4.1.3 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 106. 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 */

};

20.4.1.4 Error handling

When an overrun occurs from either side of the communication, the appropriate error flag is set in the CSW register. The state machine hardware prevents further communication in either direction. The debugger must start with a new resynchronization request to clear the error flag.

20.4.2 DM-AP command

20.4.2.1 All DM-AP command

This table lists all DM-AP commands supported by this device.

Table 107. DM-AP commands

Command	ID	Parameter/Response	Description
Start DM-AP (Legacy command)	0x01	Parameters: None *Response*: 32-bit status	Cause the device to enter DM-AP command mode. This must be done prior to sending other commands.
Get ROP Level	0x02	Parameters: None *Response*: 32-bit status	Return the ROP level value
Bulk Erase (Legacy command)	0x03	Parameters: None *Response*: 32-bit status	Erase the entire on-chip flash memory and IFR0, sector 0.
Exit DM-AP (Legacy command)	0x04	Parameters: None *Response*: 32-bit status	Cause the device to exit DM-AP command mode. The Device returns to normal mode.
Enter ISP Mode ((For Extended bootloader)	0x05	Parameters: dataWordCount: 0x1 data[0]: ISP mode enum. Same as IAP API.	Enter specified ISP mode.

Table continues on the next page...

Table 107. DM-AP commands (continued)

		Response: 32-bit status	
Set FA Mode	0x06	*Parameters*: dataWordCount: 0xFE data[]: FA Request *Response*: 32-bit status	Set the part permanently in "Fault Analysis" mode to return to NXP factory.
Start Debug Session	0x07	Parameters: None *Response*: 32-bit status	This command is used to indicate ROM the intention of connecting debugger. ROM enables debug access and enters while(1) loop.
Write words to flash (New)	0x09	Parameters: dataWordCount: 0x5 data[0]: FlashAddress data[4]: data to be written *Response*: 32-bit status	Write four words (128 bits) to flash Flash address must be 16-bit aligned
Read words from flash (New)	0x0A	Parameters: dataWordCount: 0x5 data[0]: FlashAddress *Response*: 32-bit status data[4]: Data out	Read four words from flash Flash address must be 16-bit aligned
Erase one Sector (New)	0x0B	Parameters: dataWordCount: 0x1 data[0]: flash page index *Response*: 32-bit status	Erase one flash sector

20.4.2.2 Available debug mailbox commands per lifecycle

Available debug mailbox commands sets depends on lifecycle state and read out protection level. If the part is in Bricked state or in-field ROP3, then ROM will lock the part, no debug mail commands are available.

Below table summarizes available debug mail box commands corresponding to read out protection level and lifecycle state

Table 108. Debug Mailbox Command Sets Per Lifecycle State

LC MANAGEMENT	Available commands
Develop	Start DM-AP Get_ROP_Level, Bulk Erase, (user flash and IFR0 sector 0) Exit DM-AP, Enter_ISP_Mode,

Table continues on the next page...

Table 108. Debug Mailbox Command Sets Per Lifecycle State (continued)

	Set_FA_Mode, Start Debug Session, Write_Flash_Word, Read_Flash_Word, Erase_One_Sector
In-field ROP1	Start DM-AP, Get_ROP_Level, Bulk_Erase, (user flash and IFR0 sector 0) Enter_ISP_Mode, Set_FA_Mode, Exit DM-AP
In-field ROP2	Start DM-AP, Get_ROP_Level, Bulk_Erase, (user flash and IFR0 sector 0) Set_FA_Mode, Exit DM-AP
In-field ROP3	Debug mail box is not available
Bricked	Debug mail box is not available

20.5 Extended Bootloader & In-System programming (ISP)

20.5.1 Overview

After a POR or WARM reset sequence, CPU will always start ROM code execution in BOOT ROM memory. After the operations in BOOT ROM, the control of CPU is transferred to the extended bootloader in flash IFR0 memory. This extended bootloader only supports load non-secure image (image format specified in Boot ROM section) to on-ship memory, including RAM and flash. Also bootloader will integrity check if CRC check is present in the image header before jump to user image.

Please note the booting available peripherals

Bootloader features:

- Extended bootloader size can be up to 24KB (IFR0, sector 1 ~ 3)
- Booting on USB FS HID interface and LPUART interface with auto-baud detection
- Supports ISP calls path
- Low power wakeup path
- Phantom part configuration
- Image update and swap
- No need to implement Debug Mailbox commands since it is done by ROM

20.5.2 Boot peripherals and default pins

Table 109. Bootloader Peripheral Pinmux

Peripheral	Instance	Alt Mode	Port	GPIO
LPUART	0	2	LPUART0_RXD	P0_2
			LPUART0_TXD	P0_3
USB	0		USB0_DM	USB0_DM
			USB0_DP	USB0_DP

20.5.3 Functional description

20.5.3.1 Jump to user application path

With successfully integrity CRC check on user application, before jump to user application, extended bootloader needs:

- Shutdown all boot peripherals
- Restore the registers to default values, these registers are used by all modules used by ROM or extended bootloader, except MBC and GLIKEY settings
- Set GLIKEY and MBC settings based on lifecycle

20.5.3.2 Low power wakeup Path

On Deep-Sleep wake-up or Sleep wake-up, the core would continue executing application without any ROM- or HW-enforced checks. It would be up to the application to implement its own checks if a customer deems necessary.

The difference between ROM and extended bootloader on low power wakeup path is that extended bootloader run CRC check on user image if image indicates valid image type at offset 0x24 (reserved word of vector table)

The low power wake-up path is focused on reducing the time to start executing customer code when waking from Deep Power-Down mode. This reduction in boot time is primarily achieved by doing simplified integrity CRC check. With successfully pass the integrity CRC check, then jump to wakeup entry specified in IFR0; otherwise enter into ISP path in extended bootloader.

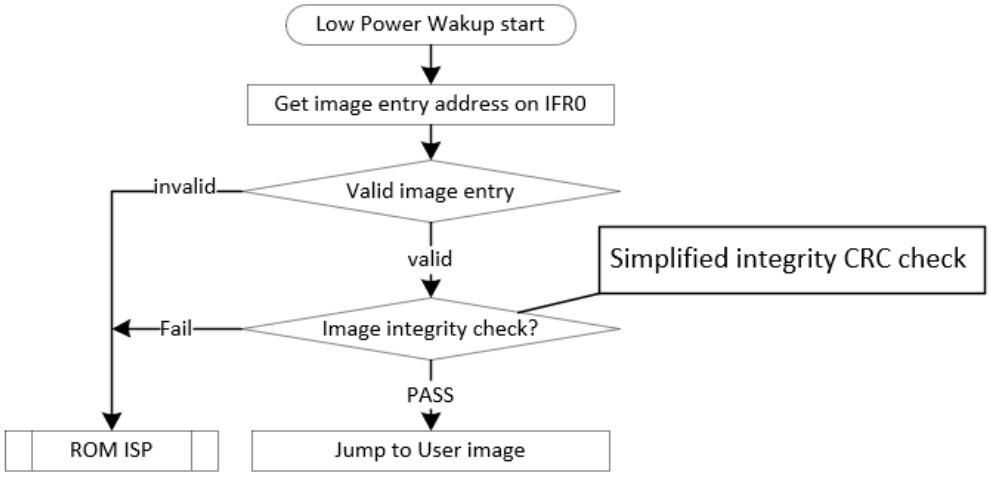


Figure 4-5 Extended Bootloader DPD Wakeup Flow

20.5.3.3 Supports ISP calls path

Due to limited IFR0 memory space, extended bootloader support these commands

Full ISP command set:

- ReadMemory
- FlashEraseAll, (user flash & IFR0 Sector 0)
- FlashEraseRegion,
- Reset,
- WriteMemory,
- GetProperty,
- Execute

Reduced ISP command set:

- FlashEraseAll, (user flash & IFR0 Sector 0)
- Reset,
- GetProperty,

However available commands set highly depends on the device lifecycle. Lifecycle management is the same as in ROM section

Table 110. Available ISP commands set per Lifecycle State

LC State	Available commands set
Develop	Full ISP command set
In-field ROP1	Reduced command set if entry through: ISP_PIN, INVALID_IMAGE, DEBUG_MAILBOX Full command set if entry through: bootloader_API
In-field ROP2	Reduced command set if entry through: INVALID_IMAGE, bootloader_API
In-field ROP3	No ISP commands are available
Bricked	No ISP commands are available

20.5.4 Flash Swap path

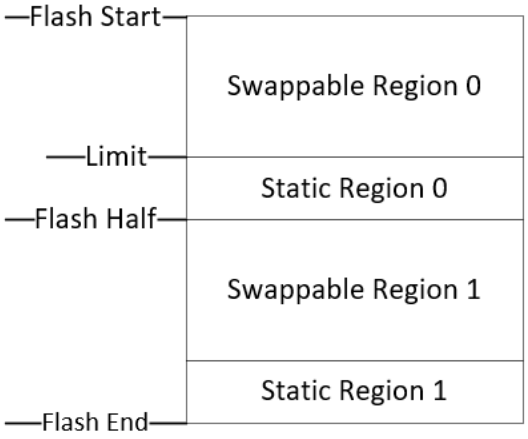
Extended bootloader supports flash swap, which is the function of FMC. On this device, there is only one bank of flash, this one bank flash is divided to four regions, two swappable regions and two static regions.

- Lower swappable region (swappable region 0): flash physical start address ~ limit
- Lower static region (static region 0): limit + 1 ~ half size of flash – 1
- Upper swappable region (swappable region 1): half size of flash ~ half size of flash + limit
- Upper static region (static region 1): half size of flash + limit + 1 ~ size of flash – 1

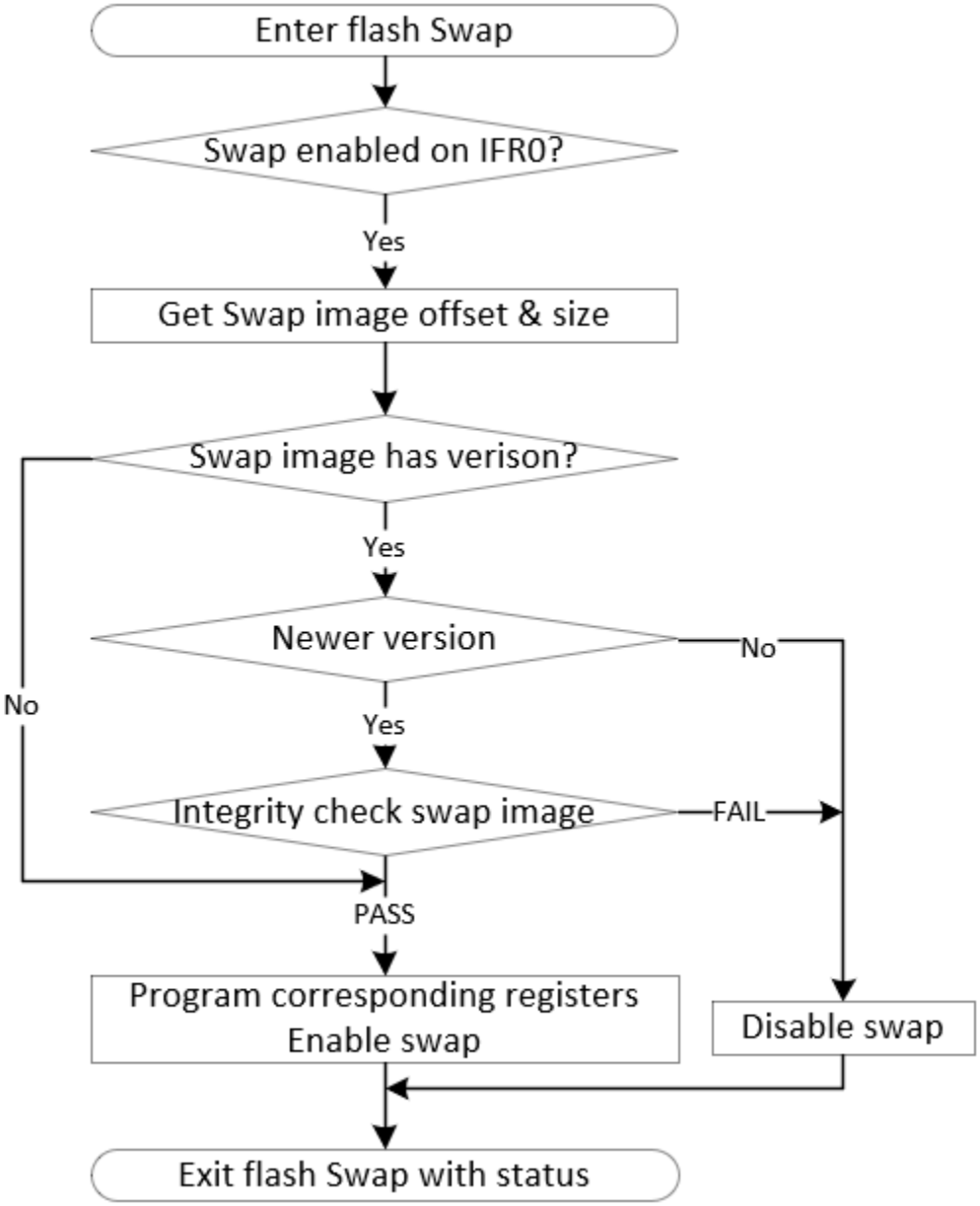
To able to run flash swap, the basic requirements are:

1. Size of swappable region 0 equals to size of swappable region 1

2. Size of swappable region 0 plus size of static region 0 equals to half size of flash



20.5.4.1 Flash swap flow chart



20.5.5 ISP (In-System Programming) Protocol

This section explains the general protocol for the packet transfers between the host and the 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.

20.5.5.1 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)

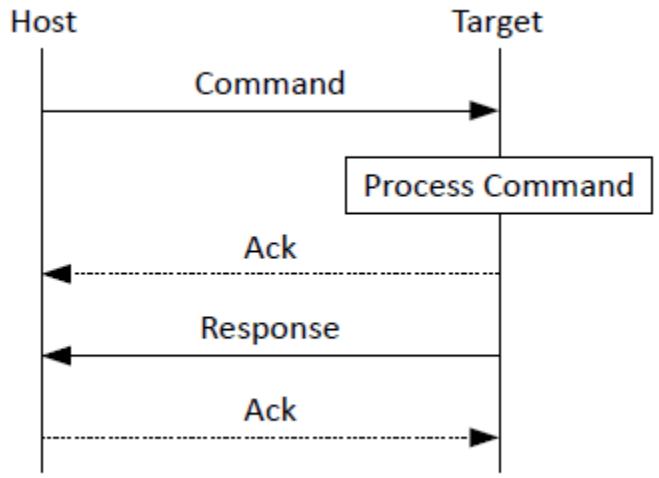


Figure 4-9 Command with No Data Phase

Remark: 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.

20.5.5.2 Command with the 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.

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.

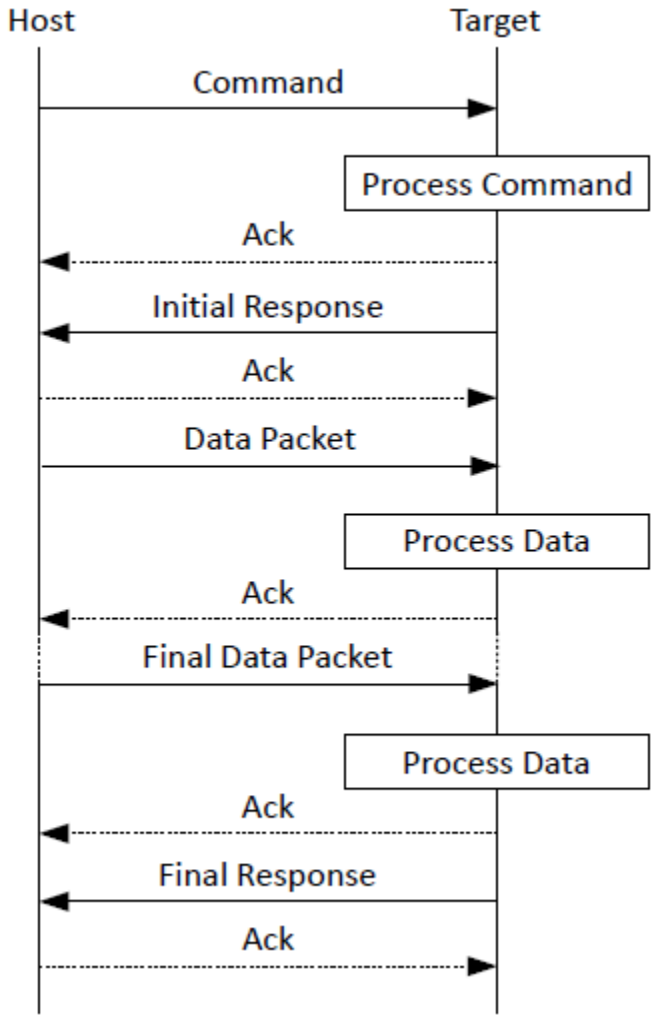


Figure 4-10 Command with Incoming Data Phase

20.5.5.3 Command with the 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).

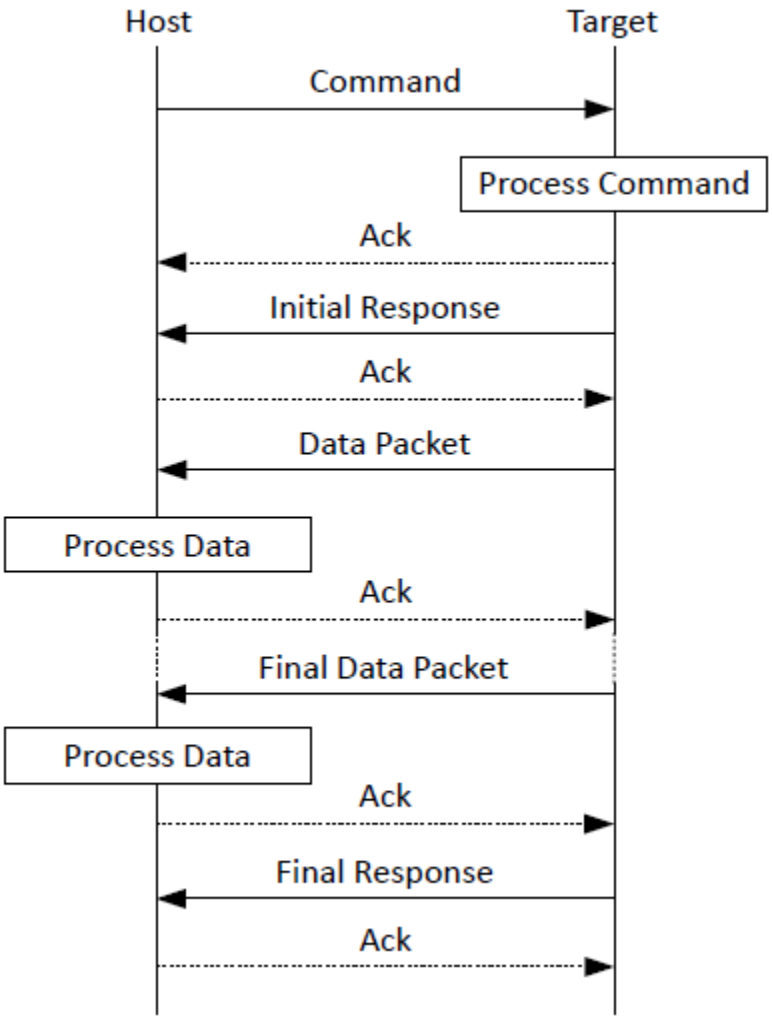


Figure 4-11 Command with Outgoing Data Phase

20.5.6 Bootloader packet types

20.5.6.1 Introduction

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.

20.5.7 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 UART, 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 the later sections.

Table 111. Ping Packet Format

Byte #	Value	Name
0	0x5A	Start byte
1	0xA6	Ping

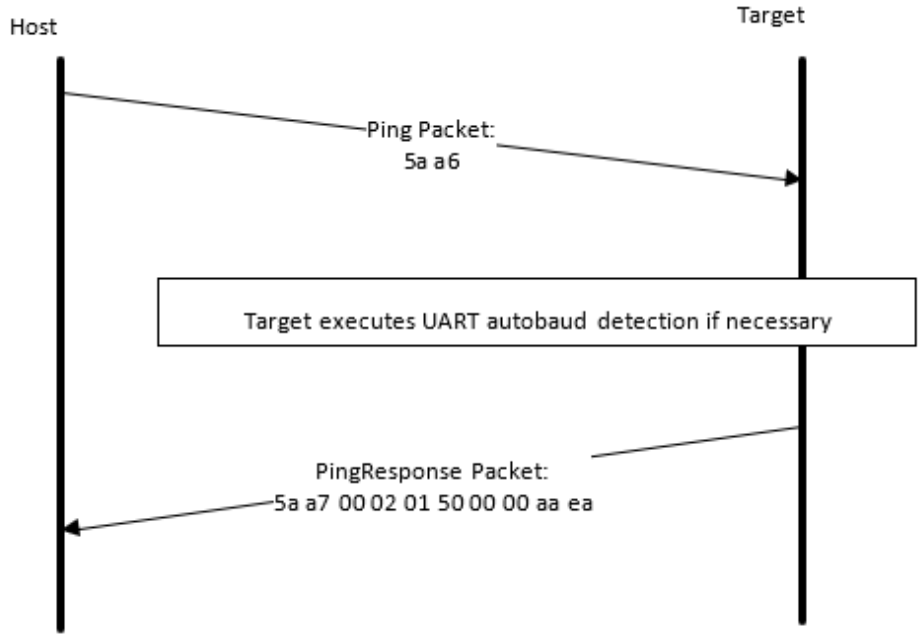


Figure 4-12 Ping Packet Protocol Sequence

20.5.7.1 Ping response packet

The target sends a Ping response packet back to the host after receiving a Ping packet. If communication is over a UART 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 112. Ping Response Packet Format

Byte	Value	Parameter
0	0x5A	Start byte
1	0xA7	Ping response code
2	0x00	Protocol bugfix

Table continues on the next page...

Table 112. Ping Response Packet Format (continued)

3	0x03	Protocol minor
4	0x01	Protocol major
5	0x50	Protocol name = 'P' (0x50)
6	0x00	Options low
7	0x00	Options high
8	0xfb	CRC16 low
9	0x40	CRC16 high

For the UART 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.

20.5.7.2 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 UART, I^2C, and SPI. The USB HID peripheral does not use framing packets. Instead, the packetization inherent in the USB protocol itself is used.

Table 113. Framing Packet Format

Byte	Value	Parameter	Description
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 Section 13.5 “CRC16 algorithm”.
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 114. 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 115. Packet Type Field

Packet type	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 UART autobaud.
0xA7	kFramingPacketType_PingResponse	A response to Ping. It contains the framing protocol version number and options.

20.5.7.3 CRC16 algorithm

The CRC is computed over each byte in the framing packet header, excluding the CRC16 field itself, and all of the payload bytes. The CRC algorithm is the XMODEM variant of

CRC16.

The characteristics of the XMODEM variants are:

Table 116. CRC16 Algorithm

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;
}
```

20.5.7.4 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 117. Command Packet Format

Command Packet Format (32 bytes)										
Command Header (4 bytes)				28 bytes for Parameters (Max 7 parameters)						
Tag	Flags	Rsvd	Param Count	Param 1 (32-bit)	Param 2 (32-bit)	Param 3 (32-bit)	Param 4 (32-bit)	Param 5 (32-bit)	Param 6 (32-bit)	Param 7 (32-bit)

Table 118. Command Header Format

Byte #	Command header field	Reset value
0	Command or Response tag	The command header is 4 bytes long with these fields.
1	Flags	

Table continues on the next page...

Table 118. Command Header Format (continued)

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 seven 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 of the transfers.

Table 119. Command Tags

Command tag	Name	Description
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	
0x07	GetProperty	
0x09	Execute	
0x0B	Reset	

Table 120. Response Tags

Response tag	Name	Description
0xA0	GenericResponse	The response tag specifies one of the responses the bootloader (target) returns to the host. The valid response tags are listed here.
0xA7	GetPropertyResponse (used for sending responses to GetProperty command only)	
0xA3	ReadMemoryResponse (used for sending responses to ReadMemory command only)	

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 seven parameters.

20.5.7.5 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses include:

- GenericResponse.

- GetPropertyResponse.
- ReadMemoryResponse.

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 121. Generic Response 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 122. GetPropertyResponse Parameters

Byte #	Value	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 two for the status code and the data byte count parameters shown below.

Table 123. 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.

20.5.8 ISP commands set

Available ISP commands set is limited per lifecycle state.

Table 124. ISP Commands Set

Lifecycle	Command sets
Develop	Full commands set: • ReadMemory • FlashEraseAll, • FlashEraseRegion, • Reset, • WriteMemory, • GetProperty, • Execute
In-field ROP1/ROP2	Partial commands set • FlashEraseAll, • Reset, • GetProperty,
In-field ROP3	NA

20.5.8.1 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 125. Parameters for GetProperty Command

Byte #	Parameter
0 - 3	Property tag See section 6.6.17 for more details.
4 - 7	External Memory Identifier (only applies to get property for external memory, or status identifier if the property tag is equal to 8).

Table 126. GetProperty Command Packet Format (Framing exmaple)

GetProperty	Parameter	Value
Framing packet	Start byte	0x5A
	PacketType	0xA4, kFramingPacketType_Command
	Length	0x0C 0x00
	Crc16	0x4B 0x33

Table 127. GetProperty Command Packet Format (example)

GetProperty	Parameter	Value
Command packet	CommandTag	0x07 – GetProperty
	Flags	0x00
	Reserved	0x00
	ParameterCount	0x02
	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). Table 4-20 shows an example of a GetPropertyResponse packet.

Table 128. 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

Table continues on the next page...

Table 128. GetProperty Response Packet Format (example) (continued)

	Status	0x00000000
	PropertyValue	0x0000014b - CurrentVersion

20.5.8.2 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 4-21.

Table 129. Parameter for FlashEraseAll Command

Byte #	Parameter	
0-3	Memory ID	
	0x000	Internal Flash

Table 4-22.

Table 130. 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 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.

20.5.8.3 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 two parameters required for the FlashEraseRegion command. The start address and byte count parameters are aligned down and aligned up respectively to be sector aligned before erase operation. 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 131. 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.

Response: The target returns a GenericResponse packet with one of the following error status codes.

Table 132. FlashEraseRegion Response Status Codes

Status code
kStatus_Success (0).
kStatus_MemoryRangeInvalid (10200).
kStatus_FlashAlignmentError (101).
kStatus_IFlashAddressError (102).
kStatus_FlashAccessError (103).
kStatus_FlashProtectionViolation (104).
kStatus_FlashCommandFailure (105).

20.5.8.4 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 133. Parameter 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.

Table 134. ReadMemory Command Packet Format (example)

ReadMemory	Parameter	Value
Framing packet	Start byte	0x5A
	PpacketType	0xA4, kFramingPacketType_Command
	Length	0x10 0x00
	Crc16	0xF4 0x1B
Command packet	CommandTag	0x03 - ReadMemory
	Flags	0x00
	Reserved	0x00
	ParameterCount	0x03
	StartAddress	0x20000400
	ByteCount	0x00000064
	Memory ID	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

20.5.8.5 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 command.
- Writing to flash requires the start address to be page aligned.
- The byte count is rounded up to a page size, 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 two parameters required for WriteMemory command. The memory ID is optional. Internal memory will be selected as default if memory ID is not specified.

Table 135. Parameters for WriteMemory Command

Byte #	Command
0-3	Start address

Table continues on the next page...

Table 135. Parameters for WriteMemory Command (continued)

4-7	Byte count
8-11	Memory ID

Table 136. WriteMemory Command Packet Format (example)

WriteMemory	Parameter	Value
Framing packet	Start byte	0x5A
	PacketType	0xA4, kFramingPacketType_Command
	Length	0x10 0x00
	Crc16	0x97 0xDD
Command packet	CommandTag	0x04 - WriteMemory
	Flags	0x01
	Reserved	0x00
	ParameterCount	0x03
	StartAddress	0x20000400
	ByteCount	0x00000064
	Memory ID	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.

20.5.8.6 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 137. 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.

20.5.8.7 Reset command

The Reset command results in the bootloader resetting the chip.

The Reset command requires no parameters.

Table 138. Reset Command Packet Format (example)

Reset	Parameter	Value
Framing packet	Start byte	0x5A
	PacketType	0xA4, kFramingPacketType_Command
	Length	0x04 0x00
	Crc16	0x6F 0x46
Command packet	CommandTag	0x0B - reset
	Flags	0x00
	Reserved	0x00
	ParameterCount	0x03

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 five seconds for the user application to start running from Flash.

20.5.9 Get Properties Command Properties

This section lists the properties of the GetProperty and SetProperty commands.

Get/Set property definitions are provided in this section.

Table 139. Properties Used by Get/SetProperty Commands, Sorted by Values

Property	Writable	Tag Value	Size	Description
Current Version	No	01h	4	Current bootloader version.
Available Peripherals	No	02h	4	The set of peripherals supported on this chip.
Available Commands	No	07h	4	The set of commands supported by the bootloader
Check Status	No	08h	4	Return the status based on specified status identifier 0 – CRC status 32-bit return value for

Table continues on the next page...

Table 139. Properties Used by Get/SetProperty Commands, Sorted by Values (continued)

				CRC Check 1. – Application CRC check failed 2. – Application CRC check is inactive 3. – Application CRC check is invalid a. – Last Error See the details of last error in later section
Max Packet Size	No	0Bh	4	Maximum supported packet size for the currently active peripheral interface.
Lifecycle State	No	11h	4	The lifecycle of the device. 0x5aa55aa5 – Device is in development lifecycle. 0xc33cc33c – Device is in deployment lifecycle.
Unique Device Identification	No	12h	16	Unique device identification
Target Version	No	18h	4	Extended bootloader version

20.5.9.1 Current Version property

The value of this property is a 4-byte structure containing the current version of the bootloader.

Table 140. Current Version Property Fields

Bit	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'K' (0x4B)	Major version	Minor version	Bugfix version

Get/Set property definitions are provided in this section.

20.5.9.2 Available Peripherals property

The value of this property is a bitfield that lists the peripherals supported by the bootloader and the hardware on which it is running.

Table 141. Peripheral Bits

Bit	[31:7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Field	Reserved	Reserved	Reserved	USB HID	Reserved	SPI Slave	I2C Slave	LPUART

If the peripheral is available, then the corresponding bit will be set in the property value. All reserved bits must be set to 0.

20.5.9.3 Available Commands property

This property value is a bitfield with set bits indicating the commands enabled in the bootloader. Only commands that can be sent from the host to the target are listed in the bitfield. Response commands such as GenericResponse are excluded.

The bit number that identifies whether a command is present is the command's tag value minus 1. 1 is subtracted from the command tag because the lowest command tag value is 0x01. To get the bit mask for a given command, use this expression: mask = 1 << (tag - 1).

Table 142. Command Bits

Bit	[Others]	[20]	[16]	[15]	[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Command																			

20.5.9.4 Target Version property

The value of this property is a 4-byte structure containing the current version of the extended bootloader.

Table 143. Target Version Property Fields

Bit	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'T' (0x4B)	Major version	Minor version	Bugfix version

Get/Set property definitions are provided in this section.

20.5.10 Serial Boot on USB path

The bootloader supports In-System Programming using the USB peripheral. The target is implemented as USB-HID device classes.

When transfer data through USB-HID device class, USB-HID does not use framing packets. Instead, the packetization, inherent in the USB protocol itself is used. The ability for the device to NAK Out transfers (until they can be received) provides the required flow control. The built-in CRC of each USB packet provides the required error detection

20.5.10.1 Device descriptor

The bootloader configures the default USB VID/PID/Strings as below:

Default VID/PID:

- VID = 0x1FC9.
- PID = 0x0155.

Default Strings:

- Manufacturer [1] = "NXP SEMICONDUCTOR INC".
- Product [2] = "USB COMPOSITE DEVICE".

The USB VID, PID, and Strings can be customized using the CMPA of the flash. For example, the USB VID and PID can be customized by writing the new VID to the usbVid field and the new PID to the usbPid field of the CMPA in flash.

20.5.10.2 Endpoints

The HID peripheral uses three endpoints:

- Control (0)
- Interrupt IN (1)
- Interrupt OUT (2)

The Interrupt OUT endpoint is optional for HID class devices, but the MCU bootloader uses it as a pipe, where the firmware can NAK send requests from the USB host.

20.5.10.3 HID Reports

There are four HID reports defined and used by the bootloader USB HID peripheral. The report ID determines the direction and type of packet sent in the report; otherwise, the contents of all reports are the same.

Table 144. HID reports assigned for the bootloader

Report ID	Packet Type	Direction
1	Command	OUT
2	Data	OUT
3	Command	IN
4	Data	IN

Each report has a maximum size of 60 bytes. The maximum payload size is 56 bytes. In addition, there is a 4-byte report header that indicates the length (in bytes) of the payload and report id sent the packet.

Note: In the future, the maximum report size may be increased, to support transfers of larger packets. Alternatively, additional reports may be added with larger maximum sizes.

The actual data sent in all of the reports looks like:

Table 145. Data format sent in USB HID packet

0	Report ID
1	Padding
2	Packet Length LSB
3	Packet Length MSB
4	Packet[0]
5	Packet[1]
6	Packet[2]
	...
N+4-1	Packet[N-1]

This data includes the Report ID, which is required if more than one report is defined in the HID report descriptor. The actual data sent and received has a maximum length of 35 bytes. The Packet Length header is written in little-endian format, and it is set to the size

(in bytes) of the packet sent in the report. This size does not include the Report ID or the Packet Length header itself. During a data phase, a packet size of 0 indicates a data phase abort request from the receiver.

20.5.11 Serial Boot on LPUART path

The bootloader integrates an autobaud detection algorithm for the UART peripheral, thereby providing flexible baud rate choices.

Autobaud feature: If UART*n* is used to connect to the bootloader, then the UART*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 UART*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 bootloader then enters a loop, waiting for bootloader commands via the UART peripheral.

NOTE

The data bytes of the ping packet must be sent continuously (with no more than 80 ms between bytes) in a fixed UART 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 ,230400 and 460800.

Packet transfer: After autobaud detection succeeds, bootloader communications can take place over the UART 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.

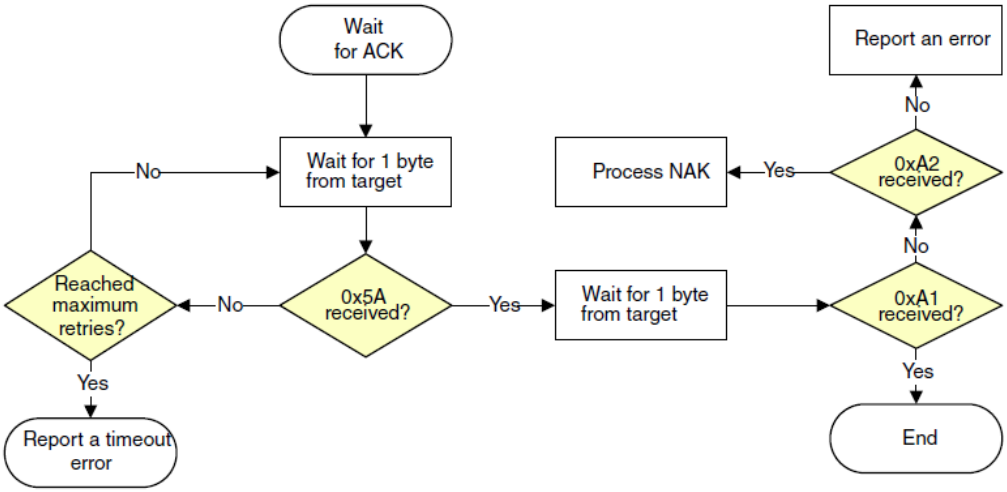


Figure 4-21 Host Reads an ACK from Target via LPUART

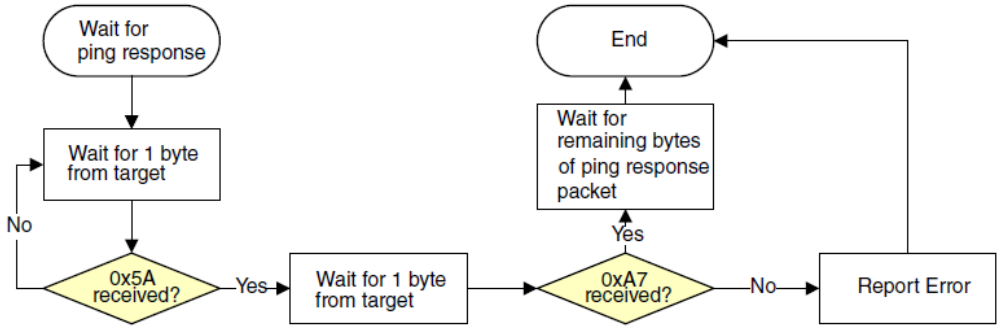


Figure 4-22 Host Reads a Ping Response from Target via LPUART

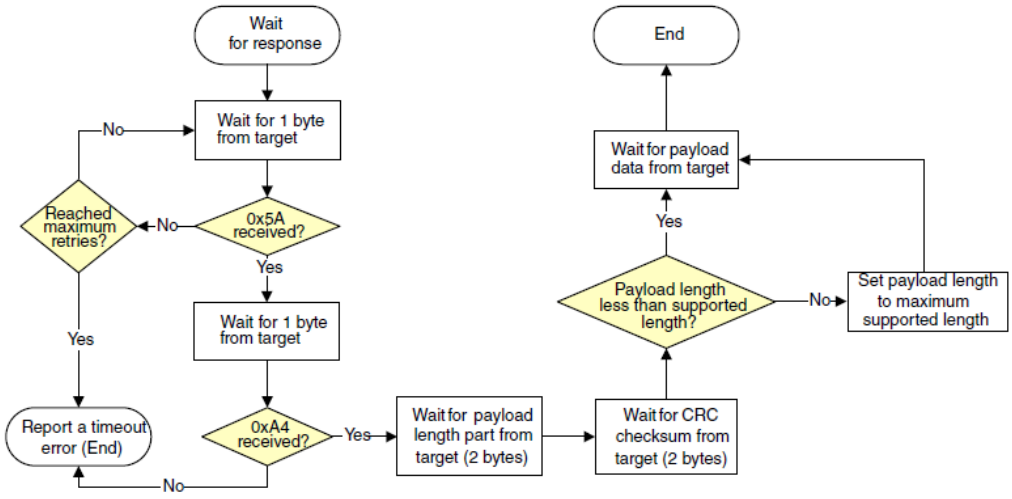


Figure 4-23 Host Reads a Command Response from Target vis LPUART

20.5.12 Extended Bootloader API

20.5.13 Bootloader Status Error Codes

This section describes the status error codes that the Bootloader returns to the host.

Table 146. Bootloader Status Error Codes, Sorted by Value

Error Code	Value	Description
kStatus_Success	0	Operation succeeded without error.
kStatus_Success	0	Operation succeeded without error.
kStatus_Fail	1	The operation failed with a generic error.
kStatus_ReadOnly	2	Requested value cannot be changed because it is read-only.
kStatus_OutOfRange	3	Requested value is out of range.
kStatus_InvalidArgument	4	The requested command's argument is undefined.
kStatus_Timeout	5	A timeout occurred.
kStatus_NoTransferInProgress	6	No send in progress
kStatus_FLASH_Success	0	API is executed successfully
kStatus_FLASH_InvalidArgument	4	An invalid argument is provided
kStatus_FlashSizeError	100	Not used.

Table continues on the next page...

Table 146. Bootloader Status Error Codes, Sorted by Value (continued)

kStatus_FlashAlignmentError	101	Address or length does not meet the required alignment.
kStatus_FlashAddressError	102	Address or length is outside addressable memory.
kStatus_FlashAccessError	103	The FTFA_FSTAT[ACCERR] bit is set.
kStatus_FlashProtectionViolation	104	The FTFA_FSTAT[FPVIOL] bit is set.
kStatus_FlashCommandFailure	105	The FTFA_FSTAT[MGSTAT0] bit is set.
kStatus_FlashUnknownProperty	106	Unknown Flash property.
kStatus_FlashEraseKeyError	107	The key provided does not match the programmed flash key.
kStatus_FlashRegionExecuteOnly	108	The area of flash is protected as execute-only.
kStatus_FLASH_ExecuteInRamFunctionNotReady	109	Execute-in-RAM function is not available
kStatus_FLASH_CommandNotSupported	111	Flash API is not supported
kStatus_FLASH_ReadOnlyProperty	112	The flash property is read-only
kStatus_FLASH_InvalidPropertyValue	113	The flash property value is out of range
kStatus_FLASH_InvalidSpeculationOption	114	The option of flash prefetch speculation is invalid
kStatus_FLASH_EccError	116	A correctable or uncorrectable error during command execution
kStatus_FLASH_CompareError	117	Destination and source memory contents do not match
kStatus_FLASH_RegulationLoss	118	A loss of regulation during read
kStatus_FLASH_InvalidWaitStateCycles	119	The wait state cycle set to r/w mode is invalid
kStatus_FLASH_OutOfDateCfpaPage	132	CFPA page version is out of date
kStatus_FLASH_BlankIfrPageData	133	Blank page cannnot be read
kStatus_FLASH_EncryptedRegionsEraseNotDoneAtOnce	134	Encrypted flash subregions are not erased at once
kStatus_FLASH_ProgramVerificationNotAllowed	135	Program verification is not allowed when the encryption is enabled
kStatus_FLASH_HashCheckError	136	Hash check of page data is failed

Table continues on the next page...

Table 146. Bootloader Status Error Codes, Sorted by Value (continued)

kStatus_FLASH_SealedFfrRegion	137	The FFR region is sealed
kStatus_FLASH_FfrRegionWriteBroken	138	The FFR Spec region is not allowed to be written discontinuously
kStatus_FLASH_NmpaAccessNotAllowed	139	The NMPA region is not allowed to be read/ written/erased
kStatus_FLASH_CmpaCfgDirectEraseNotAllowed	140	The CMPA Cfg region is not allowed to be erased directly
kStatus_FLASH_FfrBankIsLocked	141	The FFR bank region is locked
kStatus_FLASH_CfpaScratchPageInvalid	148	CFPA Scratch Page is invalid
kStatus_FLASH_CfpaVersionRollbackDisallowed	149	CFPA version rollback is not allowed
kStatus_FLASH_ReadHidingAreaDisallowed	150	Flash hiding read is not allowed
kStatus_FLASH_ModifyProtectedAreaDisallowed	151	Flash firewall page locked erase and program are not allowed
kStatus_FLASH_CommandOperationInProgress	152	The flash state is busy, indicate that a flash command in progress.
kStatus_UnknownCommand	10000	this command is not recognized
kStatus_SecurityViolation	10001	Security Violation happened when receiving disallowed commands
kStatus_AbortDataPhase	10002	Sender requested data phase abort
kStatus_Ping	10003	Ping Command Received from the host
kStatus_NoResponse	10004	No Response from the host
kStatus_NoResponseExpected	10005	Expected No Response from the host
kStatus_CommandUnsupported	10006	Unsupported command was received
kStatusRomLdrSectionOverrun	10100	means reached the end of the sb file processing
kStatusRomLdrSignature	10101	the signature or version are incorrect
kStatusRomLdrSectionLength	10102	the bootOffset/ new section count is out of range
kStatusRomLdrUnencryptedOnly	10103	the unencrypted image is disabled
kStatusRomLdrEOFReached	10104	the end of the image file is reached

Table continues on the next page...

Table 146. Bootloader Status Error Codes, Sorted by Value (continued)

kStatusRomLdrChecksum	10105	The checksum of a command tag block is invalid.
kStatusRomLdrCrc32Error	10106	The CRC-32 of the data for a load command is incorrect.
kStatusRomLdrUnknownCommand	10107	An unknown command was found in the SB file.
kStatusRomLdrIdNotFound	10108	There was no bootable section found in the SB file.
kStatusRomLdrDataUnderrun	10109	The SB state machine is waiting for more data.
kStatusRomLdrJumpReturned	10110	The function that was jumped to by the SB file has returned.
kStatusRomLdrCallFailed	10111	The call command in the SB file failed.
kStatusRomLdrKeyNotFound	10112	A matching key was not found in the SB file's key dictionary to unencrypt the section.
kStatusRomLdrSecureOnly	10113	The SB file sent is unencrypted, and security on the target is enabled.
kStatusRomLdrResetReturned	10114	The SB file reset operation has unexpectedly returned.
kStatusRomLdrRollbackBlocked	10115	An Image version rollback event detected
kStatusRomLdrInvalidSectionMacCount	10116	Invalid Section MAC count detected in the SB file
kStatusRomLdrUnexpectedCommand	10117	The command tag in the sb-file is unexpected
kStatusRomLdrBadSBKEK	10118	Bad SBKEK detected
kStatusRomLdrPendingJumpCommand	10119	Represent the jump command is pending and the actual jump is implemented in sbloader_finalize()
kStatusMemoryRangeInvalid	10200	The requested address range does not match an entry, or the length extends past the matching entry's end address.
kStatusMemoryReadFailed	10201	Memory Read Failed
kStatusMemoryWriteFailed	10202	Memory Write failed

Table continues on the next page...

Table 146. Bootloader Status Error Codes, Sorted by Value (continued)

kStatusMemoryCumulativeWrite	10203	Cumulative Write happened due to write to un-erased FLASH region
kStatusMemoryNotConfigured	10205	Memory is not configured yet before access
kStatusMemoryAlignmentError	10206	Alignment Error happened during access to memory
kStatusMemoryVerifyFailed	10207	Verifying operation failed after erasing/ programming FLASH
kStatusMemoryWriteProtected	10208	The memory to be written is protected
kStatusMemoryAddressError	10209	The Memory address is invalid/wrong
kStatusMemoryBlankCheckFailed	10210	Check the blank memory failled
kStatusMemoryBlankPageReadDisallowed	10211	The memory is blank, and read command is disallowed
kStatusMemoryProtectedPageReadDisallowed	10212	The memory is protected, and read command is disallowed
kStatusMemoryFfrSpecRegionWriteBroken	10213	The write operation to the FFR Region was broken
kStatusMemoryUnsupportedCommand	10214	The memory command is not supported
kStatus_UnknownProperty	10300	The requested property value is undefined.
kStatus_ReadOnlyProperty	10301	The requested property value cannot be written.
kStatus_InvalidPropertyValue	10302	The specified property value is invalid.
kStatus_AppCrcCheckPassed	10400	CRC check is valid and passed.
kStatus_AppCrcCheckFailed	10401	CRC check is valid but failed.
kStatus_AppCrcCheckInactive	10402	CRC check is inactive.
kStatus_AppCrcCheckInvalid	10403	CRC check is invalid because the BCA is invalid, or the CRC parameters are unset (all 0xFF bytes).
kStatus_AppCrcCheckOutOfRange	10404	CRC check is valid, but addresses are out of range.
kStatus_RomApiExecuteCompleted	0	ROM successfully process the whole sb file/ boot image

Table continues on the next page...

Table 146. Bootloader Status Error Codes, Sorted by Value (continued)

kStatus_RomApiNeedMoreData	10801	ROM needs more data to continue processing the boot image
kStatus_RomApiBufferSizeNotEnough	10802	The user buffer is not enough for use by Kboot during execution of the operation
kStatus_RomApiInvalidBuffer	10803	The user buffer is not ok for sbloader or authentication
kStatus_FLEXSPI_SequenceExecutionTimeout	6000	The FLEXSPI Sequence Execution timeout
kStatus_FLEXSPI_InvalidSequence	6001	The FLEXSPI LUT sequence invalid
kStatus_FLEXSPI_DeviceTimeout	6002	The FLEXSPI device timeout
kStatus_FLEXSPINOR_ProgramFail	20100	Page programming failure
kStatus_FLEXSPINOR_EraseSectorFail	20101	Sector Erase failure
kStatus_FLEXSPINOR_EraseAllFail	20102	Chip Erase failure
kStatus_FLEXSPINOR_WaitTimeout	20103	The execution time out
kStatus_FlexSPINOR_NotSupported	20104	Page size overflow
kStatus_FlexSPINOR_WriteAlignmentError	20105	Address alignment error
kStatus_FlexSPINOR_CommandFailure	20106	Erase/Program Verify Error
kStatus_FlexSPINOR_SFDP_NotFound	20107	Timeout occurs during the API call
kStatus_FLEXSPINOR_Unsupported_SFDP_Version	20108	Unrecognized SFDP version
kStatus_FLEXSPINOR_FLASH_NotFound	20109	Flash detection failure
kStatus_FLEXSPINOR_DTRRead_DummyProbeFailed	20110	DDR Read dummy probe failure
kStatus_IAP_Success	0	IAP API execution succedded
kStatus_IAP_Fail	1	IAP API exeution failed
kStatus_IAP_InvalidArgument	100001	Invalid argument detected during API execution
kStatus_IAP_OutOfMemory	100002	The Heap size is not enough during API execution

Table continues on the next page...

Table 146. Bootloader Status Error Codes, Sorted by Value (continued)

kStatus_IAP_ReadDisallowed	100003	The read memory operation is disallowed during API execution
kStatus_IAP_CumulativeWrite	100004	The FLASH region to be programmed is not empty
kStatus_IAP_EraseFailuire	100005	Erase operation failed
kStatus_IAP_CommandNotSupported	100006	The specific command is not supported
kStatus_IAP_MemoryAccessDisabled	100007	Memory access is disabled, typically occurred on the FLEXSPI NOR if it is not configured properly

Chapter 21
Clocking

21.1 Introduction

The main component 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.

The SCG controls the on chip clock sources, including 40MHz OSC, FRO192M and FRO12M. It also generates the clock sources for peripherals functional clock. A clock mux is implemented in SCG-Lite to select the clock source for main_clk.

The MAIN_CLK can be sourced directly from the following:

- FRO192M (It has a 48 MHz output, which is used as USB clock):
- FRO12M
- FRO16k
- 50 MHz OSC-SYS

Auto-trimming function is implemented for FRO192M. The trim source includes USB FS, SOF, SOSC. When auto-trimming is enabled, FRO192M can achieve +-0.25% accuracy. Clock monitor is implemented for SOSC. When enabled, SOSC clock is monitored by FRO12M. 1 MHz clock is derived from FRO12M, and is always on in active and sleep mode.

21.2 Configure main clock and system clock

The clock source for the registers and memories is derived from main clock. The main clock can be selected from the sources listed in step 1 below.

The CPU_CLK derived from MAIN_CLK, also called the system clock, clocks the core, the memories, and the peripherals (register interfaces and peripheral clocks).

1. Select the main clock in the SCG_RCCR register. The following options are available:
   - FRO 12 MHz output (FRO_12M) from internal oscillator.
   - FRO 16 kHz output (FRO_16k)
   - FRO high speed output (fro_hf) from FRO192M. By default, it is 48 MHz. fro_hf is the default main clock.
   - 40 MHz.
   - SYS-OSC.
2. Select the divider value for the system clock AHBCLKDIV register in the System Controller module.
3. Enable the clock for the memories and peripherals used in the application.
4. Peripheral clock must be disabled (MRCC_GLB_CC0 registers in SYSCON module when performing software reset on the peripherals via (MRCC_GLB_RST0 registers in SYSCON). Once reset assertion and de-assertion are completed, the respective peripheral clock can be enabled.

21.3 Clock generation

The system control block facilitates the clock generation. Many clocking variations are possible. Module clocking gives an overview of potential clock options. Table 147 describes signals on the clocking diagram. The maximum clock frequency is 96 MHz.

NOTE

The indicated clock multiplexers shown in Module clocking are synchronized. In order to operate, the currently selected clock must be running, and the clock to be switched to must also be running so the multiplexer can gracefully switch between the two clocks without glitches. Other clock multiplexers are not synchronized. The output divider can be stopped and restarted gracefully during switching if a glitch-free output is needed.

The FRO_12M provides a 1 MHz clock (clk_1m). It is used as Watchdog 0 independent clock source. The accuracy of this clock is limited to a couple percent over temperature (see Data sheet for specification), voltage, and silicon processing variations after trimming made during assembly. To determine the actual Watchdog oscillator output, use the frequency measure block. See Chip-specific Frequency Measurement information.

NOTE

The maximum allowed frequency for the main clock and system clock (to CPU0, AHB bus, Sync, etc.,) is 96 MHz.

Table 147. Clocking diagram signal name descriptions

Name[1]	Description
clk_16k[0]/clk_16k[1]	It is the 16.384 KHz clock output from FRO16K. It is the clock of peripherals in VSYS domain.
clk_in	It is the internal clock that comes from the external oscillator.
fro_hf	It is the clock output from FRO192M. The frequency is controlled by SCG_FIRCCFG[FREQ_SEL].
fro_hf_div	It is the clock divided by fro_hf
clk_48M	It is the 48MHz clock output from FRO192M
fro_12M	It is the 12MHz clock output from FRO12M
clk_1M	It is the 1 MHz clock output from FRO12M
main_clk	It is main clock used by the CPU, AHB bus, APB bus, IPS bus, and some peripherals. The main clock and its source selection are shown in Figure 32.
CPU_CLK	It is the clock of CPU.
SYSTEM_CLK	It is the clock of AHB bus, APB bus, IPS bus. The frequency is same with CPU_CLK
SLOW_CLK	It is the SYSTEM_CLK divided by 4

1. See the clock frequency table for max frequencies of each of these clocks.

Table 148. Clock frequency table

Clock Name	Max. Clock Frequency	
	MD mode	SD mode
clk_16k[0]/ clk_16k[1]	16.384 KHz	16.384 KHz
clk_in	SOSC OFF 50 MHz bypass mode	50 MHz
fro_hf	96 MHz	192 MHz
fro_hf_div	48 MHz	96 MHz

Table continues on the next page...

Table 148. Clock frequency table (continued)

Clock Name	Max. Clock Frequency	
	MD mode	SD mode
clk_48M	48 MHz	48 MHz
fro_12M	12 MHz	12 MHz
clk_1M	1 MHz	1 MHz
main_clk	96 MHz	192 MHz
CPU_CLK	48 MHz	96 MHz
SYSTEM_CLK	48 MHz	96 MHz
SLOW_CLK	12 MHz	24 MHz

21.3.1 Peripheral bus clock

The following peripherals use SLOW_CLK as APB/IPS clock. All other peripherals use SYSTEM_CLK as APB/IPS clock.

- LPTMR0
- CMP0
- Wake Timer
- ATX0
- AOI
- CMC
- SCG-Lite
- SPC
- WUU
- FMU

21.3.2 Peripheral clock summary

Modules	Max Functional Clock in MD mode	Max Functional Clock in SD mode
CPU and system modules		
Systick Clock	12MHz	24MHz
WWDT0 Clock	1MHz	1MHz
Trace Clock	48MHz	96MHz
Communication Modules		
FlexCAN Clock	48MHz	96MHz
FlexIO Clock	96MHz	192MHz
LPUART0~4 Clock	48MHz	96MHz
LPSPI0~1 Clock	48MHz	96MHz

Table continues on the next page...

Table continued from the previous page...

LPI2C0~3 Clock	24MHz	48MHz
I3C0 Clock	25MHz	25MHz
USB FS Clock	48MHz	48MHz
Timer Modules		
LPTMR0 Clock	12MHz	24MHz
PWM0	96MHz	192MHz
Quad Decoder	48MHz	96MHz
Ctimer 0~4	96MHz	192MHz
Ostimer Clock	1MHz	1MHz
uTimer	1MHz	1MHz
Analog Modules		
ADC0/1	24MHz	64MHz
CMP0/1	24MHz	24MHz
DAC0	24MHz	48MHz
MISC Modules		
Wake Timer	16.384KHz	16.384KHz
VBAT Clock	16.384KHz	16.384KHz

NOTE

It is the max. clock frequency of the modules in clock diagrams under different run mode

21.3.3 Module clocking

The following figures show the clock generation diagrams of the device's modules.

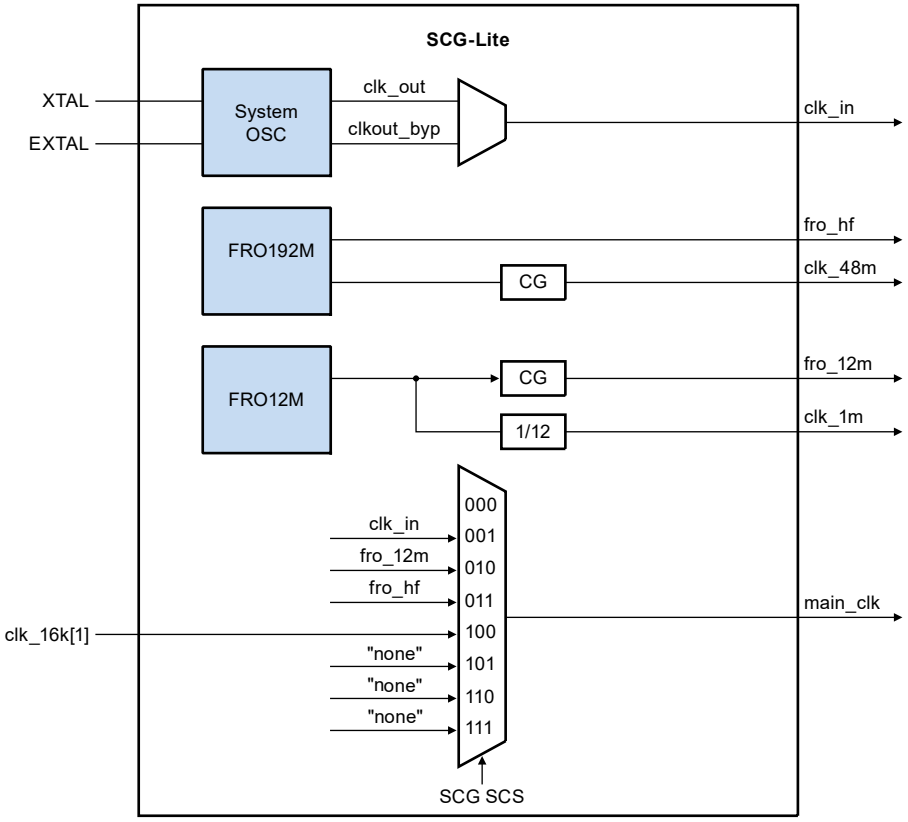


Figure 30. SCG clock

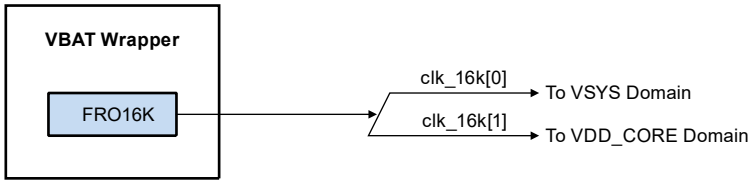


Figure 31. VBAT clock

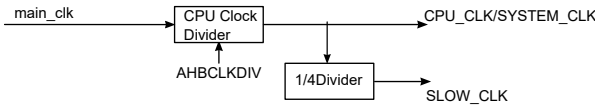


Figure 32. Main CPU clock

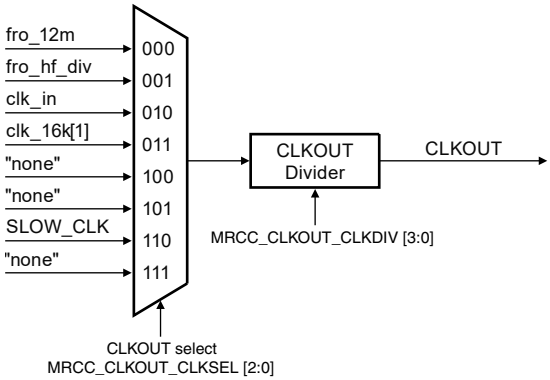


Figure 33. Clock Divider clock

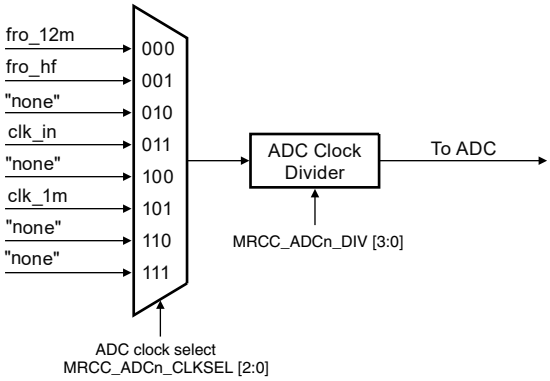


Figure 34. ADC clock

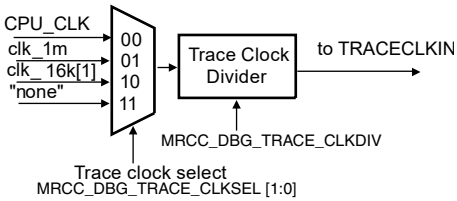


Figure 35. Trace clock

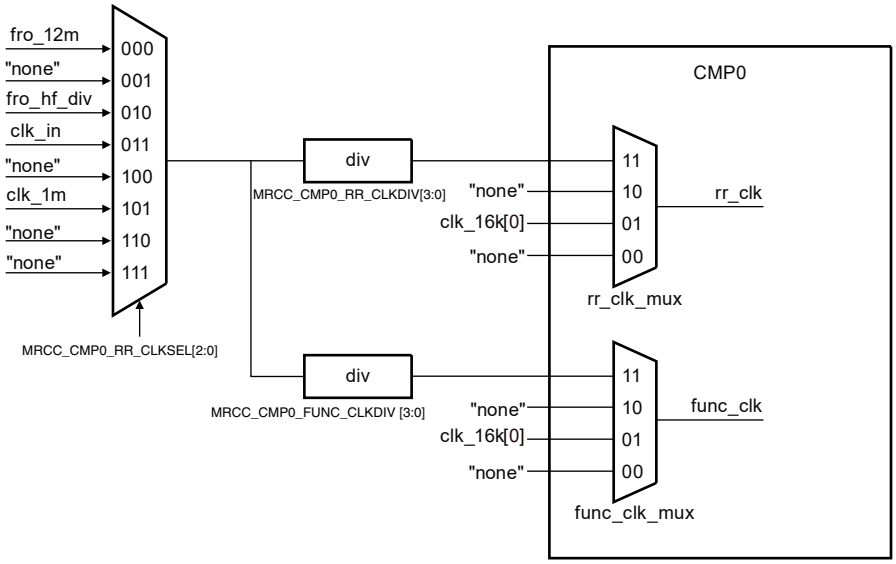


Figure 36. CMP0 clock

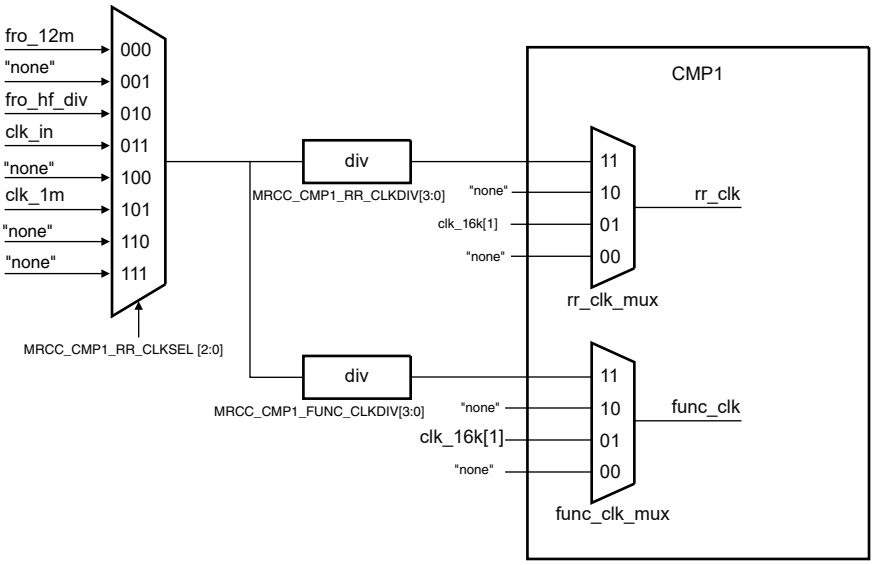


Figure 37. CMP1 clock

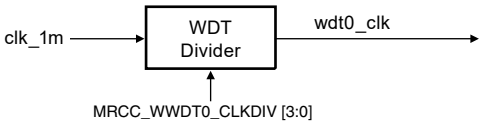


Figure 38. WWDT clock

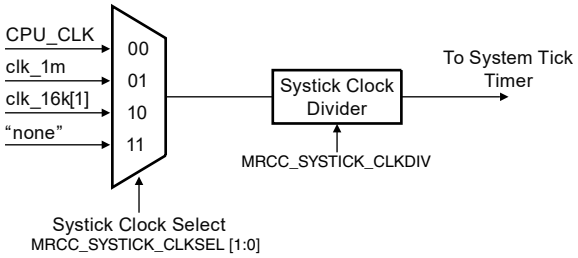


Figure 39. Systick clock

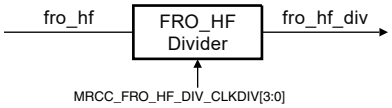


Figure 40. FRO HF clock

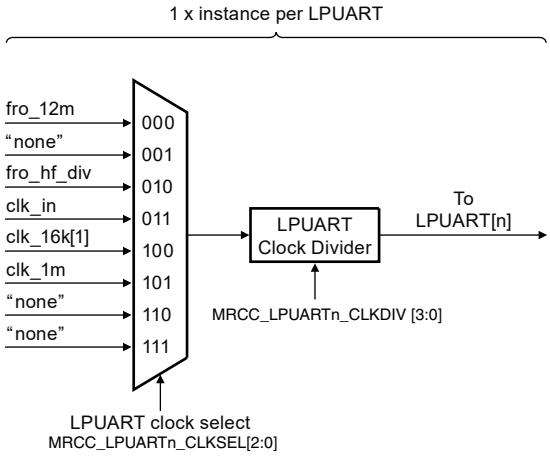


Figure 41. LPUART clock

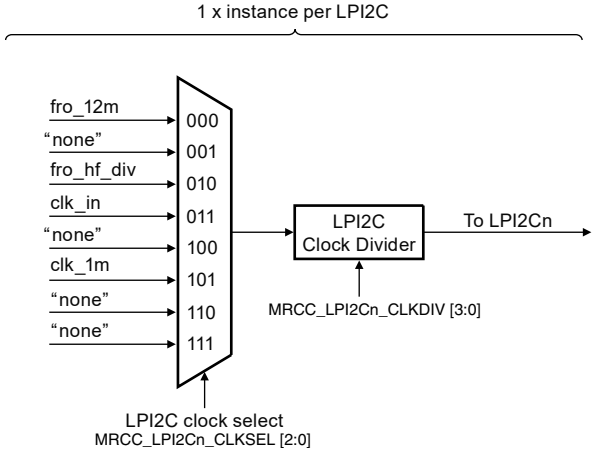


Figure 42. LPI2C clock

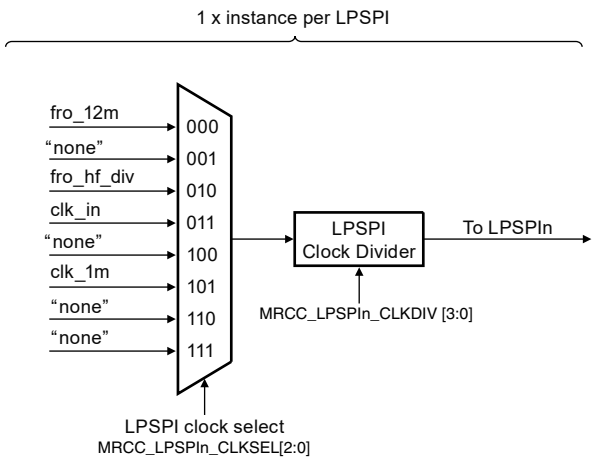


Figure 43. LPSPI clock

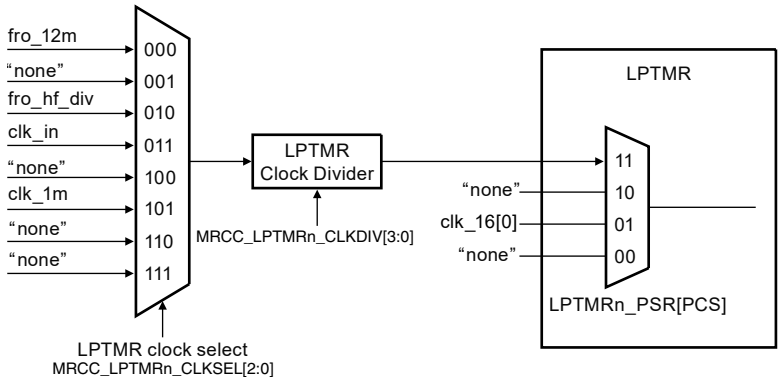


Figure 44. LPTMR clock

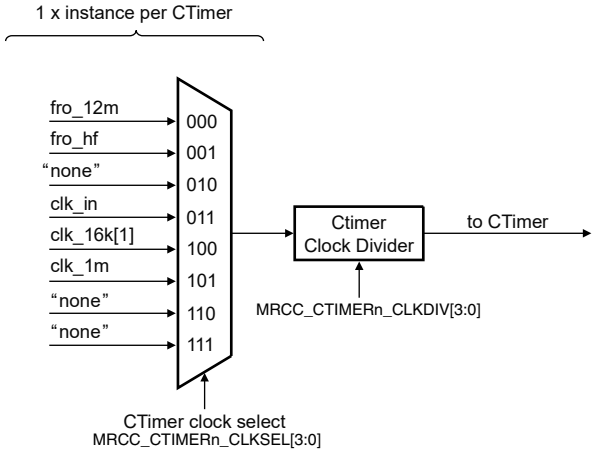


Figure 45. CTimer clock

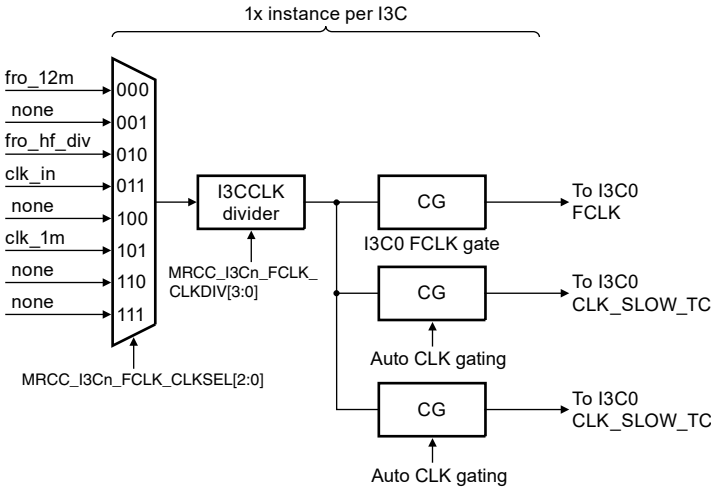


Figure 46. I3C clock

Figure 47. WUU clock

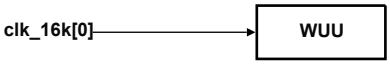


Figure 48. eFlexPWM clock

Figure 49. UTICK clock

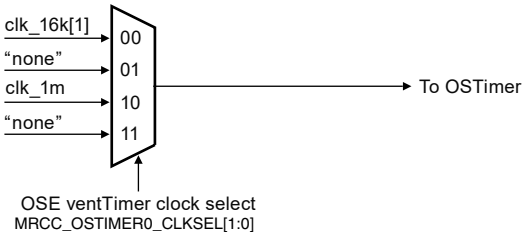


Figure 50. OStimer clock

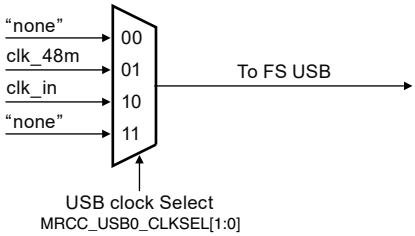


Figure 51. USB FS clock

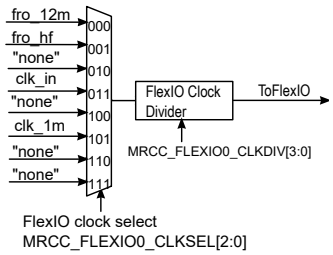


Figure 52. FlexIO clock

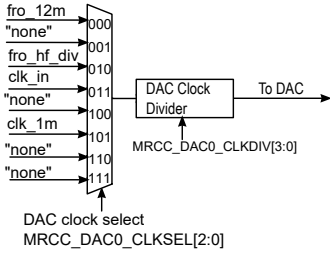


Figure 53. DAC clock

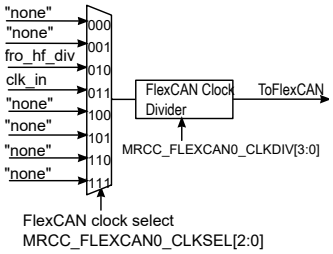


Figure 54. FlexCAN clock

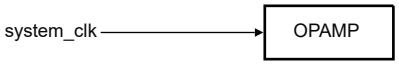


Figure 55. OPAMP clock

Chapter 22
System Clock Generator (SCG)

22.1 Chip-specific SCG information

Table 149. Reference links to related information

Topic	Related module	Reference
Full description	SCG	SCG
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

22.1.1 Module instances

This device contains one instance of the SCG module, SCG0.

22.2 Overview

The System Clock Generator (SCG) module provides the main clock and other peripheral clocks for the MCU. The SCG takes clock inputs from a variety of sources and generates the clocks that the MCU requires.

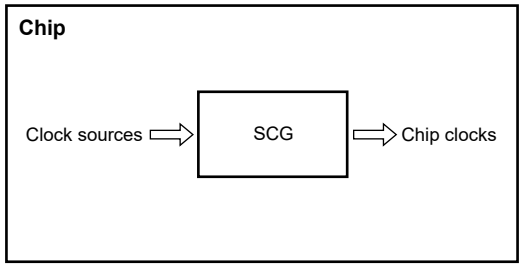


Figure 56. System Clock Generator (SCG) in MCU

The clock source inputs available for the SCG are as follows:

- System oscillator clock (SOSC)
- Slow internal reference clock (SIRC)
- Fast internal reference clock (FIRC)
- Real-Time oscillator clock (ROSC)

22.2.1 Block diagram

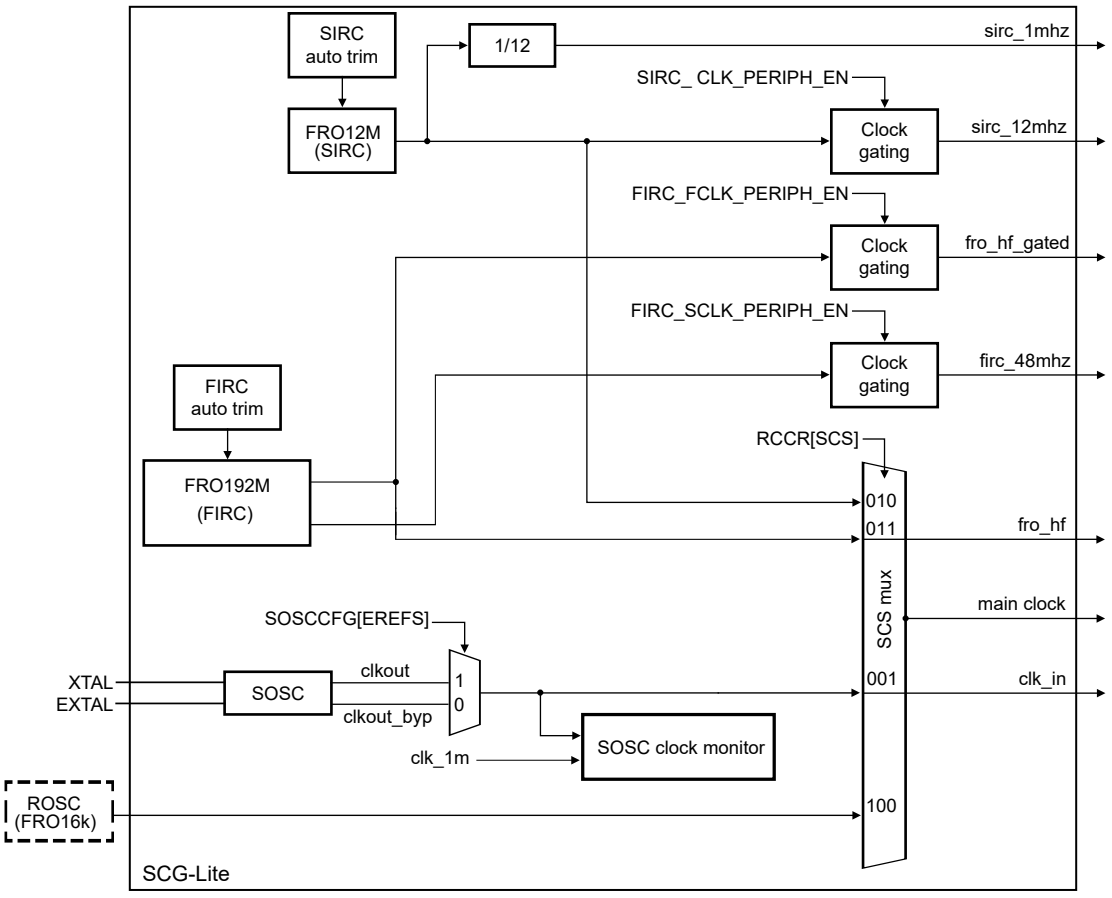


Figure 57. Block diagram

22.2.2 Features

Key features of the SCG module are:

- 2 Internal Reference Clock (IRC) generators
  - FIRC with trim capability
  - FIRC can output 192/96/64/48 MHz clock by register configuration
  - FIRC can limit output frequency by frequency phantom
  - FIRC can output 48 MHz clock continuously
  - SIRC with trim capability
  - SIRC can output 12 MHz clock and 1 MHz clock divided from it
  - Either of the IRC generators can be selected as the clock source for the system clocks
- System Crystal Oscillator Clock (SOSC)
  - Can be selected as the clock source for system clocks
- Real Time Oscillator Clock (ROSC)
  - Can be selected as the clock source for the device. See the Clocking chapter to ascertain which signal the ROSC input maps to.

- Clock monitor with reset and interrupt request capability for SOSC clocks

22.3 Functional description

22.3.1 SCG clock mode transitions

The system boots up from the FIRC source. The system clock can be switched among the following clock sources: FIRC, SIRC, SOSC, ROSC,.

The SCG restricts programming into invalid clock modes, and ignores writes to the System Clock Source (SCS) bits. When a transition between run modes or a transition from run mode into wait mode occurs, the SCG completes the switch to the corresponding clock mode as defined in the Run Clock Control Register (RCCR). After that, the system switches to the selected run/wait mode.

The following table describes the modes of operation of the SCG.

Table 150. SCG modes of operation

Mode	Description
SOSC	The SOSC mode is entered when the following occur: • the device is in run mode and 001 is written to RCCR[SCS] • SOSCCSR[SOSCEN] = 1b • SOSCCSR[SOSCVLD] = 1b In SOSC mode, the system clocks are derived from the external SOSC.
SIRC	The SIRC mode is entered when the following occur: • the device is in run mode and 010 is written to RCCR[SCS] • SIRCCSR[SIRCVLD] = 1b In SIRC mode, the system clocks are derived from SIRC.
FIRC	The FIRC mode is the default clock mode of operation or it is entered when the following occur: • the device is in run mode and 011 is written to RCCR[SCS] • FIRCCSR[FIRCEN] = 1b • FIRCCSR[FIRCVLD] = 1b In FIRC mode, the system clocks are derived from FIRC. Eight frequency settings are available for FIRC clock as described in FIRCCFG[FREQ_SEL]. The changes to FIRC frequency settings are ignored when the FIRC clock is enabled.
ROSC	The ROSC mode is entered when the following occur: • the device is in run mode and 100 is written to RCCR[SCS] • ROSCCSR[ROSCVLD] = 1b In ROSC mode, the systems clocks are derived from the external ROSC.
Stop	The Stop mode is entered whenever the MCU enters a Stop state. Power modes are chip-specific. For power mode assignments, see Power Management chapter that describes module operation in low-power modes. Entering Stop mode, all SCG clock signals are static, including SCG system clocks.

Table continues on the next page...

Table 150. SCG modes of operation (continued)

Mode	Description
	There are some exceptions. The following clocks can continue to run and stay enabled when all the specific conditions are true. SIRC is available in Deep Sleep mode when all the following conditions are true: • SIRCCSR[SIRCSTEN] = 1b FIRC is available in Deep Sleep mode when all the following conditions are true: • FIRCCSR[FIRCEN] = 1b • FIRCCSR[FIRCSTEN] = 1b SOSC is available in Deep Sleep mode when all the following conditions are true. • SOSCCSR[SOSCEN] = 1b • SOSCCSR[SOSCSTEN] = 1b **NOTE** If SPLL is not required in Deep Sleep mode then set SPLLCSR[SPLLPWREN] = 0b before entering to Deep Sleep mode and once chip is waked up and LDOCSR[VOUT_OK] = 1b then set SPLLCSR[SPLLPWREN] = 1b to enable SPLL.

22.3.2 SCG SOSC operation

The following table lists the SOSC configurations and SOSC output clock.

Table 151. SOSC mode operation

SOSCEN	EREFS	SOSC clock
0	0	Off
1	0	EXTAL Pad Clock
1	1	Crystal Oscillator Clock

22.3.3 SCG LOC operation

The SCG supports the loss of clock (LOC) monitoring for the external reference clocks including SOSC reference clock. To enable the LOC monitor, the CME bit for the clock must be programmed to 1. To monitor the external reference clock, the SCG uses an internal 31.25 KHz reference clock derived by dividing the SIRC internal clock. So prior to enabling the LOC for the external reference clock, it is required that the SIRC is programmed to be enabled.

The LOC monitors, when enabled, generate a system reset or system interrupt request depending on the settings of the CMRE (see the CMRE description for each of the clocks for further information). The frequency conditions for the external reference clocks that trigger a LOC event are listed below:

NOTE

F_{int} = SIRC divided down to 31.25 kHz

- SOSC loss of external clock minimum frequency
  - When SOSC clock frequency is above (16/5) F_{int}, the chip is never reset or generate an interrupt.

— When SOSC clock frequency is between (15/5) F_{int} and (16/5) F_{int}, the chip might reset or generate an interrupt (depending on the phase dependency between SOSC clock and reference clock)

— When SOSC clock frequency is below (15/5) F_{int}, the chip always resets or generates an interrupt.

22.3.4 Clocking

The SCG module has the following input clocks:

- Bus clock for register access.
- ROSC, clocks as one of the system clock sources

The SCG module has the following output clocks:

- Main clock
- FIRC clock (192/96/64/48 MHz)
- SIRC clock (12 MHz and 1 MHz divided from it)
- SOSC clock

22.3.5 Resets

These are the SCG resets:

1. The global reset is fed into this module. The entire module is reset by the global reset.
2. The clock monitor generates a reset in case of the following:
   - SOSCCSR[SOSCCM] = 1 & SOSCCSR[SOSCCMRE] = 1 & SOSCCSR[SOSCERR] = 1

On the assertion of any reset source, the FIRC 48 MHz clock is enabled or starts up if not running.

22.3.6 Interrupts

These are the SCG interrupts:

- Clock monitor generates interrupt when any of below happens:
  — SOSCCSR[SOSCCM] = 1 & SOSCCSR[SOSCCMRE] = 0 & SOSCCSR[SOSCERR] = 1
- Interrupt can also be generated when any of below happens:
  — SOSC clock source is not valid. SOSCCSR[SOSCVLD_IE] = 1 & SOSCCSR[SOSCVLD] = 1
  — FIRC clock source is not accurate. FIRCCSR[FIRCACC_IE] = 1 & FIRCCSR[FIRCACC] = 1
  — FIRC trimming error detected. FIRCCSR[FIRCERR_IE] = 1 & FIRCCSR[FIRCERR] = 1
  — SIRC trimming error detected. SIRCCSR[SIRCERR_IE] = 1 & SIRCCSR[SIRCERR] = 1

22.4 External signals

Table 152. External signal descriptions

Signal	Description	Mode	I/O
XTAL	SOSC Crystal output pin	SOSC	O
EXTAL	SOSC Crystal input pin	SOSC	I
CLKOUT	SCG clock output pin	SCG	O

22.5 Initialization

Out of reset, the FIRC is default enabled. FIRCCFG[FREQ_SEL] is set to 001b, and RCCR[SCS] is set to 011b, so the FRO_HF clock which is set to 48 MHz would be as the system clock source.

Out of reset, the SIRC is default enabled.

Out of reset, all the other clock sources are disabled.

22.6 Application information

22.6.1 SOSC configuration example

1. Write 1 to SOSCCFG[EREFS] to select SOSC source (internal crystal oscillator)
2. Write 0 to SOSCCFG[RANGE] to configure SOSC range
3. Write 0 to SOSCCSR[LK] to unlock SOSCCSR
4. Write 1 to SOSCCSR[SOSCCM] to enable SOSC clock monitor
5. Write 1 to SOSCCSR[SOSCSTEN] to enable SOSC in Deep Sleep mode if needed
6. Write 1 to SOSCCSR[SOSCEN] to enable SOSC
7. Read SOSCCSR[SOSCVLD] until it returns 1, indicating SOSC is valid
8. Read SOSCCSR[SOSCERR] to ensure it returns 0
9. Write 1 to RCCR[SCS] to switch main clock to SOSC
10. Read CSR[SCS] until it returns 1, indicating the switch is complete

22.6.2 SIRC configuration example

22.6.2.1 SIRC normal configuration example

1. Write 0 to SIRCCSR[LK] to unlock SIRCCSR
2. Write 1 to SIRCCSR[SIRCSTEN] to enable SIRC in Deep Sleep mode, if needed
3. Write 1 to SIRCCSR[SIRC_CLK_PERIPH_EN] to enable SIRC clock for peripheral use
4. Read SIRCCSR[SIRCVLD] until it returns 1, indicating SIRC is valid
5. Read SIRCCSR[SIRCERR] to ensure it returns 0
6. Write 2 to RCCR[SCS] to switch main clock to SIRC
7. Read CSR[SCS] until it returns 2, indicating the switch is complete
8. Write 1 to SIRCCSR[LK] to lock SIRCCSR

22.6.2.2 SIRC auto trim example

1. Follow steps 1 through 8 in SOSC configuration example
2. Write 2 to SIRCTCFG[TRIMSRC] to select SOSC as auto trim clock source
3. Write 39 to SIRCTCFG[TRIMDIV] to divide SOSC to 1 MHz
4. Write 0 to SIRCCSR[LK] to unlock SIRCCSR
5. Write 1 to SIRCCSR[SIRCTREN] to enable auto trim
6. Write 1 to SIRCCSR[SIRCTRUP] to enable update
7. Read SIRCCSR[SIRCVLD] until it returns 1, indicating SIRC is valid

8. Read SIRCCSR[SIRCERR] to ensure it returns 0
9. Read SIRCCSR]TRIM_LOCK] until it returns 1.
10. Write 1 to SIRCCSR[LK] to lock SIRCCSR

The auto trim process continues until it is disabled. The SIRCSTAT register shows the locked trim values. If there is an unlock after lock, SIRCCSR[SIRCERR] is set.

22.6.3 FIRC configuration example

22.6.3.1 FIRC normal configuration example

1. Write 111 to FIRCCFG[FREQ_SEL] to select 192 MHz
2. Write 0 to FIRCCSR[LK] to unlock FIRCCSR
3. Write 0 to FIRCCSR[FIRC_FCLK_PERIPH_EN] to disable FIRC_HF clock for peripheral use, by default FIRC_HF is enabled
4. Write 1 to FIRCCSR[FIRC_SCLK_PERIPH_EN] to enable FIRC 48 MHz clock for peripheral use if needed
5. Write 1 to FIRCCSR[FIRCSTEN] to enable FIRC clock source in Deep Sleep mode if needed
6. Read FIRCCSR[FIRCVLD] until it returns 1, indicating FIRC is valid
7. Read FIRCCSR[FIRCERR] to ensure it returns 0
8. Write 3 to RCCR[SCS] to switch main clock to FIRC
9. Read CSR[SCS] until it returns 3, indicating the switch is complete
10. Write 0 to FIRCCSR[FIRCEN] to disable FIRC, by default FIRC is enabled
11. Write 1 to FIRCCSR[LK] to lock FIRCCSR

22.6.3.2 FIRC auto trim example

1. sosc_clk_enable (); // Enable SOSC clock
2. wait_sosc_valid (); // Wait SOSC to be valid
3. Write 2 to FIRCTCFG[TRIMSRC] to select SOSC as auto trim clock source
4. Write 39 to FIRCTCFG[TRIMDIV] to divide SOSC to 1 MHz
5. Write 0 to FIRCCSR[LK] to unlock FIRCCSR
6. Write 1 to FIRCCSR[FIRCTREN] to enable auto trim
7. Write 1 to FIRCCSR[FIRCTRUP] to enable update
8. Read FIRCCSR[FIRCVLD] until it returns 1, indicating FIRC is valid
9. Read FIRCCSR[FIRCERR] to ensure it returns 0
10. Read FIRCCSR[TRIM_LOCK] until it returns 1
11. Write 1 to FIRCCSR[LK] to lock FIRCCSR

The auto trim process continues until it is disabled. The FIRCSTAT register shows the locked trim values. If there is an unlock after lock, FIRCCSR[FIRCERR] is set.

22.6.4 ROSC configuration example

1. Write 0 to ROSCCSR[LK] to unlock ROSCCSR
2. Read ROSCCSR[ROSCVLD] until it returns 1, indicating ROSC is valid
3. Read ROSCCSR[ROSCERR] to ensure it returns 0

4. Write 4 to RCCR[SCS] to switch main clock to ROSC
5. Read CSR[SCS] until it returns 4, indicating the switch is complete
6. Write 1 to ROSCCSR[LK] to lock ROSCCSR

22.7 Memory map and register definition

This section includes information about the memory map and register definitions:

- Accesses on non-implemented registers that are outside the SCG maximum illegal address range generate a transfer error.
- Read accesses on non-implemented registers that are inside the SCG maximum illegal address range do not generate a transfer error.
- Write accesses on non-implemented registers that are inside the SCG maximum illegal address range generate a transfer error.
- Write accesses on read-only registers generate a transfer error.
- Only 32-bit writes are allowed for any writable SCG registers. Writes of 8 or 16 bits result in transfer errors.

22.7.1 SCG register descriptions

22.7.1.1 SCG memory map

SCG0 base address: 4008_F000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0000_0000h
4h	Parameter (PARAM)	32	R	0000_001Eh
8h	Trim Lock (TRIM_LOCK)	32	RW	0000_0000h
10h	Clock Status (CSR)	32	R	0300_0000h
14h	Run Clock Control (RCCR)	32	RW	0300_0000h
100h	SOSC Control Status (SOSCCSR)	32	RW	0000_0000h
108h	SOSC Configuration (SOSCCFG)	32	RW	0000_0000h
200h	SIRC Control Status (SIRCCSR)	32	RW	0100_0020h
20Ch	SIRC Trim Configuration (SIRCTCFG)	32	RW	0000_0000h
210h	SIRC Trim (SIRCTRIM)	32	RW	See section
218h	SIRC Auto-trimming Status (SIRCSTAT)	32	RW	See section
300h	FIRC Control Status (FIRCCSR)	32	RW	See section
308h	FIRC Configuration (FIRCCFG)	32	RW	0000_0003h
30Ch	FIRC Trim Configuration (FIRCTCFG)	32	RW	0000_0000h
310h	FIRC Trim (FIRCTRIM)	32	RW	See section
318h	FIRC Auto-trimming Status (FIRCSTAT)	32	RW	See section
400h	ROSC Control Status (ROSCCSR)	32	RW	0000_0000h

22.7.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

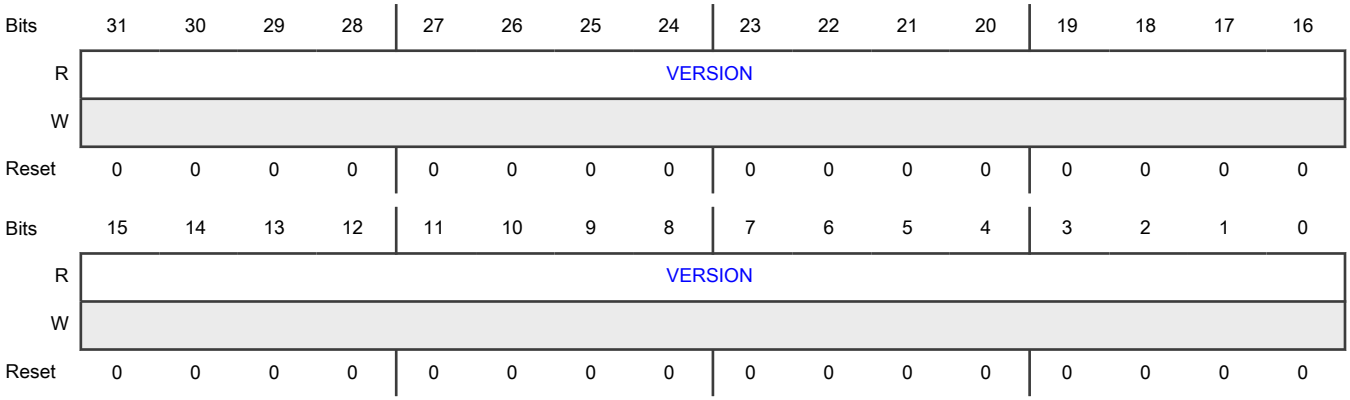
Function

The Version ID register indicates the version integrated for this instance on the device.

NOTE

Writing to this register results in a transfer error.

Diagram



Fields

Field	Function
31-0 VERSION	SCG Version Number This read-only field returns the SCG module version number.

22.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

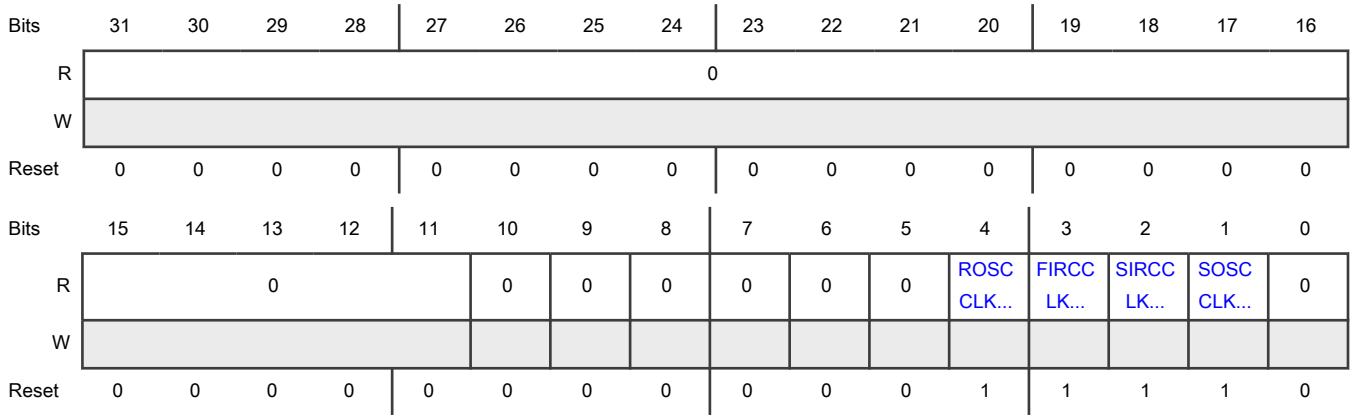
Function

The Parameter register indicates the feature parameters for this instance on the device.

NOTE

Writing to this register results in a transfer error.

Diagram



Fields

Field	Function
31-11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 ROSCCLKPRES	ROSC Clock Present Indicates that the ROSC (FRO16K) clock source is present. 0b - ROSC clock source is not present 1b - ROSC clock source is present
3 FIRCCLKPRES	FIRC Clock Present Indicates that the FIRC clock source is present.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FIRC clock source is not present 1b - FIRC clock source is present
2 SIRCCLKPRES	SIRC Clock Present Indicates that the SIRC clock source is present. 0b - SIRC clock source is not present 1b - SIRC clock source is present
1 SOSCCLKPRES	SOSC Clock Present Indicates that the SOSC clock source is present. 0b - SOSC clock source is not present 1b - SOSC clock source is present
0 —	Reserved

22.7.1.4 Trim Lock (TRIM_LOCK)

Offset

Register	Offset
TRIM_LOCK	8h

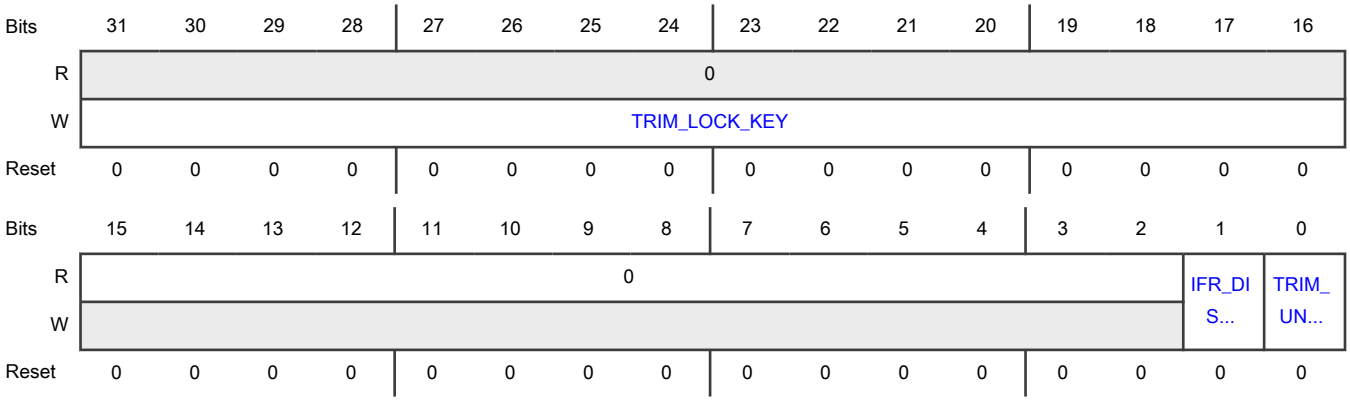
Function

The Trim Lock Register contains the TRIM_UNLOCK and IFR_DISABLE bits.

NOTE

The bits reset solely on a POR/LVD event.

Diagram



Fields

Field	Function
31-16 TRIM_LOCK_KEY	TRIM_LOCK_KEY Write value that is used to unlock writes to the IFR_DISABLE and TRIM_UNLOCK registers. **NOTE** A write value of 0x5a5a allows writes to the IFR_DISABLE and TRIM_UNLOCK to take affect. **NOTE** Writable but reads always return a 0 value.
15-2 —	Reserved
1 IFR_DISABLE	IFR_DISABLE Locks IFR write access to SCG trim registers. When set, prevents SCG trim registers from getting IFR loaded during warm reset. **NOTE** IFR_DISABLE is writable during write access to TRIM_LOCK and only when the TRIM_LOCK[31:16] write value is equal to 16'h5a5a. 0b - IFR write access to SCG trim registers not disabled. The SCG Trim registers are reprogrammed with the IFR values after any system reset. 1b - IFR write access to SCG trim registers during system reset is blocked.
0 TRIM_UNLOCK	TRIM_UNLOCK Locks user writes access to SCG Trim Configuration registers. **NOTE** TRIM_UNLOCK is writable during write access to TRIM_LOCK and only when the TRIM_LOCK[31:16] write value is equal to 16'h5a5a. 0b - SCG Trim Registers locked and not writable. 1b - SCG Trim registers unlocked and writable.

22.7.1.5 Clock Status (CSR)

Offset

Register	Offset
CSR	10h

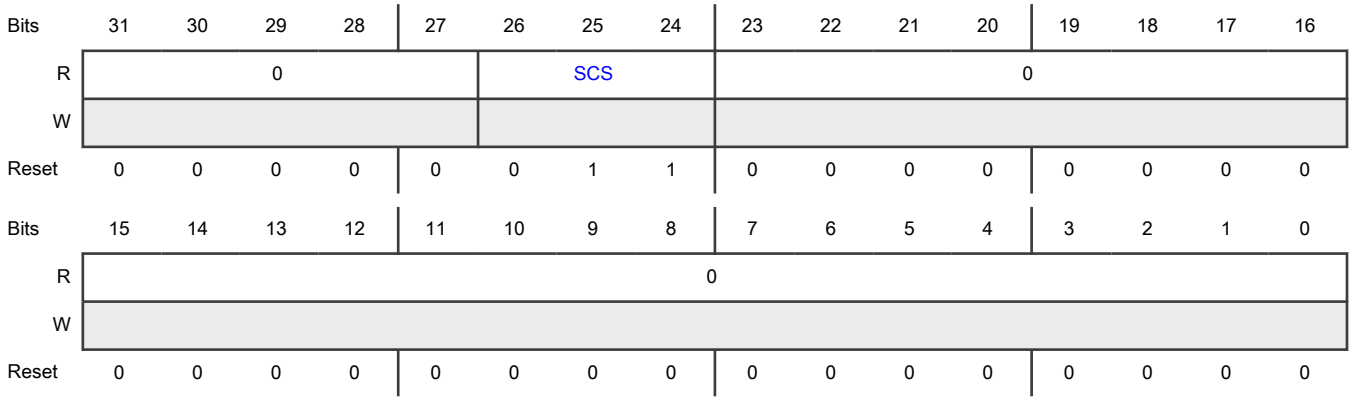
Function

The Clock Status Register shows the currently configured clock source that is generating the system clock.

NOTE

Writing to the Clock Status Register (CSR) results in a transfer error.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 SCS	System Clock Source Returns the currently configured clock source generating the system clock. Reports the configuration set by the RCCR. 001b - SOSC 010b - SIRC 011b - FIRC 100b - ROSC All other values are reserved.
23-0 —	Reserved

22.7.1.6 Run Clock Control (RCCR)

Offset

Register	Offset
RCCR	14h

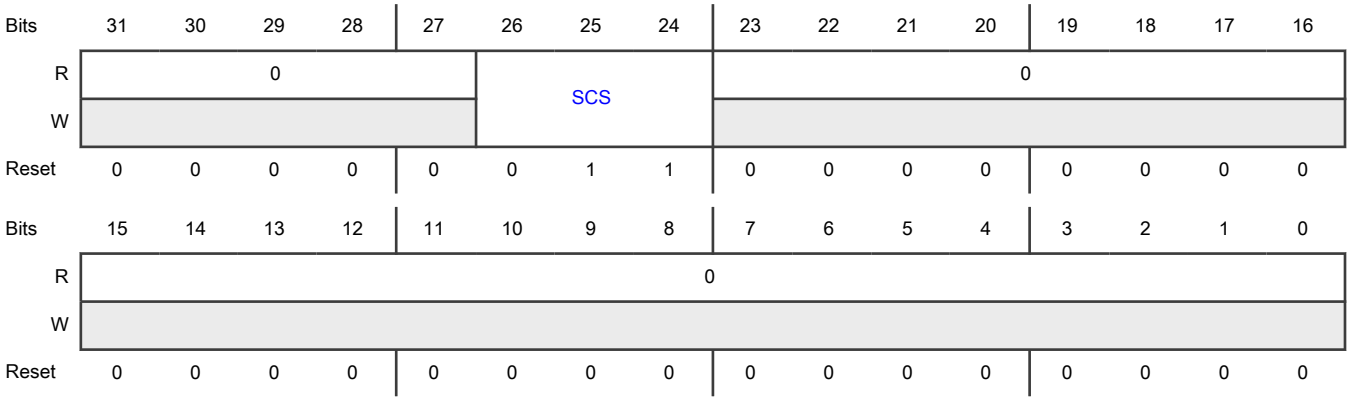
Function

The Run Clock Control Register selects the clock source generating the system clock.

NOTE

Switching to new system clock source must be done after previous RCCR changes have been completed and the Clock Status Register has been updated to match the present RCCR setting.

Diagram



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 is 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 must only be done after the previous SCS clock switch has completed. 001b - SOSC 010b - SIRC 011b - FIRC 100b - ROSC All other values are reserved.
23-0 —	Reserved

22.7.1.7 SOSC Control Status (SOSCCSR)

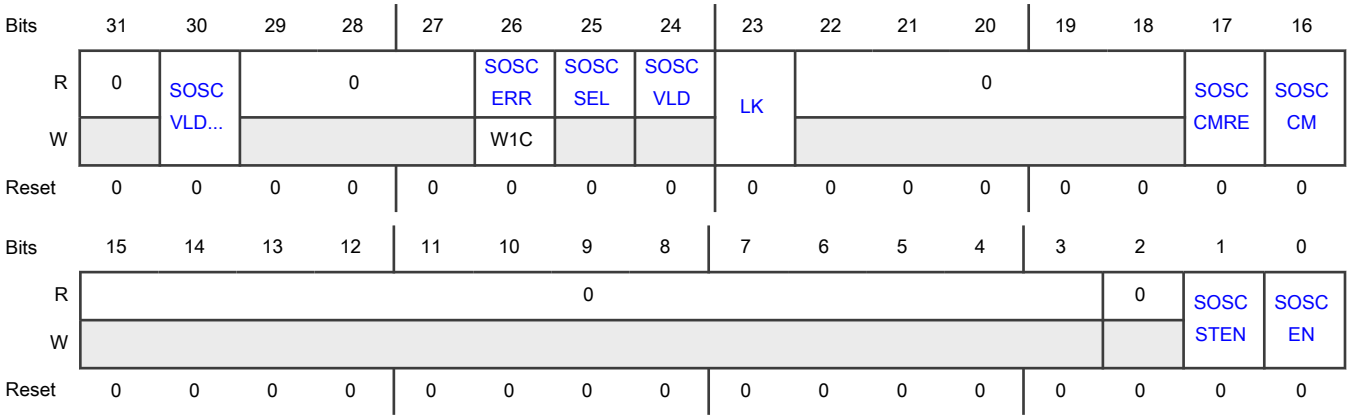
Offset

Register	Offset
SOSCCSR	100h

Function

The SOSC Control Status Register contains control and status bits for the SOSC clock source.

Diagram



Fields

Field	Function
31 —	Reserved
30 SOSCVLD_IE	SOSC Valid Interrupt Enable Generates an interrupt when SOSCVLD is asserted. 0b - SOSCVLD interrupt is not enabled 1b - SOSCVLD interrupt is enabled
29-27 —	Reserved
26 SOSCERR	SOSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one. 0b - SOSC Clock Monitor is disabled or has not detected an error 1b - SOSC Clock Monitor is enabled and detected an error
25	SOSC Selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
SOSCSEL	This flag shows the SOSC clock source is selected as the system clock source. 0b - SOSC is not the system clock source 1b - SOSC is the system clock source
24 SOSCVLD	SOSC Valid This flag shows the SOSC clock source is valid. 0b - SOSC is not enabled or clock is not valid 1b - SOSC is enabled and output clock is valid
23 LK	Lock Locks this register so that it cannot be written to. **NOTE** This 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
17 SOSCCMRE	SOSC Clock Monitor Reset Enable Enables the SOSCERR generate reset. 0b - Clock monitor generates an interrupt when an error is detected 1b - Clock monitor generates a reset when an error is detected
16 SOSCCM	SOSC Clock Monitor Enable Enables the clock monitor when SOSCVLD is set. **NOTE** SOSC clock monitor remains enabled in Deep Sleep mode only if SOSCCM and SOSCSTEN are enabled. If the clock is disabled in Deep Sleep mode, then SOSCCM must be cleared to prevent unexpected SOSC loss of clock. The clock monitor is always disabled in Power Down and Deep Power Down modes. 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 SOSC is the SIRC. This clock must be programmed to be enabled in order to monitor the SOSC. SIRC is automatically disabled in Power Down and Deep Power Down modes and the clock monitor of SOSC is disabled. 0b - SOSC Clock Monitor is disabled 1b - SOSC Clock Monitor is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-3 —	Reserved
2 —	Reserved
1 SOSCSTEN	SOSC Stop Enable Enables the SOSC clock source in Deep Sleep mode if SOSCEN is set. 0b - SOSC is disabled in Deep Sleep mode 1b - SOSC is enabled in Deep Sleep mode only if SOSCEN is set
0 SOSCEN	SOSC Enable Enables the SOSC clock source. 0b - SOSC is disabled 1b - SOSC is enabled

22.7.1.8 SOSC Configuration (SOSCCFG)

Offset

Register	Offset
SOSCCFG	108h

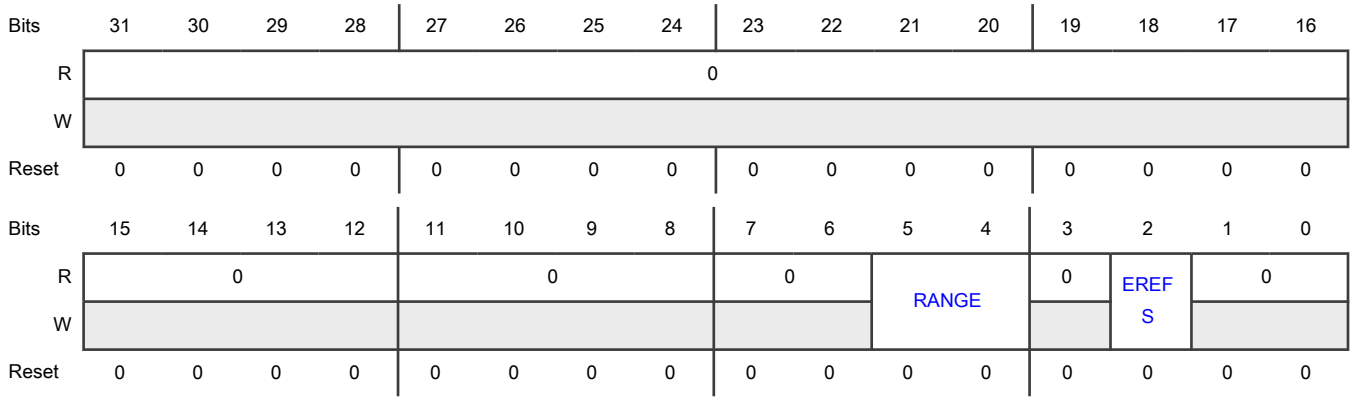
Function

The SOSC Configuration Register contains the clock frequency range select and external reference select for the SOSC clock source.

NOTE

The SOSCCFG register cannot be changed when the SOSC is enabled. When the SOSC is enabled, writes to this register are ignored, and there is no transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 —	Reserved
7-6 —	Reserved
5-4 RANGE	SOSC Range Select Selects the frequency range for the system crystal oscillator (OSC). 00b - Frequency range select of 8-16 MHz. 01b - Frequency range select of 16-25 MHz. 10b - Frequency range select of 25-40 MHz. 11b - Frequency range select of 40-50 MHz.
3 —	Reserved
2 EREFS	External Reference Select Selects the source for the external reference clock. This bit selects which clock is output from the SOSC into the SCG, whether from the crystal oscillator or from an external clock input. 0b - External reference clock selected. 1b - Internal crystal oscillator of OSC selected.
1-0 —	Reserved

22.7.1.9 SIRC Control Status (SIRCCSR)

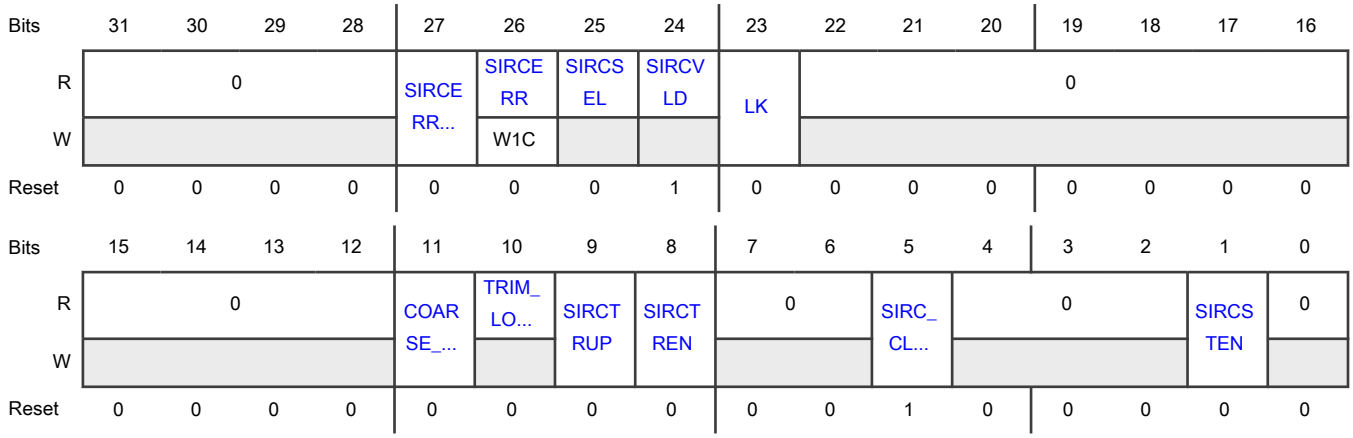
Offset

Register	Offset
SIRCCSR	200h

Function

The SIRC Control Status Register contains control and status bits for the SIRC clock source.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 SIRCERR_IE	SIRC Clock Error Interrupt Enable Generates an interrupt when SIRCERR is asserted. 0b - SIRCERR interrupt is not enabled 1b - SIRCERR interrupt is enabled
26 SIRCERR	SIRC 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 SIRC trimming 1b - Error detected with the SIRC trimming
25 SIRCSEL	SIRC Selected This flag shows the SIRC clock source is selected as the system clock source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - SIRC is not the system clock source 1b - SIRC is the system clock source
24 SIRCVLD	SIRC Valid This flag shows the SIRC clock source is valid. 0b - SIRC is not enabled or clock is not valid 1b - SIRC is enabled and output clock is valid
23 LK	Lock This 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
22-12 —	Reserved
11 COARSE_TRIM_BYPASS	Coarse Auto Trim Bypass This bit is to bypass the coarse trim step when auto-trimming. 0b - SIRC Coarse Auto Trim NOT Bypassed 1b - SIRC Coarse Auto Trim Bypassed
10 TRIM_LOCK	SIRC TRIM LOCK When SIRCTREN and SIRCTRUP are enabled, TRIM_LOCK indicates when auto trimming is complete and output SIRC frequency has locked to target SIRC range. **NOTE** • This field is automatically cleared if SIRCTREN and SIRCTRUP are not set. • TRIM_LOCK is not set when SIRCTCFG[TRIMSRC] = 2'b00, i.e., using full speed USB as the trim source. 0b - SIRC auto trim not locked to target frequency range 1b - SIRC auto trim locked to target frequency range
9 SIRCTRUP	SIRC Trim Update This bit allows the SIRCSTAT to be updated by the auto-trimming hardware. 0b - Disables SIRC trimming updates 1b - Enables SIRC trimming updates
8 SIRCTREN	SIRC 12 MHz Trim Enable (SIRCCFG[RANGE]=1) This bit enables the auto trim of SIRC 12 MHz by an external clock source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables trimming SIRC to an external clock source 1b - Enables trimming SIRC to an external clock source
7-6 —	Reserved
5 SIRC_CLK_PERIPH_EN	SIRC Clock to Peripherals Enable This bit enables the SIRC clock for peripheral use. 0b - SIRC clock to peripherals is disabled 1b - SIRC clock to peripherals is enabled
4-2 —	Reserved
1 SIRCSTEN	SIRC Stop Enable Enables the SIRC clock source in Deep Sleep mode. 0b - SIRC is disabled in Deep Sleep mode 1b - SIRC is enabled in Deep Sleep mode
0 —	Reserved

22.7.1.10 SIRC Trim Configuration (SIRCTCFG)

Offset

Register	Offset
SIRCTCFG	20Ch

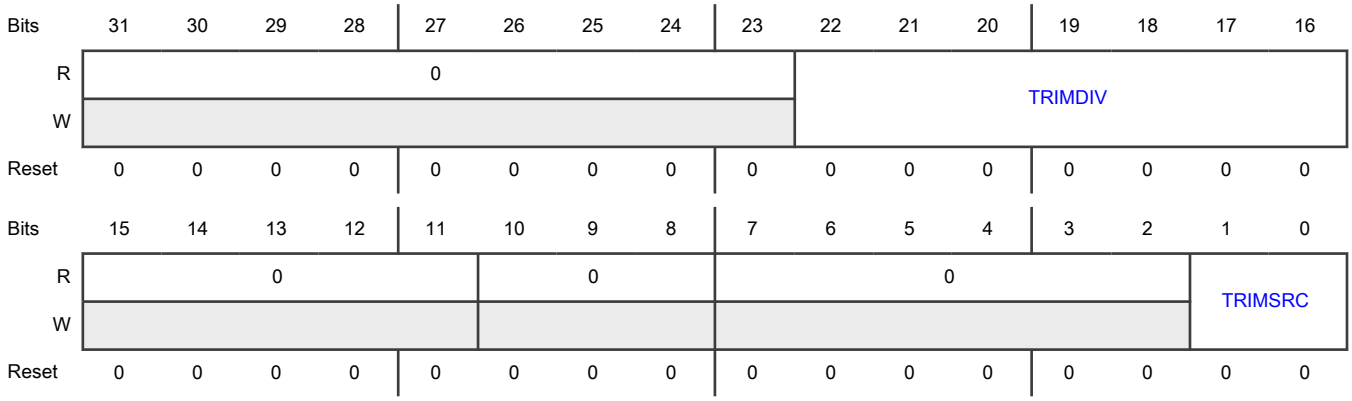
Function

The SIRC Trim Configuration Register contains the auto trim clock source select and trim clock divider control for the SIRC clock source.

NOTE

The SIRCTCFG register cannot be changed when SIRC tuning is enabled. When the SIRC tuning is enabled, writes to this register are ignored, and there is no transfer error.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 TRIMDIV	SIRC Trim Pre-divider Divider of SOSC for SIRC trimming. **NOTE** When selecting SOSC as SIRC trimming source the TRIMDIV register must be set to correct div ratio to generate 1 MHz output reference trimming clock. **NOTE** TRIMDIV is an N-Divider supporting a div ratio of 1 (TRIMDIV=0x00) to a div ratio of 128 (TRIMDIV=0x7f).
15-11 —	Reserved
10-8 —	Reserved
7-2 —	Reserved
1-0 TRIMSRC	Trim Source Configures the external clock source to tune the SIRC. TRIMSRC must be configured before programming SIRCSTAT register for trim update 00b - Reserved 01b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - SOSC. This option requires that SOSC be divided using the TRIMDIV field to get a frequency of 1 MHz. 11b - Reserved

22.7.1.11 SIRC Trim (SIRCTRIM)

Offset

Register	Offset
SIRCTRIM	210h

Function

The SIRC Trim Register holds the trim values for the SIRC clock source. This register is loaded from IFR during reset.

NOTE

This register writes are protected by TRIM LOCK register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			FVCHTRIM					0			TCTRIM				
W																
Reset	0	0	0	u[1]	u	u	u	u	0	0	0	u[1]	u	u	u	u

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

1. Reset values are loaded out of IFR.

Fields

Field	Function
31-29 —	Reserved
28-24 FVCHTRIM	Calibrates the replica voltage in FSU for CCO to get well frequency at initial period

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-21 —	Reserved
20-16 TCTRIM	Trim Temp Trim bus to calibrate the relationship between freq and temperature.
15-14 —	Reserved
13-8 CLTRIM	CL Trim Trims bus to calibrate the freq of CCO in close loop mode to within approximately ±0.6% of the target 12 MHz frequency.
7-6 —	Reserved
5-0 CCOTRIM	CCO Trim Trims bus to calibrate the freq of CCO in open loop mode to within approximately ±2.5% of the target 12 MHz frequency.

22.7.1.12 SIRC Auto-trimming Status (SIRCSTAT)

Offset

Register	Offset
SIRCSTAT	218h

Function

This register is loaded from IFR during warm reset.

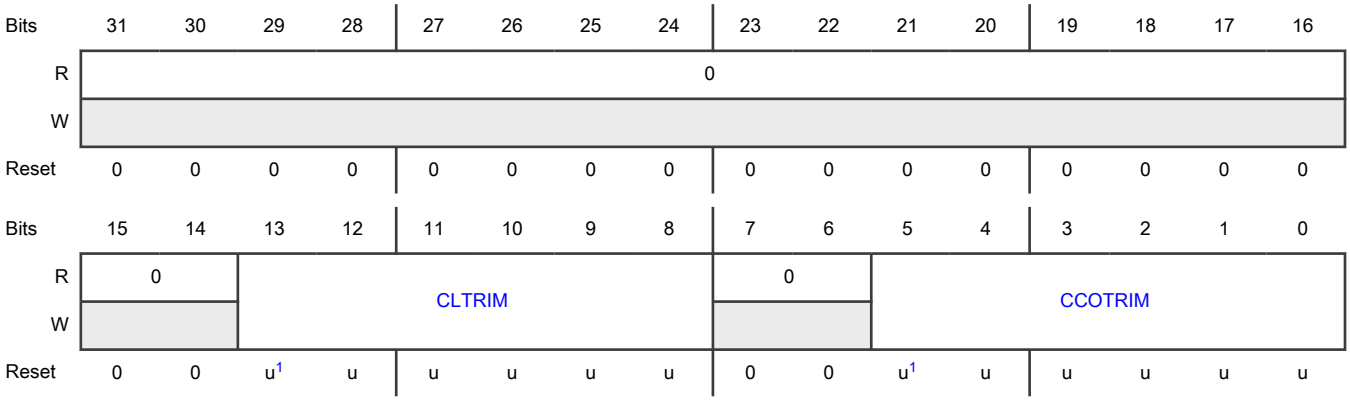
When SIRCTREN=1 and SIRCTRUP=1, this register is uploaded with the trim values generated by SIRC auto-trimming which is enabled.

When SIRCTREN=1 and SIRCTRUP=0, writing to this register is not allowed and values read from this register are used to trim SIRC clock.

NOTE

TRIMSRC needs to be programmed to TRIMSRC=10 , writes to this register are allowed and values written to this register are used to trim the SIRC clock.

Diagram



1. Reset values are loaded out of IFR.

Fields

Field	Function
31-14 —	Reserved
13-8 CLTRIM	CL Trim CLTRIM bits are to calibrate the freq of CCO in close loop mode, are used to trim the SIRC Clock to within approximately ±0.6 % of the target 12 MHz frequency.
7-6 —	Reserved
5-0 CCOTRIM	CCO Trim CCOTRIM bits are to calibrate the freq of CCO in open loop mode, are used to coarsely trim the SIRC Clock to within approximately ±2.5 % of the target 12 MHz frequency.

22.7.1.13 FIRC Control Status (FIRCCSR)

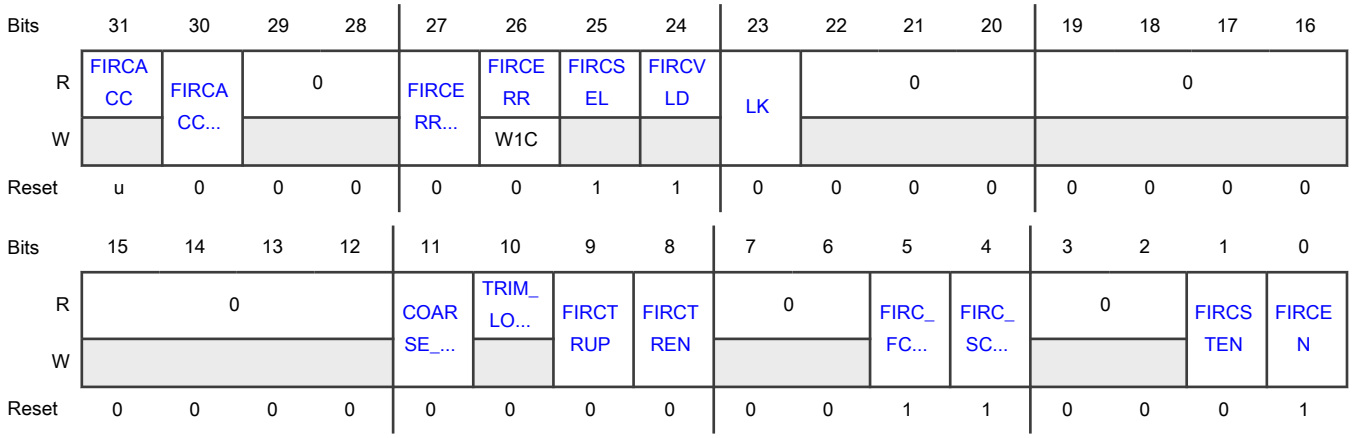
Offset

Register	Offset
FIRCCSR	300h

Function

The FIRC Control Status Register contains control and status bits for the FIRC clock source.

Diagram



Fields

Field	Function
31 FIRCACC	FIRC Frequency Accurate This flag shows the FIRC clock source is accurate. 0b - FIRC is not enabled or clock is not accurate. 1b - FIRC is enabled and output clock is accurate after some preparation time which is obtained by counting FRO_HF clock.
30 FIRCACC_IE	FIRC Accurate Interrupt Enable Generate an interrupt when FIRCACC is asserted. 0b - FIRCACC interrupt is not enabled 1b - FIRCACC interrupt is enabled
29-28 —	Reserved
27 FIRCERR_IE	FIRC Clock Error Interrupt Enable Generate an interrupt when FIRCERR is asserted. 0b - FIRCERR interrupt is not enabled 1b - FIRCERR interrupt is enabled
26 FIRCERR	FIRC 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 FIRC trimming 1b - Error detected with the FIRC trimming
25 FIRCSEL	FIRC Selected This flag shows the FIRC clock source is selected as the system clock source.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FIRC is not the system clock source 1b - FIRC is the system clock source
24 FIRCVLD	FIRC Valid status This flag shows the FIRC clock source is valid. 0b - FIRC is not enabled or clock is not valid. 1b - FIRC is enabled and output clock is valid. The clock is valid after there is an output clock from the FIRC analog.
23 LK	Lock This bit field can be cleared/set at any time. The purpose for this bitfield is to prevent runaway code from changing FIRC clock configurations. 0b - Control Status Register can be written 1b - Control Status Register cannot be written
22-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11 COARSE_TRIM_BYPASS	Coarse Auto Trim Bypass This bit is to bypass the coarse trim step when auto-trimming. 0b - FIRC Coarse Auto Trim NOT Bypassed 1b - FIRC Coarse Auto Trim Bypassed
10 TRIM_LOCK	FIRC TRIM LOCK When FIRCTREN and FIRCTRUP are enabled, TRIM_LOCK indicates when auto trimming is complete and output FIRC frequency has locked to target FIRC range. **NOTE** • This field is automatically cleared if SIRCTREN and SIRCTRUP are not set. • TRIM_LOCK is not set when SIRCTCFG[TRIMSRC] = 2'b00, i.e., using full speed USB as the trim source. 0b - FIRC auto trim not locked to target frequency range 1b - FIRC auto trim locked to target frequency range
9	FIRC Trim Update

Table continues on the next page...

Table continued from the previous page...

Field	Function
FIRCTRUP	This bit allows the FIRCSTAT to be updated by the auto-trimming hardware. 0b - Disables FIRC trimming updates 1b - Enables FIRC trimming updates
8 FIRCTREN	FRO_HF Trim Enable This bit enables the auto trim of FRO_HF by an external clock source. 0b - Disables trimming FRO_HF by an external clock source 1b - Enables trimming FRO_HF by an external clock source
7-6 —	Reserved
5 FIRC_FCLK_PERIPH_EN	FRO_HF Clock to peripherals Enable This bit enables the FRO_HF clock for peripheral use. 0b - FRO_HF to peripherals is disabled 1b - FRO_HF to peripherals is enabled
4 FIRC_SCLK_PERIPH_EN	FIRC 48 MHz Clock to peripherals Enable This bit enables the FIRC 48 MHz clock for peripheral use. 0b - FIRC 48 MHz to peripherals is disabled 1b - FIRC 48 MHz to peripherals is enabled
3-2 —	Reserved
1 FIRCSTEN	FIRC Stop Enable Enables the FIRC clock source in Deep Sleep mode if FIRCEN is set. 0b - FIRC is disabled in Deep Sleep mode 1b - FIRC is enabled in Deep Sleep mode
0 FIRCEN	FIRC Enable Enables the FIRC clock source. 0b - FIRC is disabled 1b - FIRC is enabled

22.7.1.14 FIRC Configuration (FIRCCFG)

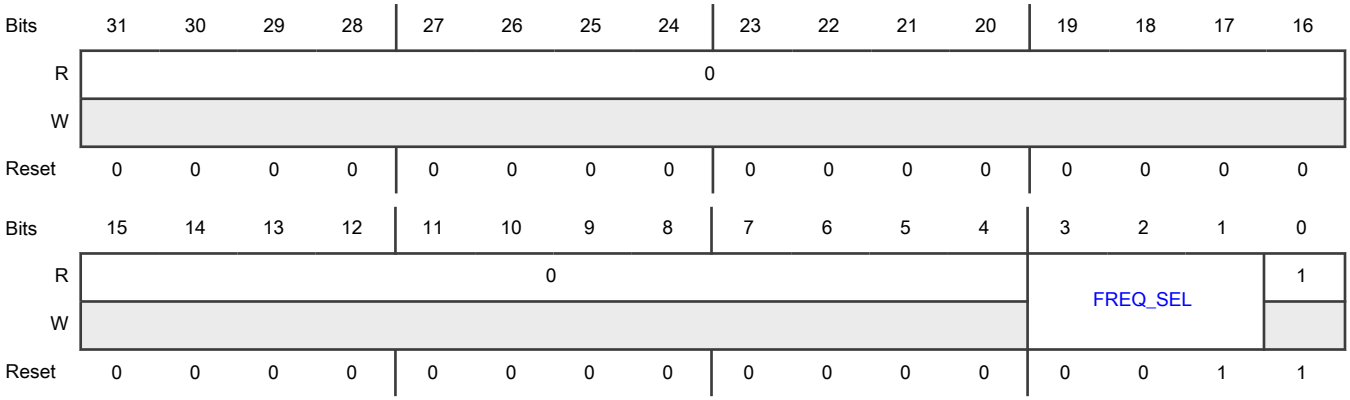
Offset

Register	Offset
FIRCCFG	308h

Function

The FIRC Configuration Register controls the clock frequency range select for the FIRC clock source.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-1 FREQ_SEL	Frequency select These bits are used to select the frequency for FRO_HF clock when used as system clock. 001b - 48 MHz FIRC clock selected, divided from 192 MHz 011b - 64 MHz FIRC clock selected 101b - 96 MHz FIRC clock selected 111b - 192 MHz FIRC clock selected
0 —	Reserved

22.7.1.15 FIRC Trim Configuration (FIRCTCFG)

Offset

Register	Offset
FIRCTCFG	30Ch

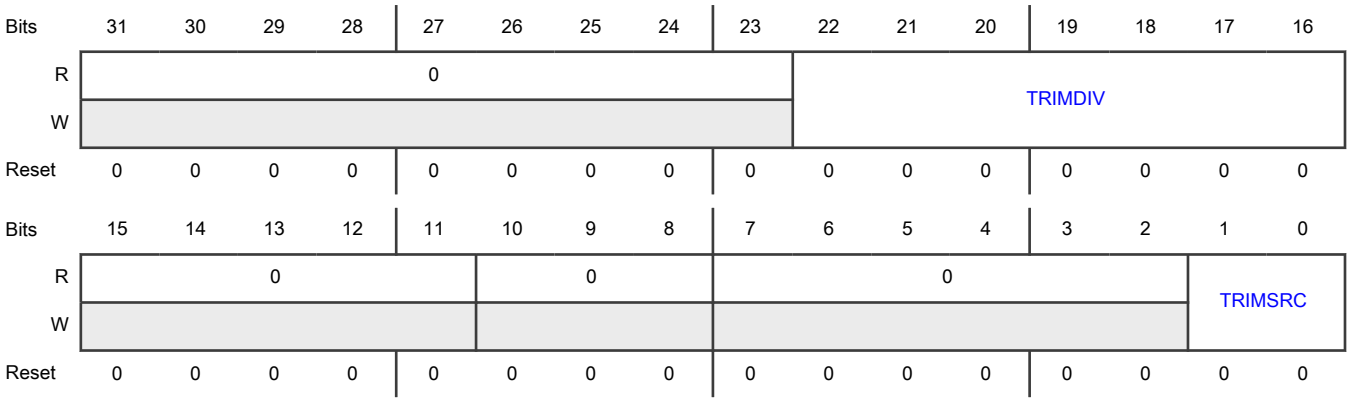
Function

The FIRC Trim Configuration Register contains the auto trim clock source select and trim clock divider control for the FIRC clock source.

NOTE

The FIRCTCFG register cannot be changed when FIRC tuning is enabled. When the FIRC tuning is enabled, writes to this register are ignored, and there is no transfer error.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 TRIMDIV	FIRC Trim Pre-divider 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 an N-Divider supporting a div ratio of 1 (TRIMDIV=0x00) to a div ratio of 128 (TRIMDIV=0x7f).
15-11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10-8 —	Reserved
7-2 —	Reserved
1-0 TRIMSRC	Trim Source Configures the external clock source to tune the FIRC. TRIMSRC must be configured before programming FIRCSTAT register for trim update. 00b - USB0 Start of Frame (1 KHz). This option does not use TRIMDIV . 01b - Reserved 10b - SOSC. This option requires that SOSC be divided using the TRIMDIV field to get a frequency of 1 MHz. 11b - Reserved

22.7.1.16 FIRC Trim (FIRCTRIM)

Offset

Register	Offset
FIRCTRIM	310h

Function

The FIRC Trim Register holds the trim values for the FIRC clock source. This register is loaded from IFR during reset. These values are used for trimming the different selected frequencies (from 48 to 192 MHz by FIRCCFG[FREQ_SEL]) of FIRC.

NOTE

This register writes are protected by TRIM LOCK register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		TRIMSTART						0				TRIMTEMP			
W																
Reset	0	0	u[1]	u	u	u	u	u	0	0	0	0	u[1]	u	u	u

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

1. Reset values are loaded out of IFR.

Fields

Field	Function
31-30 —	Reserved
29-24 TRIMSTART	Trim Start Current dac adjustment of the replica cco current of start-up circuit.
23-20 —	Reserved
19-16 TRIMTEMP	Trim Temperature Temperature coefficient compensation for FIRC .
15-14 —	Reserved
13-8 TRIMCOAR	Trim Coarse TRIMCOAR bits are used to coarsely trim the FIRC Clock to within approximately ±3.2% of the target frequency.
7-0 TRIMFINE	Trim Fine Current DAC adjustment of the CCO current. TRIMFINE bits are used to fine trim the FIRC Clock to within approximately ±0.25% of the target frequency.

22.7.1.17 FIRC Auto-trimming Status (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 FIRCTREN=1 and FIRCTRUP=0 (Note: TRIMSRC needs to be programmed to TRIMSRC= 10), writes to this register are allowed and values written to this register are used to trim FIRC clock.

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		TRIMCOAR						TRIMFINE							
W																
Reset	0	0	u[1]	u	u	u	u	u	u[1]	u	u	u	u	u	u	u

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 for the current DAC adjustment of the base current, are used to coarsely trim the FIRC Clock to within approximately ±3.2 % of the target FRO_HF clock.
7-0 TRIMFINE	Trim Fine TRIMFINE bits are for the current DAC adjustment of the CCO current, are used to trim the FIRC Clock to within approximately ±0.25 % of the target FRO_HF clock.

22.7.1.18 ROSC Control Status (ROSCCSR)

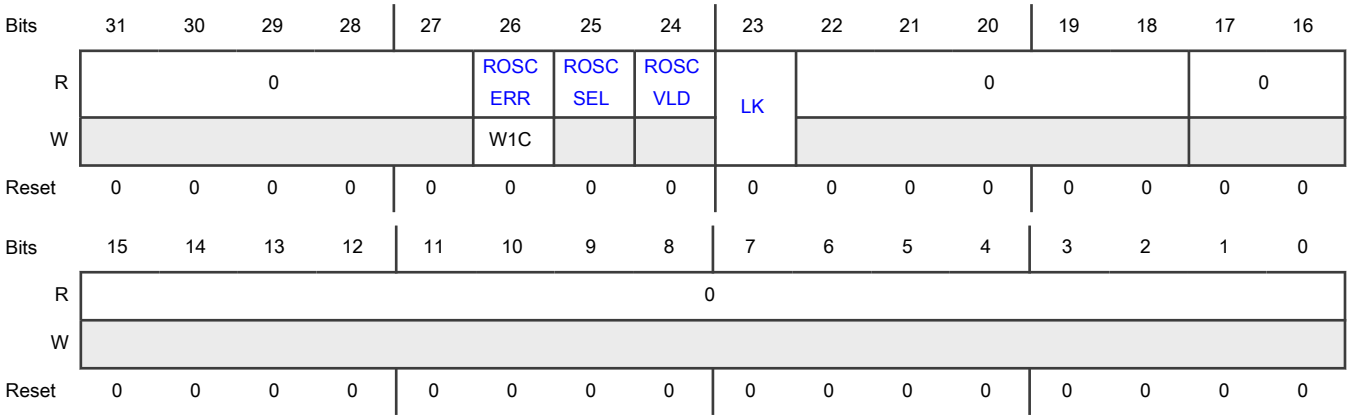
Offset

Register	Offset
ROSCCSR	400h

Function

The ROSC Control Status Register contains control and status bits for the ROSC (FRO16K) clock source.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 ROSCERR	ROSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - ROSC Clock has not detected an error 1b - ROSC Clock has detected an error
25 ROSCSEL	ROSC Selected This flag shows the ROSC clock source is selected as the system clock source. 0b - ROSC is not the system clock source 1b - ROSC is the system clock source
24 ROSCVLD	ROSC Valid This flag shows the ROSC clock source is valid. 0b - ROSC is not enabled or clock is not valid 1b - ROSC is enabled and output clock is valid
23 LK	Lock Locks this register so that it cannot be written to. **NOTE** This 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 —	Reserved
15-0 —	Reserved

Chapter 23
Power Management

23.1 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. For details about changing power modes and configuring those modes, see the CMC chapter.

23.2 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 placement of the modules within these domains.

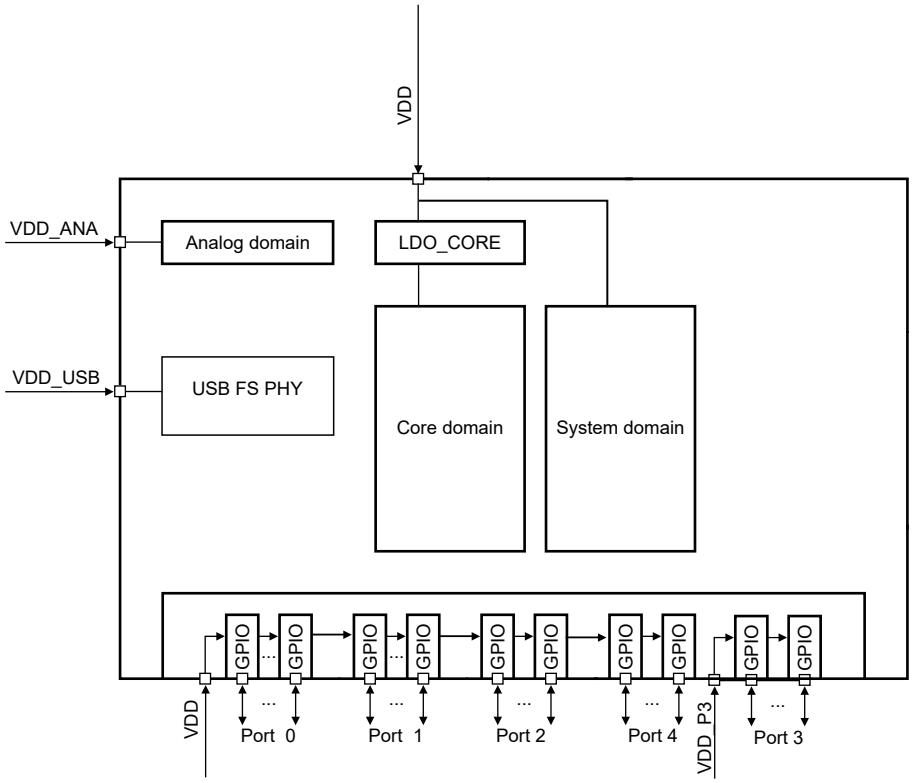


Figure 58. Power domains and voltage supplies

Domain	Voltage supply	Module
CORE	LDO_CORE	CM33
		NVIC
		Systick Timer
		Mutilayer AHB Matrix
		PBridge0/1

Table continues on the next page...

Table continued from the previous page...

		DWT
		ITM
		TPIU
		DAP
		Debug Mailbox
		SWJ
		AOI0/1
		CMC
		CDOG0
		CRC0
		DMA0
		SYSCON
		MBC
		FMC0
		FMU0
		Flash
		LPCAC
		ROM-BOOT
		FRO12M (Analog Circuitry)
		FRO192M (Analog Circuitry)
		SCG
		SOSC (Analog Circuitry)
		FlexCAN
		FlexIO
		I3C0
		LPI2C0-3
		LPSPI0/1
		LPUART0-4
		USBFS (Digital Circuitry)
		CTIMER0-4
		Frequency Measurement Unit
		OSTIMER0
		PWM0/1

Table continues on the next page...

Table continued from the previous page...

		QDC0/1
		UTICK0
		WWDT0
		ADC0-GP (Digital Circuitry)
		CMP1 (Digital Circuitry)
		DAC0 (Digital Circuitry)
		OPAMP0 (Digital Circuitry)
		GPIO0-4
		PORT0-4
SRAM	LDO_CORE / RAM_RET_LDO	Code TCM
		System TCM
SYSTEM	VDD	SPC0
		HVD/LVD/POR (Analog Circuitry)
		RAM_RET_LDO (Analog Circuitry)
		LDO_CORE (Analog Circuitry)
		VBAT
		FRO16K (Analog Circuitry)
		WUU
		LPTMR0
		Wake Timer
		CMP0/1 (Analog Circuitry)
		CMP0 (Digtal Circuitry)
		PORT0/1/2/4 (Pins)
VDD_P3	VDD_P3	PORT3 (Pins)
USB	VDD_USB	USBFS PHY (Analog Circuitry)
ANALOG	VDD_ANA	ADC0 (Analog Circuitry)
		DAC0 (Analog Circuitry)
		OPAMP0 (Analog Circuitry)

23.3 Power modes

The device supports Active, Sleep, Deep Sleep, Power Down, and Deep Power Down. Any reset brings the chip back to the Active mode.

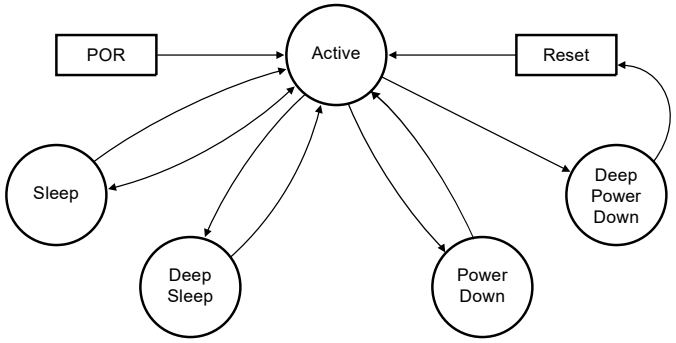


Figure 59. Power modes

23.3.1 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 CORE domain voltage level to balance power and performance.
- Gate the clocks to unused modules.

23.3.2 Sleep mode

In the Sleep mode, the CPU clock is off, and the system clock and bus clock remain ON. Most modules can remain operational. To achieve the performance requirements of a given system application while minimizing power consumption, the Sleep mode permits the following power-saving options when possible:

- Configure the CORE domain voltage level to balance power and performance.
- Gate the clocks to unused modules.

23.3.3 Deep Sleep mode

In Deep Sleep mode, CPU execution is halted. The core clock is gated off.

The Deep Sleep mode supports the following behaviors based on different clock configurations:

- 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.
- To enter Deep Sleep mode, CMC PMCTRLMAIN[LPMODE] must be configured to 0001b, and SPC LP_CFG[CORELDO_VDD_LVL] should be configured to non-zero value.
- SRAM is in deep sleep mode.

Wake from Deep Sleep mode is triggered by an interrupt or a wakeup event.

The Deep 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 the Deep Sleep mode after this module finishes its task.

To achieve the performance requirements of a given system application while minimizing power consumption, the Deep Sleep mode permits the following power-saving options when possible:

- Configure the CORE domain voltage level to balance power and performance.
- Gate the clocks to unused modules.

23.3.4 Power Down mode

Power down mode places CORE domain of the chip into a static state. It is the lowest power mode that can retain all registers.

The device wakes from Power Down mode through the interrupt routine or wake-up event.

The core clock, system clock and the bus clock are gated off.

External signals via SYSTEM domain peripherals can wake the device.

To enter power down mode, CMC PMCTRLMAIN[LPMODE] should be configured to 0011b, and SPC LP_CFG[CORELDO_VDD_LVL] should be configured to 0000b to achieve lower leakage.

Flash is powered off. SRAM arrays can also be individually configured as deep sleep or shutdown by software.

The Power Down 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 the Power Down mode after this module finishes its task.

23.4 Deep Power Down mode

The device wakes from Deep Power Down mode through the Reset routine.

In Deep Power Down mode, the whole CORE domain is power gated.

In Deep Power Down mode, the LDO_CORE is powered off. The peripherals in SYSTEM domain can be still alive.

External reset or the SYSTEM domain peripherals can wake the device.

All SRAM can be retained optionally in Deep Power Down mode.

23.5 Module operation in low power modes

The following table shows the functionality of each module in low power modes.

Module	Sleep	Deep Sleep mode	Power Down mode	Deep Power Down
Core Modules				
Bus fabric	On[1]	Static[2]	Static	Off[3]
CM33	Static	Static	Static	Off
System Modules				
AOI0/1	On	On	Static	Off
CRC	On	Static	Static	Off
CMC	On	On	Static	Off
eDMA3	On	LP[4]/Static	LP/Static	Off
EIM	On	Static	Static	Off
ERM	On	Static	Static	Off
MAU	On	Static	Static	Off
MBC	On	Static	Static	Off
Peripheral Input Mux	On	On/Static	Static	Off
RMC	On	On	On	On
SCG-Lite	On	On/Static	Static	Off

Table continues on the next page...

Table continued from the previous page...

Module	Sleep	Deep Sleep mode	Power Down mode	Deep Power Down
SPC	On	On	On	On
SYSCON	On	On/Static	Static	Off
WUU	On	On	On	On
VBAT Wrapper	On	On	On	On
Clock Source				
40M OSC	On/Off	On/Off	Off	Off
FRO192	On/Off	On/Off	Off	Off
FRO12M	On	On/Off	Off	Off
FRO16K	On/Off	On/Off	On/Off	On/Off
PMC Sub Module				
BG	On/Off	On/Off	Off	Off
LDO_CORE	LP/On	LP/On	LP	Off
PoR	On	On	On	On
RAM_Retention_LDO	On/Off	On/Off	On/Off	On/Off
VDD_LVD/HVD	On/Off	On	Off	Off
VDD_CORE_LVD	On/Off	On/Off	Off	Off
VDD_USB LVD	On/Off	On	Off	Off
Memory				
LPCAC	Static	Static	Static	Off
Flash Array	On	Static	Off (flash array VDD is power down)	Off
FMC	On	Static	Static	Off
FMU	On	Static	Static	Off
SRAM	On	Static	Static/OFF	Static/OFF
ROM	On	Static	Static	Off
Timers				
CDOG	Static	Static	Static	Off
CTimer0~5	On	Static/LP	Static	Off
FlexPWM0/1	Static	Static	Static	Off
FME	On	Static	Static	Off
LPTMR	On	Static/LP	Static/LP	Static/LP
OS Event Timer	On	Static/LP	Static	Off

Table continues on the next page...

Table continued from the previous page...

Module	Sleep	Deep Sleep mode	Power Down mode	Deep Power Down
QDC0/1	On	Static	Static	Off
uTICK	On	Static/LP	Static	Off
Wake Timer	On	Static/LP	Static/LP	Static/LP
WWDT	On	Static/LP	Static	Off
Communication				
FlexCAN	On	Static	Static	Off
FlexIO	On	Static/LP	Static	Off
I3C	On	Static/LP	Static	Off
LPI2C0~3	On	Static/LP	Static	Off
LPSPI0~1	On	Static/LP	Static	Off
LPUART0~4	On	Static/LP	Static	Off
USB FS	On	Static/LP	Static	Off
USB FS PHY	On/Off	On/Off	On/Off	On/Off
Analog				
12bit HS ADC Digital	On	Static/LP	Static	Off
12bit HS ADC Analog	On/Off	On/Off	Static	Off
DAC0	Optional	Optional	Static	Off
OPAMP0	Optional	Optional	Off	Off
CMP0 Digital	On	Static/LP	Static/LP	Static/LP
CMP0 Analog	On/Off	On/Off	On/Off	On/Off
CMP1 Digital	On	Static/LP	Static	Off
CMP1 Analog	On	On/Off	On/Off	On/Off
HMI				
GPIO0~4	On	Static/LP	Static	Off
PORT0~4	On	Static	Static	Off
IO	On	On	Static	Static
Debug and Test				
JTAG and SWD	On	Static	Static	Off
SWO	On	Static	Static	Off

1. On means module is functional and accessible via memory map. For analog module, means it is enabled.
2. Static means the module is not active
3. Off means the module can be powered off.
4. LP module in low power state (clock gated, asynchronous operation, etc). For digital module,

   it can be active with async functional clock.

NOTE

User cannot disable the regulator which powers the chip in Power Down mode and higher power mode.

23.6 Power optimization

23.6.1 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:

In Active and Sleep mode, VDD_CORE can be configured to either 1.0V or 1.1 V in SPC. When VDD_CORE is 1.0 V, it is Mid Drive (MD) mode. When VDD_CORE is 1.1 V, it is Standard Drive (SD) Mode. Lower VDD_CORE in Power Down mode, can further reduce VDD_CORE level below 1.0 V in SPC. The leakage can be reduced. Core domain can be retained, but all logics in CORE domain are static. CMC Power Mode Control Register must be configured to 4'b0011.

23.6.2 SRAM Configuration

The SRAM is partitioned, allowing the mode of each SRAM partition to be independently configured:

SRAM power mode	Description
Active (On)	SRAM can be accessed normally.
Deep Sleep (Static)	SRAM cannot be accessed, but the data is retained.
Shutdown (Off)	SRAM cannot be accessed, and the data is not retained.

SPC implements RAM_RET_LDO which can retain SRAM in Power Down(PD) and Deep Power Down (DPD) mode. There are registers in SPC to control which SRAM can be retained in PD and DPD mode.

23.6.3 Flash low power options

The internal flash memory can enter low power mode:

- in Active mode under software control (FLASHCR[FLASHDIS])
- optionally in Sleep mode (FLASHCR[FLASHDOZE])
- in Deep Sleep mode

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.

23.6.4 Peripheral clock gating

To conserve power, the clocks to modules can be turned off or divided by configuring the SYSCON module registers.

23.6.5 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.

23.6.6 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 Deep Sleep/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
- Interrupt 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.

Chapter 24
VBAT

24.1 Chip-specific VBAT information

Table 153. Reference links to related information

Topic	Related module	Reference
Full description	VBAT	VBAT
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

24.1.1 Module instances

This device contains one instance of the VBAT module, VBAT0.

24.1.2 Unsupported features

This device only supports FRO16K feature of VBAT. All other features are not supported and references to RAM LDO, bandgap timer, and switch, should be ignored.

24.2 Overview

VBAT implements the 16 kHz internal clock source via FRO16K.

24.2.1 Block diagram

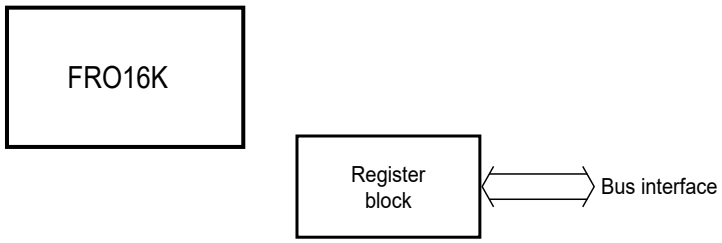


Figure 60. Block diagram

24.2.2 Features

- Supports internal 16 kHz free-running oscillator (FRO16K)

24.3 Functional description

The following sections describe the functional details of VBAT.

24.3.1 Operations

This section describes the operations of VBAT.

24.3.1.1 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.

24.3.2 Low-power modes

VBAT remains functional in all low-power modes.

24.3.3 Debug mode

Debug modes do not affect VBAT.

24.3.4 Clocks

VBAT implements the following clock domains:

- Bus interface clock - access the registers
- FRO16K internal clock

24.3.5 Reset

Only the POR resets VBAT.

24.4 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.

24.5 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).

24.5.1 VBAT register descriptions

24.5.1.1 VBAT memory map

VBAT0 base address: 4009_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0101_0001h
200h	FRO16K Control A (FROCTLA)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
218h	FRO16K Lock A (FROLCKA)	32	RW	0000_0000h
220h	FRO16K Clock Enable (FROCLKE)	32	RW	0000_0000h
700h	Wakeup 0 Register A (WAKEUP0A)	32	RW	0000_0000h
708h	Wakeup 0 Register A (WAKEUP1A)	32	RW	0000_0000h
7F8h	Wakeup Lock A (WAKLCKA)	32	RW	0000_0000h

24.5.1.2 Version ID (VERID)

Offset

Register	Offset
VERID	0h

Function

Records the specific version of VBAT in 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	0	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	1

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 FEATURE	Feature Specification Number Returns the feature set number, indicating the feature set present in this instance of VBAT.

24.5.1.3 FRO16K Control A (FROCTLA)

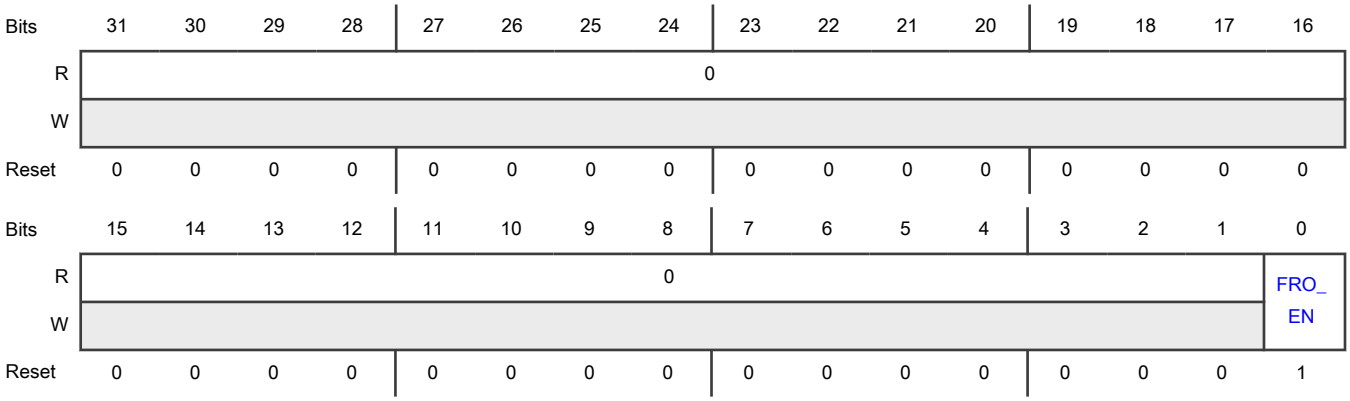
Offset

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

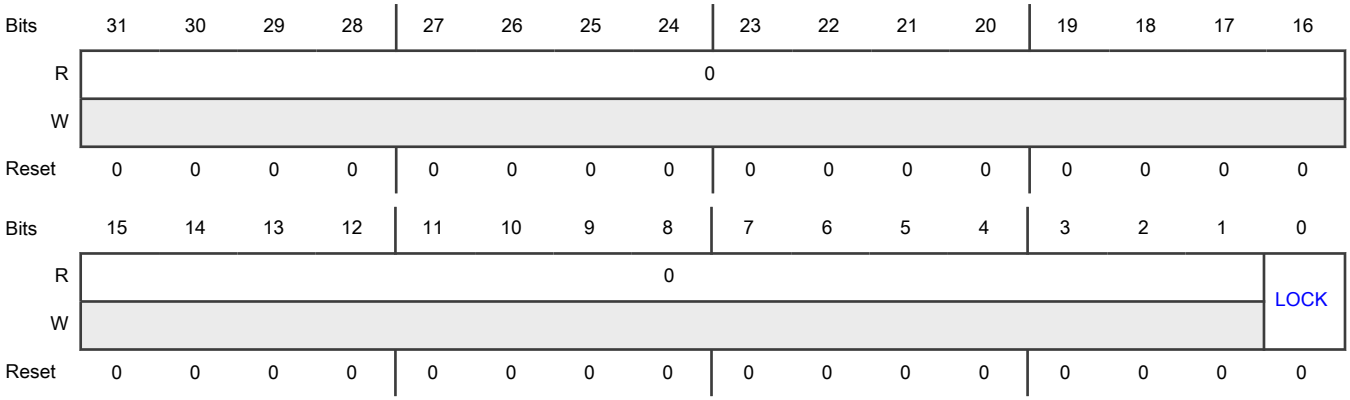
24.5.1.4 FRO16K Lock A (FROLCKA)

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

24.5.1.5 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
31-2 —	Reserved
1-0 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.

24.5.1.6 Wakeup 0 Register A (WAKEUP0A - WAKEUP1A)

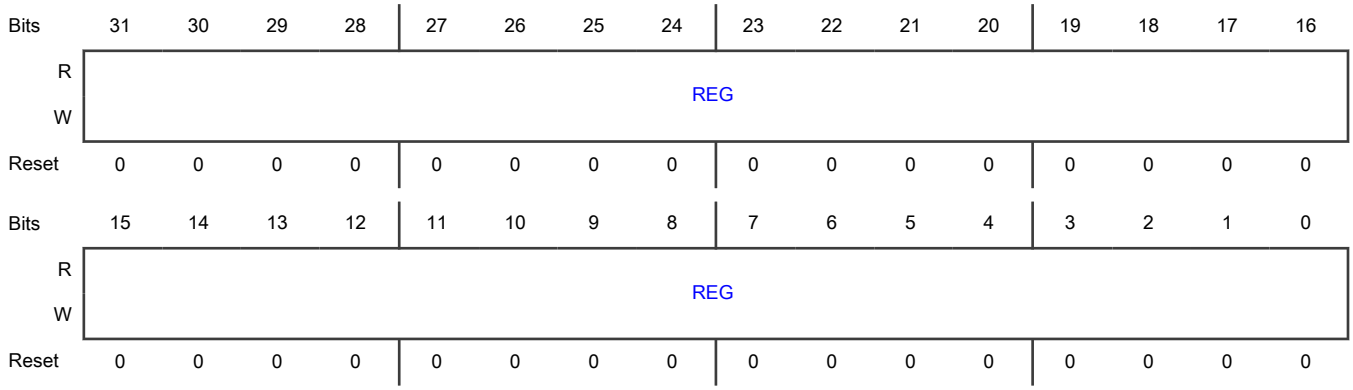
Offset

Register	Offset
WAKEUP0A	700h
WAKEUP1A	708h

Function

Contains software writable bits. Writes to this register are blocked when WAKEUP lock register is set.

Diagram



Fields

Field	Function
31-0 REG	Register Software writable value.

24.5.1.7 Wakeup Lock A (WAKLCKA)

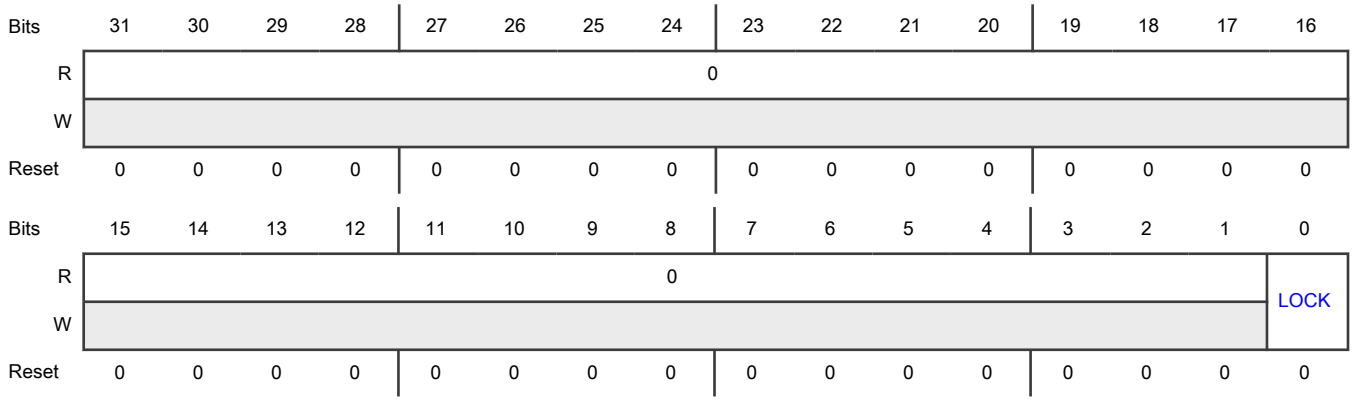
Offset

Register	Offset
WAKLCKA	7F8h

Function

Contains the lock bits. Writes to this register are blocked when Wakeup lock registers are set.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 LOCK	Lock When set, blocks writes to the Wakeup registers. 0b - Lock is disabled 1b - Lock is enabled

Chapter 25
System Power Control (SPC)

25.1 Chip-specific SPC information

Table 154. Reference links to related information

Topic	Related module	Reference
Full description	SPC	SPC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

25.1.1 Module instances

This device has one instance of the SPC module, SPC0.

25.1.2 Voltage regulators

This section describes the various on-chip regulators.

Table 155. On-chip regulators

Regulator name	Description
LDO_CORE	The LDO_CORE can be used to power VDD_CORE domain.
RAM_RET_LDO	Supports four SRAM retention switches. Each switch supports up to 16KB SRAM. • Switch 0 (SRAM_RET_EN[0]): Retain RAM X0 and X1 • Switch 1 (SRAM_RET_EN[1]): Retain RAM A0 • Switch 2 (SRAM_RET_EN[2]): Retain RAM A1, A2, A3, FlexCAN RAM • Switch 3 (SRAM_RET_EN[3]): Retain RAM B0, B1, B2

25.1.3 SOC_CNTRL fields

ACTIVE_CFG1[SOC_CNTRL] field controls which analog modules are enabled and disabled in Active or Sleep modes. LP_CFG1[SOC_CNTRL] field controls which analog modules are enabled and disabled in other low-power modes (like Deep Sleep, PowerDown, or Deep PowerDown). The following table lists the analog modules that these fields control.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Table continues on the next page...

Table continued from the previous page...

R/W											CMP1_DAC_EN	CMP0_DAC_EN			CMP1_EN	CMP0_EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W								OPAMP0_EN				DAC0_EN			USB3v_DET_EN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

25.1.4 Low power current reference enable bit LP_CFG[LP_IREFEN]

The low power current reference is used by analog modules CMP0/1, and FRO12M. The bit LP_IREFEN should be set if these analog modules are used in the Deep Power Down mode. For CMP0-1, this bit should be set if using nano mode in the Deep Power Down mode.

25.1.5 Power modes

This device doesn't support Power Down (PD) mode. ALl references to PD mode should be ignored throughout the chapter. See the Power management chapter of the device, for details.

25.1.6 Low Power Wake-Up Delay

Table 156. LPWKUP Delay

Regulator	LP_CFG Voltage	ACTIVE_CFG Voltage	Minimum LPWKUP_DELAY
LDO_CORE	0.6V	1.0V	0x5B
LDO_CORE	0.6V	1.1V	0x5B
LDO_CORE	1.0V	1.0V	0
LDO_CORE	1.0V	1.1V	0x5B

25.2 Overview

SPC contains and controls the following components:

- One low-drop-out (LDO) voltage regulator (1.0 V LDO_CORE)
- Circuits that monitor and assure correct voltage levels:
  - Power-on reset (POR)

— Low-voltage detect (LVD)

— 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.

25.2.1 Block diagram

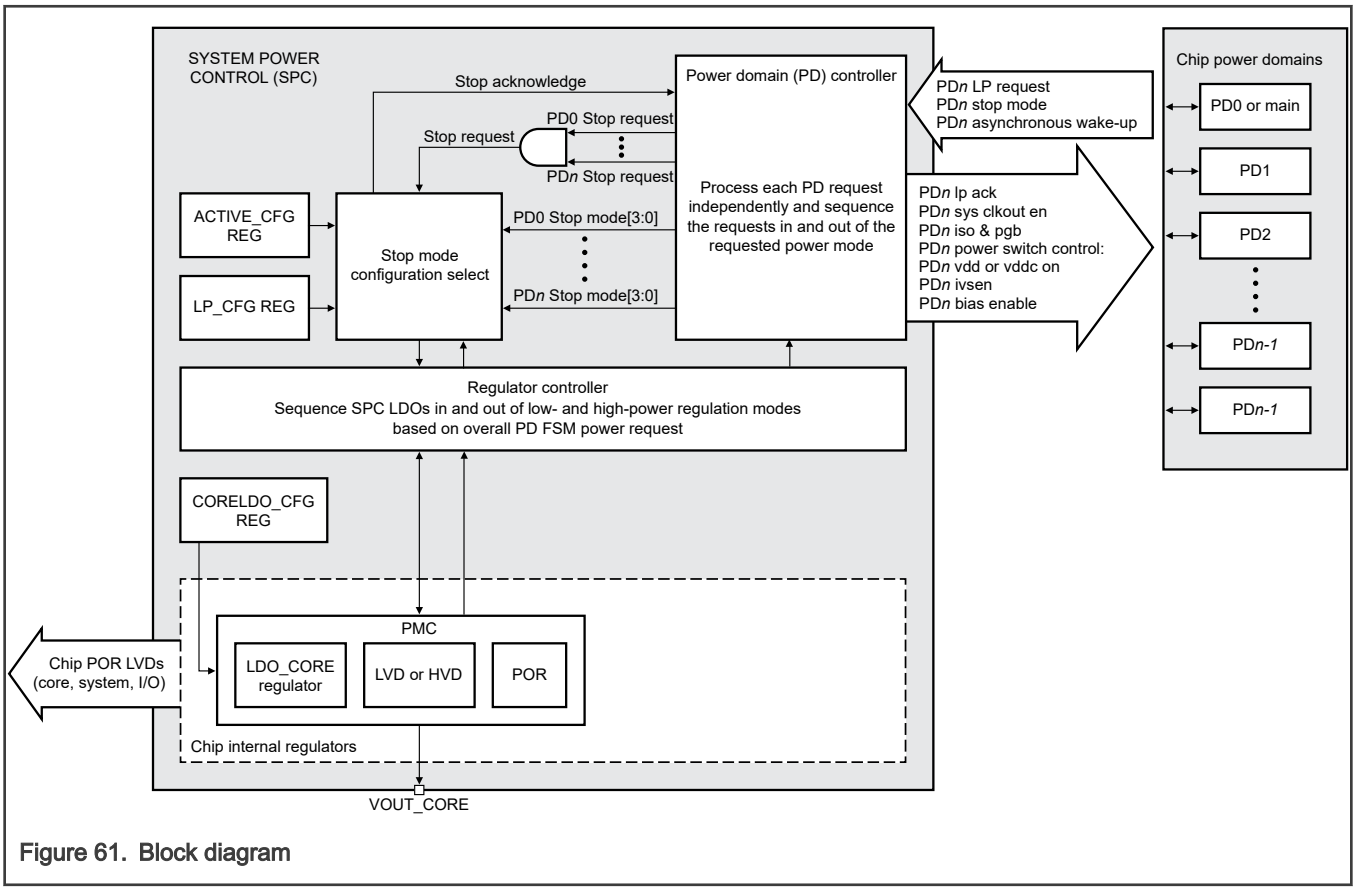


Figure 61. Block diagram

25.2.2 Features

- Selectable core internal voltage regulator (1.2 V LDO_CORE)
- 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

25.3 Functional description

25.3.1 Power mode transitions

Figure 62 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.

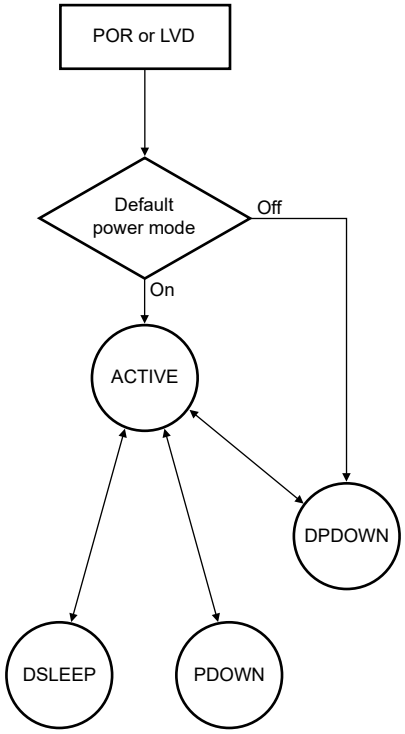


Figure 62. Power mode state diagram

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 63 defines the low-power entry and exit flow for each of the SPC power-down domains.

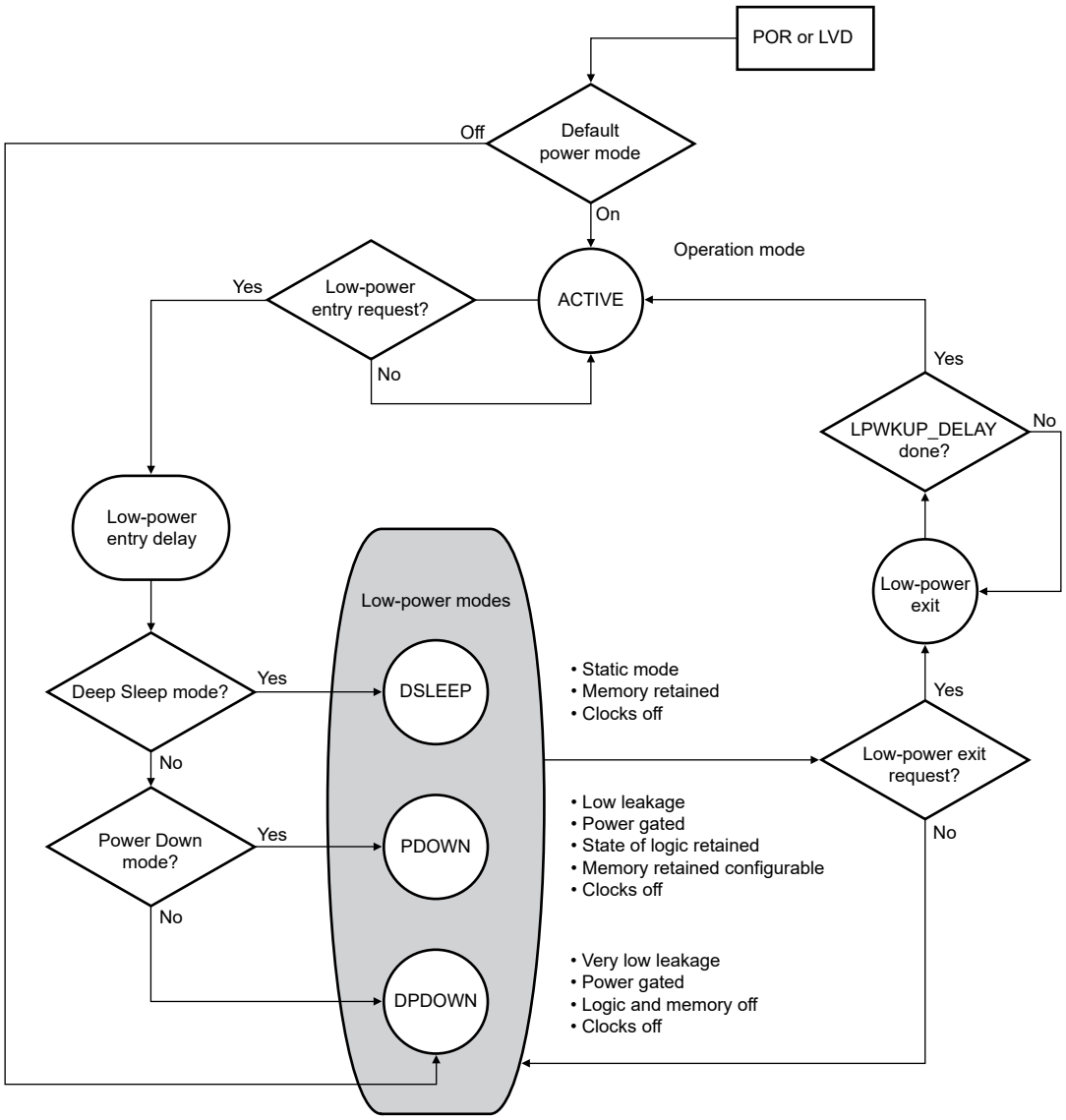


Figure 63. SPC regulator power mode state diagram

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.

25.3.2 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]).
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]).
8. Wait for voltage change to complete (SC[BUSY] = 0).
9. You can continue execution.

25.3.3 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.

25.3.4 Clocking

Table 157. 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.

25.3.5 Reset

25.3.5.1 Reset sources

25.3.5.1.1 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.

25.3.5.1.2 Low-voltage detect (LVD)

SPC supports the following LVD circuits:

- Core VDD
- System 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.

25.3.5.1.3 High-voltage detect (HVD)

SPC supports the following HVD circuit:

- System 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.

25.3.5.2 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.

25.3.6 Interrupts

SPC generates a single interrupt. The trigger for this interrupt comes from:

- Any of the LVD or HVD circuits when they are not configured for reset and their corresponding HVDIE or LVDIE register.

25.4 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

25.5 Initialization

SPC does not require any special initialization.

25.6 Application information

To disable Bandgap in Active mode:

1. Disable all LVD's and HVD's in ACTIVE_CFG[29:24]= 0x00
2. Configure LDO's to Low Drive Strength in ACTIVE_CFG register
3. 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. Configure LDO's to Low Drive Strength in LP_CFG register
3. Configure LP_CFG[BGMODE] = 0x0

25.7 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.

25.7.1 SPC memory map

SPC0 base address: 4009_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
10h	Status Control (SC)	32	RW	0000_0000h
1Ch	Low-Power Request Configuration (LPREQ_CFG)	32	RW	0000_0000h
30h	SPC Power Domain Mode Status (PD_STATUS0)	32	RW	0000_0000h
40h	SRAM Control (SRAMCTL)	32	RW	0000_0001h
54h	SRAM Retention Reference Trim (SRAMRETLDO_REFTRIM)	32	RW	0000_0017h
58h	SRAM Retention LDO Control (SRAMRETLDO_CNTRL)	32	RW	0000_0F01h
100h	Active Power Mode Configuration (ACTIVE_CFG)	32	RW	1310_0005h
104h	Active Power Mode Configuration 1 (ACTIVE_CFG1)	32	RW	0000_0002h
108h	Low-Power Mode Configuration (LP_CFG)	32	RW	0008_0004h
10Ch	Low Power Mode Configuration 1 (LP_CFG1)	32	RW	0000_0002h
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
138h	System Voltage Detect Configuration (VD_SYS_CFG)	32	RW	0000_0001h
140h	External Voltage Domain Configuration (EVD_CFG)	32	RW	0000_0000h
300h	LDO_CORE Configuration (CORELDO_CFG)	32	RW	0000_0000h

25.7.2 Version ID (VERID)

Offset

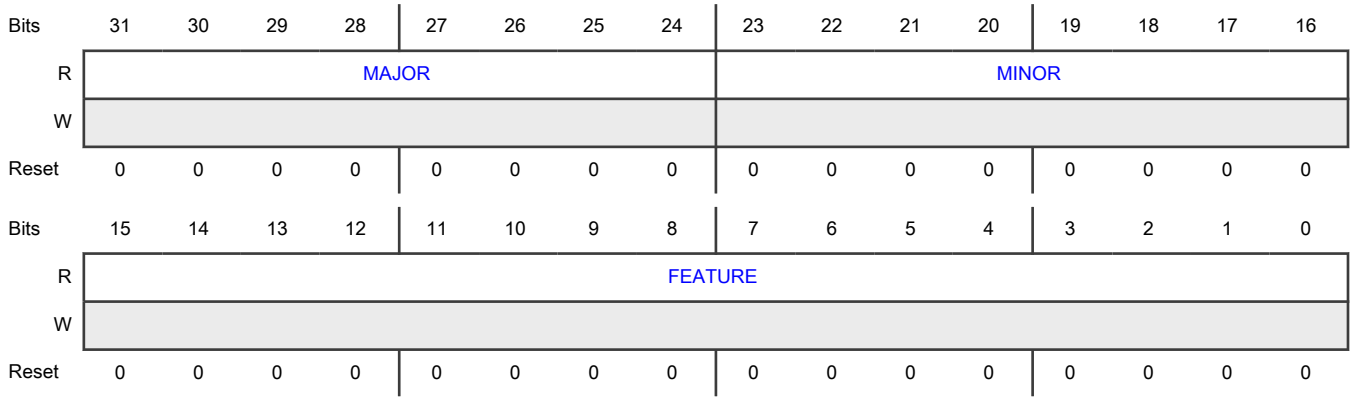
Register	Offset
VERID	0h

Function

Indicates:

- The version integrated for this instance on the chip.
- Inclusion or exclusion of several optional features.

Diagram



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.

25.7.3 Status Control (SC)

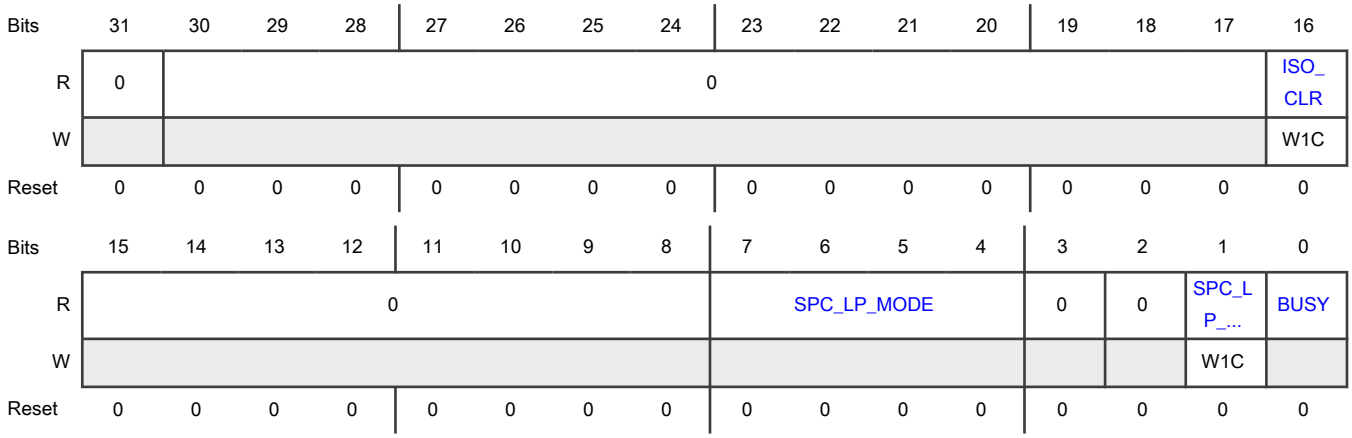
Offset

Register	Offset
SC	10h

Function

Indicates the current SPC status.

Diagram



Fields

Field	Function
31 —	Reserved
30-17 —	Reserved
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0001b - DSLEEP with system clock off 0010b - PDOWN with system clock off 0100b - Reserved 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. **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] changes while in Active mode. **NOTE** Wait until this flag is clear before changing power-mode configuration registers. 0b - Not busy 1b - Busy

25.7.4 Low-Power Request Configuration (LPREQ_CFG)

Offset

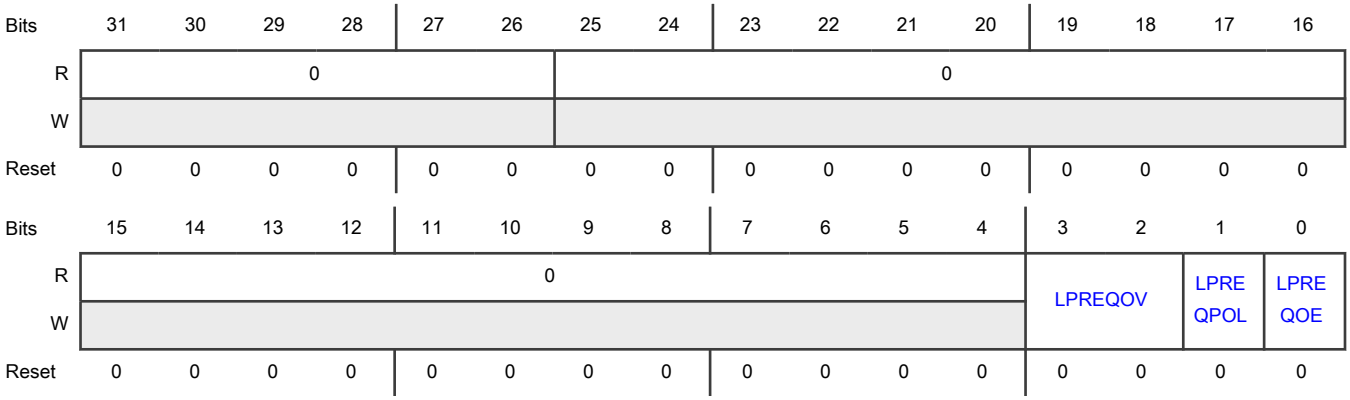
Register	Offset
LPREQ_CFG	1Ch

Function

Configures the low-power output request pin.

The register resets only after a POR, LVD, or HVD event.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-4 —	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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - High 1b - Low
0 LPREQOE	Low-Power Request Output Enable Enables the low-power request output pin. 0b - Disable 1b - Enable

25.7.5 SPC Power Domain Mode Status (PD_STATUS0)

Offset

Register	Offset
PD_STATUS0	30h

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	Power Domain Low Power Mode Request

Table continues on the next page...

Table continued from the previous page...

Field	Function
LP_MODE	Indicates the last low-power mode that the power domain requested. 0000b - SLEEP with system clock running 0001b - DSLEEP with system clock off 0010b - PDOWN with system clock off 0100b - Reserved 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

25.7.6 SRAM Control (SRAMCTL)

Offset

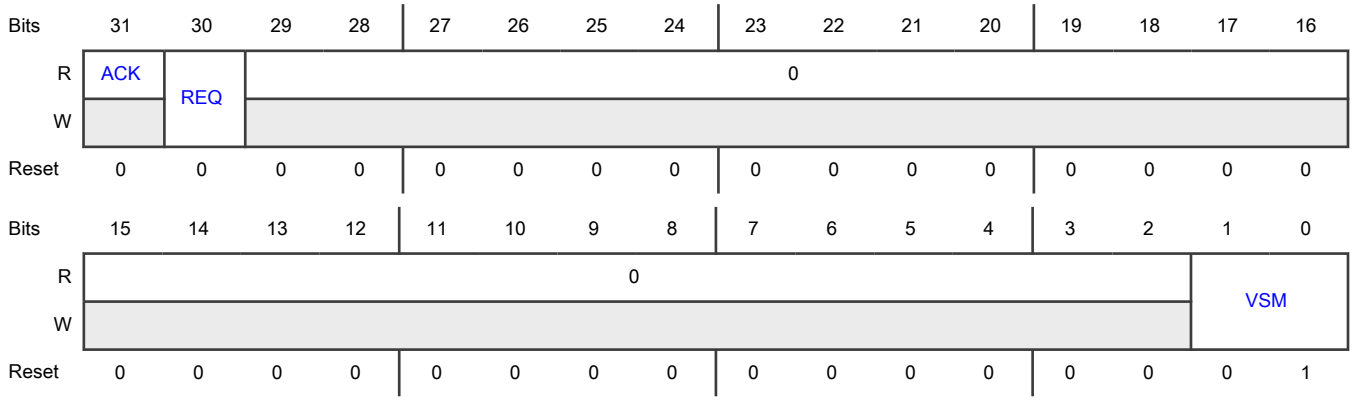
Register	Offset
SRAMCTL	40h

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 - Reserved

25.7.7 SRAM Retention Reference Trim (SRAMRETLDO_REFTRIM)

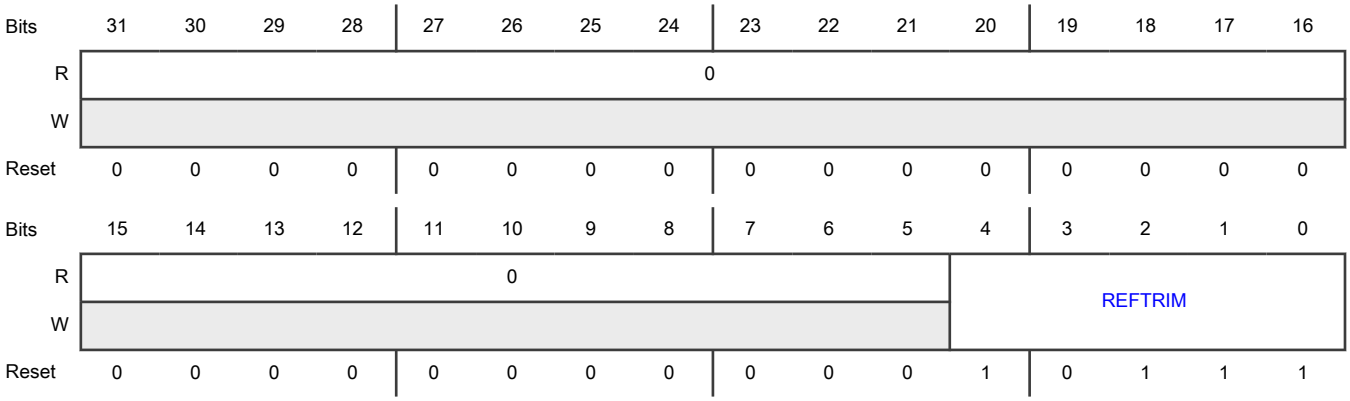
Offset

Register	Offset
SRAMRETLDO_REFTRIM	54h

Function

Specifies the trim for the SRAM retention regulator reference.

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 REFTRIM	Reference Trim. Voltage range is around 0.48V - 0.85V. Trim step is 12 mV.

25.7.8 SRAM Retention LDO Control (SRAMRETLDO_CNTRL)

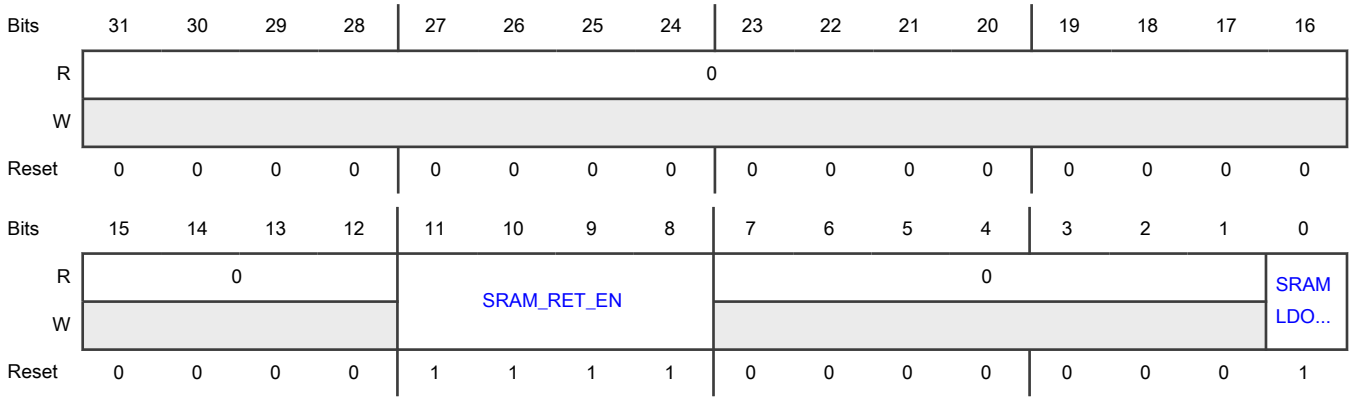
Offset

Register	Offset
SRAMRETLDO_CNTRL	58h

Function

Controls the SRAM retention LDO operation.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 SRAM_RET_EN	SRAM Retention Configures the retention of each SRAM array in low-power modes. When SRAMLDO_ON is 1, you must retain at least one SRAM array. When SRAMLDO_ON is 0, SRAM_RET_EN specifies whether the SRAM array remains powered from VDD_CORE. See the chip-specific SPC information for SRAM retention.
7-1 —	Reserved
0 SRAMLDO_ON	SRAM LDO Regulator Enable 0b - Disable 1b - Enable

25.7.9 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 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		0	SYS_HVDE	0	0	SYS_LVDE	CORE_LV...	VDD_VD_...	0	BGMODE		0	0	0	
W																
Reset	0	0	0	1	0	0	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	0		0	0		0		0	0	0	0	CORELDO_VD_LVL		0	CORELDO...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-30 —	Reserved
29 —	Reserved
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 —	Reserved
26 —	Reserved
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 VDD_VD_DISABLE	VDD Voltage Detect Disable Disables VDD voltage detect. Mask all LVDs and HVDs when you change the active voltage levels. If this field is 1, it will not mask and allow LVDs and HVDs conditions to generate a system reset or interrupt. If this field is 0, it will mask all LVDs and HVDs when changing the regulator voltage levels during Active mode. 0b - Enable 1b - Disable
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 10b - Bandgap enabled, buffer enabled 11b - Reserved
19 —	Reserved
18 —	Reserved
17-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14-13 —	Reserved
12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
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. 00b - Reserved 01b - Regulate to mid voltage (1.0 V)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Regulate to normal voltage (1.1 V) 11b - Reserved
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

25.7.10 Active Power Mode Configuration 1 (ACTIVE_CFG1)

Offset

Register	Offset
ACTIVE_CFG1	104h

Function

Enables different analog modules in Active mode. To see what analog modules each of these bits are controlling, see the chip-specific SPC information.

Diagram

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

Fields

Field	Function
31-0 SOC_CNTRL	Active Config Chip Control Enables analog modules in Active mode. Bit combinations with 0: The associated analog modules are disabled. Bit combinations with 1: The associated analog modules are enabled.

25.7.11 Low-Power Mode Configuration (LP_CFG)

Offset

Register	Offset
LP_CFG	108h

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		0	SYS_HVDE	0	0	SYS_LVDE	CORE_LV...	LP_IREF...	0	BGMODE		SRAMLDO...	0	0	0
W																
Reset	0	0	0	0	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	0		0	0		0		0	0		0	CORELDO_VDD_LVL		0	CORELDO...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-30 —	Reserved
29 —	Reserved
28	System High Voltage Detect Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYS_HVDE	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 —	Reserved
26 —	Reserved
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 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. 00b - Bandgap disabled 01b - Bandgap enabled, buffer disabled 10b - Bandgap enabled, buffer enabled 11b - Reserved
19 SRAMLDO_DPD_ON	SRAM_LDO Deep Power Low Power IREF Enable When set keeps the retention SRAM **NOTE** See the chip specific section for the modules that require the low power IREF to be enabled in DPDOWN mode, if the module is being used in DPDOWN mode. 0b - Low Power IREF is disabled for power saving in Deep Power Down mode 1b - Low Power IREF is enabled
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14-13 —	Reserved
12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-5 —	Reserved
4 —	Reserved
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. 00b - Reserved 01b - Mid voltage (1.0 V) 10b - Normal voltage (1.1 V) 11b - Reserved
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

25.7.12 Low Power Mode Configuration 1 (LP_CFG1)

Offset

Register	Offset
LP_CFG1	10Ch

Function

Enables different analog modules in Low-Power mode.

See the chip-specific SPC information for a list of analog modules that this register controls.

Diagram

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

Fields

Field	Function
31-0 SOC_CNTRL	Low-Power Configuration Chip Control Provides control bits to the chip to enable analog modules in Low Power mode. Bit combinations with 0: The associated analog modules are disabled. Bit combinations with 1: The associated analog modules are enabled.

25.7.13 Low Power Wake-Up Delay (LPWKUP_DELAY)

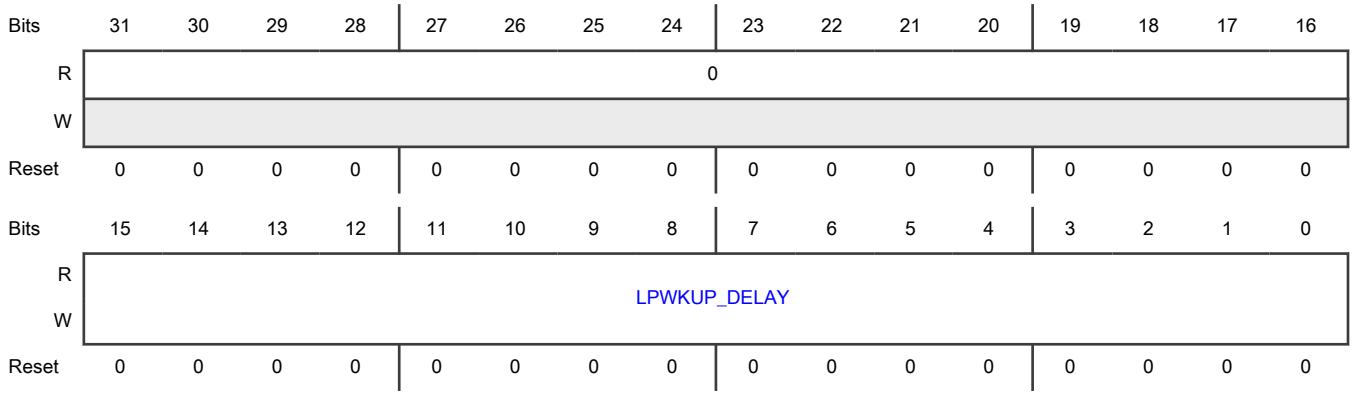
Offset

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
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.

25.7.14 Active Voltage Trim Delay (ACTIVE_VDELAY)

Offset

Register	Offset
ACTIVE_VDELAY	124h

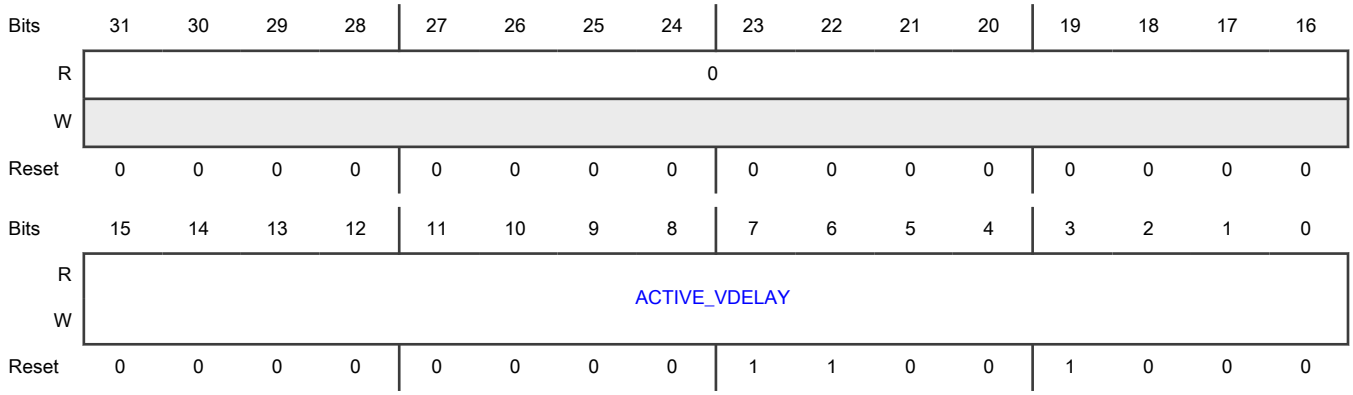
Function

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



Fields

Field	Function
31-16 —	Reserved
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.

25.7.15 Voltage Detect Status (VD_STAT)

Offset

Register	Offset
VD_STAT	130h

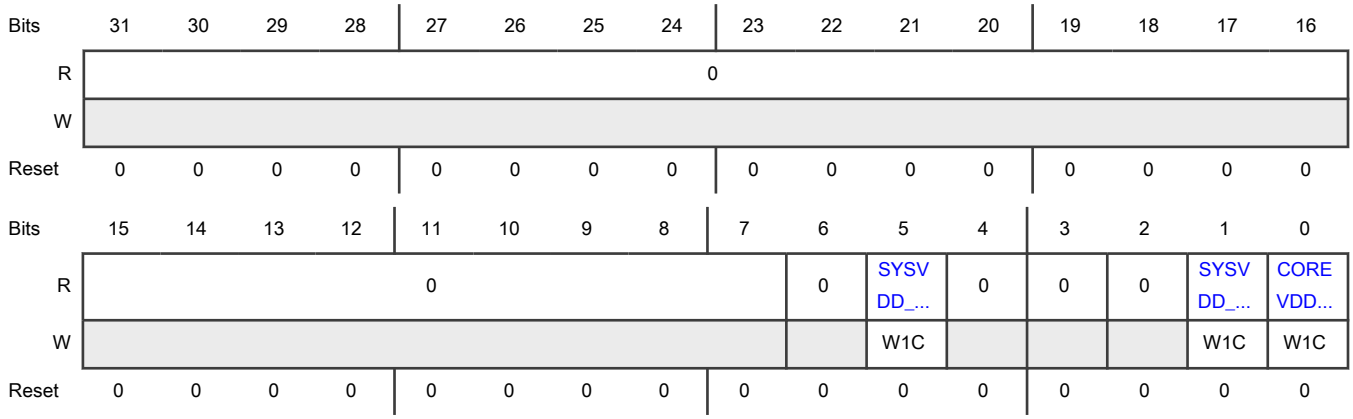
Function

Displays the LVD or HVD captured for:

- Core VDD.
- System VDD.
- IO VDD.

This register resets only after a POR event.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-7 —	Reserved
6 —	Reserved
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 —	Reserved
3 —	Reserved
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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

25.7.16 Core Voltage Detect Configuration (VD_CORE_CFG)

Offset

Register	Offset
VD_CORE_CFG	134h

Function

Provides these functions:

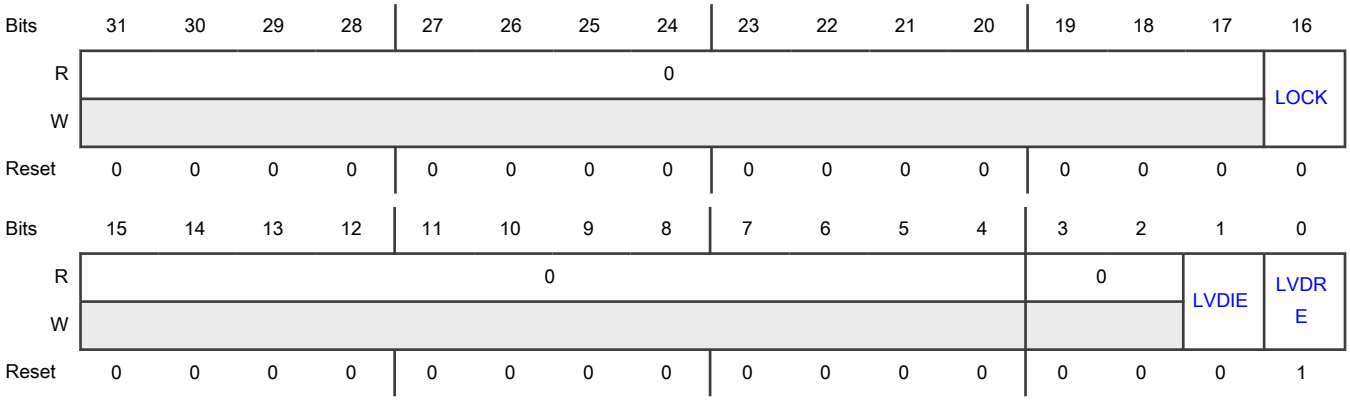
- Configures the low-voltage trip point.
- Enables the core LVD reset and interrupts.

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
3-2 —	Reserved
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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

25.7.17 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

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															LOCK
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	Reserved	0				HVDIE	HVDRE	LVDIE	LVDRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
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. 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

25.7.18 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 158.

Table 158. Mapping of bits to supplies

Bit position in field	Description
0	VDD
1	Reserved
2	VDD_P3
3	VDD_P3

NOTE

Bit 0 VDD 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[1]	0	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
31-20 —	Reserved
19-16 EVDSTAT	External Voltage Domain Status Indicates the status of the external voltage domains. See Table 158 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
15-12 —	Reserved
11-8 EVDLPISO	External Voltage Domain Low-Power Isolation Isolates the external voltage domain in low-power modes. Use this field if you use the SPC_LPREQ pin to power off any voltage domain in low-power modes. See Table 158 for the mapping of each bit of this field. Each bit's value has the following meaning: 0b - Not isolated 1b - Isolated
7-4 —	Reserved
3-0 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	See Table 158 for the mapping of each bit of this field. Each bit's value has the following meaning: 0b - Not isolated 1b - Isolated

25.7.19 LDO_CORE Configuration (CORELDO_CFG)

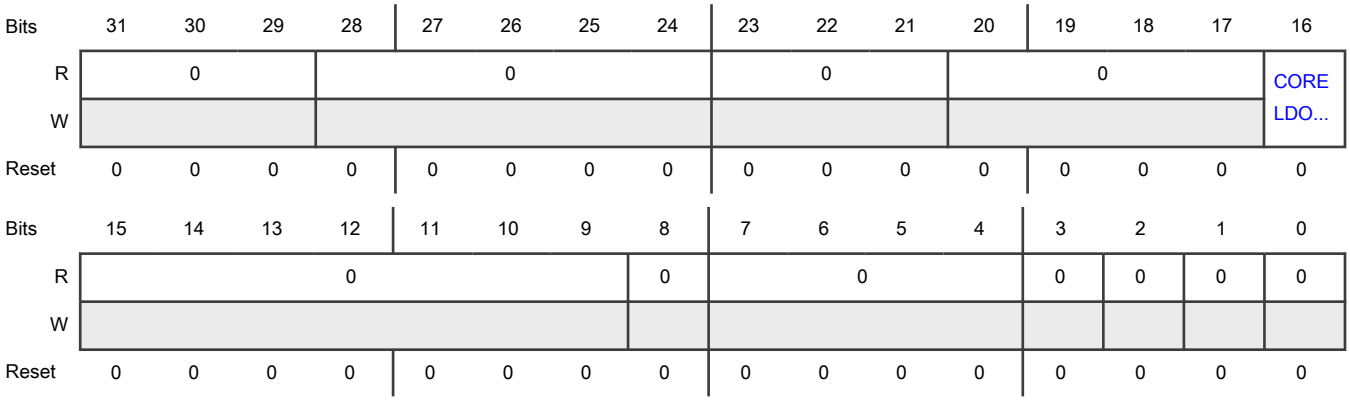
Offset

Register	Offset
CORELDO_CFG	300h

Function

Configures LDO_CORE.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 —	Reserved
23-21 —	Reserved
20-17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 CORELDO_SPARE0	CORELDO SPARE0
15-9 —	Reserved
8 —	Reserved
7-4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

Chapter 26
Standard counter/timers (CTIMER)

26.1 Chip-specific CTIMER information

Table 159. Reference links to related information

Topic	Related module	Reference
Full description	CTIMER	CTIMER
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

26.1.1 Module instances

This device has five instances of the CTIMER module, CTIMER0, CTIMER1, CTIMER2, CTIMER3 and CTIMER4. Each timer has up to 4x capture and 4x match registers with corresponding inputs and outputs.

CTimer supports async function clock. The async function clock speed can be higher than APB bus clock.

26.1.2 Signals

The tables below give a summary of each of the Timer/Counter related pins and the recommended PORT settings. Also note that different part number and package variations may provide different CTIMER related pin functions.

Table 160. Timer/Counter pin description

Pin	Type	Description
CTIMER0_CAP3:0 CTIMER1_CAP3:0 CTIMER2_CAP3:0 CTIMER3_CAP3:0 CTIMER4_CAP3:0	Input	Capture Signals- A transition on a capture pin can be configured to load one of the Capture Registers with the value in the Timer Counter and optionally generate an interrupt. Capture functionality can be selected from a number of pins. Timer/Counter block can select a capture signal as a clock source instead of the APB bus clock. For more details, see CTCR register.
CT0_MAT3:0 CT1_MAT3:0 CT2_MAT3:0 CT3_MAT3:0 CT4_MAT3:0	Output	External Match Output - When a match register (MR3:0) equals the timer counter (TC) this output can either toggle, go low, go high, or do nothing. The External Match Register (EMR) controls the functionality of this output. Match Output functionality can be selected on a number of pins in parallel.

26.1.3 External global enable

When the timer is enabled by the CTIMER global start register in SYSCON, (CTIMER Global Start Enable (CTIMERGLOBALSTARTEN)), it is equivalent to writing 1 to the TCR[CEN] bit in CTIMER. The external global start register is used to enable multiple timers at the same time. This operation is allowed by writing a 1 to each Timer's TCR[AGCEN] bit. If the Timer control register's CEN bit is already 1, the external global start enable register has no effect (See CTIMER Global Start Enable (CTIMERGLOBALSTARTEN)).

26.1.4 Peripheral Input Multiplexers

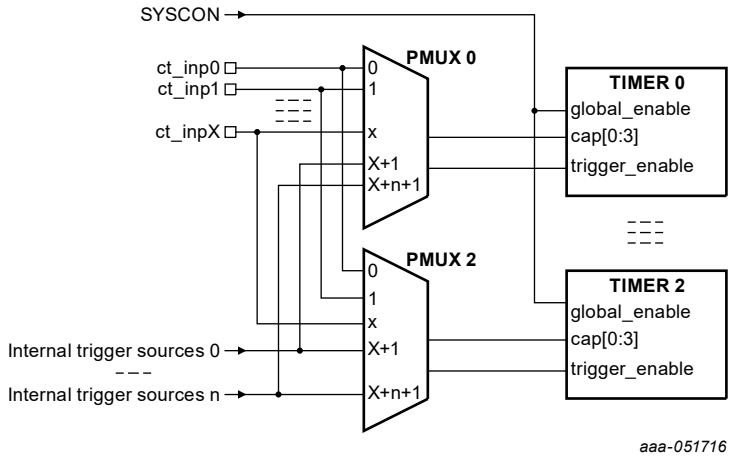


Figure 64. Peripheral Input Multiplexers for CTimers

26.1.5 Initialization

1. Select a clock source for the CTIMER using CTIMER0 clock selection control (MRCC_CTIMER0_CLKSEL), CTIMER1 clock selection control (MRCC_CTIMER1_CLKSEL), CTIMER2 clock selection control (MRCC_CTIMER2_CLKSEL), CTIMER3 clock selection control (MRCC_CTIMER3_CLKSEL) and CTIMER4 clock selection control (MRCC_CTIMER4_CLKSEL) registers.
2. Enable the clock to the CTIMER via the CTIMER0_CLK_EN, CTIMER1_CLK_EN, CTIMER2_CLK_EN, CTIMER3_CLK_EN and CTIMER4_CLK_EN fields.
3. Clear the CTIMER peripheral reset using the (Peripheral Reset Control 0 (MRCC_GLB_RST0) registers.
4. Each CTIMER provides interrupts to the NVIC. See MCR and CCR registers in the CTIMER register section for match and capture events. For interrupt connections, see the attached NVIC spreadsheet.
5. Select timer pins and pin modes as needed through the relevant PORT registers.
6. The CTIMER DMA request lines are connected to the DMA trigger inputs via the DMAC0_ITRIG_INMUX registers (See Memory map and register definition). Note that timer DMA request outputs are connected to DMA trigger inputs.

26.2 Overview

Each CTIMER is designed to count cycles of the CTIMER function clock or an externally supplied clock. A CTIMER can optionally generate an interrupt or perform other actions at specified timer values based on the settings of Match (MR0 - MR3). Each CTIMER also includes capture input pins (see Capture mode) to capture the timer value when an input signal transitions, optionally generating an interrupt.

In PWM mode, three match registers can be used to provide a single-edge controlled PWM output on the match output pins. One match register is used to control the PWM cycle length. All match registers can optionally be auto-reloaded from a companion shadow register (see Match Shadow (MSR0 - MSR3)) whenever the counter is reset to zero. This feature permits modifying the

match values for the next counter cycle without disrupting PWM waveforms during the current cycle. When enabled, match reload occurs whenever the counter is reset, either because of a match event or by writing 1 to TCR[CRST].

26.2.1 Block diagram

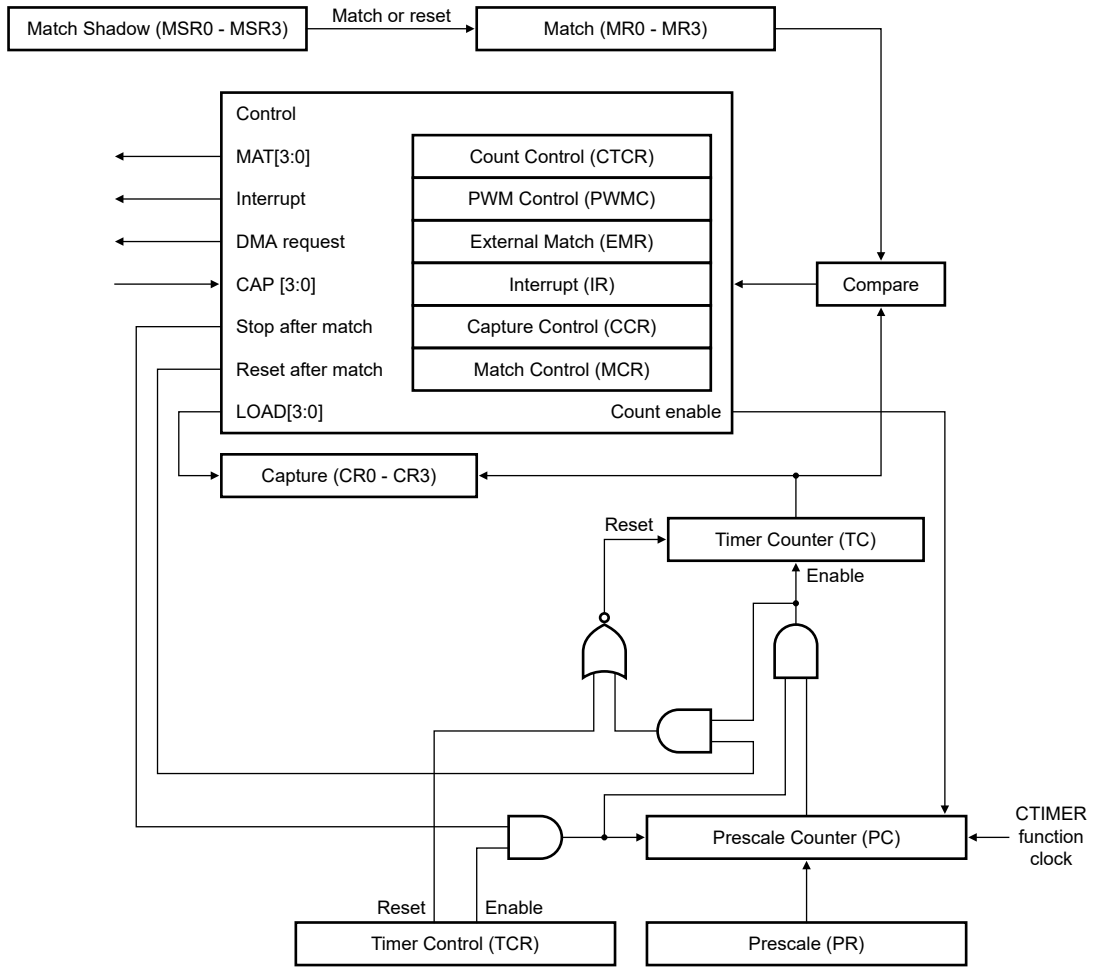


Figure 65. Block diagram

26.2.2 Features

- Contains a 32-bit CTIMER with a programmable 32-bit prescaler (the timers include external capture and match pin connections)
- Contains Timer Control (TCR) and External trigger enable, either of which you can use to enable each timer (you can also use an external global start enable register to globally enable one or more timers)
- Performs counter or timer operations
- Selects the function clocks that may be asynchronous to other system clocks
- Includes up to four 32-bit captures (see Capture mode) to take a snapshot of the timer value when an input signal transitions (a capture event may also optionally generate an interrupt, and the number of capture inputs available for each timer on a chip may vary)
- Enables you to configure the timer and prescaler to be cleared on a designated capture event (this feature permits easy pulse-width measurement by clearing the timer on the leading edge of an input pulse, capturing the timer value on the trailing edge)

- Includes Match (MR0 - MR3) to allow:
  - Continuous operation with optional interrupt generation after a match occurs
  - Optional auto-reload from Match Shadow (MSR0 - MSR3) when the counter is reset
  - Stop timer after a match with optional interrupt generation
  - Reset timer after a match with optional interrupt generation
- Generates up to four external outputs corresponding to Match (MR0 - MR3), for each timer, with the following capabilities (the number of match outputs available for each timer on a chip may vary):
  - Go low when matched
  - Go high when matched
  - Toggle when matched
  - Do nothing
- Enables you to configure up to four match registers for PWM operation, allowing up to three single-edged controlled PWM outputs (the number of match outputs available for each timer on a chip may vary)
- Uses up to two match registers to generate DMA requests

26.3 Functional description

Figure 66 shows a timer configured to reset the count and generate an interrupt after a match. The value of Prescale (PR) is 2 and the value of Match (MR0 - MR3) is 6. At the end of the timer cycle where the match occurs, the timer count is reset and gives a full-length cycle to the match value. The interrupt indicating that a match has occurred is generated in the next clock after the timer reaches the match value. Timer mode is selected in this case.

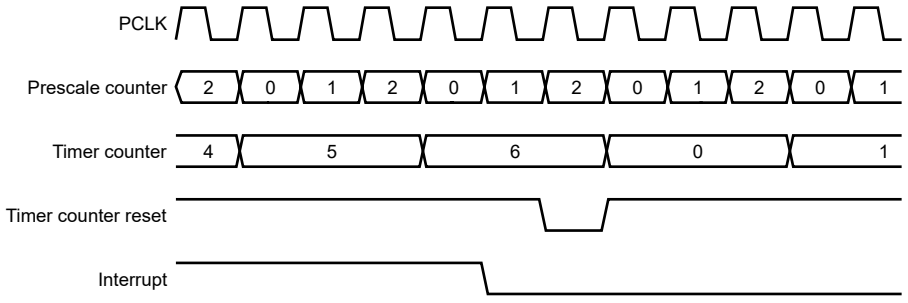


Figure 66. Timer cycle with PR = 2, MR*x* = 6, and both interrupt and reset enabled after match

Figure 67 shows a timer configured to stop and generate an interrupt after a match. The value of Prescale (PR) is 2 and the value of Match (MR0 - MR3) is 6. In the next clock after the timer reaches the match value, TCR[CEN] becomes 0 and the interrupt indicating that a match has occurred is generated.

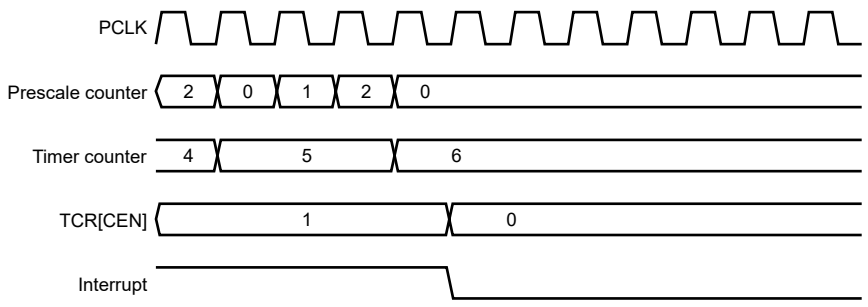


Figure 67. Timer cycle with PR = 2, MR*x* = 6, and both interrupt and stop enabled after match

26.3.1 Operation sections

All single-edge controlled PWM outputs go low at the beginning of each PWM cycle (timer is set to zero) unless their match value is zero.

Each PWM output goes high when its match value is reached. If no match occurs (that is, the match value is greater than the PWM cycle length), the PWM output remains continuously low.

If a match value larger than the PWM cycle length is written to Match (MR0 - MR3), and the PWM signal is already high, the PWM signal is cleared at the start of the next PWM cycle.

If a match register contains the same value as the timer reset value (the PWM cycle length), the PWM output is reset to low on the next clock tick after the timer reaches the match value. Therefore, the PWM output always consists of one clock tick wide positive pulse with a period that the PWM cycle length determines (that is, the timer reload value).

If Match (MR0 - MR3) = 0, the PWM output goes to high the first time the timer goes back to zero, and remains high continuously.

The following figure shows sample PWM waveforms with a PWM cycle length of 100 that MR3 specifies and PWM outputs that PWM Control (PWMC) enables. Counter mode is selected in this case.

NOTE

When you select match outputs to perform as PWM outputs, you must write 0 to MCR[MR*n*R] and MCR[MR*n*S], except to the match register setting the PWM cycle length. For this register, write 1 to MCR[MR*n*R] to enable the timer reset when the timer value matches the value of the corresponding match register.

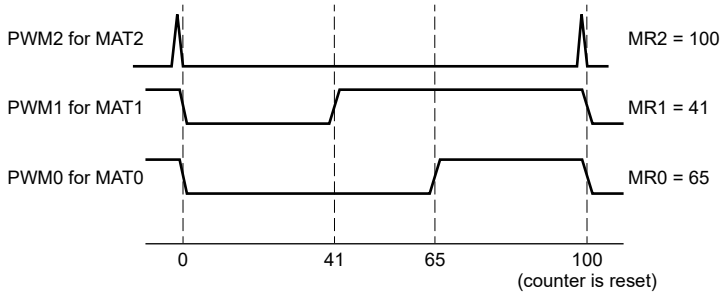


Figure 68. Sample PWM waveforms with a PWM cycle length of 100 (selected by MR3) and MAT3:0 enabled as PWM outputs by the PWM Control register.

26.3.2 Mode sections

26.3.2.1 Capture mode

You can configure a CAP to load Capture (CR0 - CR3) with the value in the counter or timer and optionally generate an interrupt. One of the pins generates the capture signal with a capture function. Each CAP is connected to one capture channel of the timer.

CTIMER can select a CAP as a clock source instead of selecting the APB CTIMER function clock. See Count Control (CTCR) for more information.

26.3.2.2 Match mode

When the value of a match register (Match (MR0 - MR3)) equals that of Timer Counter (TC), the corresponding match output can either toggle, go low, go high, or do nothing. External Match (EMR) and PWM Control (PWMC) control the functionality of this output.

26.3.3 External trigger enable

When trigger_enable is asserted (rising edge), it is equivalent to writing 1 to TCR[CEN]. If TCR[CEN] is already 1, trigger_enable rising edge events have no effect.

26.3.4 Clocking

Table 161. CTIMER clocks

Type of clock	Description
Functional Clock(ctclk)	Asynchronous to the bus clock, provide clock to perform counter or timer operations.
Bus Clock(pclk)	The bus clock is only used for bus accesses to the control and configuration registers.

26.3.5 Reset

TCR[CRST] resets Timer Counter (TC) and Prescale Counter (PC).

26.3.6 Interrupts

This module has capture and match interrupts. See the Interrupt (IR)

26.3.7 DMA

DMA requests are generated when the value of Timer Counter (TC) either matches the value of MR0 or MR1 (see Match (MR0 - MR3) for more information). This DMA request is not connected to the operation of the match outputs (see Match mode) controlled by External Match (EMR). Each match sets a DMA request flag, which is connected to the DMA controller. You must configure the DMA controller correctly.

When a timer is initially set up to generate a DMA request, the request may already be asserted before a match condition occurs. An initial DMA request may be avoided by writing 1 to the interrupt flag location twice, as if clearing a timer interrupt. See Interrupt (IR) for more information. A DMA request is cleared automatically when the DMA controller manages it.

NOTE

Timer DMA requests are generated whenever the timer value is equal to the related match register value. DMA requests are always generated when the timer is running, unless the match register value is higher than the upper count limit of the timer. It is important not to select and enable timer DMA requests in the DMA block unless you configure the timer correctly to generate valid DMA requests.

26.4 External signals

Table 162. CTIMER pin descriptions

Pin	Type	Description
CTIMER*m*_CAP*n*	Input	Capture signals—You can configure a transition on a capture pin to load one of the capture registers (Capture (CR0 - CR3)) with the value of Timer Counter (TC) and optionally generate an interrupt. You can select the capture functionality for a number of pins. CTIMER can select a capture signal as a clock source instead of the APB bus clock. See Count Control (CTCR) for more information.
CTIMER*m*_MAT*n*	Output	External match output—When the value of a match register (Match (MR0 - MR3)) equals the value of Timer Counter (TC), this output can either toggle, go low, go high, or do nothing. External Match (EMR) controls the functionality of this output. You can select the match output functionality for a number of pins in parallel.

26.4.1 Multiple capture (CAP) and match (MAT) pins

You can configure the pins of the chip to be used for this module. These pins are generally reused, and you can connect several internal module pins selectively. However, you can connect only one internal pin at a time.

NOTE

Match conditions may be used internally without the use of a chip's pin.

26.5 Initialization

See the Chip-specific CTIMER information for the initialization section for more details or steps.

26.6 Application information

CTIMER register descriptions discusses the following:

- Interval timer for counting internal events
- PWM outputs
- Pulse width demodulator via capture inputs
- Free running timer

26.7 CTIMER register descriptions

26.7.1 CTIMER memory map

CTIMER0 base address: 4000_4000h

CTIMER1 base address: 4000_5000h

CTIMER2 base address: 4000_6000h

CTIMER3 base address: 4000_7000h

CTIMER4 base address: 4000_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Interrupt (IR)	32	RW	See section
4h	Timer Control (TCR)	32	RW	See section
8h	Timer Counter (TC)	32	RW	0000_0000h
Ch	Prescale (PR)	32	RW	0000_0000h
10h	Prescale Counter (PC)	32	RW	0000_0000h
14h	Match Control (MCR)	32	RW	See section
18h - 24h	Match (MR0 - MR3)	32	RW	0000_0000h
28h	Capture Control (CCR)	32	RW	See section
2Ch - 38h	Capture (CR0 - CR3)	32	R	0000_0000h
3Ch	External Match (EMR)	32	RW	See section
70h	Count Control (CTCR)	32	RW	See section
74h	PWM Control (PWMC)	32	RW	See section
78h - 84h	Match Shadow (MSR0 - MSR3)	32	RW	0000_0000h

26.7.2 Interrupt (IR)

Offset

Register	Offset
IR	0h

Function

Provides flags for match interrupts and capture interrupts. If an interrupt is generated, it sets the corresponding flag in this register. Otherwise, the flag remains cleared. Writing 1 to the corresponding IR field resets the interrupt. Writing 0 has no effect. Clearing an interrupt for a timer match also clears any corresponding DMA request. You can write to this register to clear interrupts, and can read this register to identify which interrupt sources are pending.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
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	Reserved								CR3INT	CR2INT	CR1INT	CR0INT	MR3INT	MR2INT	MR1INT	MR0INT
Reset	u	u	u	u	u	u	u	u	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-4 CRnINT	Interrupt Flag for Capture Channel n Event Controls an interrupt flag for capture channel *n* events.
3-0 MRnINT	Interrupt Flag for Match Channel n Event Controls an interrupt flag for match channel *n* events.

26.7.3 Timer Control (TCR)

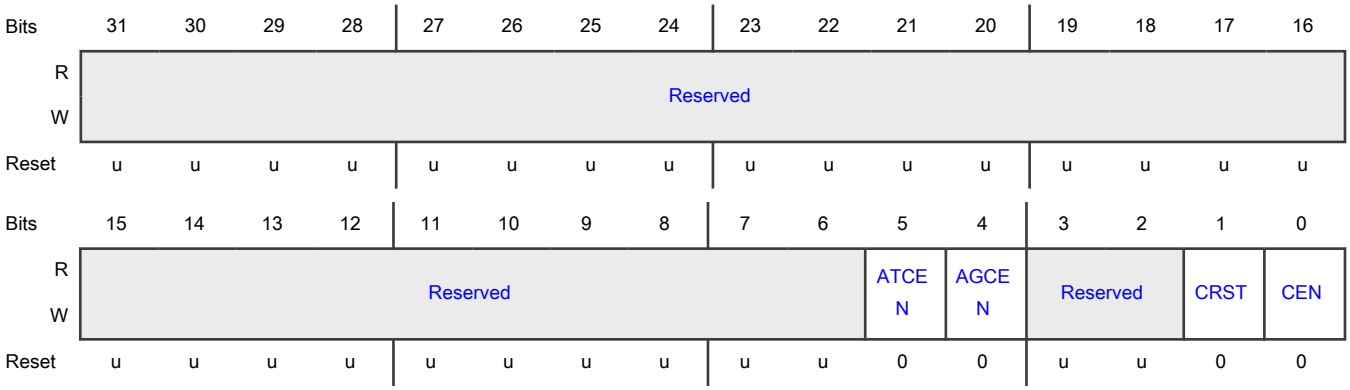
Offset

Register	Offset
TCR	4h

Function

Controls timer counter functions. You can disable or reset Timer Counter (TC) by using this register.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 ATCEN	Allow Trigger Count Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the input trigger. If this field is 1, it allows the input trigger_enable = 1 action to take effect (see External trigger enable). If this field is 0, the action is not allowed. 0b - Disable 1b - Enable
4 AGCEN	Allow Global Count Enable Enables the global count. If this field is 1, it allows the input global_enable = 1 action to take effect. If this field = 0, the action is not allowed. 0b - Disable 1b - Enable
3-2 —	Reserved
1 CRST	Counter Reset Enable Enables the counter reset. If this field is 1, Timer Counter (TC) and Prescale Counter (PC) are synchronously reset after the next positive edge of the CTIMER function clock. The counters remain reset until this field becomes 0. If this field is 0, no effect takes place. 0b - Disable 1b - Enable
0 CEN	Counter Enable Enables the counters. If this field is 1, Timer Counter (TC) and Prescale Counter (PC) are enabled. When an external trigger enables the timer or the external global start enable register enables it, this field becomes 1 automatically. 0b - Disable 1b - Enable

26.7.4 Timer Counter (TC)

Offset

Register	Offset
TC	8h

Function

Specifies the timer counter.

This register is incremented when Prescale Counter (PC) reaches its terminal count. Unless it is reset before reaching its upper limit, Timer Counter (TC) counts up through the value FFFF_FFFFh, and then wraps back to the value 0000_0000h. This event does not cause an interrupt, but you can use Match (MR0 - MR3) to detect an overflow if needed.

Timer Counter (TC) is incremented after every Prescale (PR) + 1 cycle of the CTIMER function clock. Timer Control (TCR) controls the functioning of this register.

Diagram

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

Fields

Field	Function
31-0 TCVAL	Timer Counter Value Specifies the value of the timer counter.

26.7.5 Prescale (PR)

Offset

Register	Offset
PR	Ch

Function

Specifies the prescale value.

If Prescale Counter (PC) is equal to the prescale value, the next clock timer increments Timer Counter (TC) and turns the value of Prescale Counter (PC) to 0.

Diagram

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

Fields

Field	Function
31-0 PRVAL	Prescale Reload Value Specifies the value of the prescale reload.

26.7.6 Prescale Counter (PC)

Offset

Register	Offset
PC	10h

Function

Controls the division of the CTIMER function clock by some constant value before it is applied to Timer Counter (TC). This allows control of the relationship between the resolution of the timer and the maximum time before the timer overflows.

Prescale Counter (PC) is incremented on every CTIMER function clock. When it reaches the value stored in Prescale (PR), Timer Counter (TC) is incremented and Prescale Counter (PC) is reset on the next CTIMER function clock. This causes Timer Counter (TC) to increment on every CTIMER function clock when Prescale (PR) = 0, every two CTIMER function clocks when Prescale (PR) = 1, and so on.

You can use the bus interface to observe and control Prescale Counter (PC).

Diagram

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

Fields

Field	Function
31-0 PCVAL	Prescale Counter Value Specifies the value of the prescale counter.

26.7.7 Match Control (MCR)

Offset

Register	Offset
MCR	14h

Function

Controls the operations that are performed when one of the match registers (Match (MR0 - MR3)) matches Timer Counter (TC).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R					MR3RL	MR2RL	MR1RL	MR0RL								
W	Reserved								Reserved							
Reset	u	u	u	u	0	0	0	0	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					MR3S	MR3R	MR3I	MR2S	MR2R	MR2I	MR1S	MR1R	MR1I	MR0S	MR0R	MR0I
W	Reserved															
Reset	u	u	u	u	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 MRnRL	Reload MR Reloads with the contents of the match n shadow register (see Match Shadow (MSR0 - MSR3)) when Timer Counter (TC) is reset to zero (either via a match event or by writing 1 to TCR[CRST]). 0b - Does not reload 1b - Reloads
23-12 —	Reserved
11 MR3S	Stop on MR3 Stops Timer Counter (TC) and Prescale Counter (PC), and turns TCR[CEN] to 0 if MR3 matches Timer Counter (TC). 0b - Does not stop 1b - Stops

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MR3R	Reset on MR3 Resets Timer Counter (TC) if MR3 matches its value. 0b - Does not reset 1b - Resets
9 MR3I	Interrupt on MR3 Generates an interrupt when MR3 matches the value in Timer Counter (TC). 0b - Does not generate 1b - Generates
8 MR2S	Stop on MR2 Stops Timer Counter (TC) and Prescale Counter (PC), and turns TCR[CEN] to 0 if MR2 matches Timer Counter (TC). 0b - Does not stop 1b - Stops
7 MR2R	Reset on MR2 Resets Timer Counter (TC) if MR2 matches its value. 0b - Does not reset 1b - Resets
6 MR2I	Interrupt on MR2 Generates an interrupt when MR2 matches the value in Timer Counter (TC). 0b - Does not generate 1b - Generates
5 MR1S	Stop on MR1 Stops Timer Counter (TC) and Prescale Counter (PC), and turns TCR[CEN] to 0 if MR1 matches Timer Counter (TC). 0b - Does not stop 1b - Stops
4 MR1R	Reset on MR1 Resets Timer Counter (TC) if MR1 matches its value. 0b - Does not reset 1b - Resets
3 MR1I	Interrupt on MR1 Generates an interrupt when MR1 matches the value in Timer Counter (TC).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Does not generate 1b - Generates
2 MR0S	Stop on MR0 Stops Timer Counter (TC) and Prescale Counter (PC), and turns TCR[CEN] to 0 if MR0 matches Timer Counter (TC). 0b - Does not stop 1b - Stops
1 MR0R	Reset on MR0 Resets Timer Counter (TC) if MR0 matches its value. 0b - Does not reset 1b - Resets
0 MR0I	Interrupt on MR0 Generates an interrupt when MR0 matches the value in Timer Counter (TC). 0b - Does not generate 1b - Generates

26.7.8 Match (MR0 - MR3)

Offset

Register	Offset
MR0	18h
MR1	1Ch
MR2	20h
MR3	24h

Function

Continuously compares the value of Match (MR0 - MR3) with the value of TC[TCVAL]. When both the values are equal, actions such as the following can be triggered automatically:

- Generation of an interrupt
- Resetting of Timer Counter (TC)
- Stopping of the timer

The settings defined in Match Control (MCR) control these actions.

If MCR[MRnRL] = 1, Match (MR0 - MR3) is automatically reloaded with the current contents of its corresponding Match Shadow (MSR0 - MSR3) whenever Timer Counter (TC) becomes 0. This transfer takes place on the same clock edge that advances Timer Counter (TC) to 0.

NOTE

Timer Counter (TC) resets in response to an occurrence of a match on MATCH being used to set the cycle counter rate. A reset can also occur if you write 1 to TCR[CRST].

Diagram

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

Fields

Field	Function
31-0 MATCH	Timer Counter Match Value Triggers an action automatically when the value of this field is the same as that of TC[TCVAL].

26.7.9 Capture Control (CCR)

Offset

Register	Offset
CCR	28h

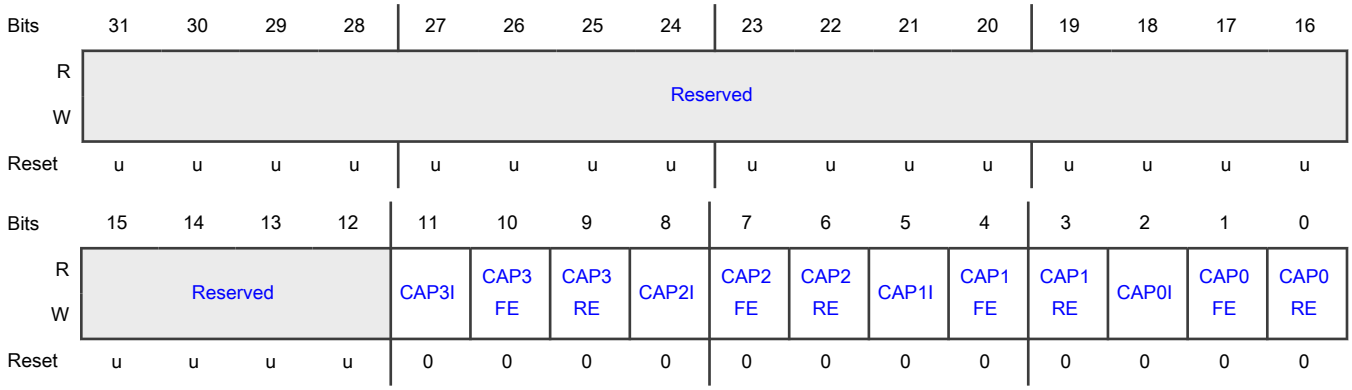
Function

Controls whether one of the capture registers (Capture (CR0 - CR3)) is loaded with the value in Timer Counter (TC) when the capture event occurs. The register also controls whether the capture event generates an interrupt. When you configure both the rising and falling bits at the same time, which is a valid configuration, it results into a capture event for both edges. In the description below, "*n*" represents the timer number, 0 or 1.

NOTE

If you select Counter mode for a particular CAP*n* input in Count Control (CTCR), you must program the fields in this register as 0. You can select both capture and interrupt for the other CAP*n* inputs.

Diagram



Fields

Field	Function
31-12 —	Reserved
11 CAP3I	Generate Interrupt on Channel 3 Capture Event Generates an interrupt on channel 3 capture event using a CR3 load. 0b - Does not generate 1b - Generates
10 CAP3FE	Falling Edge of Capture Channel 3 Loads CR3 with the contents of Timer Counter (TC) when you write 1 and then 0 to this field. 0b - Does not load 1b - Loads
9 CAP3RE	Rising Edge of Capture Channel 3 Loads CR3 with the contents of Timer Counter (TC) when you write 0 and then 1 to this field. 0b - Does not load 1b - Loads
8 CAP2I	Generate Interrupt on Channel 2 Capture Event Generates an interrupt on channel 2 capture event using a CR2 load. 0b - Does not generate 1b - Generates
7 CAP2FE	Falling Edge of Capture Channel 2 Loads CR2 with the contents of Timer Counter (TC) when you write 1 and then 0 to this field. 0b - Does not load 1b - Loads

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 CAP2RE	Rising Edge of Capture Channel 2 Loads CR2 with the contents of Timer Counter (TC) when you write 0 and then 1 to this field. 0b - Does not load 1b - Loads
5 CAP1I	Generate Interrupt on Channel 1 Capture Event Generates an interrupt on channel 1 capture event using a CR1 load. 0b - Does not generates 1b - Generates
4 CAP1FE	Falling Edge of Capture Channel 1 Loads CR1 with the contents of Timer Counter (TC) when you write 1 and then 0 to this field. 0b - Does not load 1b - Loads
3 CAP1RE	Rising Edge of Capture Channel 1 Loads CR1 with the contents of Timer Counter (TC) when you write 0 and then 1 to this field. 0b - Does not load 1b - Loads
2 CAP0I	Generate Interrupt on Channel 0 Capture Event Generates an interrupt on channel 0 capture event using a CR0 load. 0b - Does not generate 1b - Generates
1 CAP0FE	Falling Edge of Capture Channel 0 Loads CR0 with the contents of Timer Counter (TC) when you write 1 and then 0 to this field. 0b - Does not load 1b - Loads
0 CAP0RE	Rising Edge of Capture Channel 0 Loads CR0 with the contents of Timer Counter (TC) when you write 0 and then 1 to this field. 0b - Does not load 1b - Loads

26.7.10 Capture (CR0 - CR3)

Offset

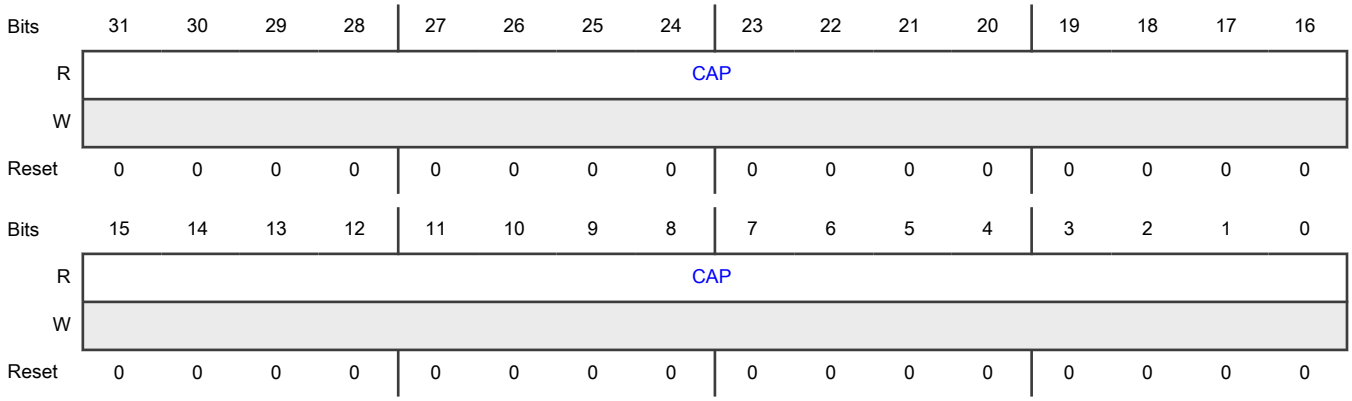
Register	Offset
CR0	2Ch
CR1	30h
CR2	34h
CR3	38h

Function

Works with a capture channel and may load with the value of Timer Counter (TC) when a specified event occurs on the signal defined for that capture channel. The signal could originate from an external pin or an internal source.

This register determines whether the capture function is enabled, and whether a capture event happens on the rising edge, falling edge, or on both edges of the associated signal. The register is loaded with the value of Timer Counter (TC) when there is an event on the CAP*n* input.

Diagram



Fields

Field	Function
31-0 CAP	Timer Counter Capture Value Determines whether the capture function is enabled and defines the edge on which the capture function occurs.

26.7.11 External Match (EMR)

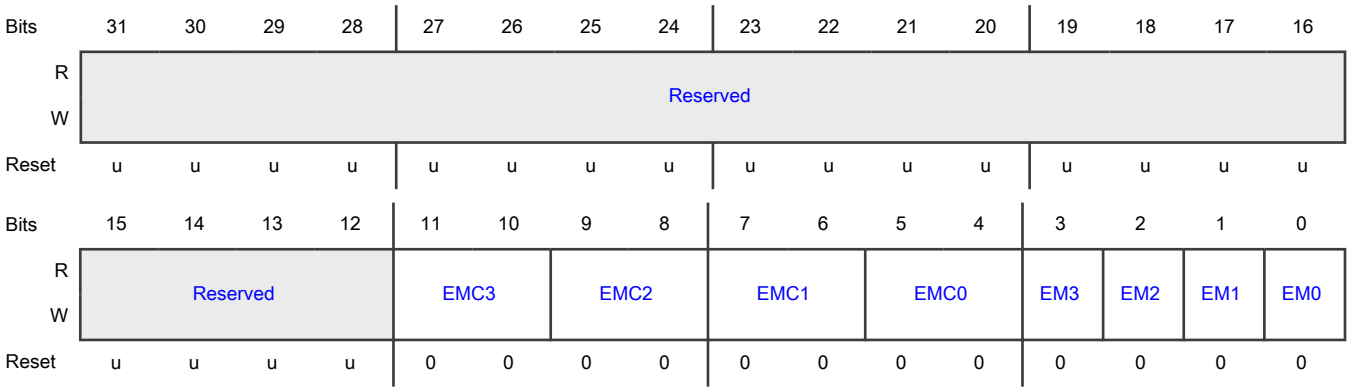
Offset

Register	Offset
EMR	3Ch

Function

Provides both control and status of the external match pins. If the match outputs (see Match mode) are configured as PWM outputs, PWM rules determine the function of this register.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-10 EMC3	External Match Control 3 Determines the functionality of EM3: • If this field is 1, the corresponding external match field or output becomes 0 (if pinned out MAT3 pin is low). • If this field is 10, the corresponding external match field or output becomes 1 (if pinned out MAT3 pin is high). • If this field is 11, it toggles the corresponding external match field or output. 00b - Does nothing 01b - Goes low 10b - Goes high 11b - Toggles
9-8 EMC2	External Match Control 2 Determines the functionality of EM2: • If this field is 1, the corresponding external match field or output becomes 0 (if pinned out MAT2 pin is low). • If this field is 10, the corresponding external match field or output becomes 1 (if pinned out MAT2 pin is high). • If this field is 11, it toggles the corresponding external match field or output. 00b - Does nothing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Goes low 10b - Goes high 11b - Toggles
7-6 EMC1	External Match Control 1 Determines the functionality of EM1: • If this field is 1, the corresponding external match field or output becomes 0 (if pinned out MAT1 pin is low). • If this field is 10, the corresponding external match field or output becomes 1 (if pinned out MAT1 pin is high). • If this field is 11, it toggles the corresponding external match field or output. 00b - Does nothing 01b - Goes low 10b - Goes high 11b - Toggles
5-4 EMC0	External Match Control 0 Determines the functionality of EM0: • If this field is 1, the corresponding external match field or output becomes 0 (if pinned out MAT0 pin is low). • If this field is 10, the corresponding external match field or output becomes 1 (if pinned out MAT0 pin is high). • If this field is 11, it toggles the corresponding external match field or output. 00b - Does nothing 01b - Goes low 10b - Goes high 11b - Toggles
3 EM3	External Match 3 Reflects the state of output MAT3—whether this output is connected to a pin. If a match occurs between Timer Counter (TC) and MR3 (see Match (MR0 - MR3)), this field can either toggle, go low, go high, or do nothing, as selected by EMC3. This field is driven to the MAT pins if the match function is selected via IOPCTL. 0b - Low 1b - High
2 EM2	External Match 2 Reflects the state of output MAT2—whether this output is connected to a pin. If a match occurs between Timer Counter (TC) and MR2 (see Match (MR0 - MR3)), this field can either toggle, go low, go high, or

Table continues on the next page...

Table continued from the previous page...

Field	Function
	do nothing, as selected by EMC2. This field is driven to the MAT pins if the match function is selected via IOPCTL. 0b - Low 1b - High
1 EM1	External Match 1 Reflects the state of output MAT1—whether this output is connected to a pin. If a match occurs between Timer Counter (TC) and MR1 (see Match (MR0 - MR3)), this field can either toggle, go low, go high, or do nothing, as selected by EMC1. This field is driven to the MAT pins if the match function is selected via IOPCTL. 0b - Low 1b - High
0 EM0	External Match 0 Reflects the state of output MAT0—whether this output is connected to a pin. If a match occurs between Timer Counter (TC) and MR0 (see Match (MR0 - MR3)), this field can either toggle, go low, go high, or do nothing, as selected by EMC0. This field is driven to the MAT pins if the match function is selected via IOPCTL. 0b - Low 1b - High

26.7.12 Count Control (CTCR)

Offset

Register	Offset
CTCR	70h

Function

Selects between Timer and Counter mode, and in Counter mode, selects the signal and edges for counting.

When Counter mode is selected as the mode of operation, the CAP input (selected by CINSEL) is sampled on every rising edge of the CTIMER function clock. After comparing two consecutive samples of this CAP input, one of the following four events is recognized:

- Rising edge
- Falling edge
- Either of the edges
- No change in the selected CAP input level

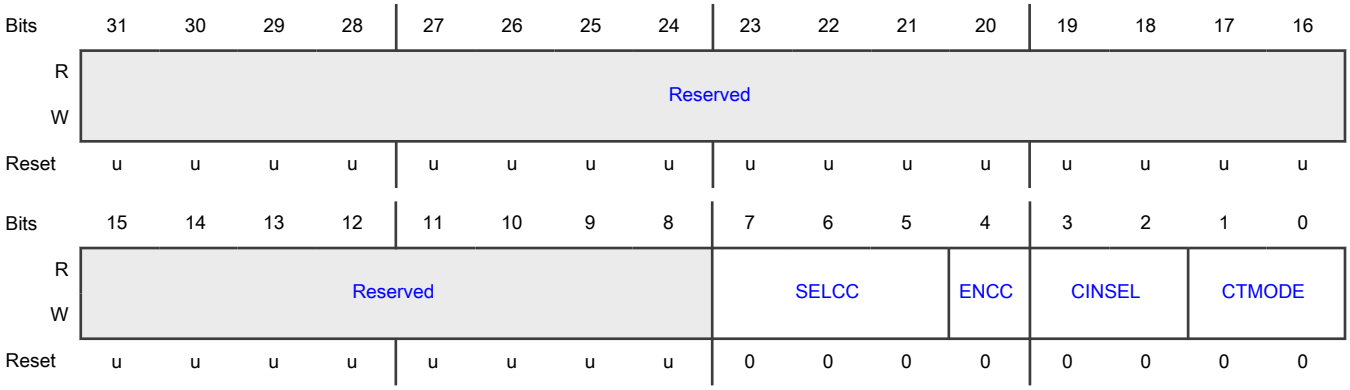
Only if the identified event occurs and the event corresponds to the one selected by CTMODE, Timer Counter (TC) is incremented.

Effective processing of the externally supplied clock to the counter has some limitations:

- Because two successive rising edges of the CTIMER function clock are used to identify only one edge on the CAP selected input, the frequency of the CAP input cannot exceed one half of the CTIMER function clock.
- Consequently, the duration of the high or low levels on the same CAP input in this case cannot be shorter than the CTIMER function clock duration.

You can also use ENCC and SELCC to enable and configure the capture-clears-timer feature. This feature allows for a designated edge on a particular CAP input to reset the timer to all zeros. Using this mechanism to clear the timer on the leading edge of an input pulse and perform a capture on the trailing edge permits direct pulse-width measurement using a single capture input without the need to perform a subtraction operation in software.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-5 SELCC	Edge Select Selects which input edge (rising or falling) causes the timer and prescaler to be cleared, when ENCC is 1. These fields have no effect when ENCC is 0. 000b - Capture channel 0 rising edge 001b - Capture channel 0 falling edge 010b - Capture channel 1 rising edge 011b - Capture channel 1 falling edge 100b - Capture channel 2 rising edge 101b - Capture channel 2 falling edge 110b - Capture channel 3 rising edge 111b - Capture channel 3 falling edge
4 ENCC	Capture Channel Enable Enables capture channel. Writing 1 to this field enables clearing of the timer and the prescaler when the capture-edge event specified in SELCC occurs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 CINSEL	Count Input Select Selects which CAP pin is sampled for clocking when CTMODE is not 0. **NOTE** If Counter mode is selected for a particular CAP*n* input in Count Control (CTCR), you must program the corresponding fields for that input in Capture Control (CCR) as 0. However, you can select both capture and interrupt for the other three CAP*n* inputs in the same timer. 00b - Channel 0, CAPn[0] for CTIMERn 01b - Channel 1, CAPn[1] for CTIMERn 10b - Channel 2, CAPn[2] for CTIMERn 11b - Channel 3, CAPn[3] for CTIMERn
1-0 CTMODE	Counter Timer Mode Selects between Timer and Counter mode, and in Counter mode, selects the signal and edges for counting: • If this field is 0, every rising CTIMER function clock edge is incremented. • If this field is 1, Timer Counter (TC) is incremented on the rising edges of the CAP input selected by CINSEL. • If this field is 10, Timer Counter (TC) is incremented on the falling edges of the CAP input selected by CINSEL. • If this field is 11, Timer Counter (TC) is incremented on both edges of the CAP input selected by CINSEL. 00b - Timer mode 01b - Counter mode rising edge 10b - Counter mode falling edge 11b - Counter mode dual edge

26.7.13 PWM Control (PWMC)

Offset

Register	Offset
PWMC	74h

Function

Configures match outputs (see Match mode) as PWM outputs. You can independently set each match output to perform either as a PWM output or match output controlled by External Match (EMR).

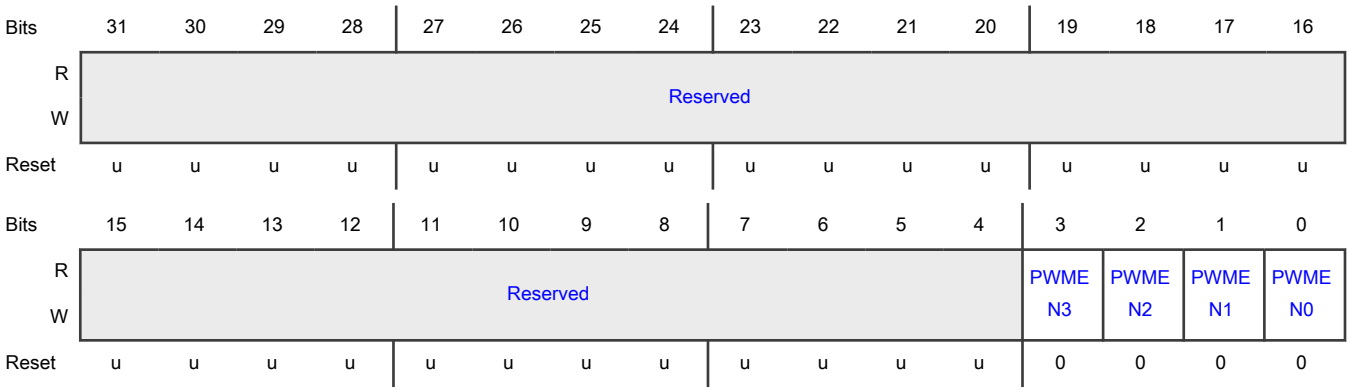
NOTE

Different part numbers and package variations may provide different match output pin functions.

For each timer, you can select a maximum of three single-edge controlled PWM outputs on the MAT*n*[2:0] outputs. One additional match register (see Match (MR0 - MR3)) determines the PWM cycle length. When a match occurs in any of the other match registers, the PWM output is set to high. The match register resets the timer that you can configure to set the PWM cycle length. When the timer is reset to zero, all currently high match outputs configured as PWM outputs are cleared.

This register enables PWM mode for the external match pins.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 PWMEN3	PWM Mode Enable for Channel 3 Enables PWM mode for channel 3. Use match channel 3 to set the PWM cycle length. If this field is 0, EM3 controls CTIMER*n*_MAT3. If this field is 1, PWM mode is enabled for CTIMER*n*_MAT3. 0b - Disable 1b - Enable
2 PWMEN2	PWM Mode Enable for Channel 2 Enables PWM mode for channel 2. If this field is 0, EM2 controls CTIMER*n*_MAT2. If this field is 1, PWM mode is enabled for CTIMER*n*_MAT2. 0b - Disable 1b - Enable
1 PWMEN1	PWM Mode Enable for Channel 1 Enables PWM mode for channel 1. If this field is 0, EM1 controls CTIMER*n*_MAT1. If this field is 1, PWM mode is enabled for CTIMER*n*_MAT1. 0b - Disable 1b - Enable
0	PWM Mode Enable for Channel 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
PWMEN0	Enables PWM mode for channel 0. If this field is 0, EM0 controls CTIMER*n*_MAT0. If this field is 1, PWM mode is enabled for CTIMER*n*_MAT0. 0b - Disable 1b - Enable

26.7.14 Match Shadow (MSR0 - MSR3)

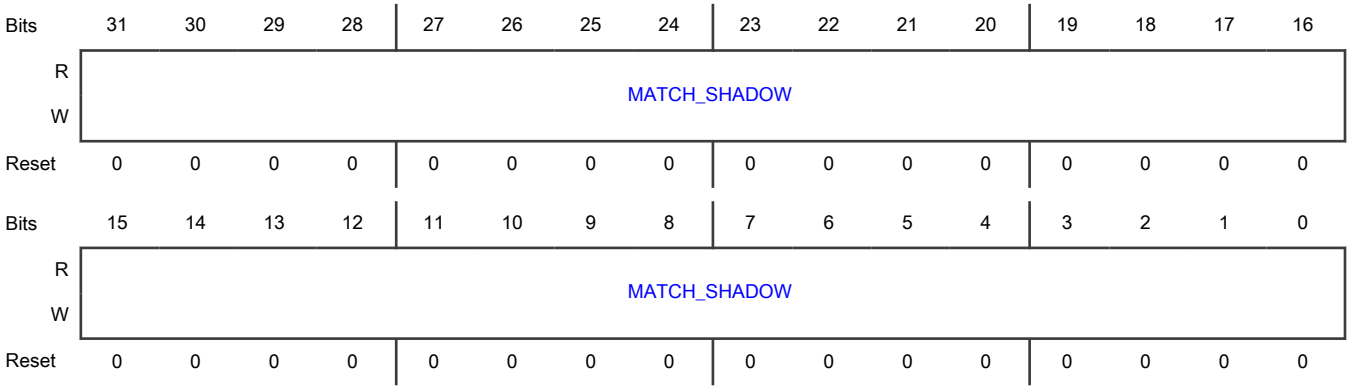
Offset

Register	Offset
MSR0	78h
MSR1	7Ch
MSR2	80h
MSR3	84h

Function

Contains the value that the corresponding Match (MR0 - MR3) are (optionally) reloaded with at the start of each new counter cycle. Typically, the match that causes the counter to be reset (and instigates the match reload) is also programmed to generate an interrupt or DMA request. Software or the DMA engine then has one full counter cycle to modify the contents of this register before the next reload occurs.

Diagram



Fields

Field	Function
31-0 MATCH_SHADOW	Timer Counter Match Shadow Value Reloads automatically with the contents of this register whenever Timer Counter (TC) is reset to zero.

Chapter 27
Windowed Watchdog Timer (WWDT)

27.1 Chip-specific WWDT information

Table 163. Reference links to related information

Topic	Related module	Reference
Full description	WWDT	WWDT
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

27.1.1 Module instances

This device has one instance of the WWDT module, WWDT0.

27.1.2 LOCK function

WWDT supports lock function. Once WWDT is locked, WWDT configuration, including clock, can’t be changed.

27.2 Overview

WWDT helps you reset or interrupt a microcontroller within a programmable time, if the microcontroller (or core) is stuck in an infinite loop or is executing an unintended code. If an application fails to reload or feed a watchdog timer within the predefined duration of time, it generates a watchdog reset (if enabled).

27.2.1 Block diagram

In the following WWDT block diagram, the synchronization logic (APB bus clock to WDCLK) is not shown. See the chip-specific WWDT information section or clocking chapter for more details.

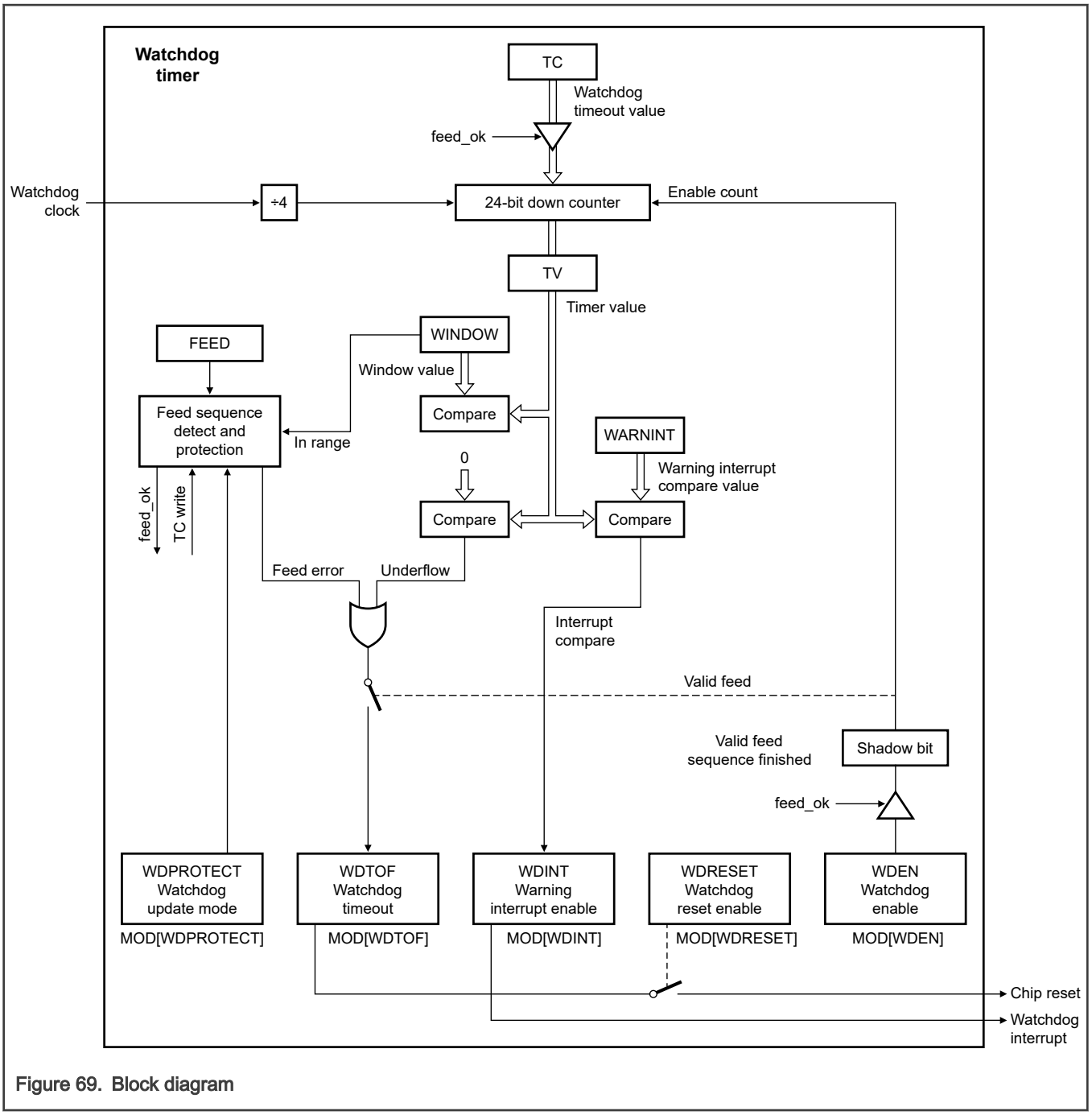


Figure 69. Block diagram

27.2.2 Features

- Can reset or interrupt the microcontroller within a programmable time
- Watchdog reset indication flag
- Clocking features:
  - Programmable 24-bit timer with an internal fixed (divided by 4) prescaler
  - Selectable time period in increments of four watchdog clocks: From 1024 watchdog clocks (T_{WDCLK} x 256 x 4) to over 67 million watchdog clocks (T_{WDCLK} x 2^{24} x 4)

  - Watchdog clock (WDCLK) source having a selectable frequency in the range of 6 kHz to 1.5 MHz (see the chip-specific information section and the Clocking chapter for detailed setting in this device)
- Can be configured to run in Low-Power mode (see the chip-specific information section and the Power Management chapter for detailed setting in this device)
- Other configurable features:
  - Windowed operation requiring a watchdog reload to occur between a programmable minimum and maximum timeout period
  - Generation of a warning interrupt at a programmable time before watchdog timeout
  - Protection of the watchdog reload value so that you can change it only after the warning interrupt time is reached

27.3 Functional description

When you program a watchdog window (a timing window), an early watchdog reload is also treated as a watchdog event, which prevents situations where a system failure may still reload the watchdog timer. For example, the application code could be stuck in an interrupt service that contains a watchdog reload (feed). You must set the timing window in a way that this situation causes an early watchdog timer reload, which generates a watchdog event, and then enables the system to recover from this situation.

WWDT consists of a fixed (divided by 4) prescaler and a 24-bit counter. The minimum value of the counter is FFh; if you write a value lower than FFh, the counter automatically loads with the value FFh.

- Minimum watchdog interval is (T_{WDCLK} X 256 X 4).
- Maximum watchdog interval is (T_{WDCLK} X 2^{24} X 4) in multiples of (T_{WDCLK} X 4).

27.3.1 Operating modes

This section describes all functional operating modes of the WWDT module: Disabled, Watchdog Interrupt, and Watchdog Reset.

Table 164. Watchdog operating modes selection

MOD[WDEN]	MOD[WDRESET]	Mode	Mode description
0	X (0 or 1)	Disabled	Debugs or operates without the watchdog running.
1	0	Watchdog Interrupt	Generates a watchdog warning interrupt but does not enable the watchdog reset. A watchdog counter equal to the value of WARNINT[WARNINT] sets MOD[WDINT] and generates a watchdog interrupt request.
1	1	Watchdog Reset	Enables both the watchdog interrupt and watchdog reset: • A watchdog counter equal to the value of WARNINT[WARNINT] sets MOD[WDINT] and generates a watchdog interrupt request. • A watchdog counter equal to 0 resets the microcontroller. • A watchdog feed before reaching the value of WINDOW[WINDOW] also causes a watchdog reset.

27.3.2 Example timing diagrams to show WWDT in operation

A feed is correct when both of the following conditions are true:

- A valid feed sequence completes, writing AAh followed by 55h to FEED[FEED].
- The value of TV[COUNT] is not greater than the value of WINDOW[WINDOW].

If either of these conditions is not true, a feed error occurs.

In the correct watchdog feed with Windowed mode enabled, the sequence of writing AAh followed by 55h occurs when the counter value (11FCh) is not greater than the value of WINDOW[WINDOW] (1200h). It is a correct feed (reload) event. After the correct feed event occurs, the watchdog counter reloads with the TC[COUNT] value (2000h).

In addition to the above two conditions, when MOD[WDPROTECT] = 1, the watchdog counter must not be greater than the value of WARNINT[WARNINT] for a correct feed to occur. See the following figure for a pictorial representation.

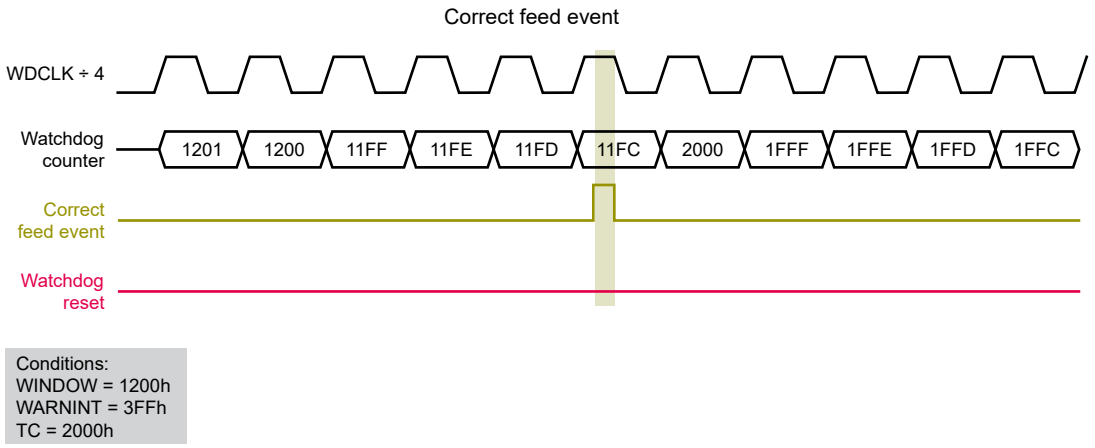


Figure 70. Correct watchdog feed with Windowed mode enabled

A correct feed sequence can only occur when the value of TV[COUNT] is not greater than the value of WINDOW[WINDOW] (1200h). Otherwise, a feed error occurs.

The following figure shows a reset trigger when you feed the watchdog too early.

An early feed event occurs if the feed sequence occurs when the watchdog counter value (1257h) is greater than the value of WINDOW[WINDOW] (1200h). This feed error can generate a reset if MOD[WDRESET] = 1.

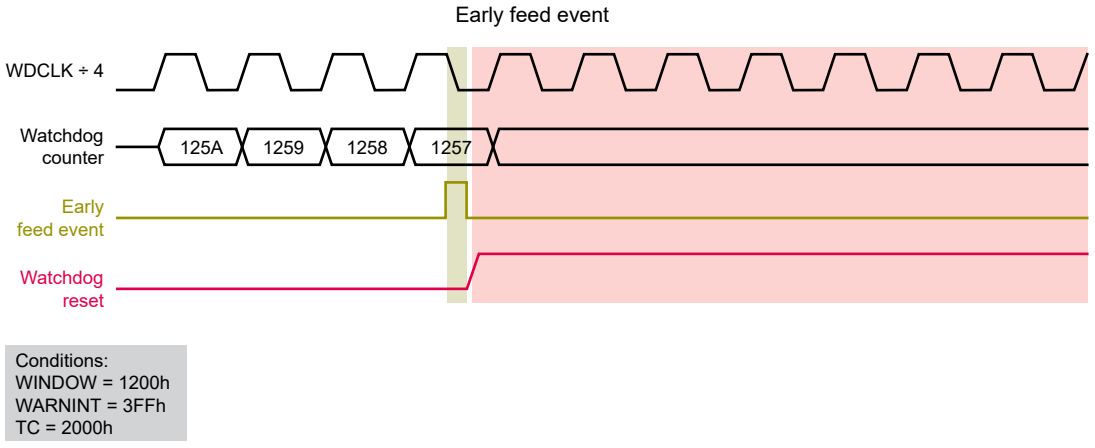


Figure 71. Early watchdog feed with Windowed mode enabled

The following figure shows WWDT generating a warning.

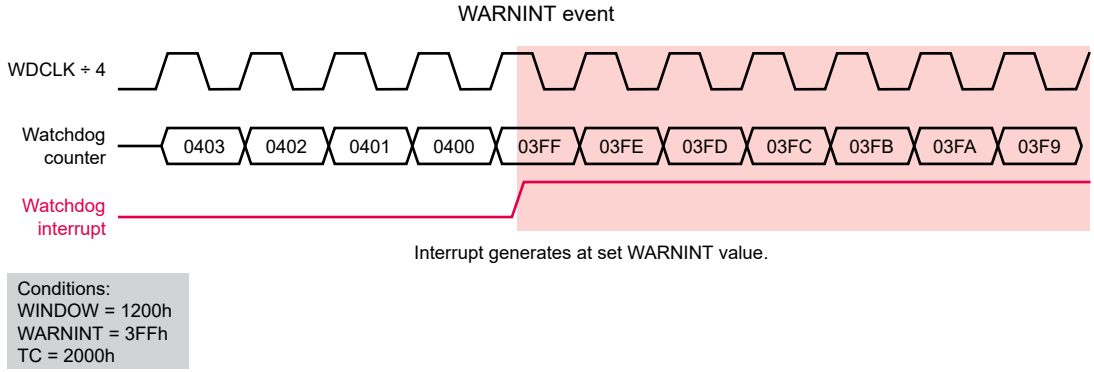


Figure 72. Watchdog warning interrupt

27.3.3 Clocking

WWDT uses two clocks: APB bus clock and WDCLK.

- The system clock derives the APB bus clock that is used for APB accesses to the watchdog registers.
- The watchdog oscillator derives WDCLK that is used for watchdog timer counting.

The synchronization logic between the two clock domains works as follows:

- When updating Mode (MOD) and Timer Constant (TC), the new value takes effect in three WDCLK cycles on the logic in the WDCLK clock domain.
- When the watchdog timer is counting on WDCLK, the synchronization logic first locks the value of the counter on WDCLK and then synchronizes it with the APB bus clock, so that the CPU can read Timer Value (TV).

NOTE

Because of the synchronization step, you must add a delay of three WDCLK clock cycles between the feed sequence and the time that you take to enable the functionality of MOD[WDPROTECT]. The length of the delay depends on the selected watchdog clock, WDCLK.

27.3.4 Using the WWDT lock

WWDT supports lock features to ensure continuous operation. These features prevent:

- The disabling of the WWDT clock source.
- The changing of the WWDT reload value.

27.3.4.1 Preventing the disabling of the WWDT clock source

If MOD[LOCK] = 1, the WWDT clock source is locked. You cannot disable the clock source when entering Sleep or Deep-Sleep modes. Therefore, enable the watchdog oscillator for each Power mode before writing 1 to MOD[LOCK].

27.3.4.2 Preventing the changing of the WWDT reload value

If MOD[WDPROTECT] = 1, then any attempt to change the value of TC[COUNT] with a feed sequence, before the watchdog counter value becomes less than the values of WARNINT[WARNINT] and WINDOW[WINDOW], sets MOD[WDTOF] and causes a watchdog reset.

Any type of reset disables the reload overwrite lock mechanism.

27.4 External signals

This module has no external signals.

27.5 Initialization

Perform the following steps to initialize WWDT:

1. Enable and configure the watchdog oscillator.
2. Set the watchdog timer constant reload value using TC[COUNT].
3. Set the watchdog timer operating mode using MOD[WDPROTECT].
4. Set a value for the watchdog window time using WINDOW[WINDOW], if the windowed operation is desired.
5. Set a value for the watchdog warning interrupt using WARNINT[WARNINT] and configure the system interrupt controller, if a warning interrupt is desired.
6. Enable the watchdog by writing AAh followed by 55h to FEED[FEED].

To prevent a watchdog event, you must feed or reload WWDT again before the watchdog counter reaches 0. If you set a value for the watchdog window time, then the feed or reload must also occur after the watchdog counter passes the value of WINDOW[WINDOW].

After you configure WWDT in a way that a watchdog event causes a reset (MOD[WDRESET] = 1), or when the counter reaches 0 (it causes a reset), then the microcontroller (or core) is also reset (loading the stack pointer and program counter from the vector table, just like the way it happens for an external reset).

27.6 Memory map and register definition

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

27.6.1 WWDT register descriptions

27.6.1.1 WWDT memory map

WWDT0 base address: 4000_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Mode (MOD)	32	RW	0000_0000h
4h	Timer Constant (TC)	32	RW	0000_00FFh
8h	Feed Sequence (FEED)	32	W	See section
Ch	Timer Value (TV)	32	R	0000_00FFh
14h	Warning Interrupt Compare Value (WARNINT)	32	RW	0000_0000h
18h	Window Compare Value (WINDOW)	32	RW	00FF_FFFFh

27.6.1.2 Mode (MOD)

Offset

Register	Offset
MOD	0h

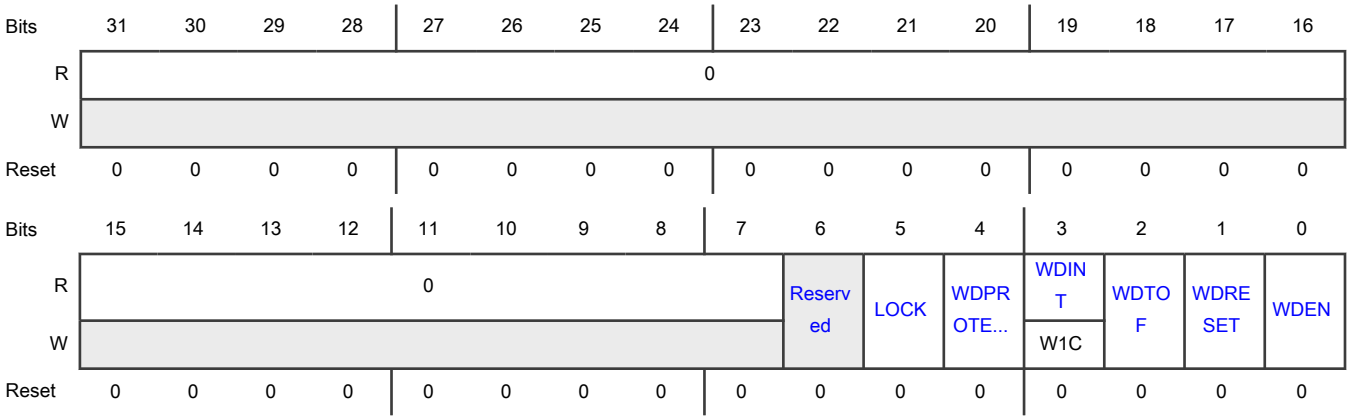
Function

Controls the operation of WWDT. You must perform a watchdog feed before any changes to this register take effect.

NOTE

After you write 1 to MOD[WDEN], MOD[WDPROTECT], or MOD[WDRESET], you can return those fields to 0 only with an external reset or a WWDT reset. Until those resets occur, writing 0 to the aforementioned fields has no effect.

Diagram



Fields

Field	Function
31-7 —	Reserved
6 —	Reserved
5 LOCK	Lock Prevents disabling or powering down of the watchdog oscillator after you write 1 to this field and perform a watchdog feed. You can write 1 to this field, which can become 0 only with a reset. 0b - No Lock 1b - Lock
4 WDPROTECT	Watchdog Update Mode Protects the value of TC[COUNT] from changing at any time. In Flexible mode (when this field is 0), you can change the value of TC[COUNT] at any time. In Threshold mode (when this field is 1), you can change the value of TC[COUNT] only after the counter is less than the individual values of WARNINT[WARNINT] and WINDOW[WINDOW]. Any attempt to change the value of TC[COUNT] with a feed sequence, before the watchdog counter value becomes less than the values of WARNINT[WARNINT] and WINDOW[WINDOW], sets MOD[WDTOF] and causes a watchdog reset. You can write 1 to this field, which can become 0 only with a reset (see the note in MOD register).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flexible 1b - Threshold
3 WDINT	Warning Interrupt Flag Sets the flag when the timer is at or below the value defined in WARNINT[WARNINT]. Note that you cannot clear this flag while the value of WARNINT[WARNINT] is equal to the value of TV[COUNT]. This can occur if the value of both WARNINT[WARNINT] and MOD[WDRESET] is 0 when TV[COUNT] decrements to 0. When you configure WWDT to generate a warning interrupt, the interrupt occurs when the counter is no longer greater than the value defined by WARNINT[WARNINT]. If a feed sequence occurs when the watchdog counter value is greater than the value of WINDOW[WINDOW], this flag is set. Any reset clears this flag. 0b - No flag 1b - Flag
2 WDTOF	Watchdog Timeout Flag The bit could be set if: • Watchdog timer timeout has occurred • Feed error has occurred You must clear this flag by writing 0 to it. An external reset or POR clears this flag too. 0b - Watchdog event has not occurred. 1b - Watchdog event has occurred (causes a chip reset if WDRESET = 1).
1 WDRESET	Watchdog Reset Enable Allows a watchdog timeout to cause a chip reset, if this field = 1. After you write 1 to this field, you cannot write 0 to it (see the note in MOD register). **NOTE** If you write 0 to this field, and after an interrupt occurs, do not write 1 to this field immediately unless you refresh the watchdog. 0b - Interrupt 1b - Reset
0 WDEN	Watchdog Enable Enables or disables the watchdog timer. After writing 1 to this field, perform a watchdog feed (reload) to enable the watchdog timer. After you write 1 to this field, you cannot write 0 to it (see the note in MOD register). 0b - Timer stopped

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Timer running

27.6.1.3 Timer Constant (TC)

Offset

Register	Offset
TC	4h

Function

Specifies the desired timeout value. Transferring the value of this register into the watchdog counter requires a feed sequence. The register resets to 00_00FFh. Writing a value below FFh causes 00_00FFh to load into the register. Therefore, the minimum timeout interval is T_{WDCLK} X 256 X 4.

If MOD[WDPROTECT] = 1, then any attempt to change the value of this register with a feed sequence, before the watchdog counter value becomes less than the individual values of WARNINT[WARNINT] and WINDOW[WINDOW], sets MOD[WDTOF] and causes a watchdog reset.

Diagram

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

Fields

Field	Function
31-24 —	Reserved
23-0 COUNT	Watchdog Timeout Value Specifies the desired timeout value for WWDT.

27.6.1.4 Feed Sequence (FEED)

Offset

Register	Offset
FEED	8h

Function

Reloads the watchdog timer with the TC[COUNT] value, if you write AAh followed by 55h to FEED[FEED]. This operation also starts WWDT, if the watchdog timer is enabled by writing 1 to MOD[WDEN].

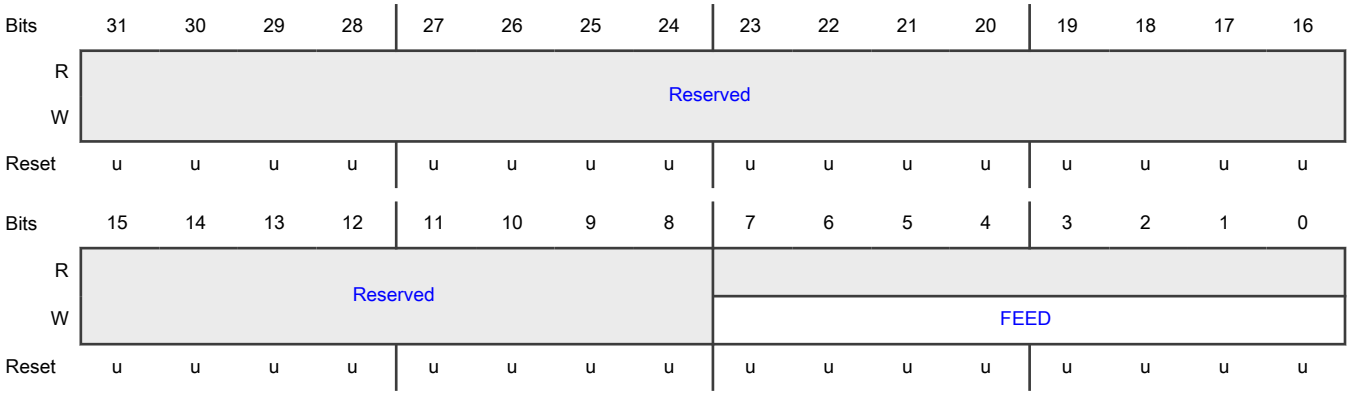
However, writing 1 to MOD[WDEN] is not sufficient to enable WWDT. Before the watchdog can generate a reset, MOD[WDEN] must be 1, followed by a valid feed sequence that completes itself. Until then, the watchdog ignores feed errors.

After writing AAh to FEED[FEED], access to any watchdog register other than writing 55h to FEED[FEED] causes an immediate reset or interrupt when WWDT is enabled and sets MOD[WDTOF]. An incorrect access to a watchdog register during a feed sequence generates the reset during the second APB bus clock.

To avoid an unintended interrupt, it is a good practice to disable interrupts around a feed sequence. Disable interrupts if they may result in rescheduling processor control away from the current task in the middle of the feed, and then lead to some other access to WWDT before control returns to the interrupted task.

If the value of WINDOW[WINDOW] is less than the default value, it may limit the time for which a watchdog feed is allowed.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 FEED	Feed Value Specifies feed value. The value must be AAh followed by 55h.

27.6.1.5 Timer Value (TV)

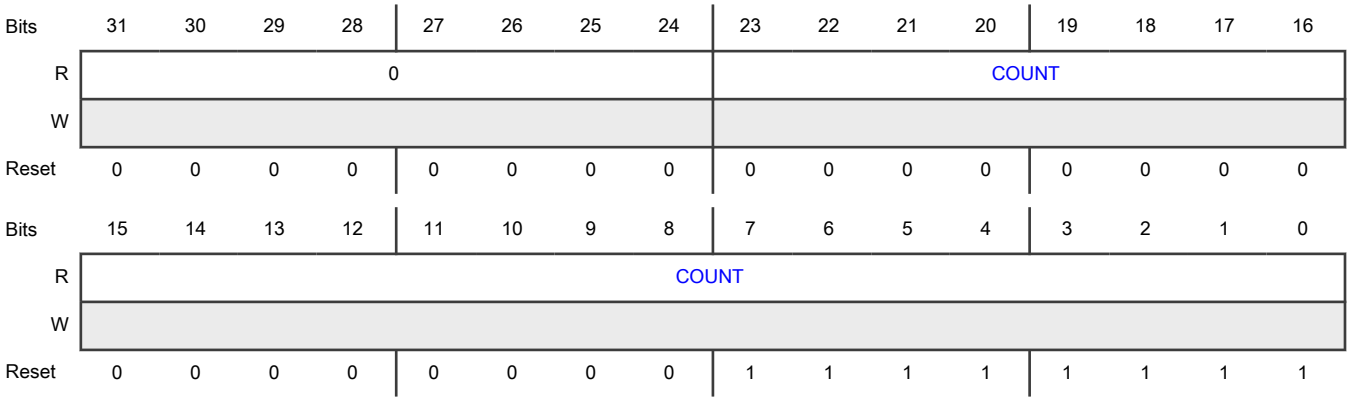
Offset

Register	Offset
TV	Ch

Function

Contains the current value of the WWDT counter. When the core (or CPU) reads this register, the value of this register is older than the actual value of the timer. Reading the timer using the lock and synchronization procedure takes up to six WDCLK cycles plus six APB bus clock cycles.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 COUNT	Counter Timer Value Specifies the current value of the watchdog timer.

27.6.1.6 Warning Interrupt Compare Value (WARNINT)

Offset

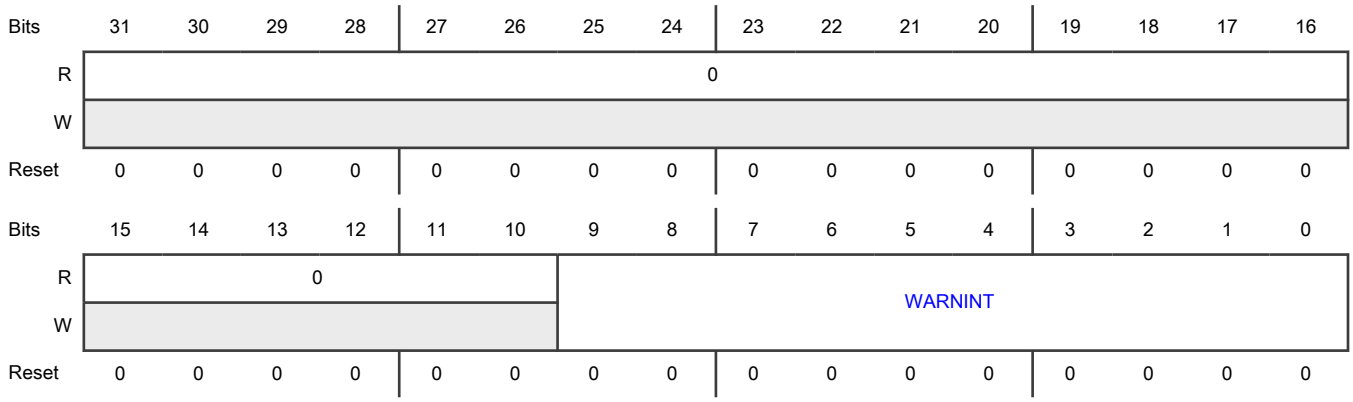
Register	Offset
WARNINT	14h

Function

Determines the value of TV[COUNT] that generates a watchdog interrupt. When this value is less than or equal to the value defined by WARNINT, an interrupt generates after the subsequent WDCLK.

A match of the WWDT counter to WARNINT occurs when the lower 10 bits of the timer counter have the same value as the lower 10 bits of WARNINT, and the remaining upper bits of the counter are all 0. This gives a maximum time of 1023 watchdog timer counts (4096 watchdog clocks) for the interrupt to occur before a watchdog event. If WARNINT is 0, then the interrupt occurs at the same time as the watchdog event.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 WARNINT	Watchdog Warning Interrupt Compare Value Specifies a timer value that generates a warning interrupt.

27.6.1.7 Window Compare Value (WINDOW)

Offset

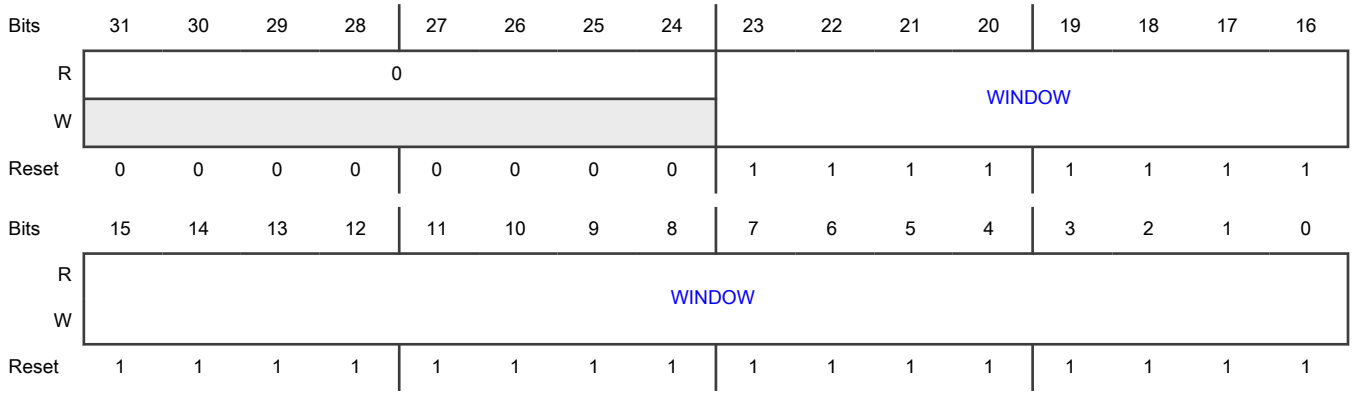
Register	Offset
WINDOW	18h

Function

Determines the highest timer value (TV[COUNT]) allowed during a watchdog feed. If a feed sequence occurs when the value of TV[COUNT] is greater than the value in WINDOW[WINDOW], a watchdog event occurs.

WINDOW[WINDOW] resets to the maximum value of TV[COUNT], where windowing is not in effect.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 WINDOW	Watchdog Window Value Specifies the highest timer value in which a watchdog feed can occur.

Chapter 28
Micro-Tick Timer (UTICK)

28.1 Chip-specific UTICK information

Table 165. Reference links to related information

Topic	Related module	Reference
Full description	UTICK	UTICK
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

28.1.1 Module instances

This device has one instance of the UTICK module, UTICK0.

28.2 Overview

UTICK is a 31-bit timer that counts down to 0 and then generates an interrupt. Thus, it provides a fixed time interval between interrupts. This is a simple, ultra-low-power timer that can run and wake up the chip from Low-power mode. See chip specific information for low power modes UTICK can wakeup.

28.2.1 Block diagram

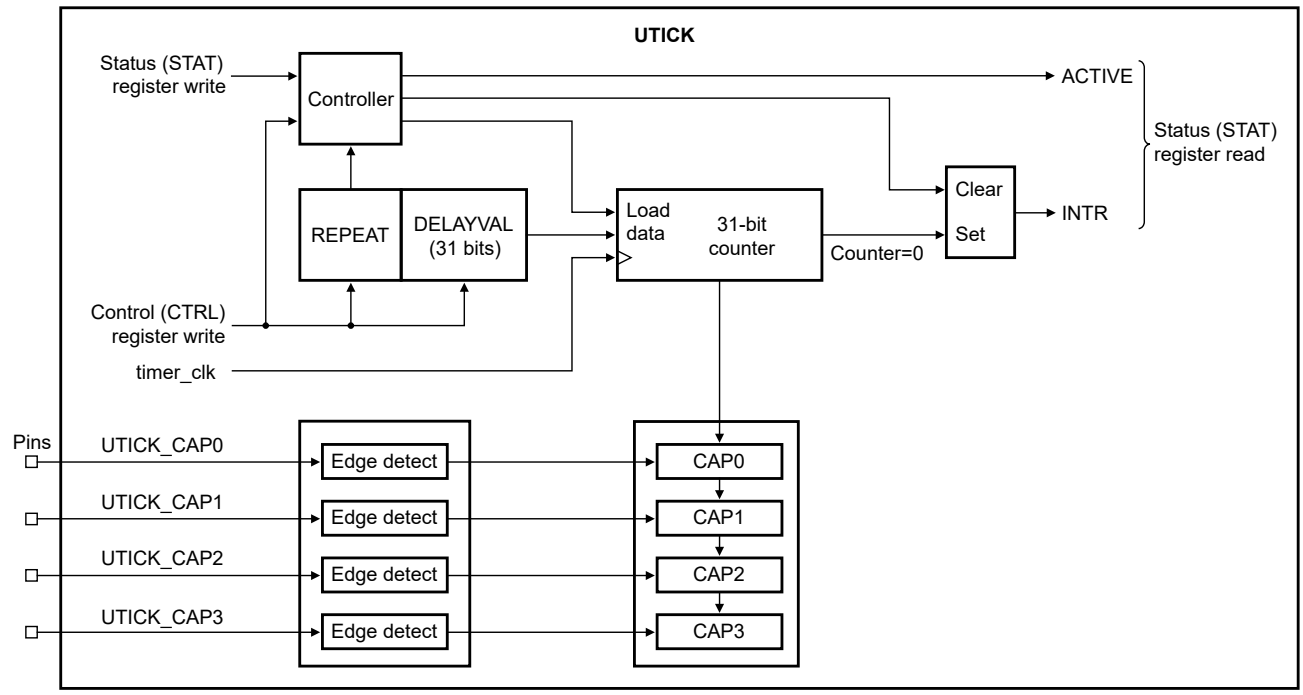


Figure 73. Block diagram

28.2.2 Features

- Wakes up the chip from Low-power mode.
- Starts with a nonzero value written to CTRL[DELAYVAL].
- Operates using interrupt or software polling mode.
- Includes Capture (CAP0 - CAP3) registers that use external pin transitions as a trigger.

28.3 Functional description

28.3.1 Operations

Perform the following procedure before using UTICK:

1. Configure CTRL[REPEAT] to either perform a one-time delay or continuous delay.
2. Configure CTRL[DELAYVAL] to specify the delay time value in terms of the timer clock.
3. Configure CFG[CAPEN*n*] to decide the number of events to capture (the maximum is four).
4. Configure CFG[CAPPOL*n*] to capture the polarity (whether to capture on a positive or negative edge of a signal).

28.3.2 Clocking

The UTICK clock is selectable from the following clock inputs:

Table 166. Clocking

Clock	Description
Global functional clock (timer_clk)	This clock is supposed to be on during normal operation. This clock is treated as asynchronous to pclk. **NOTE** See the Chip specific section for more details on UTICK clock.
System bus clock (pclk)	This clock is used only for register reads and writes.

28.3.3 Interrupts

The interrupt is generated when the tick_count reaches 1h so that on the next rising edge of timer_clk, it doesn't trigger again automatically. Its associated status field is STAT[INTR].

28.4 External signals

Signal	I/O	Description
UTICK_CAP0, UTICK_CAP1, UTICK_CAP2, UTICK_CAP3	I	Capture inputs. Configure the selected transition on a capture pin to load Capture (CAP0 - CAP3) with the counter value.

28.5 Initialization

To initialize UTICK:

1. Enable the clock to the UTICK register interface.
2. Enable UTICK interrupts for waking up from Low-power mode, configure Control (CTRL) register.
3. Configure Capture Configuration (CFG) register to enable UTICK capture functions and selects the polarity of capture triggers.
4. Enable the oscillator that provides the UTICK clock, after writing values corresponding registers UTICK timer works normally.

28.6 UTICK register descriptions

28.6.1 UTICK memory map

UTICK0 base address: 4000_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	32	RW	0000_0000h
4h	Status (STAT)	32	RW	See section
8h	Capture Configuration (CFG)	32	RW	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	Capture Clear (CAPCLR)	32	W	See section
10h - 1Ch	Capture (CAP0 - CAP3)	32	R	0000_0000h

28.6.2 Control (CTRL)

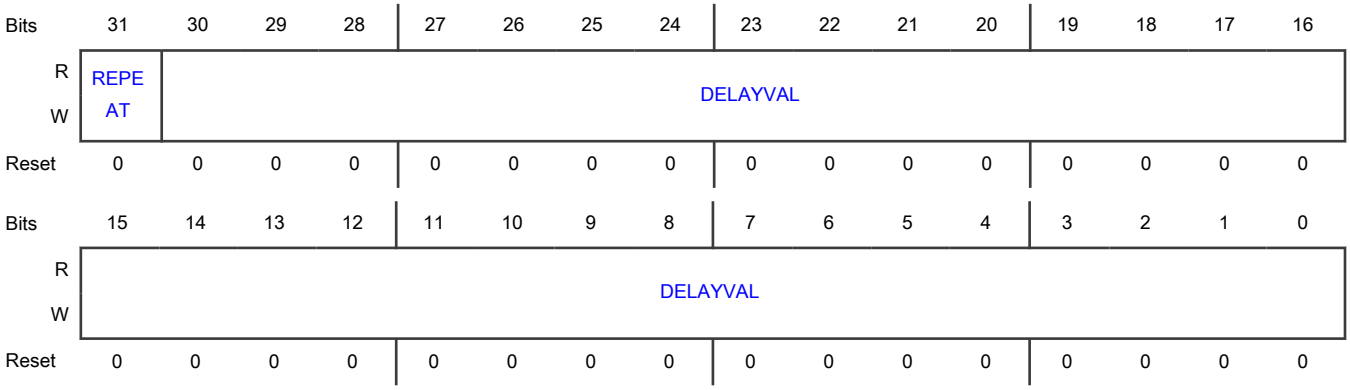
Offset

Register	Offset
CTRL	0h

Function

Resets the counter, which means a new interval is measured if one is in progress.

Diagram



Fields

Field	Function
31 REPEAT	Repeat Delay Specifies the frequency of repeat delay. 0b - One-time delay 1b - Delay repeats continuously
30-0 DELAYVAL	Tick Interval Specifies the tick interval. The resulting delay is (DELAYVAL + 1) periods of the timer clock. The minimum usable tick interval is one clock cycle, for a delay of two timer clocks. The timer will be stopped when the tick interval is equal to zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000_0000_0000_0000_0000_0000_0000_0000b - Reserved All other values - Clock cycles as defined in the description

28.6.3 Status (STAT)

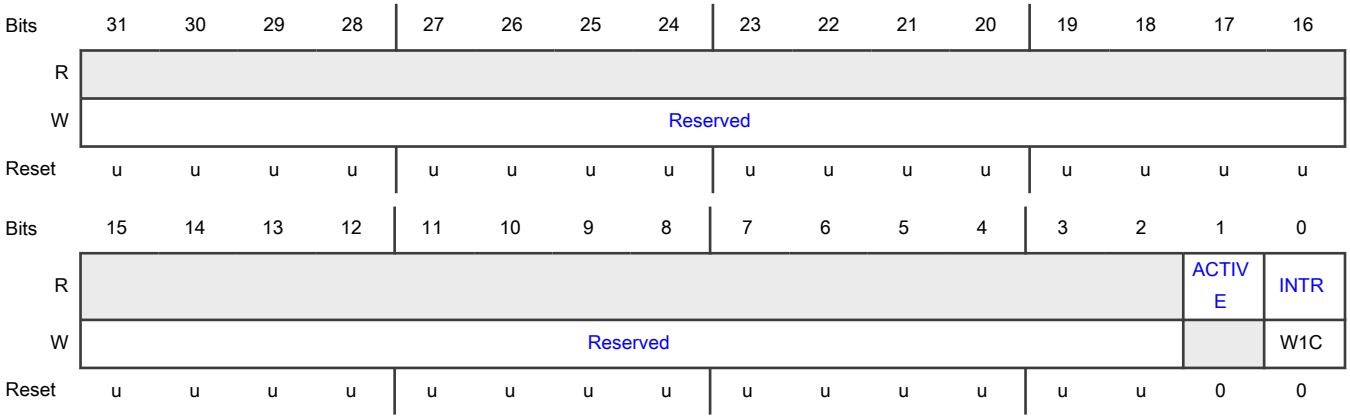
Offset

Register	Offset
STAT	4h

Function

Indicate the micro timer is working or stopping and any event on output compare channels. The bits are write one to clear.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 ACTIVE	Timer Active Flag Indicates whether UTICK is currently active. 0b - Inactive (stopped) 1b - Active
0 INTR	Interrupt Flag Indicates whether an interrupt is pending.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not pending 1b - Pending

28.6.4 Capture Configuration (CFG)

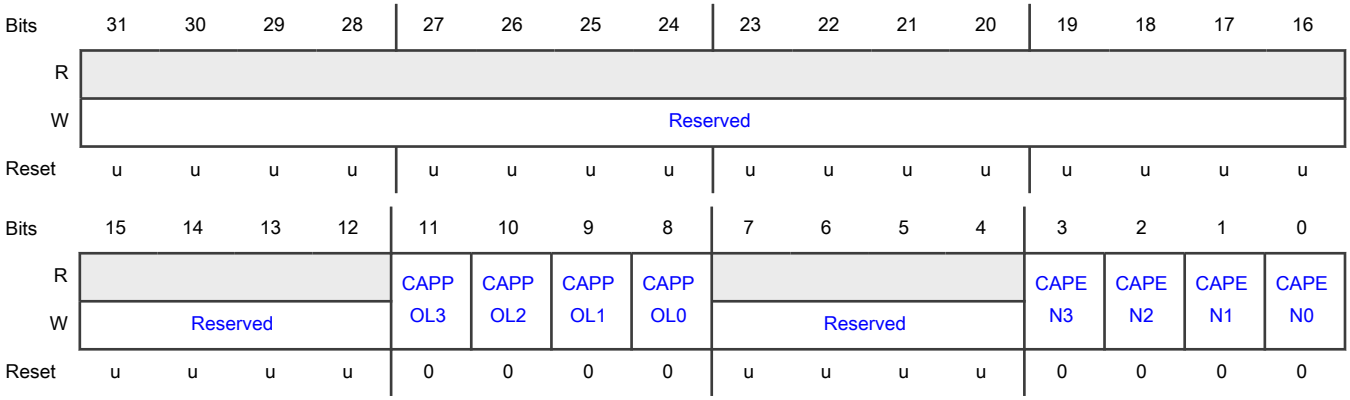
Offset

Register	Offset
CFG	8h

Function

Enables micro-tick capture functions and selects the polarity of capture triggers.

Diagram



Fields

Field	Function
31-12 —	Reserved
11 CAPPOL3	Capture Polarity 3 Specifies the capture edge. 0b - Positive 1b - Negative
10 CAPPOL2	Capture Polarity 2 Specifies the capture edge.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Positive 1b - Negative
9 CAPPOL1	Capture-Polarity 1 Specifies the capture edge. 0b - Positive 1b - Negative
8 CAPPOL0	Capture Polarity 0 Specifies the capture edge. 0b - Positive 1b - Negative
7-4 —	Reserved
3 CAPEN3	Enable Capture 3 0b - Disable 1b - Enable
2 CAPEN2	Enable Capture 2 0b - Disable 1b - Enable
1 CAPEN1	Enable Capture 1 0b - Disable 1b - Enable
0 CAPEN0	Enable Capture 0 0b - Disable 1b - Enable

28.6.5 Capture Clear (CAPCLR)

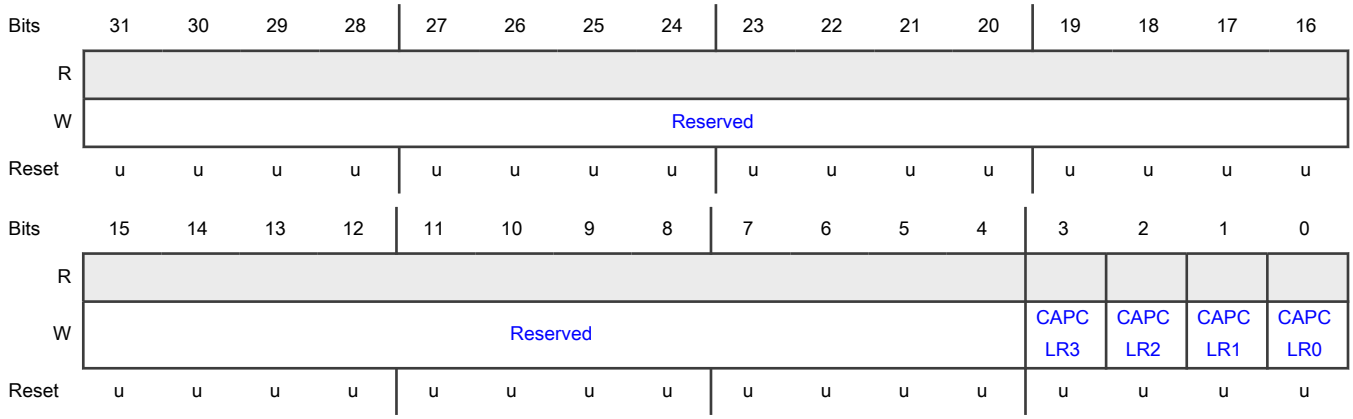
Offset

Register	Offset
CAPCLR	Ch

Function

Allows you to clear previous capture values, enabling new captures to take place.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 CAPCLR3	Clear Capture 3 0b - Does nothing 1b - Clears the CAP3 register value
2 CAPCLR2	Clear Capture 2 0b - Does nothing 1b - Clears the CAP2 register value
1 CAPCLR1	Clear Capture 1 0b - Does nothing 1b - Clears the CAP1 register value
0 CAPCLR0	Clear Capture 0 0b - Does nothing 1b - Clears the CAP0 register value

28.6.6 Capture (CAP0 - CAP3)

Offset

Register	Offset
CAP0	10h
CAP1	14h
CAP2	18h

Table continues on the next page...

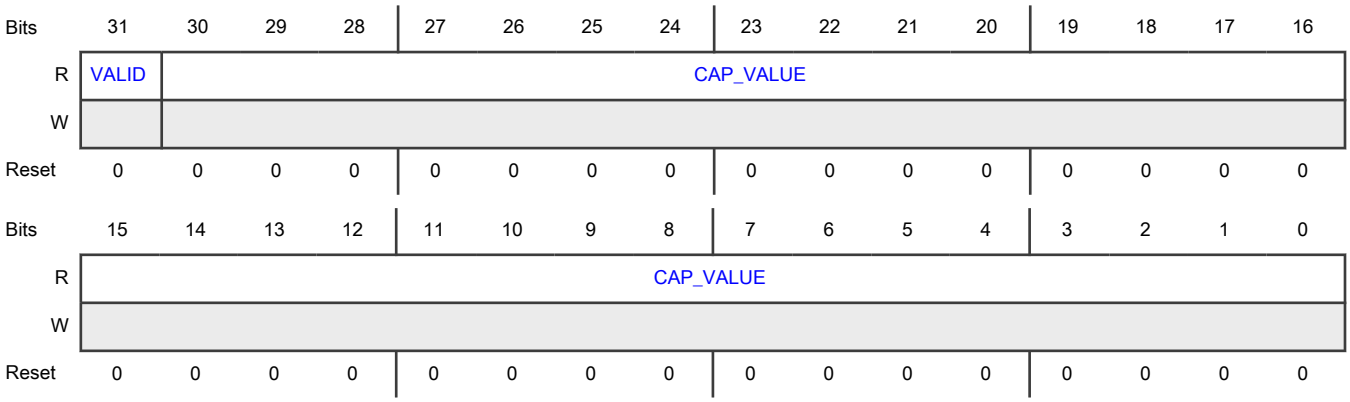
Table continued from the previous page...

Register	Offset
CAP3	1Ch

Function

Contains the micro-tick time value based on any previously captured events. Each capture register is associated with one of the capture trigger inputs.

Diagram



Fields

Field	Function
31 VALID	Captured Value Valid Flag Specifies whether a valid value is captured, based on a transition of the related UTICK_CAP*n* pin. Write 1 to the related field in Capture Clear (CAPCLR) to clear this flag. 0b - Valid value not captured 1b - Valid value captured
30-0 CAP_VALUE	Captured Value for the Related Capture Event Specifies the captured value for the related capture event. The captured value is 1 lower than the actual value of UTICK at the moment of the capture event.

Chapter 29
OS Event Timer (OSTIMER)

29.1 Chip-specific OSTIMER information

Table 167. Reference links to related information

Topic	Related module	Reference
Full description	OSTIMER	OSTIMER
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

29.1.1 Module instances

This device has one instance of the OSTIMER module, OSTIMER0.

29.2 Overview

OSTIMER includes a shared 42-bit timer and separate 42-bit match and capture functions for the Cortex-M33 core.

29.2.1 Block diagram

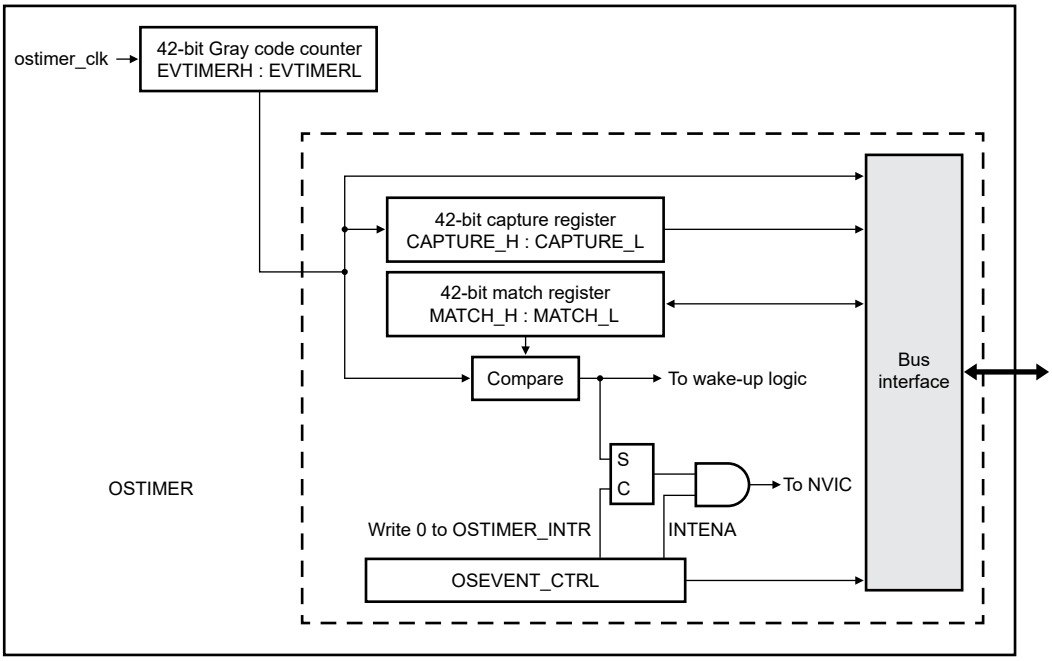


Figure 74. Block diagram

29.2.2 Features

- Provides a 42-bit Gray code counter (known as central EVTimer). Using Gray code means that the timer can run at a frequency unrelated to the CPU clock and can still be read by the CPU without a synchronization delay. Gray code is a reflected binary code that changes in a single-bit position for each increment.
- Supports separate functions for CPU:
  - A capture register can copy the main counter value when triggered by a CPU request.
  - A match register can be compared to the main counter and can optionally generate an interrupt or wake-up event.

29.3 Functional description

The following sections describe the functional details of this module.

29.3.1 Shared event or timestamp timer

The 42-bit shared EVTimer initializes after chip power-up and then counts up continuously. The typical clock for this timer is the 1 MHz low-power oscillator.

This timer is implemented as a Gray code counter that enables the counter to be read asynchronously by any of the processing domains or captured by a capture register running on an asynchronous clock. The output of this shared EVTimer is connected to the processor-specific submodules.

29.3.2 CPU-specific match, capture, and interrupt generation

The submodules associated with CPU include:

- A separate bus interface.
- 42 bits of capture values, available in CAPTURE_L and CAPTURE_H.
- 42 bits of match values, available in MATCH_H and MATCH_L.
- A control register, which includes an interrupt request flag and an interrupt enable field.

Capture registers	42 bits of capture values are available in CAPTURE_L and CAPTURE_H for the CPU. When the CPU issues a capture command, capture registers store the current value of the central EVTimer.
Match registers and interrupt request	42 bits of match values are available in MATCH_H and MATCH_L for each bus interface. The EVTimer output is compared against this combined value for interrupt or wake-up generation. A match to this register pair sets OSEVENT_CTRL[OSTIMER_INTRFLAG], which you can enable to generate an interrupt or wake-up request. You must write values to the match registers in Gray code. OSEVENT_CTRL[MATCH_WR_RDY] must be 0 before you write to the match registers. When you write a new value to the match registers, you must write to MATCH_L before the write to MATCH_H.

29.3.3 Clocking

The OSTIMER uses three clock signals: hclk, capture_clk, and ostimer_clk.

1. Hclk is the IPS interface clock for both the Arm and DSP sub-module, being used in all the registers and write/read operations.
2. Ostimer_clk is the counter clock, being used in the timer itself and in the intr_wakeup signals. It is expected to be running in a low frequency (~1MHz).
3. capture_clk is the Arm and DSP clock, it is used to register the capture signal “capture_load”.

29.3.4 Reset

For OSTIMER, the reset function is more complicated as compared to most other functions.

- Only the POR and software reset can reset the shared EVTimer. "Software reset" refers to a complete reset of the module.
- The capture and match registers are reset by a system reset. "System reset" refers to a reset from the RESET*n* pin, SYSRESETREQ from the CPU, a POR, or any other internal low-voltage resets.
- The control register (OSEVENT_CTL) is reset by the system reset or by software reset. However, the MATCH_WR_RDY bit can only be reset by system reset.

The following table shows a list of resets for OSTIMER registers.

Table 168. OSTIMER registers and their corresponding resets

Registers	POR	Software reset
EVTIMERL and EVTIMERH	Yes	Yes
CAPTURE_L and CAPTURE_H	Yes	No
OSEVENT_CTRL	Yes	Yes

29.3.5 Interrupts

If the interrupt request is enabled (i.e., OSEVENT_CTRL[OSTIMER_INTENA]= 1), the OSTIMER will generate interrupt when the central 42-bit EVTimer and the value programmed in MATCH_L and MATCH_H matches.

No interrupt generates if the interrupt request is disabled (i.e., OSEVENT_CTRL[OSTIMER_INTENA]= 0).

29.4 External signals

This module has no external signals.

29.5 Initialization

Perform this procedure to initialize OSTIMER:

1. Enable the OSTIMER clock. This enables the register interface and the peripheral function clock.
2. Select a clock source for OSTIMER.
3. Clear the OSTIMER peripheral reset.
4. Enable the interrupt that OSTIMER provides to the interrupt controller (set OSEVENT_CTRL[OSTIMER_INTRFLAG]= 1). This allows the chip to wake up from Deep Sleep mode.

29.6 Memory map and register definition

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

29.6.1 OSTIMER register descriptions

29.6.1.1 OSTIMER memory map

OSTIMER0 base address: 400A_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	EVTIMER Low (EVTIMERL)	32	R	0000_0000h
4h	EVTIMER High (EVTIMERH)	32	R	0000_0000h
8h	Local Capture Low for CPU (CAPTURE_L)	32	R	0000_0000h
Ch	Local Capture High for CPU (CAPTURE_H)	32	R	0000_0000h
10h	Local Match Low for CPU (MATCH_L)	32	RW	FFFF_FFFFh
14h	Local Match High for CPU (MATCH_H)	32	RW	FFFF_FFFFh
1Ch	OSTIMER Control for CPU (OSEVENT_CTRL)	32	RW	0000_0008h

29.6.1.2 EVTIMER Low (EVTIMERL)

Offset

Register	Offset
EVTIMERL	0h

Function

Contains the immediate value of the central EVTimer. This is a Gray encoded counter register that is not synchronized with the bus clock.

Because two 32-bit reads are required to retrieve the entire counter value, it is possible for the counter to roll over between the reads of EVTIMERL and EVTIMERH. Gray encoding ensures that even if this roll over occurs, the value read represents either the counter value immediately preceding or immediately following the rollover, if the bus clock rate is at least twice the module's clock rate (nominally 1 MHz), or if the bus is clocked by the same 1 MHz LP oscillator that provides the clock to the module.

Diagram

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

Fields

Field	Function
31-0 EVTIMER_COUNT_VALUE	EVTimer Count Value Specifies the current value of the lower 32 bits of the central 42-bit EVTimer when you read this register. **NOTE** Only one physical EVTimer exists and is readable from all domains.

29.6.1.3 EVTIMER High (EVTIMERH)

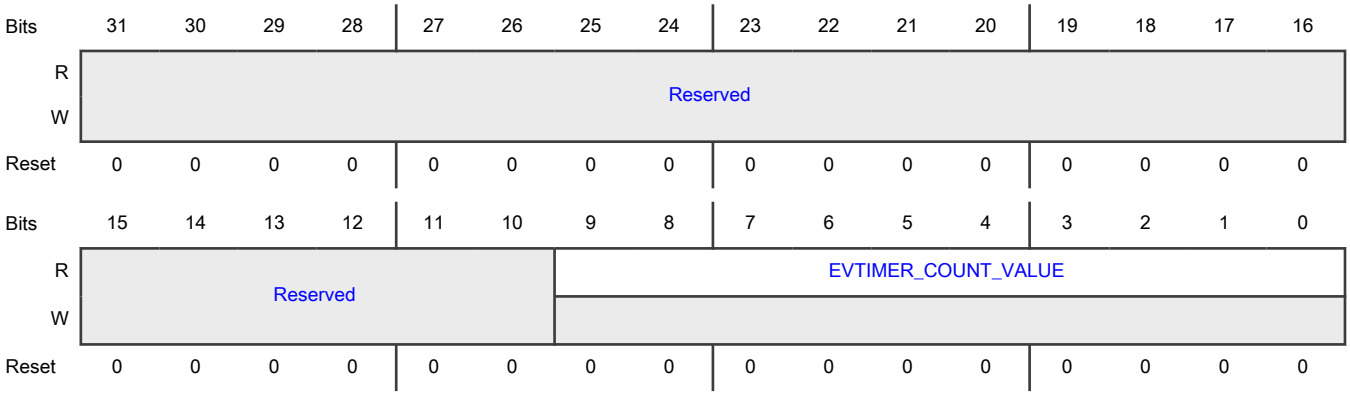
Offset

Register	Offset
EVTIMERH	4h

Function

Contains the higher value of the central EVTimer.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 EVTIMER_COUNT_VALUE	EVTimer Count Value Specifies the current value of the upper 10 bits of the central 42-bit EVTimer when you read this register. **NOTE** Only one physical EVTimer exists and is readable from all domains.

29.6.1.4 Local Capture Low for CPU (CAPTURE_L)

Offset

Register	Offset
CAPTURE_L	8h

Function

Contains the value of the central EVTimer when captured by the CPU. This register is Gray encoded to match EVTimer. You must not execute a new capture command between reading CAPTURE_L and CAPTURE_H, to avoid retrieving incoherent results.

Diagram

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

Fields

Field	Function
31-0 CAPTURE_VALUE	EVTimer Capture Value Specifies the value of the lower 32 bits of the central 42-bit EVTimer when you read this register, at the time that the CPU generated the last capture signal. A separate pair of CAPTURE_L and CAPTURE_H is implemented for the CPU. CPU reads its own capture value at the same pair of addresses.

29.6.1.5 Local Capture High for CPU (CAPTURE_H)

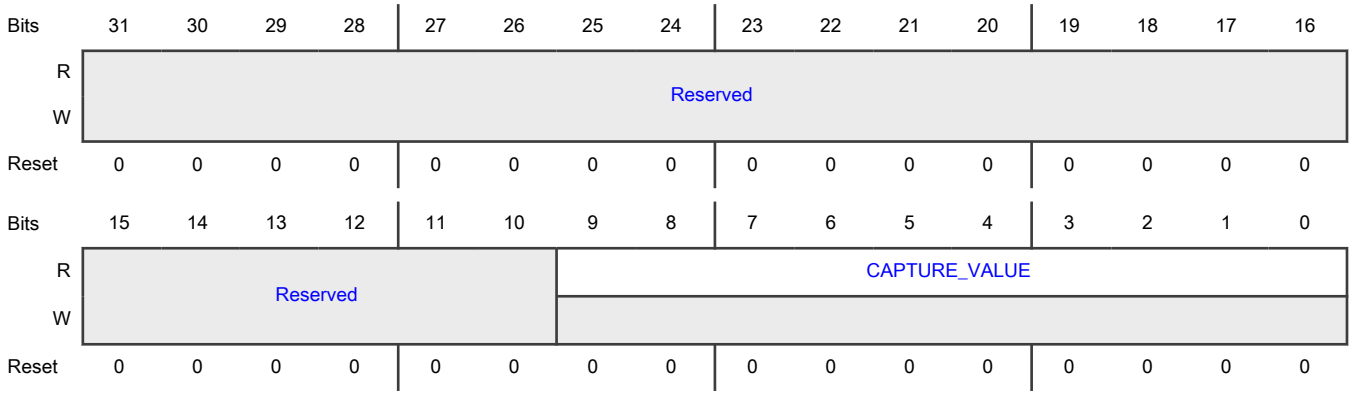
Offset

Register	Offset
CAPTURE_H	Ch

Function

Contains the value of the central EVTimer when captured by the CPU. This register is Gray encoded to match EVTimer. You must not execute a new capture command between reading CAPTURE_L and CAPTURE_H, to avoid retrieving incoherent results.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 CAPTURE_VALUE	EVTimer Capture Value Specifies the value of the upper 10 bits of the central 42-bit EVTimer when you read this register, at the time the CPU generated the last capture signal (using CMSIS C function "__SEV();").

29.6.1.6 Local Match Low for CPU (MATCH_L)

Offset

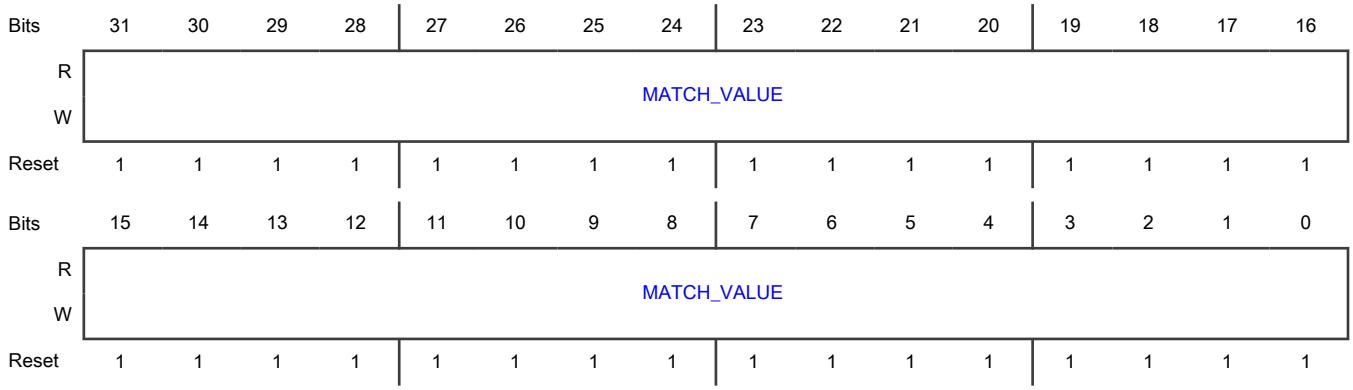
Register	Offset
MATCH_L	10h

Function

Compares the match value with the central EVTimer value. This register is Gray encoded to match EVTimer. The value in this pair of registers is stable, so there is no need for multiple reads. When writing to the match register pair, you must write to MATCH_L first, followed by the write to MATCH_H.

You must not initiate a second write to the match register pair until the first write is complete (which is a minimum of three bus clocks followed by a write to MATCH_H). OSEVENT_CTRL[MATCH_WR_RDY] indicates when it is safe to reload the match registers. You do not need to read OSEVENT_CTRL[MATCH_WR_RDY] if an interrupt has already occurred because of the previous match value, or the required time period has elapsed.

Diagram



Fields

Field	Function
31-0 MATCH_VALUE	EVTimer Match Value Compares the value (lower 32 bits) of the MATCH_L and MATCH_H register pair with the central EVTimer value. If both the values match, the CPU generates an interrupt request if the interrupt is enabled. A separate pair of MATCH_L and MATCH_H is implemented for the CPU reading its own local value at the same pair of addresses.

29.6.1.7 Local Match High for CPU (MATCH_H)

Offset

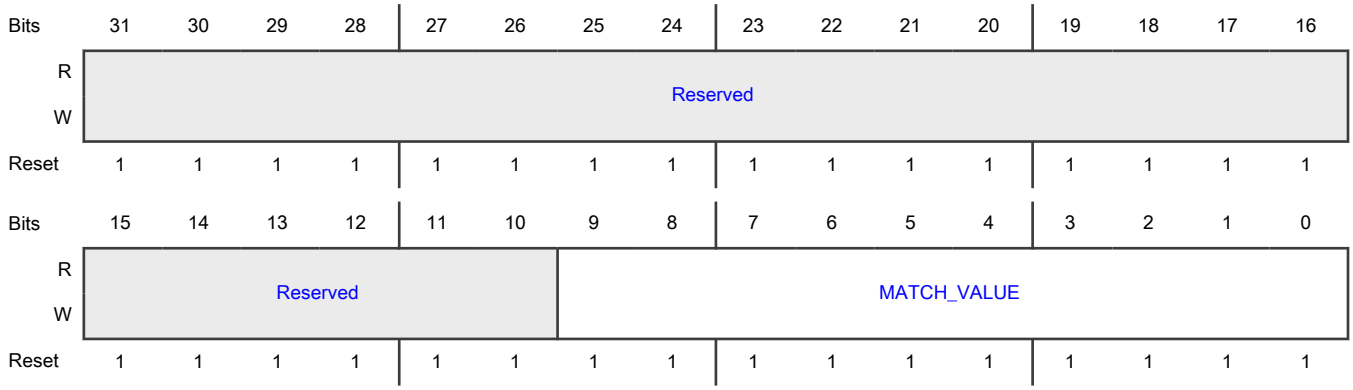
Register	Offset
MATCH_H	14h

Function

Compares the match value with the central EVTimer value. This register is Gray encoded to match EVTimer. The value in this pair of registers is stable, so there is no need for multiple reads. When writing to the match register pair, you must write to MATCH_L first, followed by the write to MATCH_H.

You must not initiate a second write to the match register pair until the first write is complete (which is a minimum of three bus clocks followed by a write to MATCH_H). OSEVENT_CTRL[MATCH_WR_RDY] indicates when it is safe to reload the match registers. You do not need to read OSEVENT_CTRL[MATCH_WR_RDY] if an interrupt has already occurred because of the previous match value, or the required time period has elapsed.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-0 MATCH_VALUE	EVTimer Match Value Compares the value (upper 10 bits) of the MATCH_L and MATCH_H register pair with the central EVTimer value. If both the values match, the CPU generates an interrupt request if the interrupt is enabled.

29.6.1.8 OSTIMER Control for CPU (OSEVENT_CTRL)

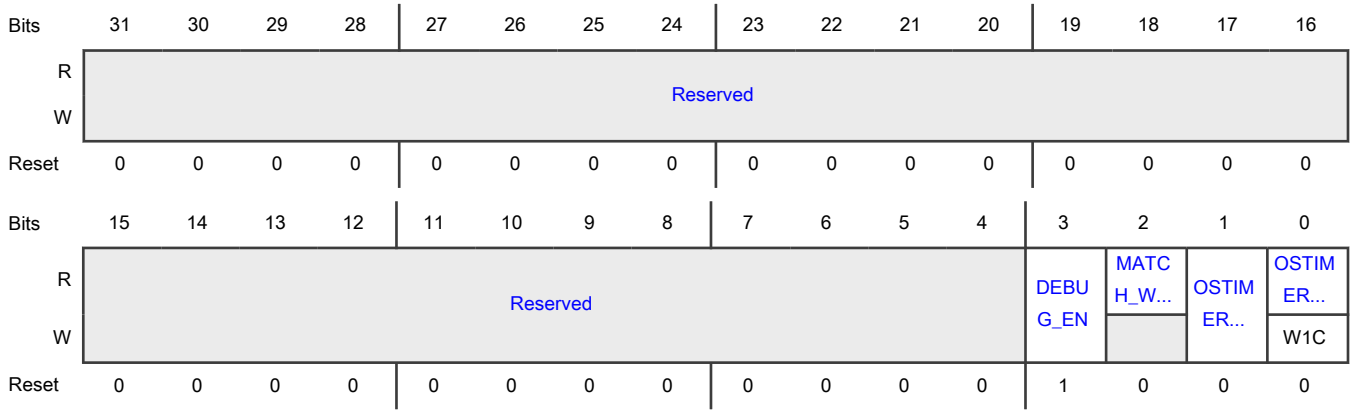
Offset

Register	Offset
OSEVENT_CTRL	1Ch

Function

Provides the interrupt flag and interrupt enable signals for the CPU. A separate OSEVENT_CTRL register is implemented for the CPU. CPU reads its own local value at the same address.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 DEBUG_EN	Debug Enable Enables OSTIMER to operate when the chip is in Debug mode. 0b - Disables 1b - Enables
2 MATCH_WR_RDY	EVTimer Match Write Ready Specifies that when this field is low, it is safe to reload (write) to the match registers. In typical applications, it is not necessary to read this field. The 42-bit match register value is transferred from a pair of shadow registers to the active match registers after you write to MATCH_H. You must not initiate a second write to MATCH_L and MATCH_H until after this transfer completes. This field becomes 0 after the transfer completes. It is not necessary to read this status field, if an interrupt has already occurred because of the first match value, or if it is certain (via some other means) that the required period of time (3 bus clocks) has elapsed.
1 OSTIMER_INTENA	Interrupt or Wake-Up Request This bit is to enable/disable interrupt request. If the value is 1, then this field asserts an interrupt or wake-up request to the domain processor. Otherwise, the interrupt or wake-up requests are blocked. 0b - Interrupts blocked 1b - Interrupts enabled
0 OSTIMER_INTRFLAG	Interrupt Flag Sets an interrupt flag when a match occurs between the central 42-bit EVTimer and the value programmed in the MATCH_L and MATCH_H register pair for the CPU. Writes to clear this field are asynchronous. You must write 1 to this field before writing a new match value into MATCH_L and MATCH_H.

Chapter 30
Wake Timer

30.1 Chip-specific Wake-up Timer information

Table 169. Reference links to related information

Topic	Related module	Reference
Full description	WAKEUPTIMER	Wake-up Timer
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

30.1.1 Module instances

This device has one instance of the Wake Timer module, WAKEUPTIMER0.

30.2 Overview

The wake timer is clocked by the 1 kHz clock which is the output of the WAKE_TIMER_CTRL[OSC_DIV_ENA], the clock divider of the clock source.

30.2.1 Block diagram

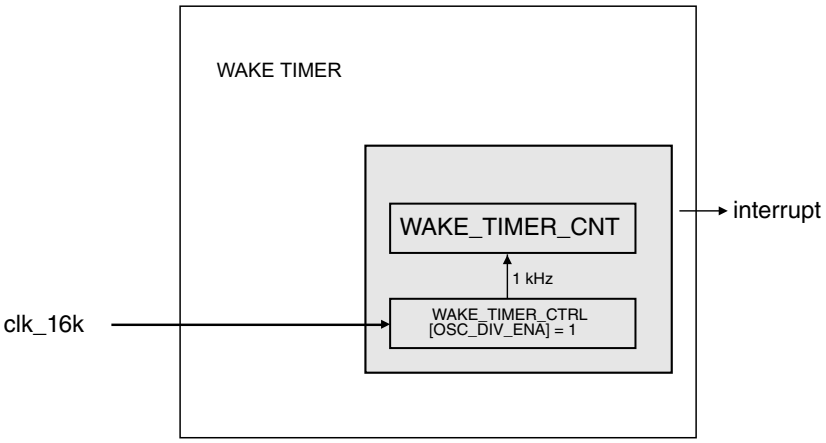


Figure 75. WAKE TIMER block diagram

30.2.2 Features

- 32-bit wake timer counter to wake the chip.

30.3 Functional description

The wake timer is a 32-bit down counter, which is clocked at a 1 kHz rate when enabled. Writing any non-zero value to this timer automatically enables the counter and launches a countdown sequence. When the counter is being used as a wake timer, this write can occur prior to entering a low-power mode.

When a starting count value loads, the wake timer turns on, counts from the pre-loaded value down to zero, generates an interrupt and/or a wake command, and then turns itself off until relaunched by a subsequent software write.

30.4 External signals

This module has no external signals.

30.5 Memory map and register description

This section includes the WAKE TIMER memory map and detailed descriptions of all registers.

30.5.1 WAKE_TIMER register descriptions

30.5.1.1 WAKE_TIMER memory map

WAKETIMER0 base address: 400A_E000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Wake Timer Control (WAKE_TIMER_CTRL)	32	RW	0000_0000h
Ch	Wake Timer Counter (WAKE_TIMER_CNT)	32	RW	0000_0000h

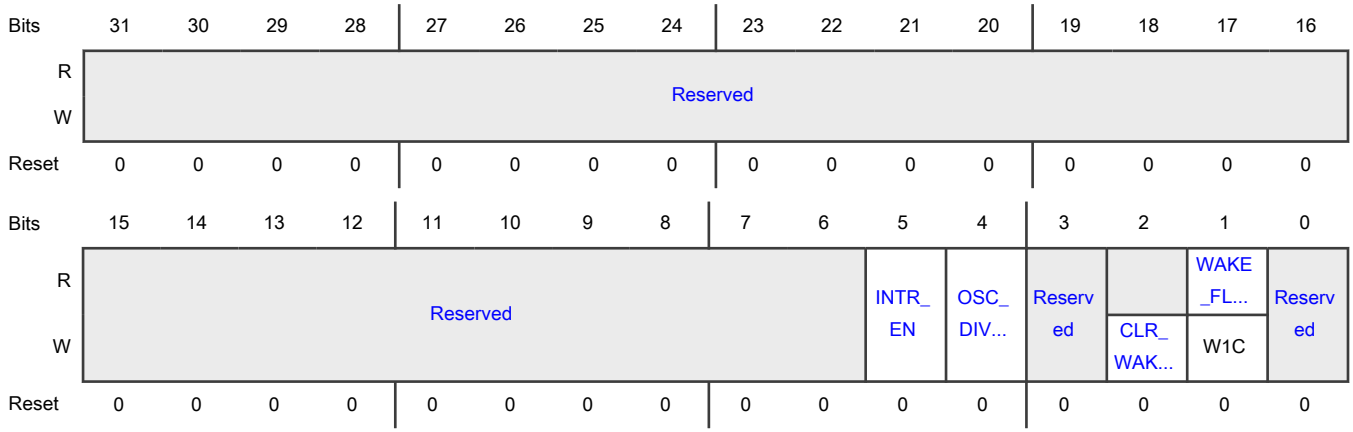
30.5.1.2 Wake Timer Control (WAKE_TIMER_CTRL)

Offset

Register	Offset
WAKE_TIMER_CTRL	0h

Function
The wake timer is clocked by the 1 kHz clock which is the output of the 4-bit ripple counter with the 16 kHz clock as the clock source. This register controls the wake timer counter register.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 INTR_EN	Enable Interrupt Enable interrupt when WAKE_FLAG is set. 0b - Disabled 1b - Enabled
4 OSC_DIV_ENA	OSC Divide Enable Enables the 4-bit clock divider to divide down the 16 kHz input clock to generate the 1 kHz clock source for the wake timer. OSC_DIV_ENA must be set to 1 for the wake timer to run. 0b - Disabled 1b - Enabled
3 —	Reserved
2 CLR_WAKE_TIMER	Clear Wake Timer **NOTE** Reading this bit always returns 0. 0b - No effect. 1b - Clears the wake timer counter and halts operation until a new count value is loaded.
1 WAKE_FLAG	Wake Timer Status Flag Writing 1 clears this status bit. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Wake timer has not timed out. 1b - Wake timer has timed out.
0 —	Reserved

30.5.1.3 Wake Timer Counter (WAKE_TIMER_CNT)

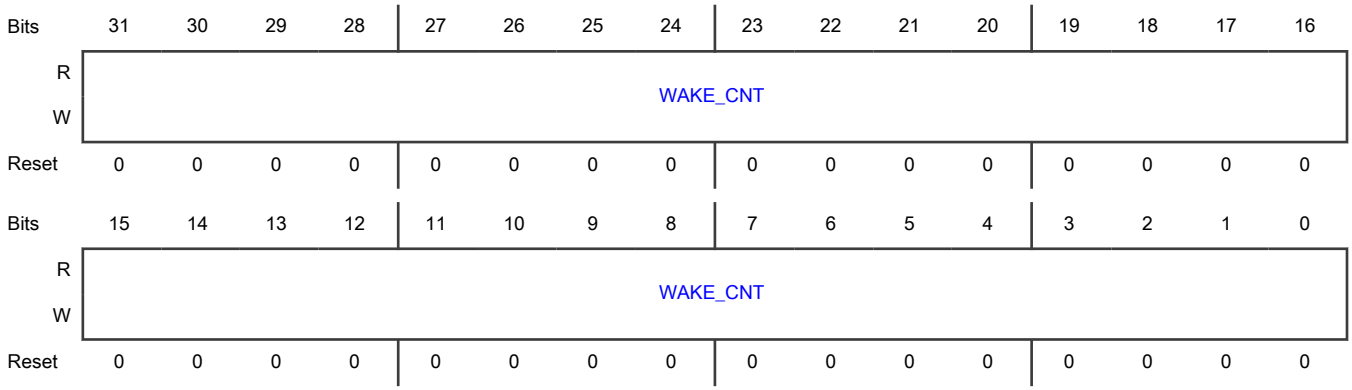
Offset

Register	Offset
WAKE_TIMER_CNT	Ch

Function

Do not write to WAKE_TIMER_CNT while counting is in progress.

Diagram



Fields

Field	Function
31-0 WAKE_CNT	Wake Counter A read reflects the current value of the wake timer. Due to clock domain crossing, read the register twice in succession and confirm that the two values are the same. If the values are not the same, repeat the procedure until the two values are the same. A write to this field pre-loads a start-count value into the timer and starts a countdown sequence.

Chapter 31
Enhanced Flex Pulse Width Modulator (eFlexPWM)

31.1 Chip-specific PWM information

Table 170. Reference links to related information

Topic	Related module	Reference
Full description	PWM	PWM
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Bus and Memories		Memory architecture
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

31.1.1 Module instances

This device supports two instances of the FlexPWM module: FlexPWM0 and FlexPWM1.

31.1.2 PWM control signals

This device doesn't support PWM_A and PWM_B capture functions (only PWM_X is supported). Any references of PWM_A and PWM_B should be ignored throughout the chapter.

31.1.3 Module clocking

FlexPWM can select between three clock signals: the IPBus clock, EXT_CLK, and AUX_CLK.

The IPBus clock comes from MAIN_CLK, which can reach 192MHz frequency.

31.2 Overview

The Pulse Width Modulator (PWM) module contains PWM submodules, each of which can be used to control a single half-bridge power stage. Fault channel support is provided.

This module generates various switching patterns, including highly sophisticated waveforms. It is ideal for controlling different Switched Mode Power Supplies (SMPS) topologies.

31.2.1 Block diagram

The following figure shows the PWM block diagram.

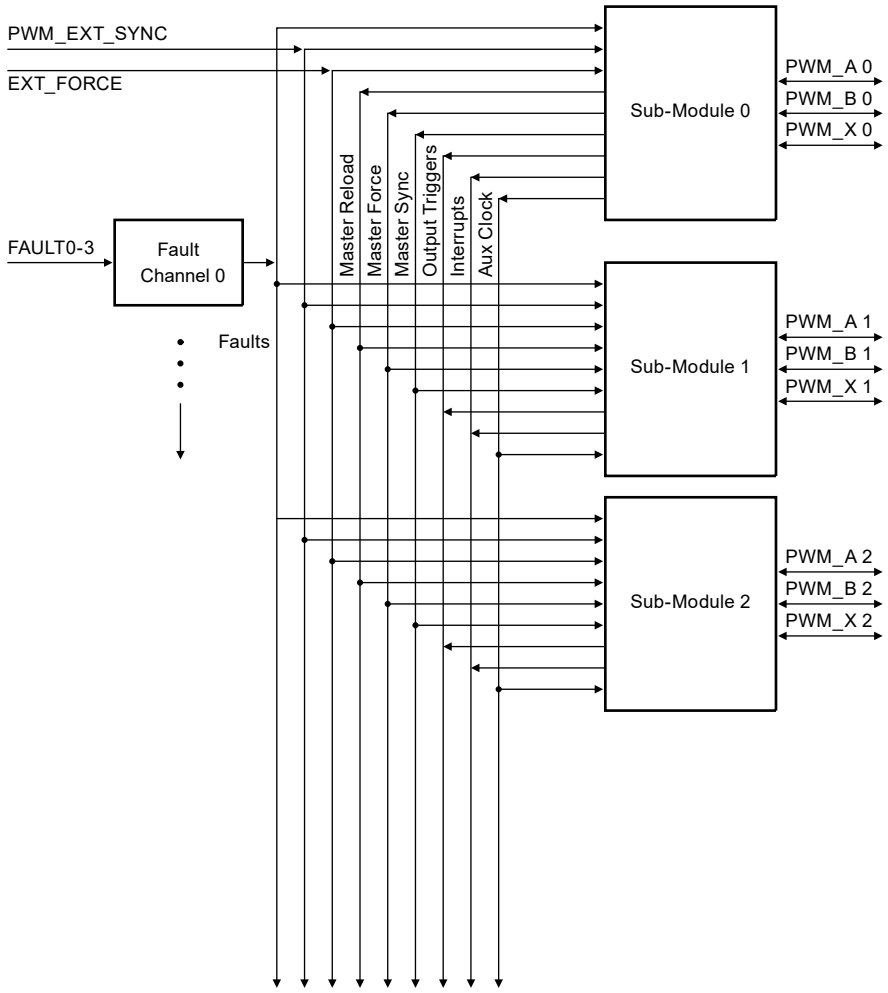


Figure 76. PWM block diagram

31.2.2 Features

Following are the features of PWM:

- 16-bit resolution for center, edge-aligned, and asymmetrical PWMs
- PWM outputs that can operate as complementary pairs or independent channels
- Ability to accept signed numbers for PWM generation
- Independent control of both edges of each PWM output
- Support for synchronization to external hardware or other PWM
- Double buffered PWM registers
  - Integral reload rates from 1 to 16
  - Half cycle reload capability
- Multiple output trigger events can be generated per PWM cycle via hardware
- Support for double switching PWM outputs
- Fault inputs can be assigned to control multiple PWM outputs
- Programmable filters for fault inputs

- Independently programmable PWM output polarity
- Independent top and bottom deadtime insertion
- Each complementary pair can operate with its own PWM frequency and deadtime values
- Individual software control for each PWM output
- All outputs can be programmed to change simultaneously via a FORCE_OUT event
- PWM_X pin can optionally output a third PWM signal from each submodule
- Channels not used for PWM generation can be used for buffered output compare functions and for input capture functions
- Enhanced dual edge capture functionality

31.3 Functional description

31.3.1 PWM submodule

The following figure shows the PWM Submodule Block Diagram.

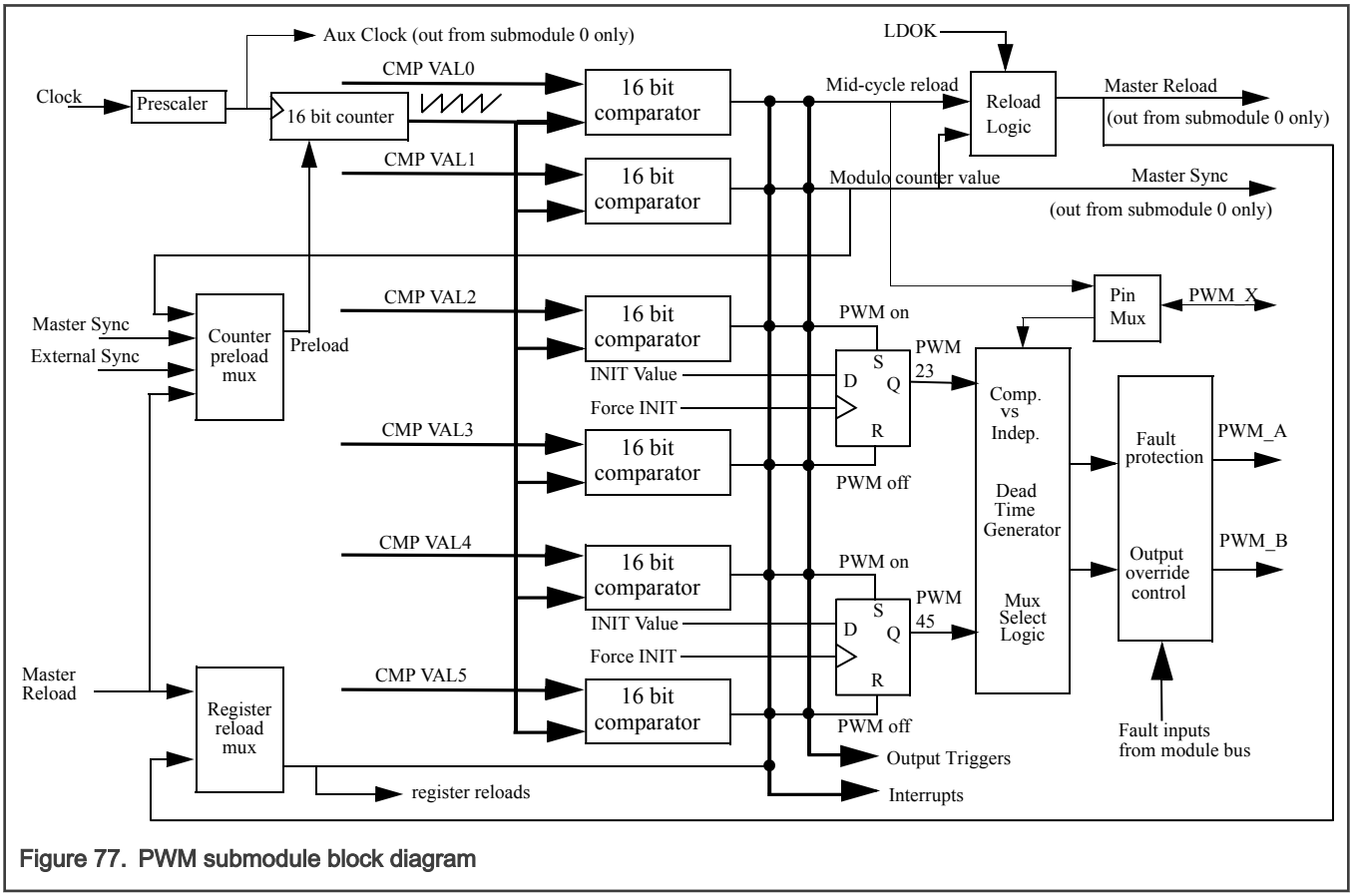


Figure 77. PWM submodule block diagram

31.3.2 PWM capabilities

This section describes some capabilities of the PWM module.

31.3.2.1 Center aligned PWMs

Each submodule has its own timer that is capable of generating PWM signals on two output pins. The edges of each of these signals are controlled independently as shown in Figure 78.

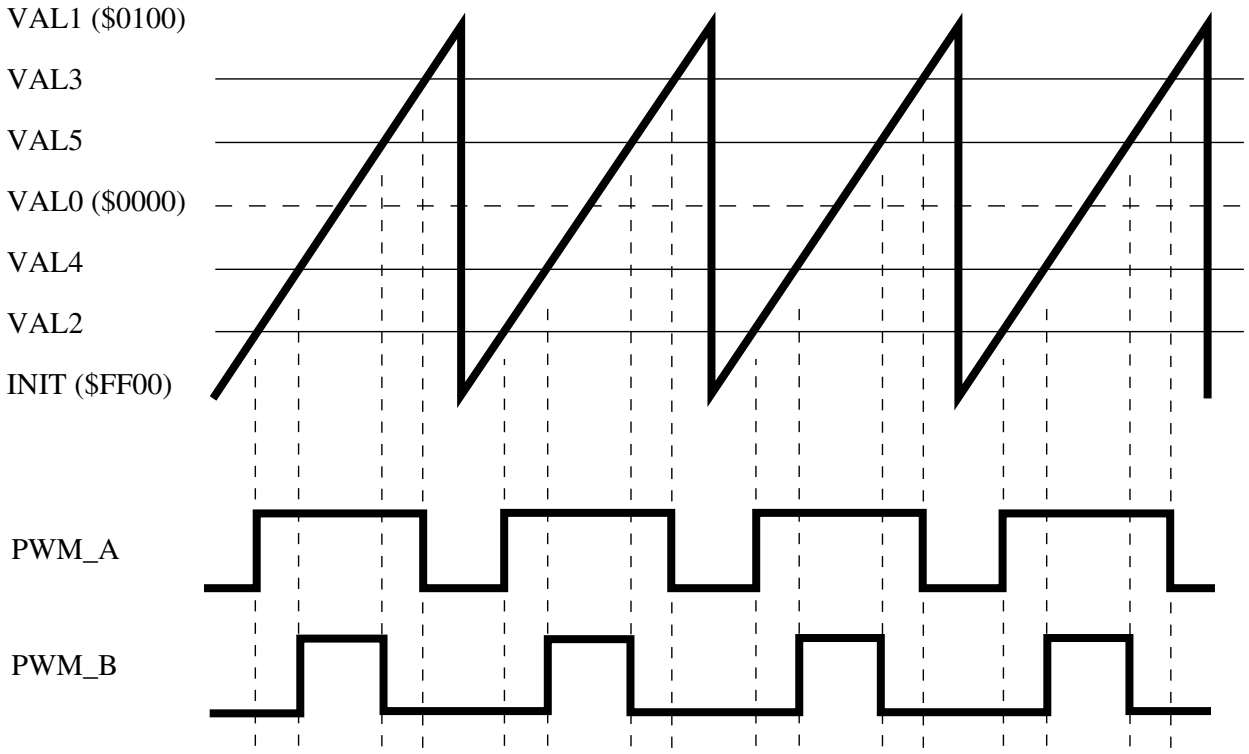


Figure 78. Center aligned PWM example

The submodule timers only count in the up direction and then reset to the INIT value. Instead of having a single value that determines pulse width, there are two values that must be specified: the turn-on edge and the turn-off edge. This double-action edge generation provides the user control over the pulse width and also the relative alignment of the signal. As a result, there is no need to support separate PWM alignment modes since the PWM alignment mode is inherently a function of the turn-on and turn-off edge values.

Figure 78 also illustrates an additional enhancement to the PWM generation process. When the counter resets, it is reloaded with a user-specified value, which may or may not be zero. If the value chosen happens to be the 2's complement of the modulus value, then the PWM generator operates in "signed" mode. This means if each PWM's turn-on and turn-off edge values are same in numbers but different in their sign, the "on" portion of the output signal is centered around a count value of zero. Therefore, only one PWM value is calculated in software and then this value and its negative are provided to the submodule as the turn-off and turn-on edges respectively. This technique results in a pulse width consists of an odd number of timer counts. If all PWM signal edge calculations follow this convention, then the signals will be center aligned with each other, which is the goal. The center alignment between the signals is not restricted to symmetry around the zero count value, as any other number would also work. However, centering on zero provides the greatest range in signed mode and also simplifies the calculations.

31.3.2.2 Edge aligned PWMs

Figure 79 shows the results of edge aligned operation when the turn-on edge for each pulse is specified to be the INIT value. Therefore, only the turn-off edge value needs to be periodically updated to change the pulse width.

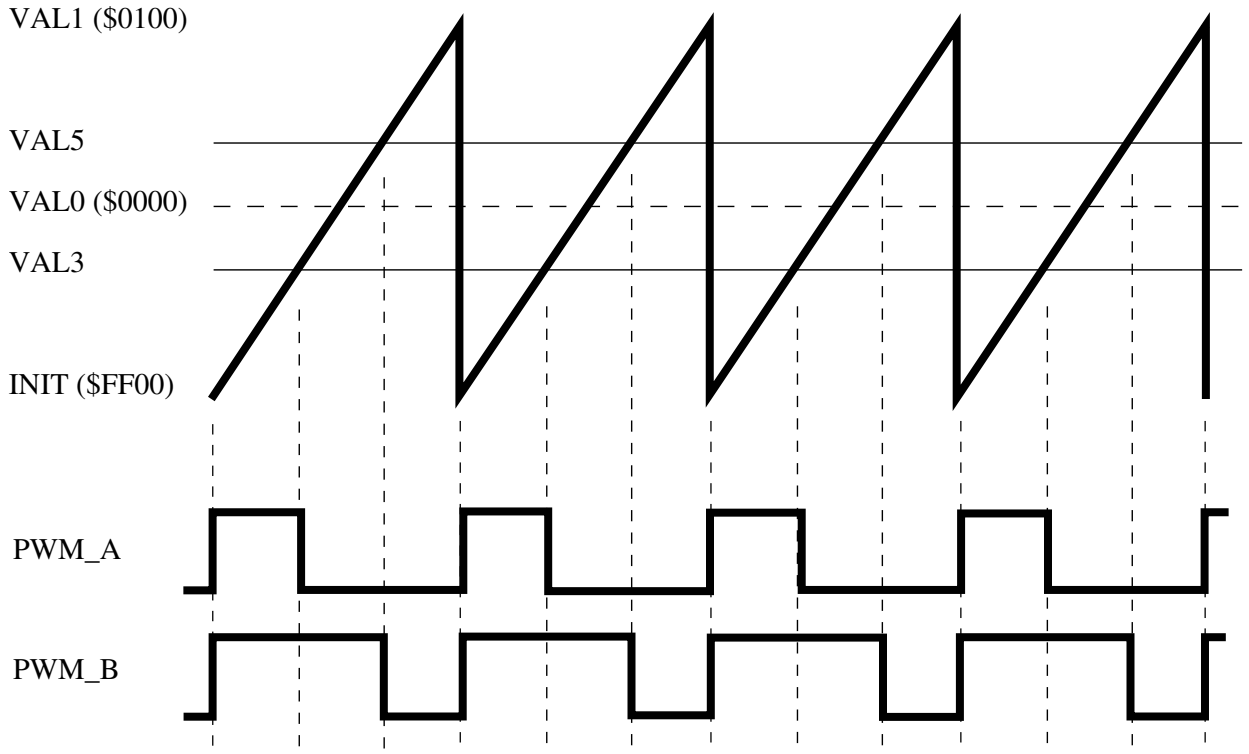


Figure 79. Edge aligned example (INIT=VAL2=VAL4)

With edge aligned PWMs, another example of the benefits of signed mode can be seen. Use "bipolar" PWMs to drive an H-bridge, where a 50% duty cycle results in zero volts on the load. Duty cycles less than 50% generate negative load voltages and duty cycles greater than 50% generate positive load voltages. If the module is set to signed mode operation (the INIT and VAL1 values are the same number with opposite signs), then there is a direct proportionality between the PWM turn-off edge value and the motor inverter voltage, including the sign. Therefore, signed mode of operation simplifies the software interface to the PWM module since no offset calculations are required to translate the output variable control algorithm to the voltage on an H-Bridge load.

31.3.2.3 Phase shifted PWMs

In the previous sections, the benefits of the signed mode of operation were discussed in the context of simplifying the required software calculations by eliminating the requirement to bias up signed variables before applying them to the module. However, if numerical biases are applied to the turn-on and turn-off edges of different PWM signals, the signals will be phase shifted to each other, as shown in Figure 80. This results in certain advantages when applied to a power stage. For example, when operating a multi-phase inverter at a low modulation index, all of the PWM switching edges from the different phases occur at the same time. This can be troublesome from a noise standpoint, especially if ADC readings of the inverter must be scheduled near those times. Phase shifting the PWM signals can open up timing windows between the switching edges to allow a signal to be sampled by the ADC. However, phase shifting does not affect the duty cycle so average load voltage is not affected.

If the outputs of submodules 1 - 2 need to be delayed from the output of submodule 0 (and from each other), instead of just creating a phase delay by adding an offset to the turn on and turn off times of the different submodules, another method is to use the PHASEDLY registers for submodules 1 - 2 to indicate their delay from the submodule 0 timing. This method can be used when the master sync signal from submodule 0 is selected as the initialization source (CTRL2[INIT_SEL]==b10). This method allows all of the submodules to be programmed with the same turn on and turn off time but submodules 1 - 2 can still be delayed the time from submodule 0.

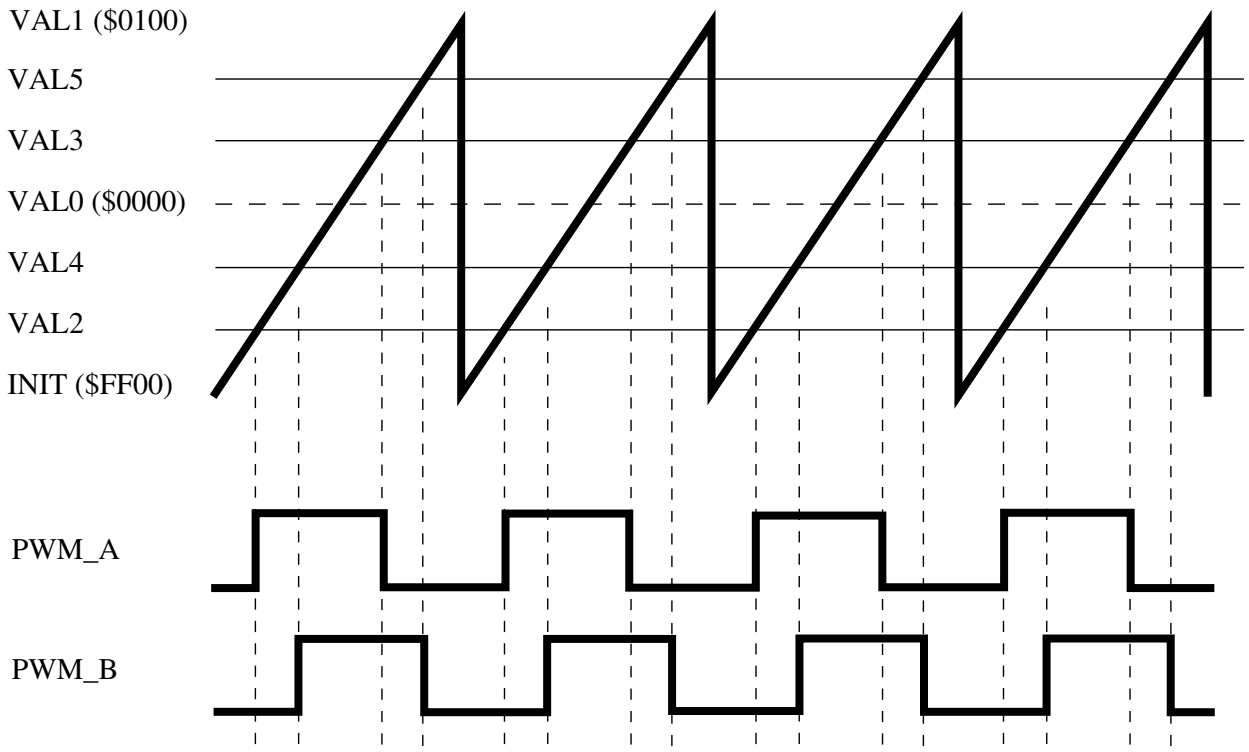


Figure 80. Phase shifted outputs - example for 1 submodule

An additional benefit of phase shifted PWMs is shown in Figure 81. In this case, an H-Bridge circuit is driven by 4 PWM signals to control the voltage waveform on the primary of a transformer. Both left and right side PWMs are configured to generate a square wave with 50% duty cycle. This works for the H-Bridge since no narrow pulse widths are generated reducing the high frequency switching requirements of the transistors. The RMS value of this waveform is directly controlled by the amount of phase shift of the square waves. Regardless of the phase shift, no DC component appears in the load voltage as long as the duty cycle of each square wave remains at 50% suitable for transformer loads. As a result, this topology is frequently used in industrial welders to adjust the amount of energy delivered to the weld arc.

NOTE

The square wave on the right side of the H-Bridge is phase shifted compared to the left side of the H-Bridge. As a result, the transformer primary sees the bottom waveform across its terminals.

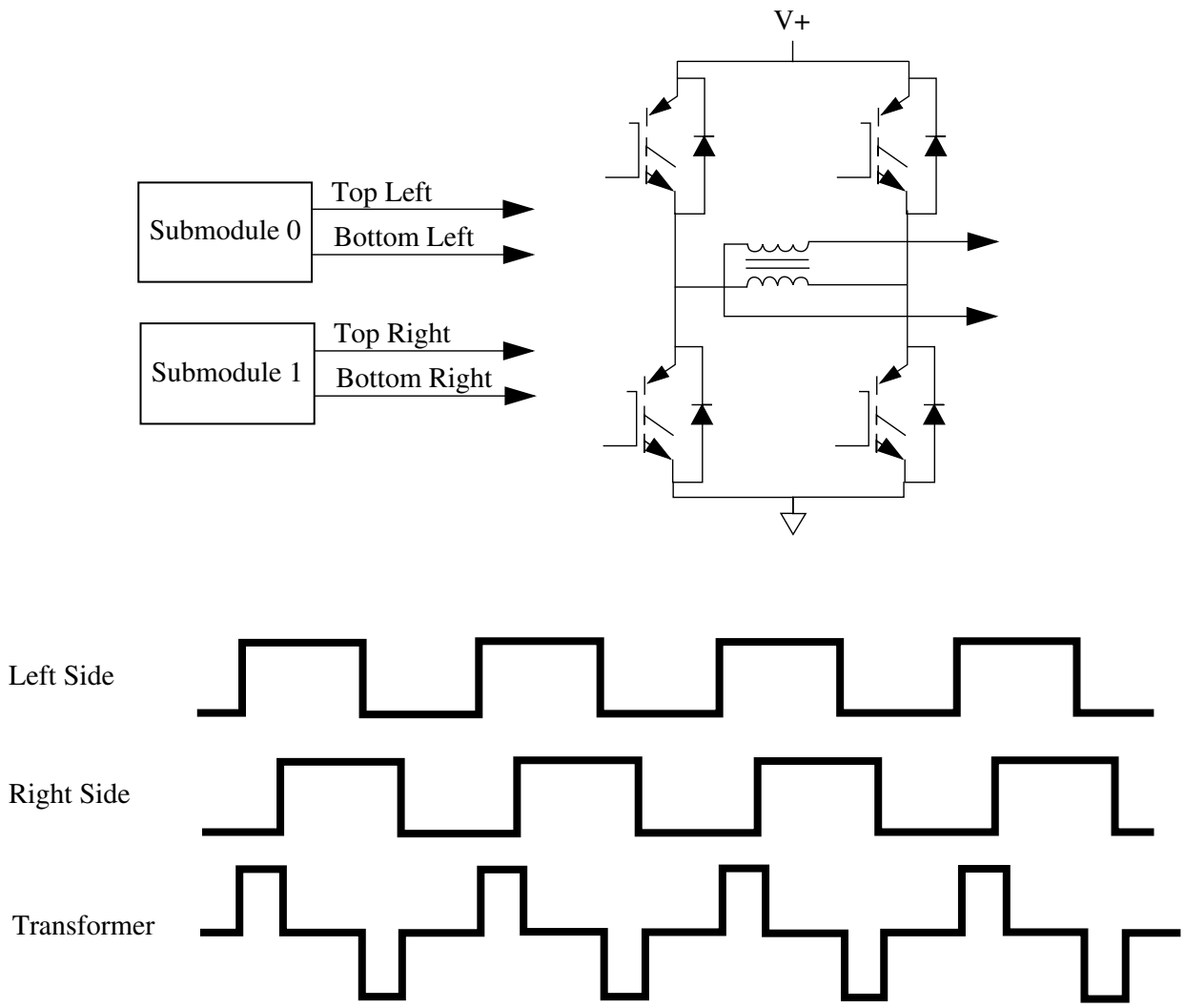


Figure 81. Phase Shifted PWMs Applied to a Transformer Primary

31.3.2.4 Double switching PWMs

Double switching PWM output is supported to aid in single shunt current measurement and three-phase reconstruction. This method supports two independent rising edges and two independent falling edges per PWM cycle. The VAL2 and VAL3 registers are used to generate the even channel (labeled as PWM_A in Figure 82) while VAL4 and VAL5 are used to generate the odd channel. The two channels (PWM23 or PWM_A and PWM45 or PWM_B from force out logic) are combined using XOR logic (force out logic) as the following figure shows. The DBLPWM signal can be run through the deadtime insertion logic.

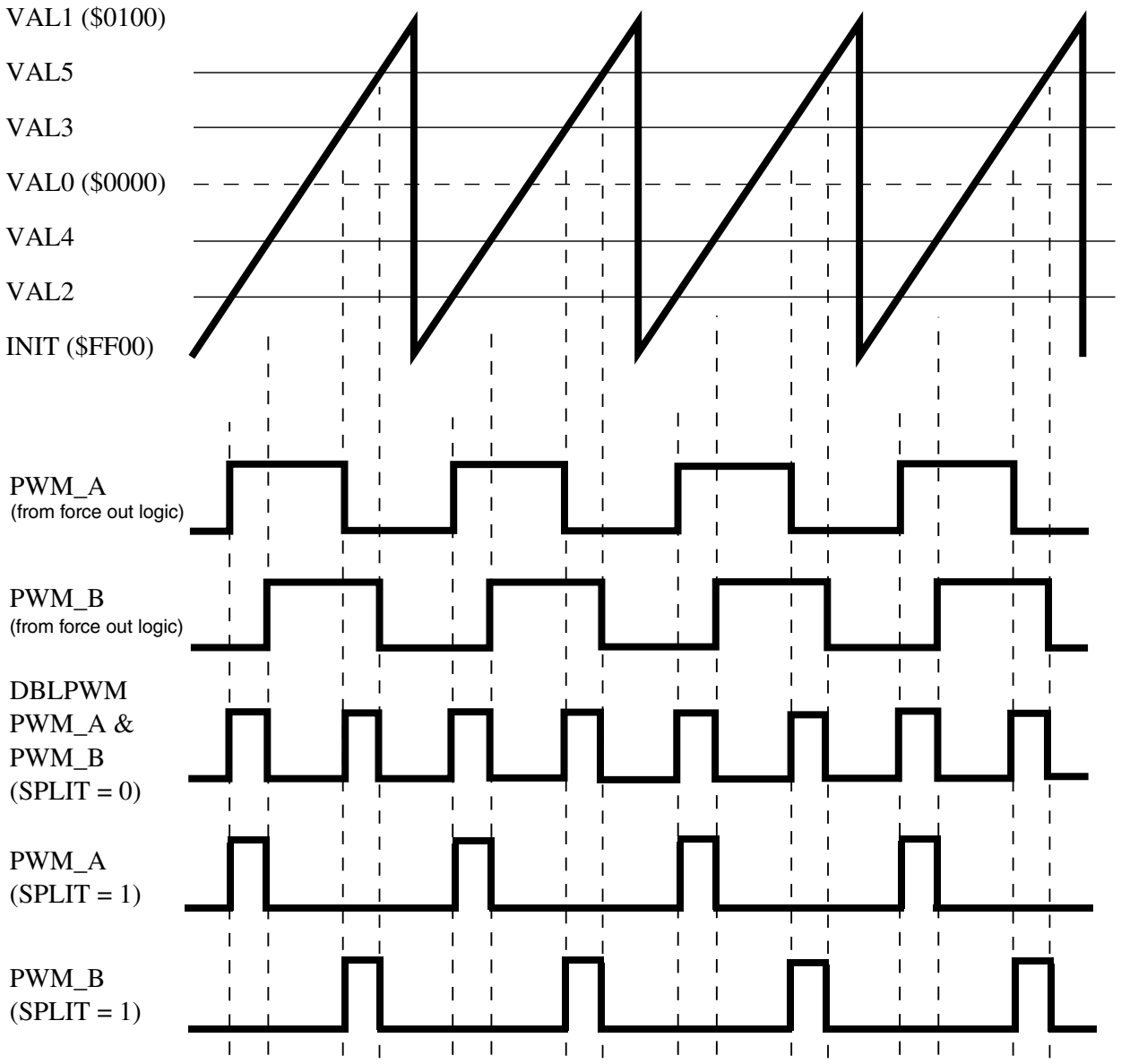


Figure 82. Double switching output example

31.3.2.5 ADC triggering

In cases where the timing of the ADC triggering is critical, it must be scheduled as a hardware event instead of software activated. With this PWM module, multiple ADC triggers can be generated in hardware per PWM cycle without the requirement of another timer module. Figure 83 shows how this is accomplished. When specifying a complementary mode of operation, only two edge comparators are required to generate the output PWM signals for a given submodule. This means the other comparators are free to perform other functions. In this example, the software does not need to quickly respond after the first conversion to set up other conversions that must occur in the same PWM cycle.

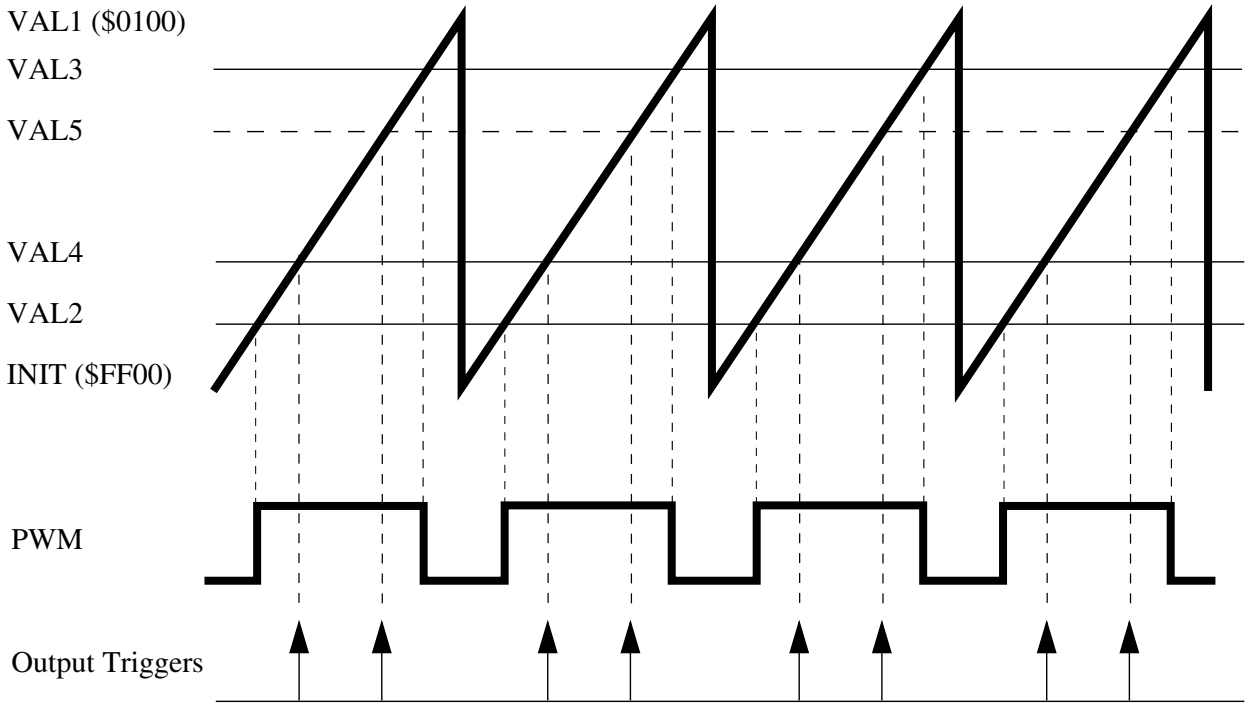


Figure 83. Multiple output trigger generation in hardware

Because each submodule has its own timer, it is possible for each submodule to run at a different frequency. One of the options possible with this PWM module is to have one or more submodules running at a lower frequency, but still synchronized to the timer in submodule0. Figure 84 shows how this feature can be used to schedule ADC triggers over multiple PWM cycles. You can use the lower-frequency submodule to control the sampling frequency of the software control algorithm where multiple ADC triggers can now be scheduled over the entire sampling period. In Figure 84, *all* submodule comparators are shown being used for ADC trigger generation.

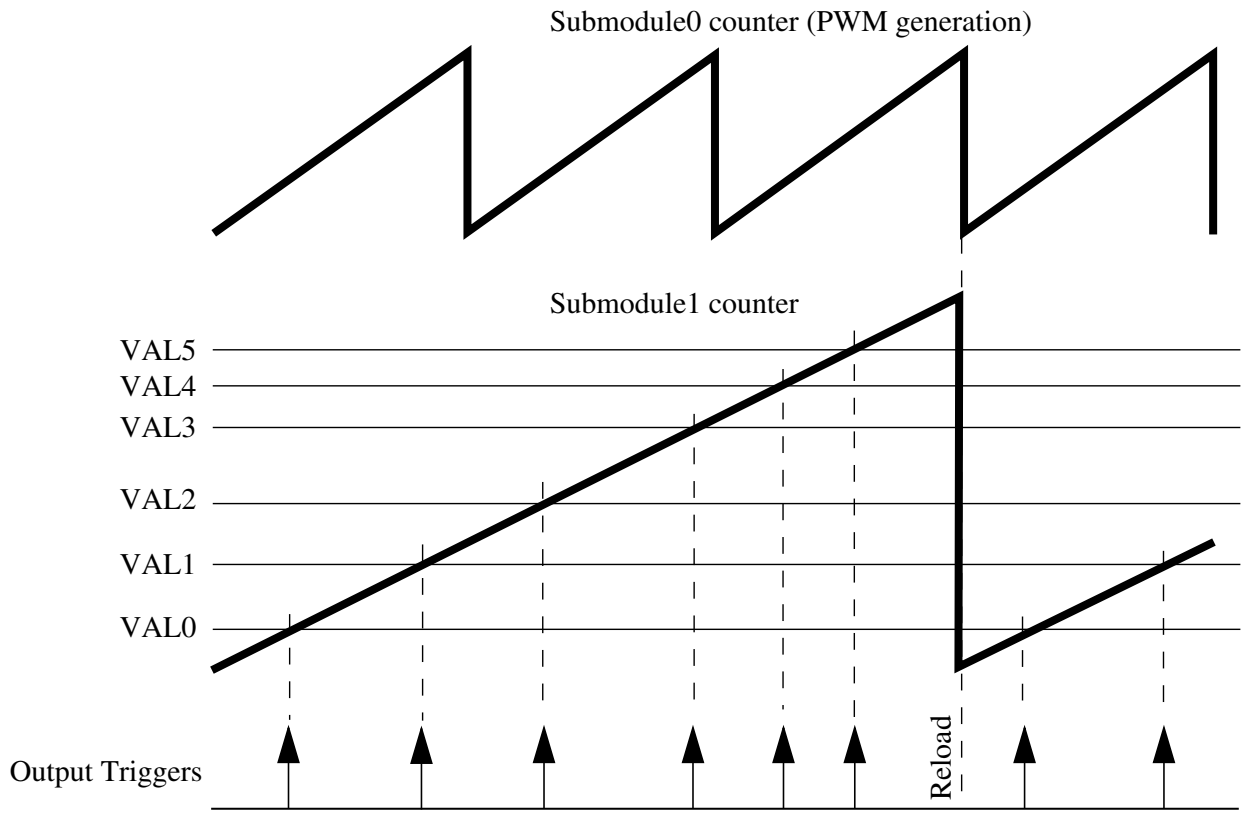


Figure 84. Multiple output triggers over several PWM cycles

31.3.2.6 Enhanced capture capabilities (E-Capture)

When a PWM pin is not used for PWM generation, it can be used to perform input captures. For PWM generation, both edges of the PWM signals are specified via separate compare register values. When programmed for input capture, both of these registers work on the same pin to capture multiple edges, toggling from one to the other in either a free running or one-shot fashion. Programming the desired edge of each capture circuit, period, and pulse width of an input signal can easily be measured without the requirement to re-arm the circuit. In addition, each edge of the input signal can clock an 8-bit counter where the counter output is compared to a user specified value (EDGCMP). When the counter output equals EDGCMP, the value of the submodule timer is captured and the counter is automatically reset. This feature counts a specified number of edge events and then performs a capture and interrupt. The following figure illustrates some of the functionality of the E-Capture circuit.

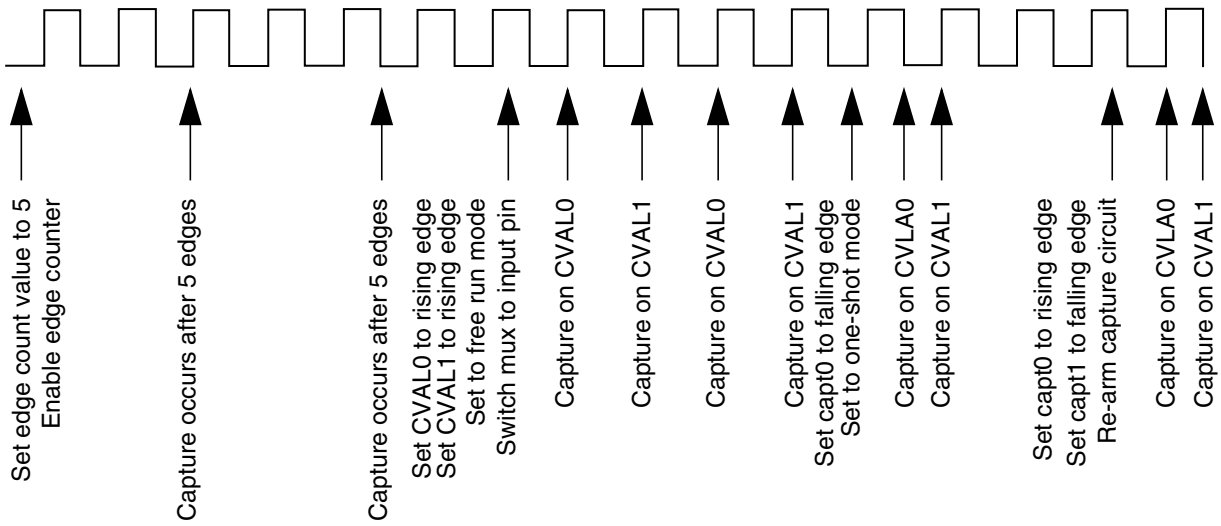


Figure 85. Capture capabilities of the E-Capture circuit

When a submodule is used for PWM generation, its timer counts up to the modulus value used to specify the PWM frequency and then is re-initialized. Therefore, using this timer for input captures on one of the other pins (for example, PWM_X) has limited utility since it does not count through all of the numbers and the timer reset represents a discontinuity in the 16-bit number range. However, when measuring a signal that is synchronous to the PWM frequency, the timer modulus range is suited for the application. As shown in Figure 86, the output of a PWM power stage is connected to the PWM_X pin that is configured for free running input captures. Specifically, the CVAL0 capture circuitry is programmed for rising edges and the CVAL1 capture circuitry is set for falling edges. This results in new load pulse width data acquired every PWM cycle. To calculate the pulse width, subtract the CVAL0 register value from the CVAL1 register value. This measurement is beneficial when performing deadtime distortion correction on a half bridge circuit driving an inductive load. Also, these values can be directly compared to the VALx registers responsible for generating the PWM outputs to obtain a measurement of system propagation delays. For details, refer to the separate discussion of deadtime distortion correction.

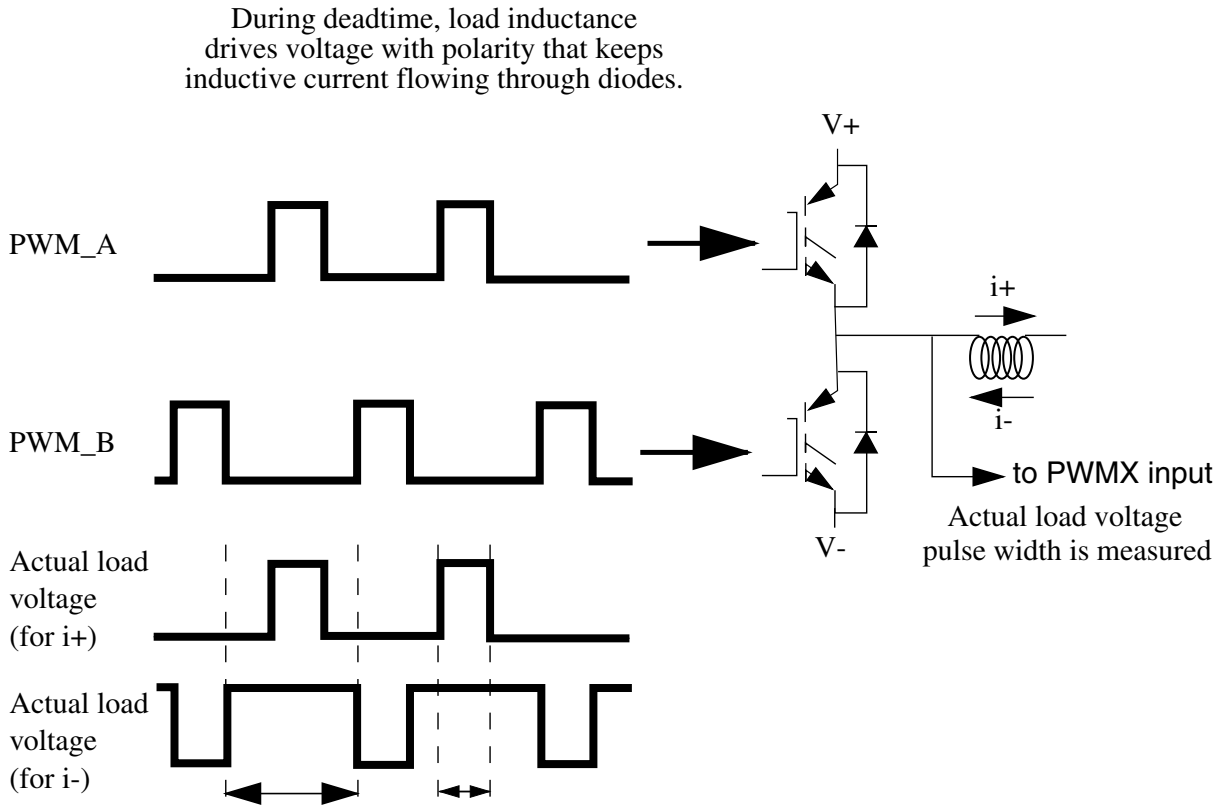


Figure 86. Output pulse width measurement possible with the E-capture circuit

31.3.2.7 Synchronous switching of multiple outputs

Before the PWM signals are routed to the output pins, they are processed by a hardware block that permits all submodule outputs to be switched synchronously. This feature is useful in commutated motor applications where the next commutation state can be laid in ahead of time and then immediately switched to the outputs when the appropriate condition or time is reached. All the changes occur immediately after the trigger event occurs eliminating any interrupt latency and also the changes occurs synchronously on all submodule outputs.

The synchronous output switching is accomplished via a signal called FORCE_OUT. This signal originates from the local FORCE bit within the submodule, from submodule0, or from external to the PWM module, and in most cases, is supplied from an external timer channel configured for output compare. In a typical application, software sets up the desired states of the output pins in preparation for the next FORCE_OUT event. This selection lays dormant until the FORCE_OUT signal transitions and then all outputs are switched simultaneously. The signal switching is performed upstream from the deadtime generator so that any abrupt changes that occur do not violate deadtime on the power stage when in complementary mode.

Figure 87 shows an application that can benefit from this feature. On a brushless DC motor in many cases, it is required to spin the motor without need of hall-effect sensor feedback. Instead, the back EMF of the motor phases is monitored and this information is used to schedule the next commutation event. The top waveforms of Figure 87 represent these back EMF signals. Timer compare events (represented by the long vertical lines in the diagram) are scheduled based on the zero crossings of the back-EMF waveforms. The PWM module is configured via software ahead of time with the next state of the PWM pins in anticipation of the compare event. When it happens, the output compare of the timer drives the FORCE_OUT signal which immediately changes the state of the PWM pins to the next commutation state with no software latency.

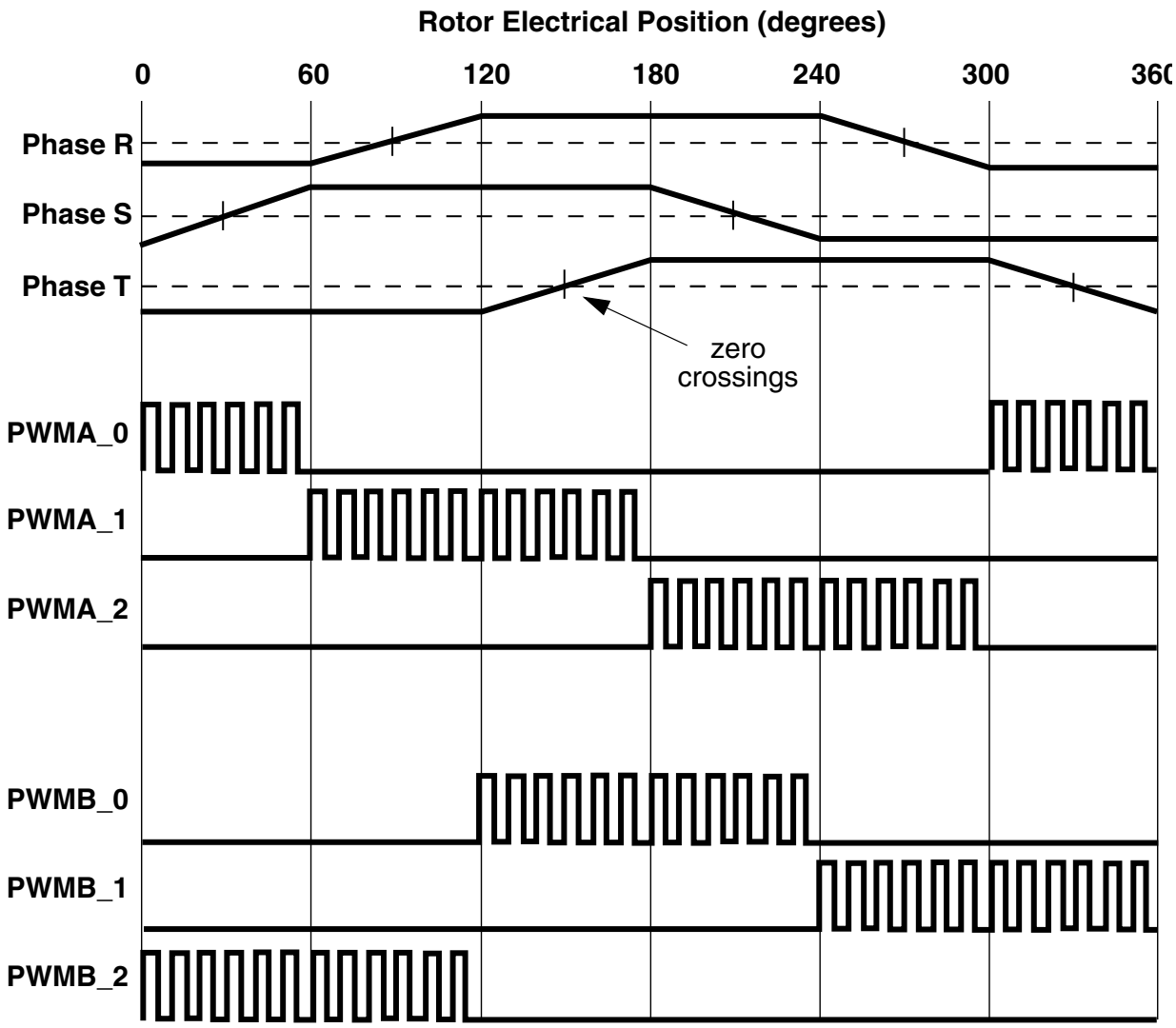


Figure 87. Sensorless BLDC commutation using the force out function

31.3.3 Operation

This section describes the implementation of various sections of the PWM in detail.

The following figure is a high-level block diagram of output PWM generation.

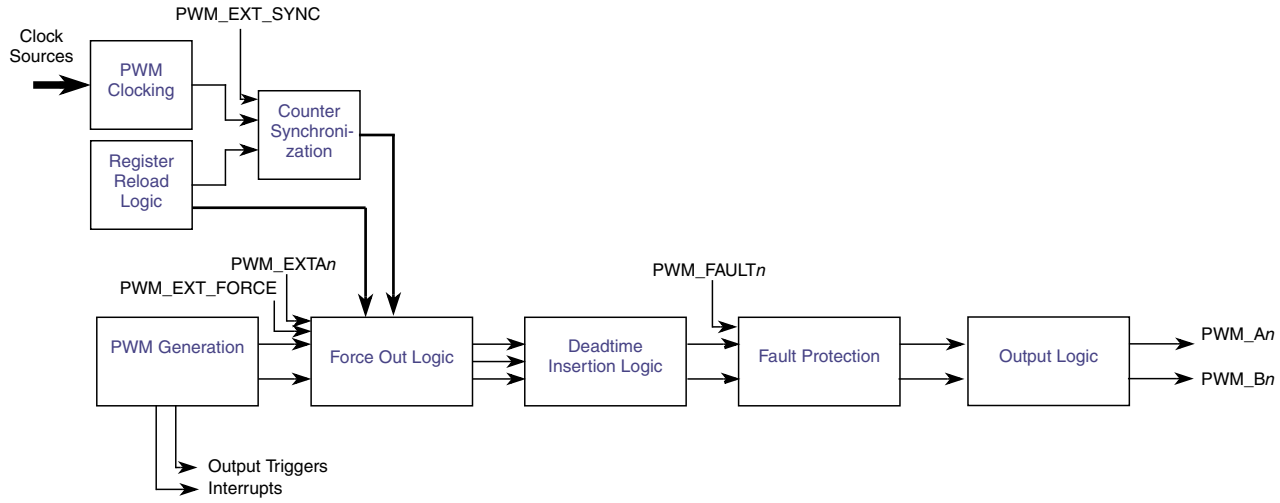


Figure 88. High-level output PWM generation block diagram

31.3.3.1 Register reload logic

The register reload logic is used to determine when the outer set of registers for all double buffered register pairs will be transferred to the inner set of registers. The register reload event can be scheduled to occur every "n" PWM cycles using CTRL[LDFQ] and CTRL[FULL], which is defined by VAL1 register. A half cycle reload option is also supported (CTRL[HALF]) where the reload can take place in the middle of a PWM cycle. The half cycle point is defined by the VAL0 register and does not have to be exactly in the middle of the PWM cycle.

As shown in Figure 89 the reload signal from submodule0 can be broadcast as the Master Reload signal allowing the reload logic from submodule0 to control the reload of registers in other submodules.

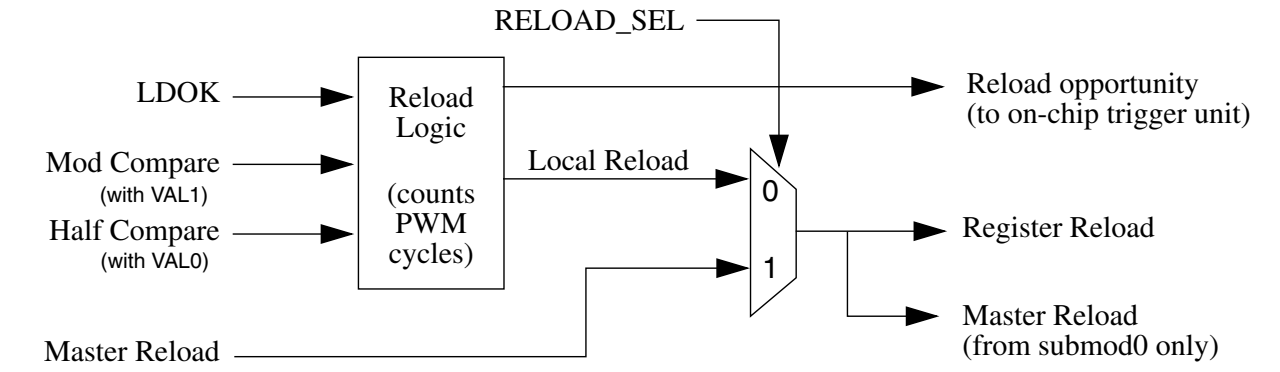


Figure 89. Register reload logic of each submodule

31.3.3.2 Counter synchronization

As shown in Figure 90, the 16-bit counter counts up until its output equals VAL1 which is used to specify the counter modulus value. The resulting compare causes a rising edge to occur on the Local Synchronization signal which is one of four possible sources used to cause the 16-bit counter to be initialized with INIT. If Local Synchronization is selected as the counter initialization signal, VAL1 within the submodule effectively controls the timer period (and then the PWM frequency generated by that submodule), and everything operates or functions at a local level.

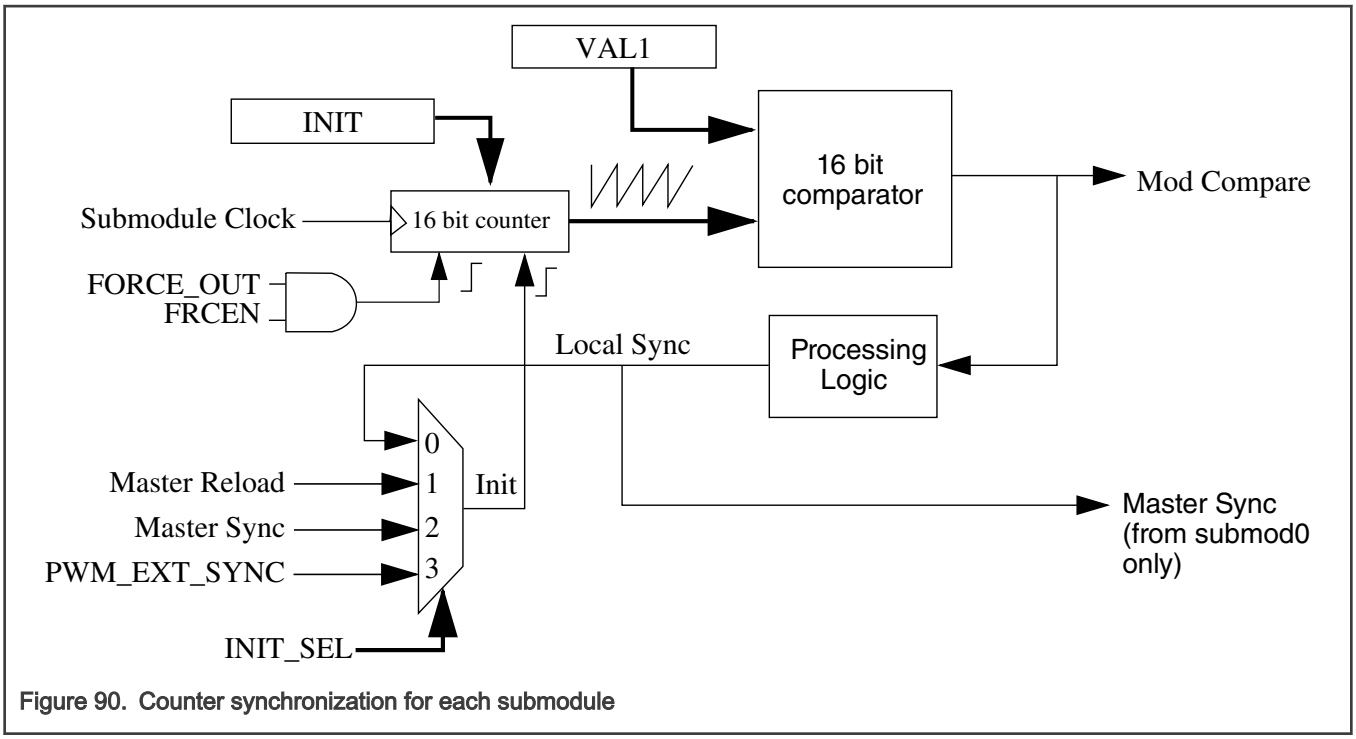


Figure 90. Counter synchronization for each submodule

The Master Synchronization signal originates as the Local Sync from submodule0. If configured to do so, the timer period of any submodule can be locked to the period of the timer in submodule0.

The PWM_EXT_SYNC signal originates on chip or off chip depending on the system architecture. This signal may be selected as the source for counter initialization so that an external source can control the period of all submodules.

If the Master Reload signal is selected as the source for counter initialization, then the period of the counter will be locked to the register reload frequency of submodule0. Since the reload frequency is commensurate to the sampling frequency of the software control algorithm, the submodule counter period is equal the sampling period. As a result, this timer can be used to generate output compares or output triggers over the entire sampling period which may consist of several PWM cycles. The Master Reload signal can only originate from submodule0.

The counter can optionally initialize upon the assertion of the FORCE_OUT signal assuming that CTRL2[FRCEN] is set. As shown in Figure 90, this constitutes a second initialization input into the counter, which causes the counter to initialize regardless of which signal is selected as the counter initialization signal. A forced initialization causes a register reload if MCTRL[LDOK] is set.

The counter can be initialized by FORCE_OUT signal only when MCTRL[RUN] = 1 or CTLR2[CLK_SEL] = 2 which chooses auxiliary clock from SM0.

The FORCE_OUT signal is provided mainly for commutated applications. When PWM signals are commutated on an inverter controlling a brushless DC motor, it is necessary to restart the PWM cycle at the beginning of the commutation interval. This action effectively resynchronizes the PWM waveform to the commutation timing. Otherwise, the average voltage applied to a motor winding integrated over the entire commutation interval will be a function of the timing between the asynchronous commutation event for the PWM cycle. The effect is more critical at higher motor speeds where each commutation interval may consist of only a few PWM cycles. If the counter is not initialized at the start of each commutation interval, the result is an oscillation caused by the beating between the PWM frequency and the commutation frequency.

31.3.3.3 PWM generation

Figure 91 illustrates how PWM generation is accomplished in each submodule. In each case, two comparators and associated VALx registers are utilized for each PWM output signal. One comparator and VALx register are used to control the turn-on edge, while a second comparator and VALx register control the turn-off edge.

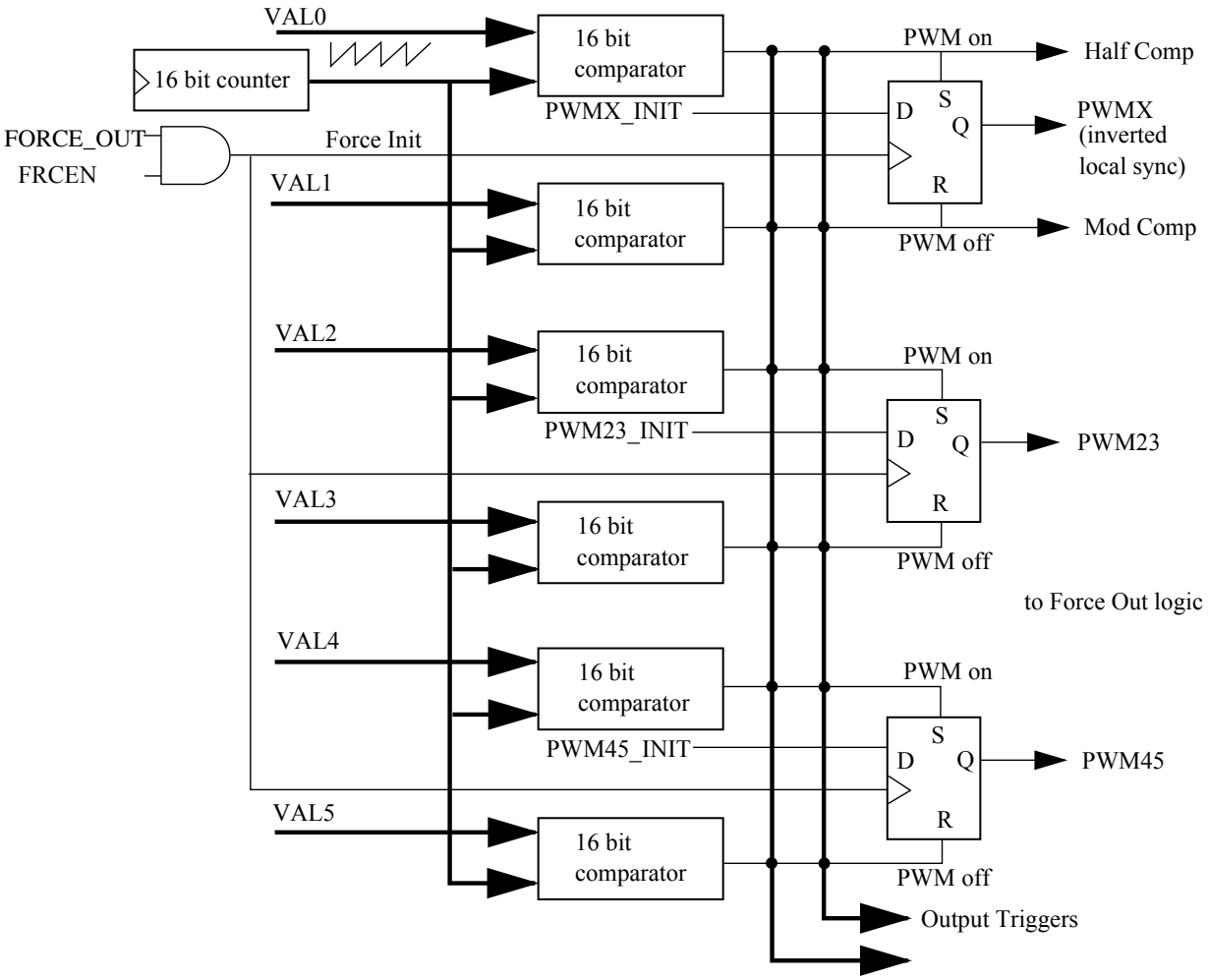


Figure 91. PWM Generation Hardware of each submodule

The generation of the Local Synchronization signal is performed exactly the same way as the other PWM signals in the submodule. While comparator 0 causes a falling edge of the Local Synchronization signal, comparator 1 generates a rising edge. Comparator 1 is also hardwired to the reload logic to generate the full cycle reload indicator.

If VAL1 is controlling the modulus of the counter and VAL0 is half of the VAL1 register minus the INIT value, then the half cycle reload pulse occurs exactly half way through the timer count period and the Local Synchronization will have a 50% duty cycle. On the other hand, if the VAL1 and VAL0 registers are not required for register reloading or counter initialization, they can be used to modulate the duty cycle of the Local Synchronization signal, effectively turning it into an auxiliary PWM signal (PWM_X) assuming that the PWM_X pin is not being used for another function such as input capture or deadtime distortion correction. Including the Local Synchronization signal, each submodule is capable of generating three PWM signals where software has complete control over each edge of each of the signals.

If the comparators and edge value registers are not required for PWM generation, they can be used for other functions such as output compares, generating output triggers, or generating interrupts at timed intervals.

The 16-bit comparators shown in Figure 91 are "equal to" comparators. In addition, if both the set and reset of the flip-flop are asserted, then the flop output goes to 0.

31.3.3.4 Output compare capabilities

By using the VALx registers in conjunction with the submodule timer and 16-bit comparators, buffered output compare functionality can be achieved with no additional hardware required. Specifically, the following output compare functions are possible:

- An output compare sets the output high
- An output compare sets the output low
- An output compare generates an interrupt
- An output compare generates an output trigger

In PWM generation, an output compare is initiated by programming a VALx register for a timer compare, which in turn causes the output of the D flip-flop to either set or reset. For example, if an output compare is desired on the PWM_A signal that sets it high, VAL2 would be programmed with the counter value where the output compare should take place. However, to prevent the D flip-flop from being reset again after the compare has occurred, the VAL3 register must be programmed to a value out of the modulus range of the counter. Therefore, a comparison that would result in resetting the D flip-flop output would never occur. Conversely, if an output compare is desired on the PWM_A signal that sets it low, the VAL3 register is programmed with the appropriate count value and the VAL2 register is programmed with a value out of the counter modulus range. Regardless of whether a high compare or low compare is programmed, an interrupt, or output trigger can be generated when the compare event occurs.

31.3.3.5 Force out logic

For each submodule, the software can select between eight signal sources for the FORCE_OUT signal depending on the chip architecture:

1. Local CTRL2[FORCE]
2. Master Force signal from submodule0
3. Local Reload signal
4. Master Reload signal from submodule0
5. Local Synchronization signal
6. Master Synchronization signal from submodule0
7. EXT_SYNC signal from on or off chip
8. EXT_FORCE signal from on or off chip

The local signals are used to change the signals on the output pins of the submodule without regard for synchronization with other submodules. However, if it is required that all signals on all submodule outputs change at the same time, the Master, EXT_SYNC, or EXT_FORCE signals must be selected.

Figure 92 illustrates the Force logic. The SEL23 and SEL45 fields each choose from one of four signals that can be supplied to the submodule outputs: the PWM signal, the inverted PWM signal, a binary level specified by software via the OUT23 and OUT45 bits, or the PWM_EXTA alternate external control signals. The selection can be determined ahead of time, and when a FORCE_OUT event occurs, these values are presented to the signal selection mux that immediately switches the requested signal to the output of the mux for further processing downstream.

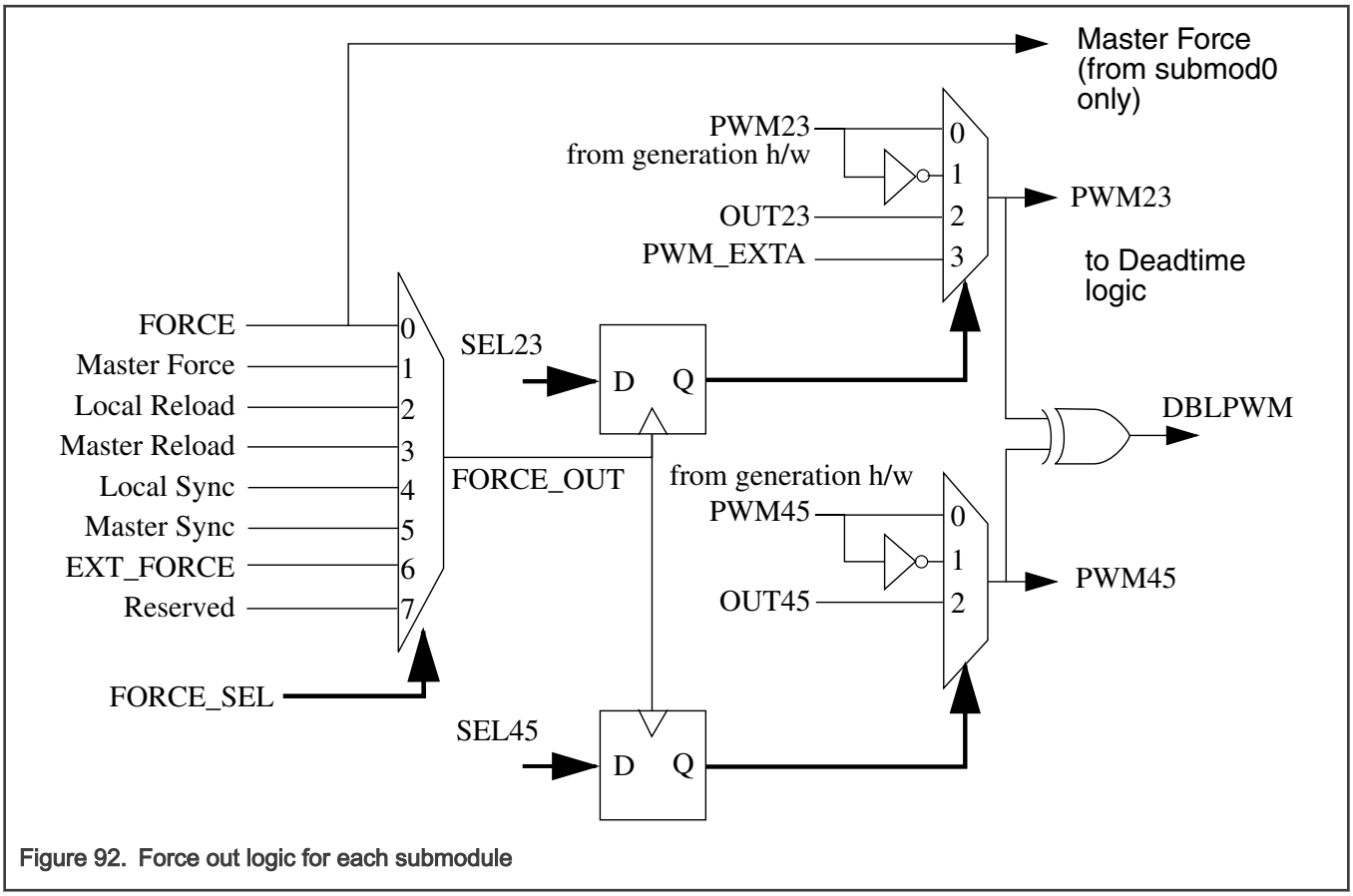


Figure 92. Force out logic for each submodule

The local CTRL2[FORCE] signal of submodule0 can be broadcast as the Master Force signal to other submodules. This feature allows the CTRL2[FORCE] of submodule0 to synchronously update all of the submodule outputs at the same time. The EXT_FORCE signal originates from outside the PWM module from a source such as a timer or digital comparators in the Analog-to-Digital Converter.

31.3.3.6 Independent or complementary channel operation

Writing a logic one to CTRL2[INDEP] configures the pair of PWM outputs as two independent PWM channels. Each PWM output is controlled by its own VALx pair operating independently of the other output.

Writing a logic zero to CTRL2[INDEP] configures the PWM output as a pair of complementary channels. The PWM pins are paired as shown in Figure 93 in complementary channel operation. The signal that is connected to the output pin (PWM23 or PWM45) is determined by MCTRL[IPOL].

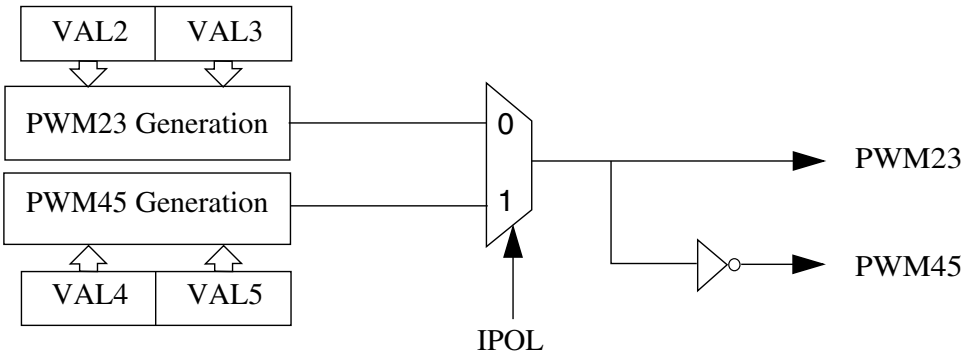


Figure 93. Complementary channel pair for each submodule

The complementary channel operation is for driving top and bottom transistors in a motor drive circuit, as shown in Figure 94.

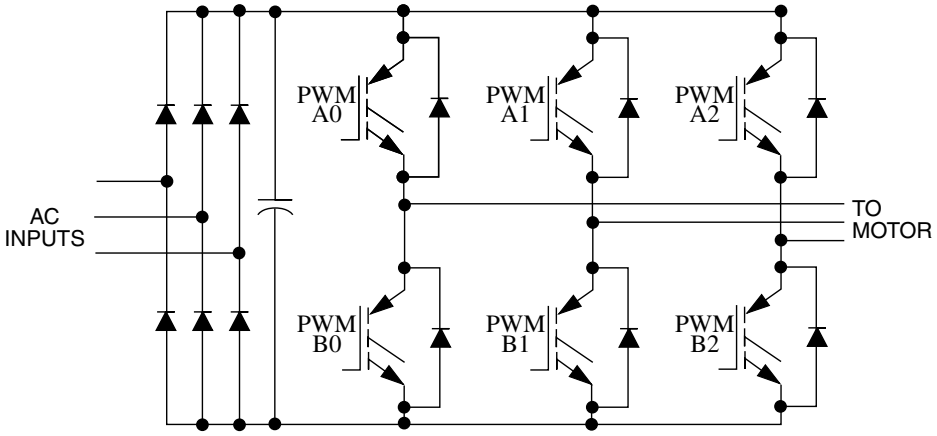


Figure 94. Typical 3 phase AC motor drive

The complementary operation allows the use of the deadtime insertion feature.

31.3.3.7 Deadtime insertion logic

The following figure shows the deadtime insertion logic of each submodule which is used to create non-overlapping complementary signals when not in independent mode.

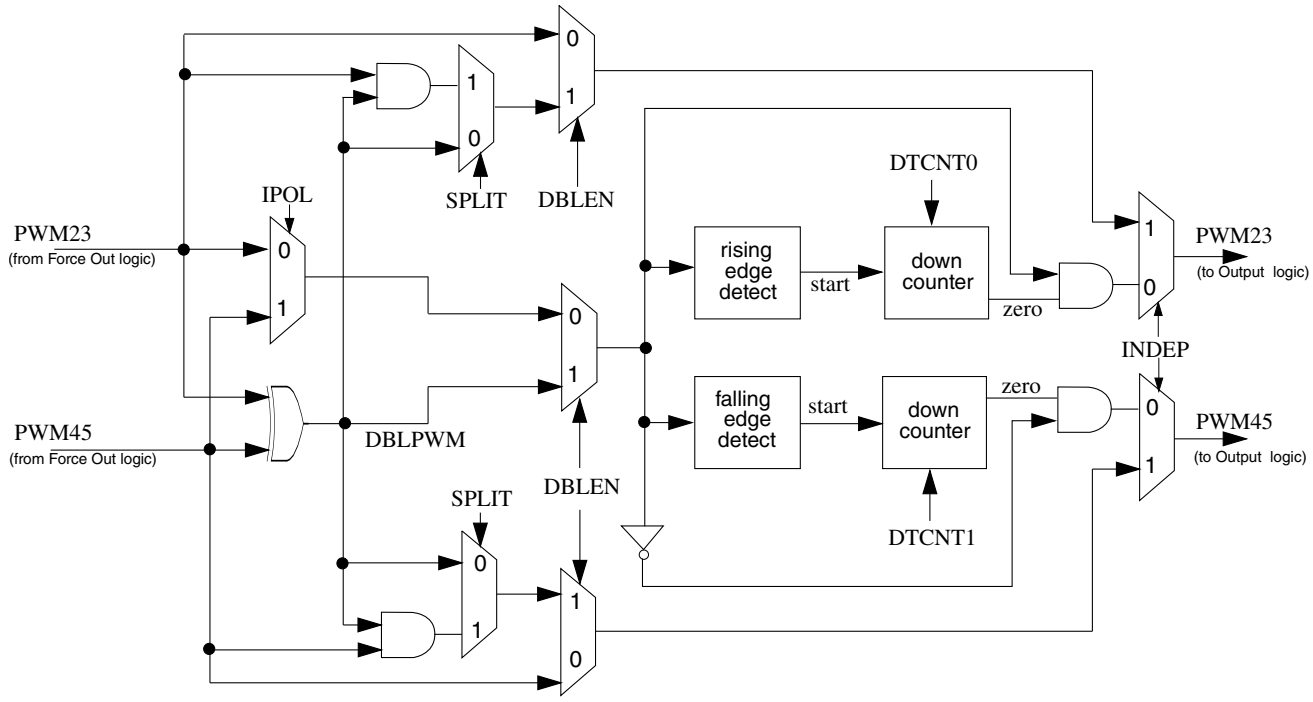


Figure 95. Deadtime Insertion Logic for each submodule

While in the complementary mode, a PWM pair can be used to drive top/bottom transistors, as shown in the figure. When the top PWM channel is active, the bottom PWM channel is inactive, and vice versa.

NOTE

To avoid short-circuiting the DC bus and endangering the transistor, there must be no overlap of conducting intervals between the top and bottom transistors. But the transistor's characteristics make its switching-off time longer than switching-on time. To avoid the conducting overlap of top and bottom transistors, deadtime needs to be inserted in the switching period, as shown in Figure 96.

The deadtime generators automatically insert software-selectable activation delays into the pair of PWM outputs. The deadtime registers (DTCNT0 and DTCNT1) specify the number of IPBus clock cycles to use for deadtime delay. Every time the deadtime generator inputs change state, deadtime is inserted. Deadtime forces both PWM outputs in the pair to the inactive state.

When deadtime is inserted in complementary PWM signals connected to an inverter driving an inductive load, the PWM waveform on the inverter output will have a different duty cycle than what appears on the output pins of the PWM module. This results in a distortion in the voltage applied to the load. A method of correcting this, adding to or subtracting from the PWM value used, is discussed next.

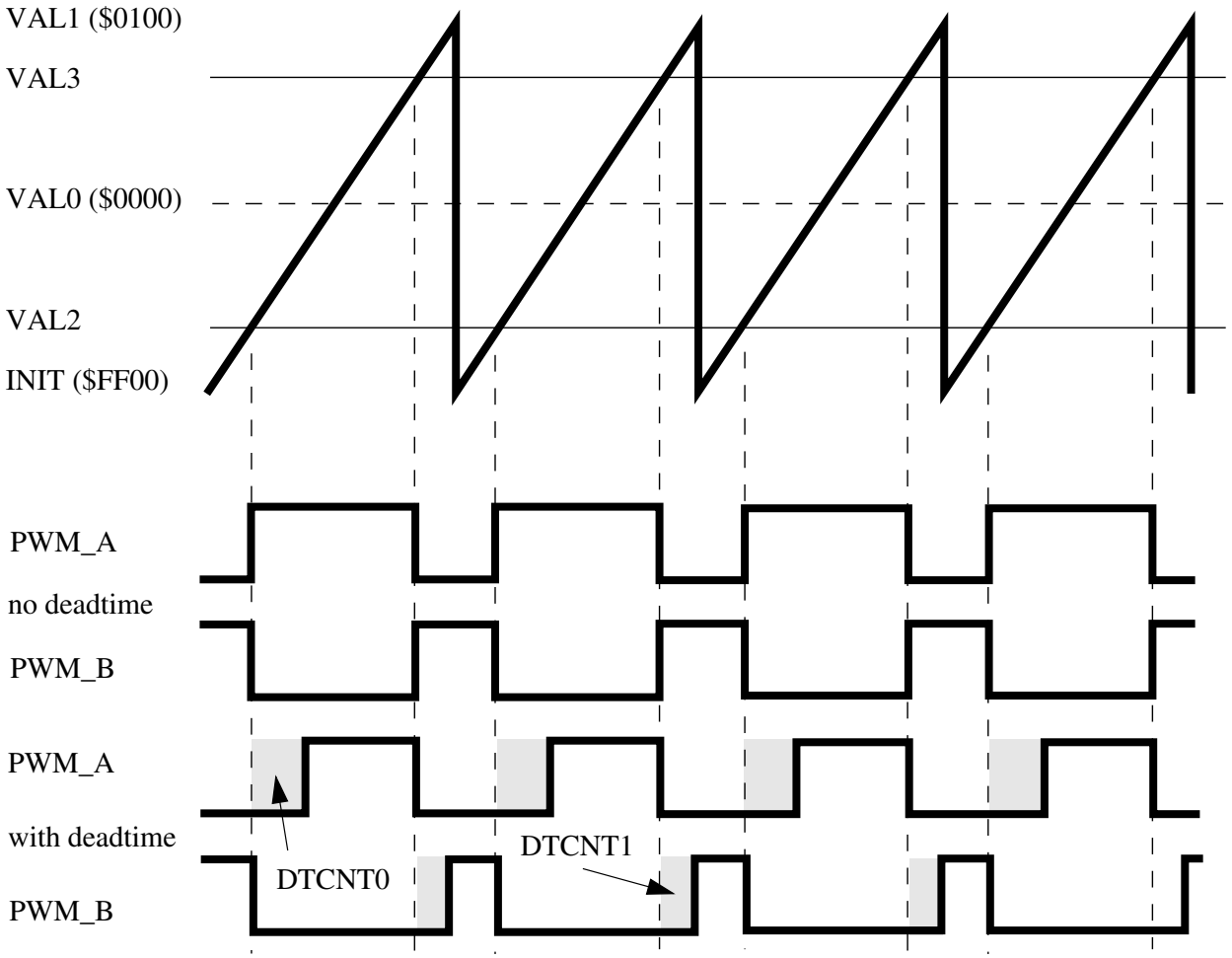


Figure 96. Deadtime insertion

31.3.3.7.1 Top/Bottom correction

In complementary mode, either the top or the bottom transistor controls the output voltage. However, deadtime has to be inserted to avoid overlap of conducting interval between the top and bottom transistor. Both transistors in complementary mode are off during deadtime, allowing the output voltage to be determined by the current status of load and introducing distortion in the output voltage, as shown in Figure 97. On AC induction motors running open-loop, the distortion typically manifests itself as poor low-speed performance, such as torque ripple and rough operation.

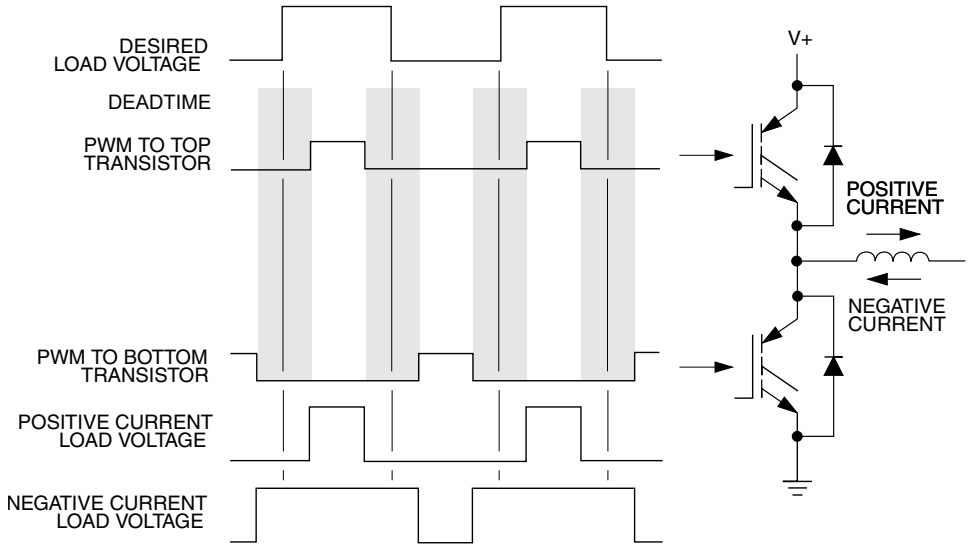


Figure 97. Deadtime distortion

During deadtime, load inductance distorts output voltage by keeping current flowing through the diodes. This deadtime current flow creates a load voltage that varies with the current direction. With a positive current flow, the load voltage during deadtime is equal to the bottom supply, putting the top transistor in control. With a negative current flow, the load voltage during deadtime is equal to the top supply putting the bottom transistor in control.

Remembering that the original PWM pulse widths were shortened by deadtime insertion, the averaged sinusoidal output is less than the desired value. However, when deadtime is inserted, it distorts in the motor current waveform inverter outputs. This distortion is aggravated by different turn-on and turn-off delays of each of the transistors. By giving the PWM module information on which transistor is controlling at a given time this distortion can be corrected.

For a typical circuit in complementary channel operation, only one of the transistors is effective in controlling the output voltage at any given time. This depends on the direction of the motor inverter current for that pair, as shown in Figure 97. To correct distortion one of two different factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the software is responsible for calculating both compensated PWM values prior to placing them in the VALx registers. Either the VAL2/VAL3 or the VAL4/VAL5 register pair controls the pulse width at any given time. For a given PWM pair, whether the VAL2/VAL3 or VAL4/VAL5 pair is active depends on either:

- The state of the current status pin, PWMX, for that driver
- The state of the odd/even correction bit, MCTRL[IPOL], for that driver

To correct deadtime distortion, the software can decrease or increase the value in the appropriate VALx register.

- In edge-aligned operation, decreasing or increasing the PWM value by a correction value equal to the deadtime typically compensates for deadtime distortion.
- In center-aligned operation, decreasing or increasing the PWM value by a correction value equal to one-half the deadtime typically compensates for deadtime distortion.

31.3.3.7.2 Manual correction

To detect the current status, the voltage on each PWMX pin is sampled twice in a PWM period, at the end of each deadtime. The value is stored in CTRL[DT]. CTRL[DT] is a timing marker indicating when to toggle between PWM value registers. The software can then set MCTRL[IPOL] to switch between VAL2/VAL3 and VAL4/VAL5 register pairs according to CTRL[DT] values.

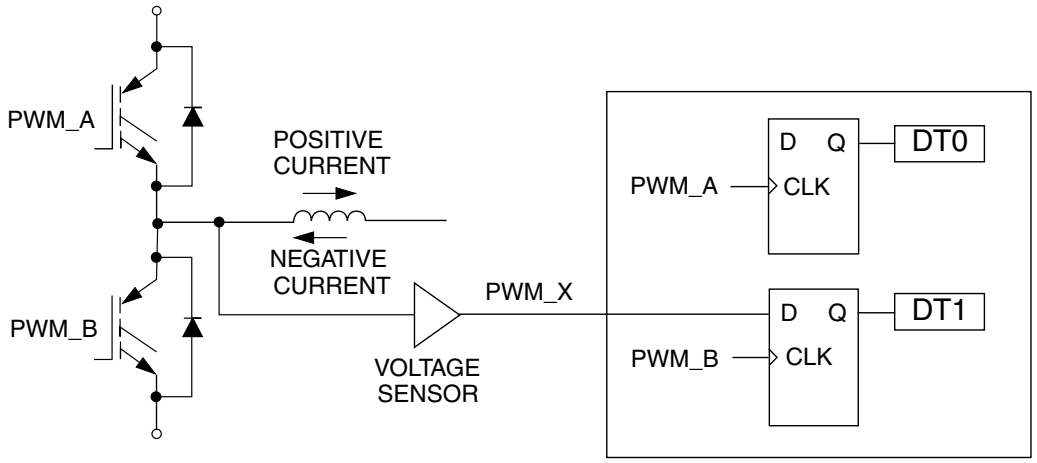


Figure 98. Current-status sense scheme for deadtime correction

Both D flip-flops latch low, CTRL[DT] = 00, during deadtime periods if the current is large and flowing out of the complementary circuit. See the preceding figure. Both D flip-flops latch the high, CTRL[DT] = 11, during deadtime periods if the current is also large and flowing into the complementary circuit.

However, under low-current, the output voltage of the complementary circuit during deadtime is between the high and low levels. The current cannot free-wheel through the opposition anti-body diode, regardless of polarity, giving additional distortion when the current crosses zero. Sampled results are CTRL[DT] = b10. Thus, the best time to change one PWM value register to another is just before the current zero crossing.

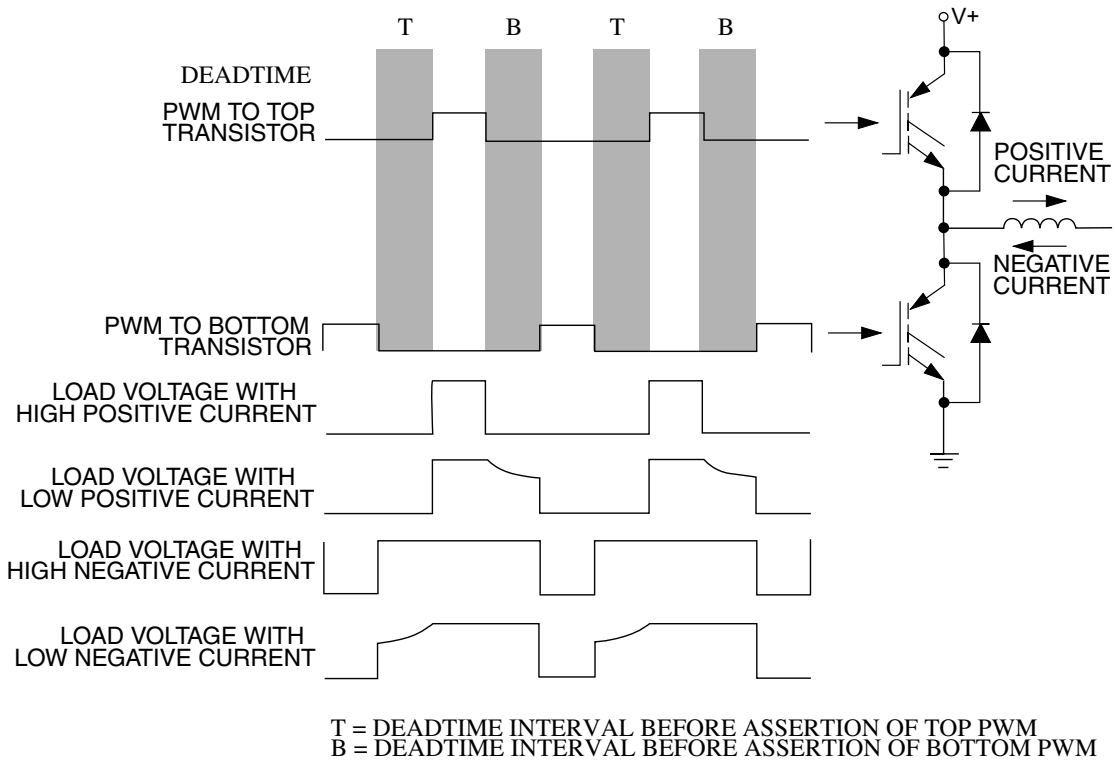


Figure 99. Output voltage waveforms

31.3.3.8 Output logic

The following figure shows the output logic of each submodule including how each PWM output has individual fault disabling, polarity control, and output enable. This allows for maximum flexibility when interfacing with the external circuitry.

The PWM23 and PWM45 signals which are output from the deadtime logic (as shown in Figure 100) are positive true signals. In other words, a high level on these signals should result in the corresponding transistor in the PWM inverter being turned ON. The voltage level required at the PWM output pin to turn the transistor ON or OFF is a function of the logic between the pin and the transistor. Therefore, it is imperative that the user program OCTRL[POLA] and OCTRL[POLB] before enabling the output pins. A fault condition can result in the PWM output being put in a high-impedance state, forced to a logic 1 state, or forced to a logic 0 state, depending on the values programmed into the OCTRL[PWMxFS] fields.

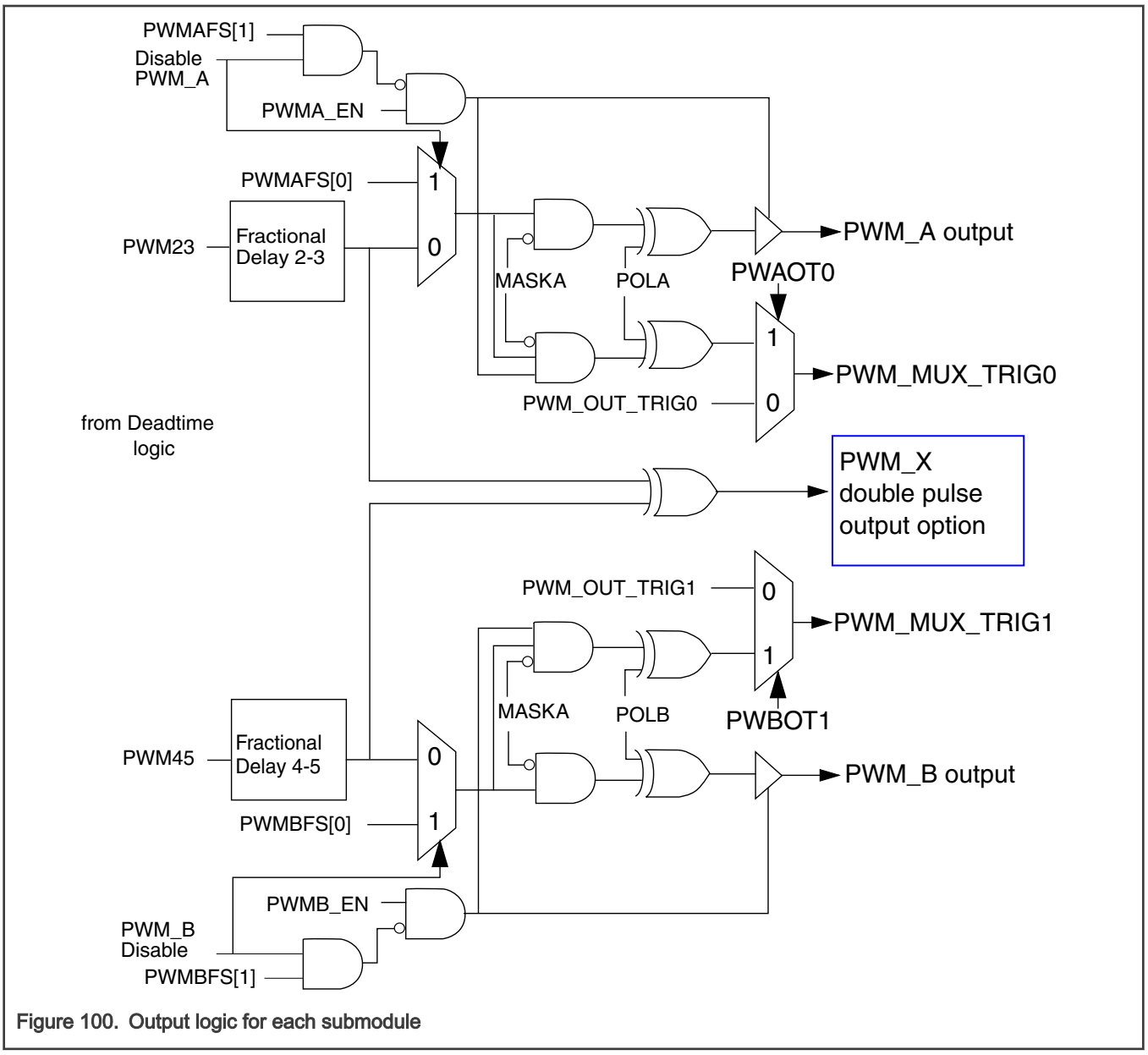


Figure 100. Output logic for each submodule

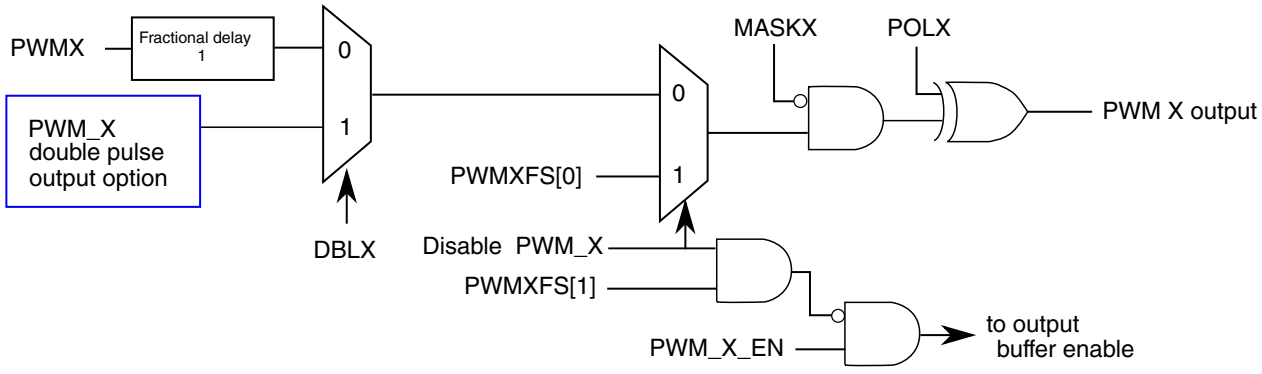


Figure 101. Output logic continued

31.3.3.9 E-Capture

The Enhanced Capture (E-Capture) logic is designed to measure both edges of an input signal. As a result, when a submodule pin is configured for input capture, the CVALx registers associated with that pin are used to record the edge values.

The following figure shows block diagram of the E-Capture circuit. Upon entering the pin input, the signal is split into two paths. One goes straight to a mux input where software can select to pass the signal directly to the capture logic for processing. The other path connects the signal to an 8-bit counter which counts both the rising and falling edges of the signal. The output of this counter is compared to an 8-bit value that is specified by the user (EDGCMPx) and when the two values are equal, the comparator generates a pulse that resets the counter. This pulse is also supplied to the mux input where software can select it to be processed by the capture logic. This feature permits the E-Capture circuit to count up to 256 edge events before initiating a capture event. This feature is useful for dividing down high frequency signals for capture processing so that capture interrupts do not overwhelm the CPU. Also, this feature can be used to generate an interrupt after "n" events have been counted.

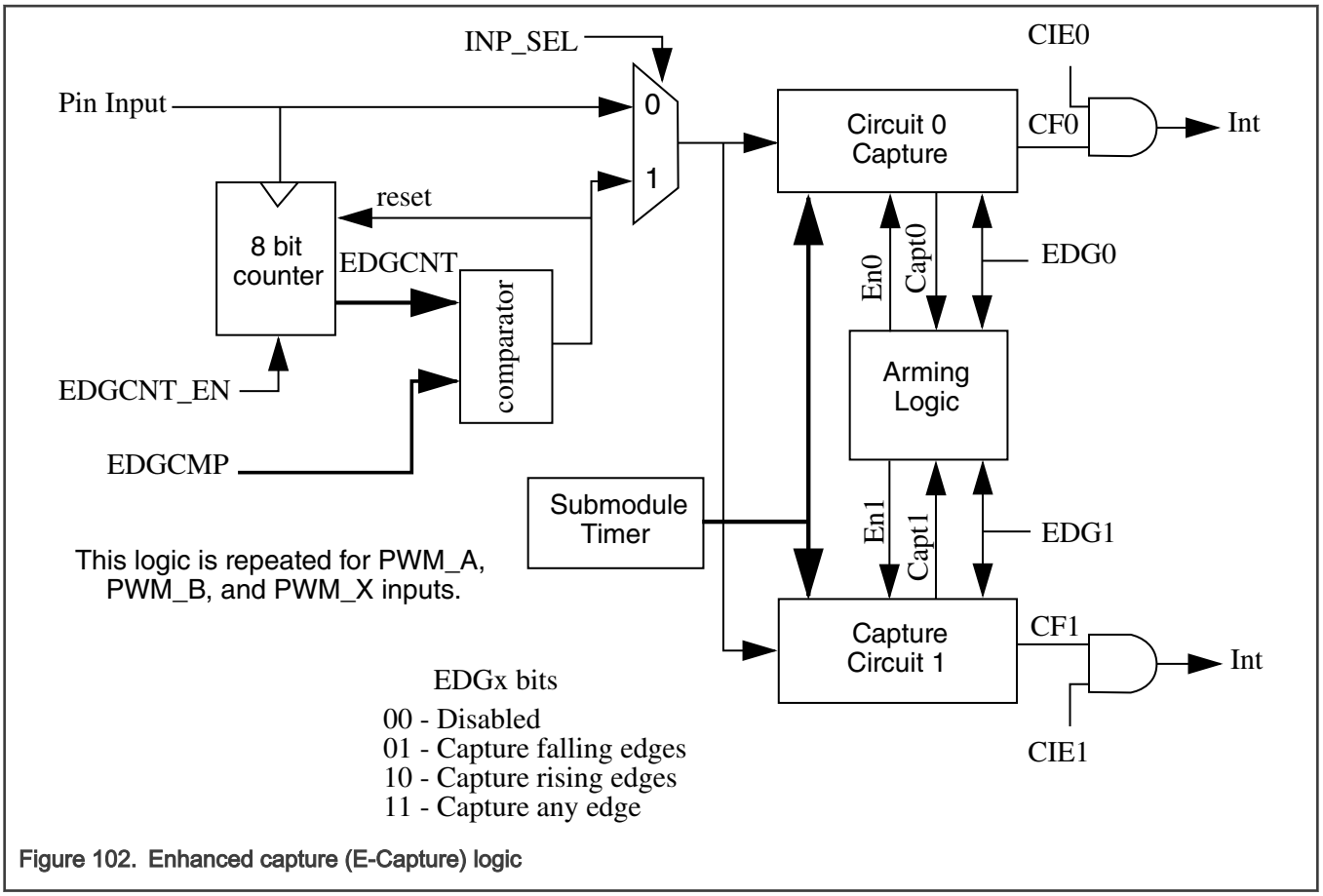


Figure 102. Enhanced capture (E-Capture) logic

Based on the mode selection, the mux selects either the pin input or the compare output from the count/compare circuit to be processed by the capture logic. The selected signal is routed to two separate capture circuits which work in tandem to capture sequential edges of the signal. The type of edge to be captured by each circuit is determined by CAPTCTRLx[EDGx1] and CAPTCTRLx[EDGx0], whose functionality is listed in the above figure. Also, controlling the operation of the capture circuits is the arming logic which allows captures to be performed in a free running (continuous) or one shot mode. In free running mode, the capture sequences will be performed indefinitely. If both capture circuits are enabled, they will work together in a ping-pong style where a capture event from one circuit leads to the arming of the other and vice versa. In one shot mode, only one capture sequence will be performed. If both capture circuits are enabled, capture circuit 0 is first armed and when a capture event occurs, capture circuit 1 is armed. Once the second capture occurs, further captures are disabled until another capture sequence is initiated. Both capture circuits are also capable of generating an interrupt to the CPU.

31.3.3.10 Fault protection

Fault protection can control any combination of PWM output pins. Faults are generated by a logic one on any of the FAULTx pins. This polarity can be changed via FCTRL[FLVL]. Each FAULTx pin can be mapped arbitrarily to any of the PWM outputs. When fault protection hardware disables PWM outputs, the PWM generator continues to run, only the output pins are forced to logic 0, logic 1, or high impedance depending on the values of OCTRL[PWMxFS].

The fault decoder disables PWM pins selected by the fault logic and the disable mapping (DISMAPn) registers. The following figure shows an example of the fault disable logic. Each bank of bits in DISMAPn control the mapping for a single PWM pin. See the following table.

The fault protection is enabled even when the PWM module is not enabled. Therefore, a fault is latched in and must be cleared to prevent an interrupt when the PWM is enabled.

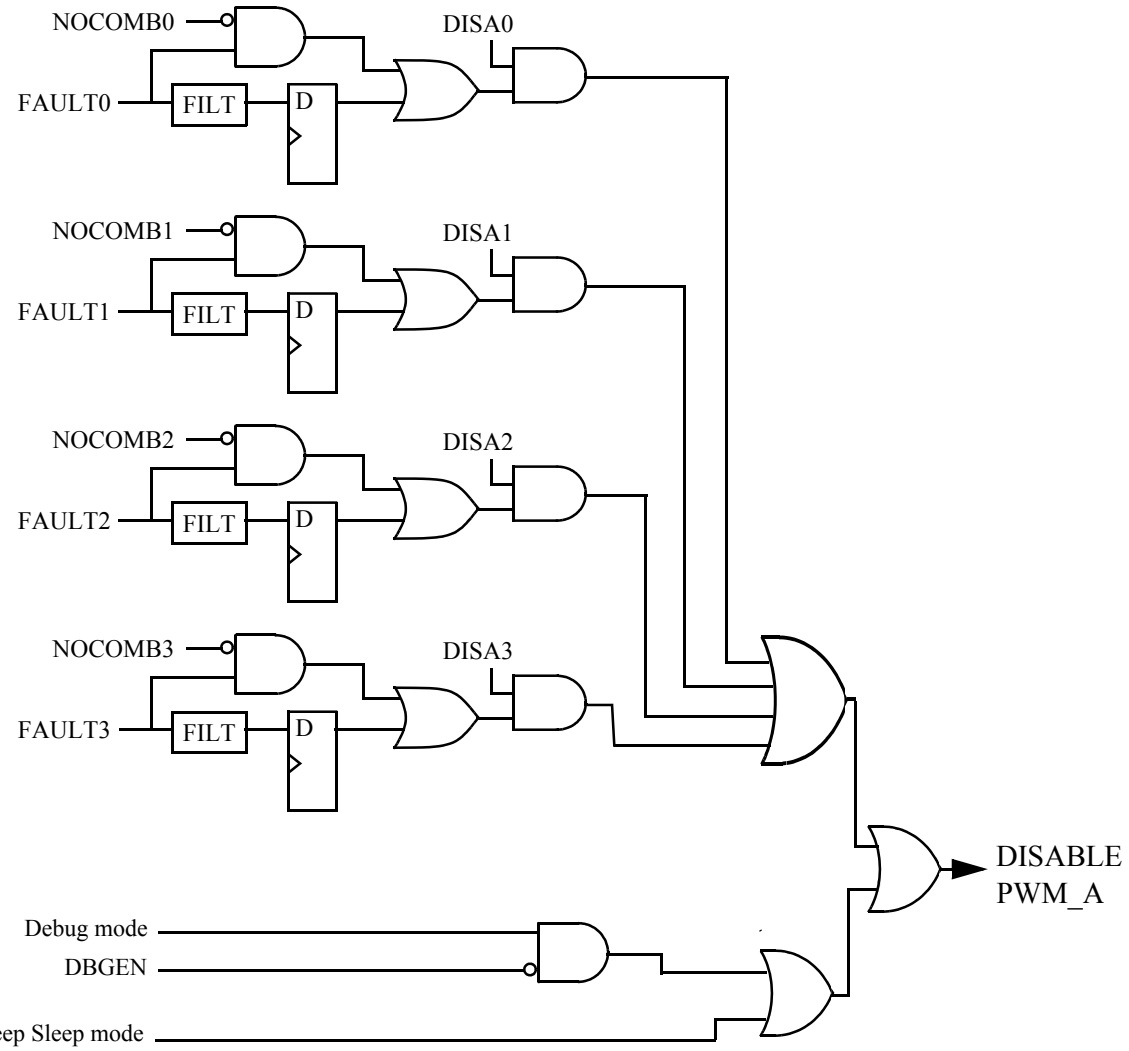


Figure 103. Fault decoder for PWM_A

Table 171. Fault Mapping

PWM Pin for 1 submodule	Controlling Register Bits
PWM_A	DISMAP0[DIS0A]
PWM_B	DISMAP0[DIS0B]
PWM_X	DISMAP0[DIS0X]

31.3.3.10.1 Fault pin filter

Each fault pin has a programmable filter that can be bypassed. The sampling period of the filter can be adjusted with FFILT[FILT_PER]. The number of consecutive samples that must agree before an input transition is recognized can be adjusted using FFILT[FILT_CNT]. Setting FFILT[FILT_PER] to all 0 disables the input filter for a given FAULTx pin.

Upon detecting a logic 0 on the filtered FAULTx pin (or a logic 1 if FCTRL[FLVLx] is set), the corresponding FSTS[FFPINx] and FSTS[FFLAGx] bits are set. FSTS[FFPINx] remains set as long as the filtered FAULTx pin is zero. Clear FSTS[FFLAGx] by writing a logic 1 to FSTS[FFLAGx].

If the FIEx, FAULTx pin interrupt enable bit is set, FSTS[FFLAGx] generates a CPU interrupt request. The interrupt request latch remains set until:

- Software clears FSTS[FFLAGx] by writing a logic one to the bit
- Software clears the FIEx bit by writing a logic zero to it
- A reset occurs

Even with the filter enabled, there is a combinational path from the FAULTx inputs to the PWM pins, which in turn bypasses the filter when FCTRL20[NOCOMBx] = 0. This logic is also capable of holding a fault condition in the event of loss of clock to the PWM module.

31.3.3.10.2 Automatic fault clearing

Setting an automatic clearing mode bit, FCTRL[FAUTOx] configures faults from the FAULTx pin for automatic clearing.

When FCTRL[FAUTOx] is set, disabled PWM pins are enabled when the FAULTx pin returns to logic one and a new PWM full or half cycle begins. See the following figure. If FSTS[FFULLx] is set and the fault condition on FAULTx disappears, then the disabled PWM pins are enabled at the start of next full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle. Clearing FSTS[FFLAGx] does not affect disabled PWM pins when FCTRL[FAUTOx] is set.

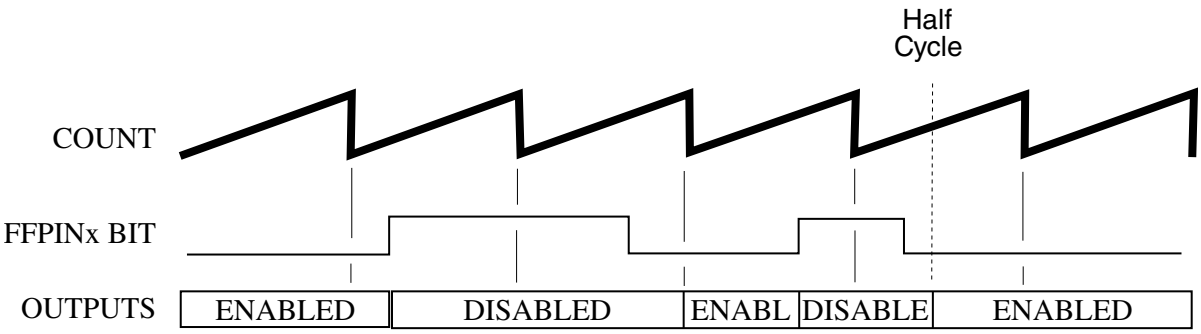


Figure 104. Automatic fault clearing

31.3.3.10.3 Manual fault clearing

Clearing the automatic clearing mode bit, FCTRL[FAUTOx], configures faults from the FAULTx pin for manual clearing:

- If the fault safety mode bits FCTRL[FSAFEx] are clear, then PWM pins disabled by the FAULTx pins are enabled when:
  - Software clears the corresponding FSTS[FFLAGx] flag
  - The pins are enabled when the next PWM full or half cycle begins regardless of the logic level detected by the filter at the FAULTx pin, as shown in Figure 105. If FSTS[FFULLx] is set, then the disabled PWM pins are enabled at the start of a full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle.
- If the fault safety mode bits FCTRL[FSAFEx] are set, then PWM pins disabled by the FAULTx pins are enabled when:
  - Software clears the corresponding FSTS[FFLAGx] flag
  - The filter detects a logic zero on the FAULTx pin at the start of the next PWM full or half cycle boundary, as shown in Figure 106. If FSTS[FFULLx] is set, then the disabled PWM pins are enabled at the start of a full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle.

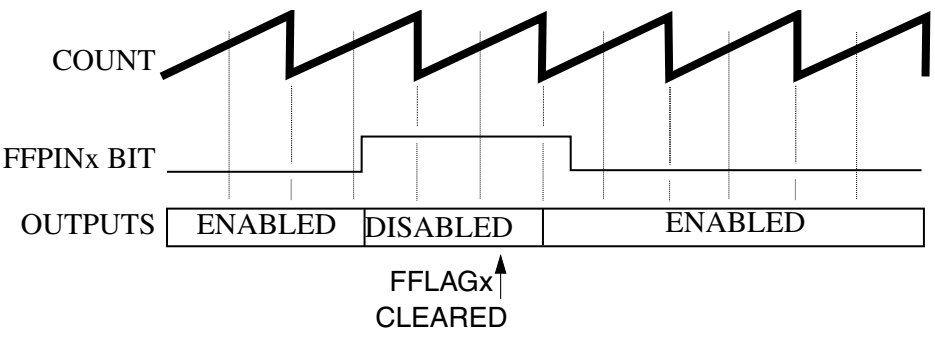


Figure 105. Manual fault clearing (FCTRL[FSAFEx] = 0, FSTS[FFULLx] = 1)

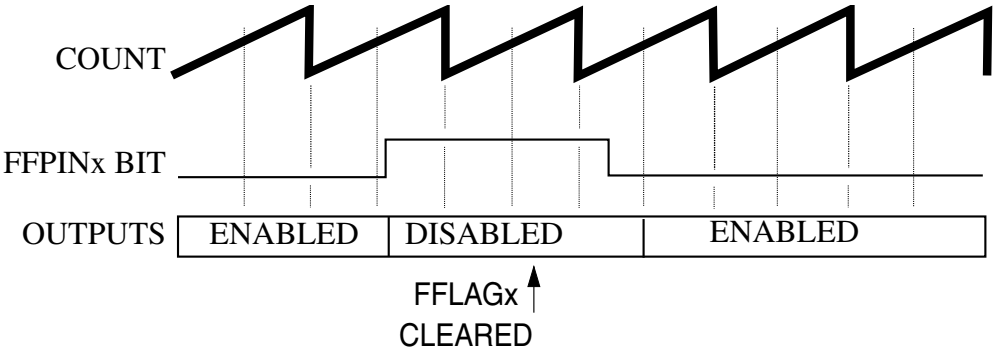


Figure 106. Manual fault clearing (FCTRL[FSAFEx] = 1, FSTS[FHALFx] = 1)

NOTE

Fault protection also applies during software output control when the SEL23 and SEL45 fields are set to select OUT23 and OUT45 bits or PWM_EXTA. Fault clearing still occurs at half or full PWM cycle boundaries while the PWM generator is engaged, MCTRL[RUN] equals one. But the OUTx bits can control the PWM pins while the PWM generator is off, MCTRL[RUN] equals zero. Thus, fault clearing occurs at IPBus cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

31.3.3.10.4 Fault testing

FTST[FTEST] is used to simulate a fault condition on each of the fault inputs within that fault channel.

31.3.3.11 PWM generator loading

31.3.3.11.1 Load enable

MCTRL[LDOK] enables loading of the following PWM generator parameters.

- The prescaler divisor: from CTRL[PRSC]
- The PWM period and pulse width: from the INIT, FRACVALx, and VALx registers

MCTRL[LDOK] allows the software to finish calculating all of these PWM parameters so they can be synchronously updated. The CTRL[PRSC], INIT, and VALx registers are loaded by software into a set of outer buffers. When MCTRL[LDOK] is set, these values are transferred to an inner set of registers at the beginning of the next PWM reload cycle to be used by the PWM generator when CTRL[LDMOD] is cleared. These values can be transferred to the inner set of registers immediately upon setting MCTRL[LDOK] if CTRL[LDMOD] is set. Set MCTRL[LDOK] by reading it when it is a logic zero and then writing a logic one to it. After loading, MCTRL[LDOK] is automatically cleared.

31.3.3.11.2 Load frequency

CTRL[LDFQ] selects an integral loading frequency of one to 16 PWM reload opportunities. CTRL[LDFQ] takes effect at every PWM reload opportunity, regardless of the state of MCTRL[LDOK]. CTRL[HALF] and CTRL[FULL] control reload timing. If CTRL[FULL] is set, a reload opportunity occurs at the end of every PWM cycle when the count equals VAL1. If CTRL[HALF] is set, a reload opportunity occurs at the half cycle when the count equals VAL0. If both CTRL[HALF] and CTRL[FULL] are set, a reload opportunity occurs twice per PWM cycle when the count equals VAL1 and when it equals VAL0.

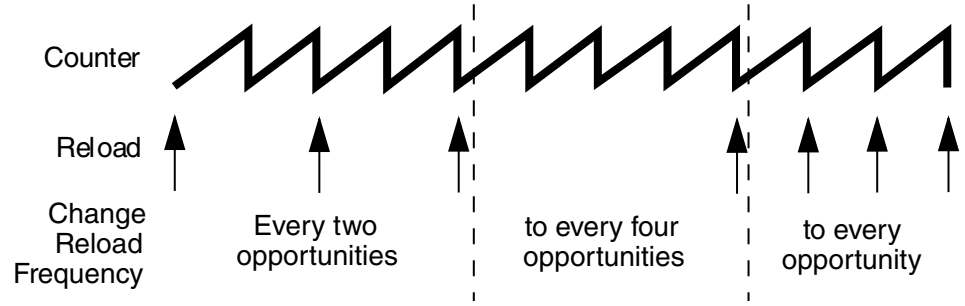


Figure 107. Full cycle reload frequency change

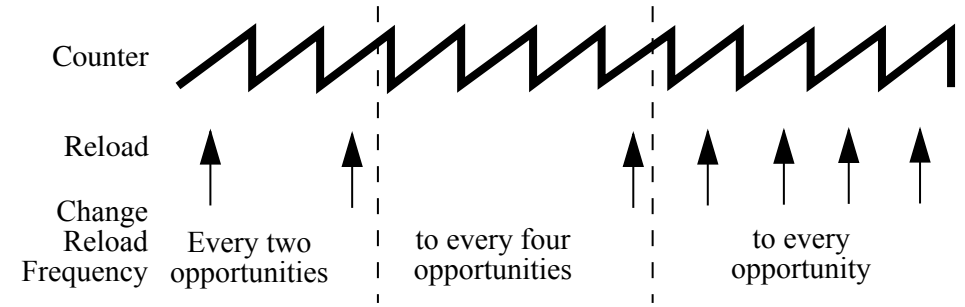


Figure 108. Half cycle reload frequency change

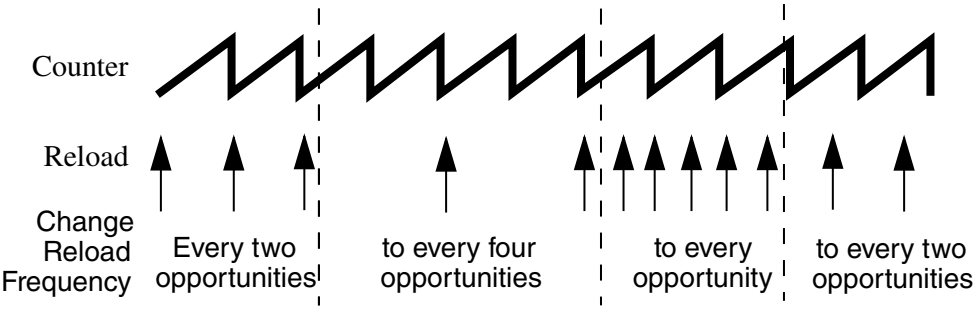


Figure 109. Full and half cycle reload frequency change

31.3.3.11.3 Reload flag

At every reload opportunity the PWM Reload Flag (STS[RF]) is set. Setting STS[RF] happens even if an actual reload is prevented by MCTRL[LDOK]. If the PWM reload interrupt enable bit INTEN[RIE] is set, the STS[RF] flag generates CPU interrupt requests allowing software to calculate new PWM parameters in real time. When INTEN[RIE] is not set, reloads still occur at the selected reload rate without generating CPU interrupt requests.

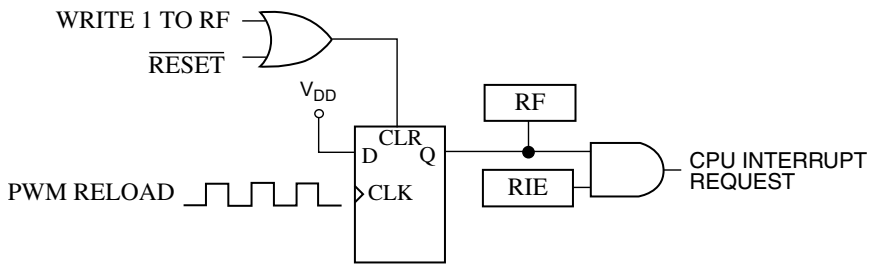


Figure 110. PWMF reload interrupt request

31.3.3.11.4 Reload errors

When one of the INIT, VALx, FRACVALx, or CTRL[PRSC] registers is updated (written by software), the STS[RUF] flag is set to indicate the data in the set of double buffered registers is not coherent. STS[RUF] is cleared by a successful reload which consists of the reload signal while MCTRL[LDOK] is set. If STS[RUF] is set and MCTRL[LDOK] is clear when the reload signal occurs, a reload error takes place and STS[REF] is set. If STS[RUF] is clear when a reload signal asserts, then the data is coherent and no error is flagged.

31.3.3.11.5 Initialization

Initialize all registers and then set MCTRL[LDOK] before setting MCTRL[RUN].

NOTE

If MCTRL[LDOK] is not set, setting MCTRL[RUN] also sets the STS[RF] flag. To prevent a CPU interrupt request, clear INTEN[RIE] before setting MCTRL[RUN].

The PWM generator uses the last values loaded if MCTRL[RUN] is cleared and then set while MCTRL[LDOK] equals zero.

When MCTRL[RUN] is cleared:

- The STS[RF] flag and pending CPU interrupt requests are not cleared
- All fault circuitry remains active
- Software/external output control remains active
- Deadtime insertion continues during software/external output control

31.3.4 Power modes

Be careful when using this module in Deep Sleep, Sleep, and Debug operating modes.

CAUTION

Some applications require regular software updates for proper operation. Failure to provide regular software updates could result in destroying the hardware setup.

To accommodate this situation, PWM outputs are placed in their inactive states in Deep Sleep mode, and they can optionally be placed in inactive states in Sleep and Debug modes. PWM outputs are reactivated (assuming they were active beforehand) when these modes are exited.

Table 172. Modes when PWM operation is restricted

Mode	Description
Deep Sleep	PWM outputs are inactive.
Debug	PWM outputs are driven or inactive as a function of CTRL2[DBGEN].

31.3.5 Clocking

Figure 111 shows the logic used to generate the main counter clock. Each submodule can select between three clock signals: the IPBus clock, EXT_CLK, and AUX_CLK. The EXT_CLK goes to all of the submodules. The AUX_CLK signal is broadcast from submodule0 and can be selected as the clock source by other submodules so that the 8-bit prescaler and MCTRL[RUN] from submodule0 can control all of the submodules.

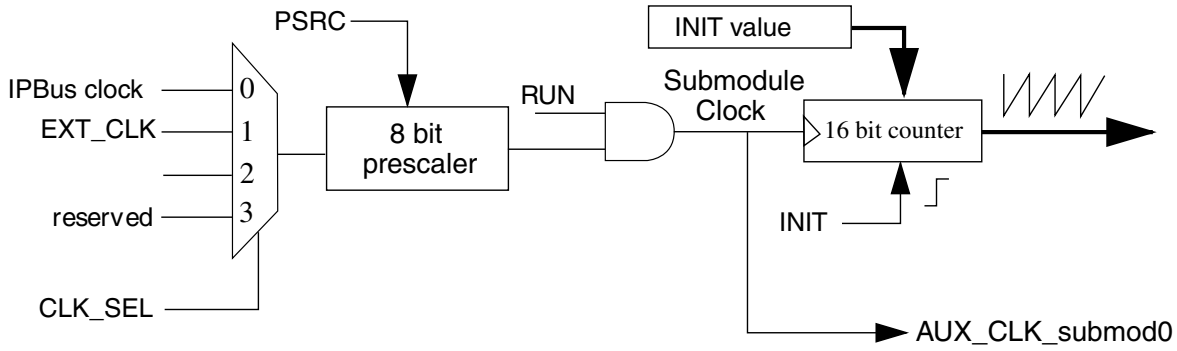


Figure 111. Clocking Block Diagram for PWM Submodule 0

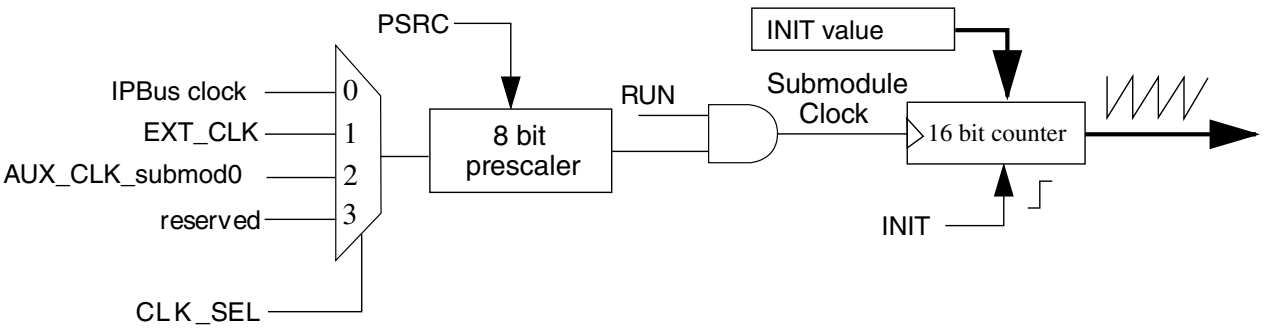


Figure 112. Clocking Block Diagram for all other PWM Submodules

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the IPBus clock frequency by 1-128. The prescaler bits, CTRL[PRSC], select the prescaler divisor. This prescaler is buffered and will not be used by the PWM generator until MCTRL[LDOK] is set and a new PWM reload cycle begins or CTRL[LDMOD] is set.

31.3.6 Resets

All PWM registers are reset to their default values upon any system reset.

The reset forces all registers to their reset states and tristates the PWM outputs.

31.3.7 Interrupts

Each of the submodules within the eFlexPWM module can generate an interrupt from several sources. The fault logic can also generate interrupts. The interrupt service routine (ISR) must check the related interrupt enables and interrupt flags to determine the actual cause of the interrupt.

Table 173. Interrupt Summary

Core Interrupt	Interrupt Flag	Interrupt Enable	Name	Description
PWM_CMP0	SM0STS[CMPF]	SM0INTEN[CMPIE]	Submodule 0 compare interrupt	Compare event has occurred
PWM_CAP0	SM0STS[CFX1], SM0STS[CFX0]	SM0INTEN[CFX1IE], SM0INTEN[CFX0IE]	Submodule 0 input capture interrupt	Input capture event has occurred
PWM_RELOAD0	SM0STS[RF]	SM0INTEN[RIE]	Submodule 0 reload interrupt	Reload event has occurred
PWM_CMP1	SM1STS[CMPF]	SM1INTEN[CMPIE]	Submodule 1 compare interrupt	Compare event has occurred
PWM_CAP1	SM1STS[CFX1], SM1STS[CFX0]	SM1INTEN[CFX1IE], SM1INTEN[CFX0IE]	Submodule 1 input capture interrupt	Input capture event has occurred
PWM_RELOAD1	SM1STS[RF]	SM1INTEN[RIE]	Submodule 1 reload interrupt	Reload event has occurred
PWM_CMP2	SM2STS[CMPF]	SM2INTEN[CMPIE]	Submodule 2 compare interrupt	Compare event has occurred
PWM_CAP2	SM2STS[CFX1], SM2STS[CFX0]	SM2INTEN[CFX1IE], SM2INTEN[CFX0IE	Submodule 2 input capture interrupt	Input capture event has occurred
PWM_RELOAD2	SM2STS[RF]	SM2INTEN[RIE]	Submodule 2 reload interrupt	Reload event has occurred
PWM_RERR	SM0STS[REF]	SM0INTEN[REIE]	Submodule 0 reload error interrupt	Reload error has occurred
	SM1STS[REF]	SM1INTEN[REIE]	Submodule 1 reload error interrupt	
	SM2STS[REF]	SM2INTEN[REIE]	Submodule 2 reload error interrupt	
PWM_FAULT	FSTS[FFLAG]	FCTRL[FIE]	Fault input interrupt	Fault condition has been detected

31.3.8 DMA

Each submodule can request a DMA read access for its capture FIFOs and a DMA write request for its double buffered registers.

Table 174. DMA summary

DMA request	DMA enable	Name	Description
Submodule 0 read request	SM0DMAEN[CX0DE]	SM0 Capture FIFO X0 read request	SM0CVAL0 contains a value to be read

Table continues on the next page...

Table 174. DMA summary (continued)

DMA request	DMA enable	Name	Description
Submodule 0 read request	SM0DMAEN[CX1DE]	SM0 Capture FIFO X1 read request	SM0CVAL1 contains a value to be read
Submodule 0 read request	SM0DMAEN[CAPTDE]	SM0 Capture FIFO read request source select	Selects source of submodule0 read DMA request
Submodule 0 write request	SM0DMAEN[VALDE]	SM0VALx write request	SM0VALx and SM0FRACVALx registers need to be updated
Submodule 1 read request	SM1DMAEN[CX0DE]	SM1 Capture FIFO X0 read request	SM1CVAL0 contains a value to be read
Submodule 1 read request	SM1DMAEN[CX1DE]	SM1 Capture FIFO X1 read request	SM1CVAL1 contains a value to be read
Submodule 1 read request	SM1DMAEN[CAPTDE]	SM1 Capture FIFO read request source select	Selects source of submodule1 read DMA request
Submodule 1 write request	SM1DMAEN[VALDE]	SM1VALx write request	SM1VALx and SM1FRACVALx registers need to be updated
Submodule 2 read request	SM2DMAEN[CX0DE]	SM2 Capture FIFO X0 read request	SM2CVAL0 contains a value to be read
Submodule 2 read request	SM2DMAEN[CX1DE]	SM2 Capture FIFO X1 read request	SM2CVAL1 contains a value to be read
Submodule 2 read request	SM2DMAEN[CAPTDE]	SM2 Capture FIFO read request source select	Selects source of submodule2 read DMA request
Submodule 2 write request	SM2DMAEN[VALDE]	SM2VALx write request	SM2VALx and SM2FRACVALx registers need to be updated

31.4 External signals

The PWM has pins named PWM_An, PWM_Bn, PWM_Xn, FAULTn, EXT_SYNC, EXT_FORCE, and PWMn_EXTA. The PWM also has an on-chip input called EXT_CLK and output signals called PWMn_OUT_TRIGx and PWMn_MUX_TRIGx.

31.4.1 PWM_An and PWM_Bn - External PWM output pair

These pins are the output pins of the PWM channels. These pins can be independent PWM signals or a complementary pair. When not needed as an output, they can be used as inputs to the input capture circuitry.

31.4.2 PWM_Xn - Auxiliary PWM output signal

These pins are the auxiliary output pins of the PWM channels. They can be independent PWM signals. When not needed as an output, they can be used as inputs to the input capture circuitry or used to detect the polarity of the current flowing through the complementary circuit at deadtime correction.

31.4.3 FAULTn - Fault Inputs

These are input pins for disabling selected PWM outputs.

31.4.4 EXT_SYNC - External synchronization signal

These input signals allow a source external to the PWM to initialize the PWM counter. Therefore, the PWM can be synchronized to external circuitry.

31.4.5 EXT_FORCE - External output force signal

This input signal allows a source external to the PWM to force an update of the PWM outputs. Therefore, the PWM can be synchronized to external circuitry.

31.4.6 PWMn_EXTA - Alternate PWM control signals

These pins allow an alternate source to control the PWM_An and PWM_Bn outputs. Typically, the PWMn_EXTA input (depending on the state of MCTRL[IPOL]) is used for the generation of a complementary pair. Typical control signals include ADC conversion high/low limits, TMR outputs, GPIO inputs, and comparator outputs.

For pin input details, see chip-specific eFlexPWM information.

31.4.7 PWMn_OUT_TRIG0 and PWMn_OUT_TRIG1 - Output triggers

These outputs allow the PWM submodules to control timing of ADC conversions. See the description of the SMnTCTRL[OUT_TRIG_EN] for information about how to enable these outputs and how the compare registers match up to the output triggers.

31.4.8 PWM[n]_MUX_TRIG0 and PWM[n]_MUX_TRIG1 - Output triggers

These outputs can be either PWMn_OUT_TRIG0/PWMn_OUT_TRIG1 signals or PWM_A/PWM_B signals from output logic. See the description of the PWAOT0 and PWBOT1 bits in the Output Trigger Control Register for information about how to enable these outputs.

31.4.9 EXT_CLK - External clock signal

This signal allows a source external to the PWM (typically a timer or an off-chip source) to control the PWM clocking. Therefore, the PWM can be synchronized to the timer, or multiple chips can be synchronized to each other.

31.5 PWM register descriptions

The address of a register is the sum of a base address and an address offset. The base address is defined at the core level, and the address offset is defined at the module level. The PWM module has a set of registers for each PWM submodule, for the configuration logic, and for each fault channel.

NOTE

While the registers are 16-bit wide, they can be accessed in pairs as 32-bit registers, if the pairs are word-aligned.

Submodule registers are repeated for each PWM submodule. To designate which submodule that they are in, register names are prefixed with SMx(x=0,1...). The base address of submodule 0 is the same as the base address for the PWM module as a whole. The base address of submodule 1 is offset 0x60 from the base address for the PWM module as a whole. This 0x60 offset is based on the number of registers in a submodule. The base address of submodule 2 is equal to the base address of submodule 1 plus this same 0x60 offset.

The base address of the configuration registers is equal to the base address of the PWM module as a whole plus an offset of 0x180.

31.5.1 PWM memory map

FlexPWM0 base address: 400A_9000h

FlexPWM1 base address: 400A_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Counter Register (SM0CNT)	16	R	0000h
2h	Initial Count Register (SM0INIT)	16	RW	0000h
4h	Control 2 Register (SM0CTRL2)	16	RW	0000h
6h	Control Register (SM0CTRL)	16	RW	0400h
Ah	Value Register 0 (SM0VAL0)	16	RW	0000h
Eh	Value Register 1 (SM0VAL1)	16	RW	0000h
12h	Value Register 2 (SM0VAL2)	16	RW	0000h
16h	Value Register 3 (SM0VAL3)	16	RW	0000h
1Ah	Value Register 4 (SM0VAL4)	16	RW	0000h
1Eh	Value Register 5 (SM0VAL5)	16	RW	0000h
22h	Output Control Register (SM0OCTRL)	16	RW	0000h
24h	Status Register (SM0STS)	16	RW	0000h
26h	Interrupt Enable Register (SM0INTEN)	16	RW	0000h
28h	DMA Enable Register (SM0DMAEN)	16	RW	0000h
2Ah	Output Trigger Control Register (SM0TCTRL)	16	RW	0000h
2Ch	Fault Disable Mapping Register 0 (SM0DISMAP0)	16	RW	FFFFh
30h	Deadtime Count Register 0 (SM0DTCNT0)	16	RW	07FFh
32h	Deadtime Count Register 1 (SM0DTCNT1)	16	RW	07FFh
3Ch	Capture Control X Register (SM0CAPTCTRLX)	16	RW	0000h
3Eh	Capture Compare X Register (SM0CAPTCOMPX)	16	RW	0000h
40h	Capture Value 0 Register (SM0CVAL0)	16	R	0000h
42h	Capture Value 0 Cycle Register (SM0CVAL0CYC)	16	R	0000h
44h	Capture Value 1 Register (SM0CVAL1)	16	R	0000h
46h	Capture Value 1 Cycle Register (SM0CVAL1CYC)	16	R	0000h
5Eh	Capture PWM_X Input Filter Register (SM0CAPTFILTX)	16	RW	0000h
60h	Counter Register (SM1CNT)	16	R	0000h
62h	Initial Count Register (SM1INIT)	16	RW	0000h
64h	Control 2 Register (SM1CTRL2)	16	RW	0000h
66h	Control Register (SM1CTRL)	16	RW	0400h
6Ah	Value Register 0 (SM1VAL0)	16	RW	0000h
6Eh	Value Register 1 (SM1VAL1)	16	RW	0000h
72h	Value Register 2 (SM1VAL2)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
76h	Value Register 3 (SM1VAL3)	16	RW	0000h
7Ah	Value Register 4 (SM1VAL4)	16	RW	0000h
7Eh	Value Register 5 (SM1VAL5)	16	RW	0000h
82h	Output Control Register (SM1OCTRL)	16	RW	0000h
84h	Status Register (SM1STS)	16	RW	0000h
86h	Interrupt Enable Register (SM1INTEN)	16	RW	0000h
88h	DMA Enable Register (SM1DMAEN)	16	RW	0000h
8Ah	Output Trigger Control Register (SM1TCTRL)	16	RW	0000h
8Ch	Fault Disable Mapping Register 0 (SM1DISMAP0)	16	RW	FFFFh
90h	Deadtime Count Register 0 (SM1DTCNT0)	16	RW	07FFh
92h	Deadtime Count Register 1 (SM1DTCNT1)	16	RW	07FFh
9Ch	Capture Control X Register (SM1CAPTCTRLX)	16	RW	0000h
9Eh	Capture Compare X Register (SM1CAPTCOMPX)	16	RW	0000h
A0h	Capture Value 0 Register (SM1CVAL0)	16	R	0000h
A2h	Capture Value 0 Cycle Register (SM1CVAL0CYC)	16	R	0000h
A4h	Capture Value 1 Register (SM1CVAL1)	16	R	0000h
A6h	Capture Value 1 Cycle Register (SM1CVAL1CYC)	16	R	0000h
B8h	Phase Delay Register (SM1PHASEDLY)	16	RW	0000h
BEh	Capture PWM_X Input Filter Register (SM1CAPTFILTX)	16	RW	0000h
C0h	Counter Register (SM2CNT)	16	R	0000h
C2h	Initial Count Register (SM2INIT)	16	RW	0000h
C4h	Control 2 Register (SM2CTRL2)	16	RW	0000h
C6h	Control Register (SM2CTRL)	16	RW	0400h
CAh	Value Register 0 (SM2VAL0)	16	RW	0000h
CEh	Value Register 1 (SM2VAL1)	16	RW	0000h
D2h	Value Register 2 (SM2VAL2)	16	RW	0000h
D6h	Value Register 3 (SM2VAL3)	16	RW	0000h
DAh	Value Register 4 (SM2VAL4)	16	RW	0000h
DEh	Value Register 5 (SM2VAL5)	16	RW	0000h
E2h	Output Control Register (SM2OCTRL)	16	RW	0000h
E4h	Status Register (SM2STS)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E6h	Interrupt Enable Register (SM2INTEN)	16	RW	0000h
E8h	DMA Enable Register (SM2DMAEN)	16	RW	0000h
EAh	Output Trigger Control Register (SM2TCTRL)	16	RW	0000h
ECh	Fault Disable Mapping Register 0 (SM2DISMAP0)	16	RW	FFFFh
F0h	Deadtime Count Register 0 (SM2DTCNT0)	16	RW	07FFh
F2h	Deadtime Count Register 1 (SM2DTCNT1)	16	RW	07FFh
FCh	Capture Control X Register (SM2CAPTCTRLX)	16	RW	0000h
FEh	Capture Compare X Register (SM2CAPTCOMPX)	16	RW	0000h
100h	Capture Value 0 Register (SM2CVAL0)	16	R	0000h
102h	Capture Value 0 Cycle Register (SM2CVAL0CYC)	16	R	0000h
104h	Capture Value 1 Register (SM2CVAL1)	16	R	0000h
106h	Capture Value 1 Cycle Register (SM2CVAL1CYC)	16	R	0000h
118h	Phase Delay Register (SM2PHASEDLY)	16	RW	0000h
11Eh	Capture PWM_X Input Filter Register (SM2CAPTFILTX)	16	RW	0000h
180h	Output Enable Register (OUTEN)	16	RW	0000h
182h	Mask Register (MASK)	16	RW	0000h
184h	Software Controlled Output Register (SWCOUT)	16	RW	0000h
186h	PWM Source Select Register (DTSRCSEL)	16	RW	0000h
188h	Master Control Register (MCTRL)	16	RW	0000h
18Ah	Master Control 2 Register (MCTRL2)	16	RW	0000h
18Ch	Fault Control Register (FCTRL0)	16	RW	0000h
18Eh	Fault Status Register (FSTS0)	16	RW	See section
190h	Fault Filter Register (FFILT0)	16	RW	0000h
192h	Fault Test Register (FTST0)	16	RW	0000h
194h	Fault Control 2 Register (FCTRL20)	16	RW	0000h

31.5.2 Counter Register (SM0CNT - SM2CNT)

Offset

Register	Offset
SM0CNT	0h

Table continues on the next page...

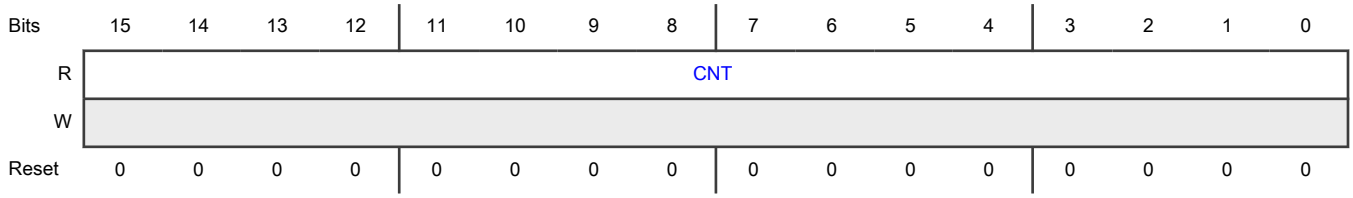
Table continued from the previous page...

Register	Offset
SM1CNT	60h
SM2CNT	C0h

Function

This field displays the state of the signed 16-bit submodule counter. This register is not byte accessible. Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CNT	Counter Register Bits

31.5.3 Initial Count Register (SM0INIT - SM2INIT)

Offset

Register	Offset
SM0INIT	2h
SM1INIT	62h
SM2INIT	C2h

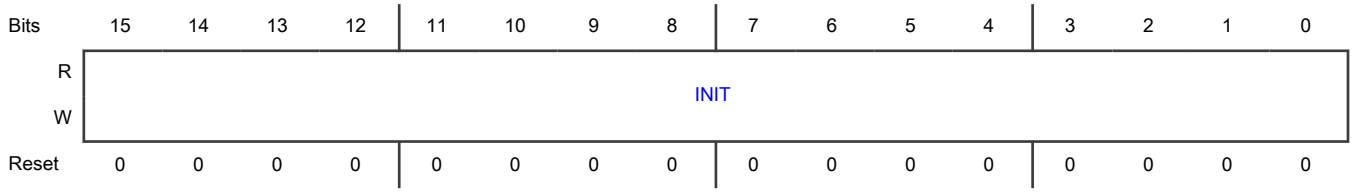
Function

The 16-bit signed value in this buffered register defines the initial count value for the PWM in PWM clock periods. This is the value loaded into the submodule counter when local sync, master sync, or master reload is asserted (based on the value of CTRL2[INIT_SEL]) or when CTRL2[FORCE] is asserted and force init is enabled. For PWM operation, the buffered contents of this register are loaded into the counter at the start of every PWM cycle. This register is not byte accessible.

NOTE

The INIT register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. This register cannot be written when MCTRL[LDOK] is set. Reading INIT reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 INIT	Initial Count Register Bits

31.5.4 Control 2 Register (SM0CTRL2 - SM2CTRL2)

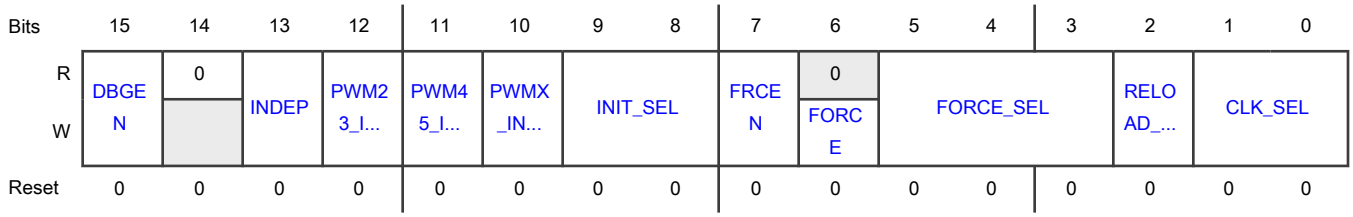
Offset

Register	Offset
SM0CTRL2	4h
SM1CTRL2	64h
SM2CTRL2	C4h

Function

Contains control fields for clock select and forcing output and initialization.This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15 DBGEN	Debug Enable When set to one, the PWM continues to run while the chip is in Debug mode. If the device enters Debug mode and this bit is zero, then the PWM outputs are disabled until Debug mode is exited. At that point, the PWM pins resume operation as programmed in the PWM registers.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 —	Reserved
13 INDEP	Independent or Complementary Pair Operation This bit determines if the PWM_A and PWM_B channels are independent PWMs or a complementary PWM pair. 0b - PWM_A and PWM_B form a complementary PWM pair. 1b - PWM_A and PWM_B outputs are independent PWMs.
12 PWM23_INIT	PWM23 Initial Value This bit determines the initial value for PWM23 and the value to which it is forced when FORCE_INIT is asserted.
11 PWM45_INIT	PWM45 Initial Value This bit determines the initial value for PWM45 and the value to which it is forced when FORCE_INIT is asserted.
10 PWMX_INIT	PWM_X Initial Value This bit determines the initial value for PWM_X and the value to which it is forced when FORCE_INIT is asserted.
9-8 INIT_SEL	Initialization Control Select These bits control the source of the INIT signal which goes to the counter. 00b - Local sync (PWM_X) causes initialization. 01b - Master reload from submodule 0 causes initialization. This setting should not be used in submodule 0 as it forces the INIT signal to logic 0. The submodule counter will only re-initialize when a master reload occurs. 10b - Master sync from submodule 0 causes initialization. This setting should not be used in submodule 0 as it forces the INIT signal to logic 0. 11b - EXT_SYNC causes initialization.
7 FRCEN	Force Enable This bit allows the CTRL2[FORCE] signal to initialize the counter without regard to the signal selected by CTRL2[INIT_SEL]. This is a software controlled initialization. A forced initialization will also assert the register reload if MCTRL[LDOK] is set. 0b - Initialization from a FORCE_OUT is disabled. 1b - Initialization from a FORCE_OUT is enabled.
6 FORCE	Force Initialization If CTRL2[FORCE_SEL] is set to 000, writing a 1 to this bit results in a FORCE_OUT event. This causes the following actions to be taken:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The PWM_A and PWM_B output pins assume values based on DTSRCSEL[SMxSEL23] and DTSRCSEL[SMxSEL45]. • If CTRL2[FRCEN] is set, the counter value is initialized with the INIT register value only when the submodule MCTRL[RUN] = 1 or CTRL2[CLK_SEL] = 2.
5-3 FORCE_SEL	Force Select This bit determines the source of the FORCE OUTPUT signal for this submodule. 000b - The local force signal, CTRL2[FORCE], from this submodule is used to force updates. 001b - The master force signal from submodule 0 is used to force updates. This setting should not be used in submodule 0 as it holds the FORCE OUTPUT signal to logic 0. 010b - The local reload signal from this submodule is used to force updates without regard to the state of LDOK. 011b - The master reload signal from submodule0 is used to force updates if LDOK is set. This setting should not be used in submodule0 as it holds the FORCE OUTPUT signal to logic 0. 100b - The local sync signal from this submodule is used to force updates. 101b - The master sync signal from submodule0 is used to force updates. This setting should not be used in submodule0 as it holds the FORCE OUTPUT signal to logic 0. 110b - The external force signal, EXT_FORCE, from outside the PWM module causes updates. 111b - The external sync signal, EXT_SYNC, from outside the PWM module causes updates.
2 RELOAD_SEL	Reload Source Select This bit determines the source of the RELOAD signal for this submodule. When this bit is set, MCTRL[LDOK[0]] for submodule 0 should be used since the local MCTRL[LDOK] will be ignored. 0b - The local RELOAD signal is used to reload registers. 1b - The master RELOAD signal (from submodule 0) is used to reload registers. This setting should not be used in submodule 0 as it forces the RELOAD signal to logic 0.
1-0 CLK_SEL	Clock Source Select These bits determine the source of the clock signal for this submodule. 00b - The IPBus clock is used as the clock for the local prescaler and counter. 01b - EXT_CLK is used as the clock for the local prescaler and counter. 10b - Submodule 0’s clock (AUX_CLK) is used as the source clock for the local prescaler and counter. This setting should not be used in submodule 0 as it forces the clock to logic 0. 11b - Reserved

31.5.5 Control Register (SM0CTRL - SM2CTRL)

Offset

Register	Offset
SM0CTRL	6h
SM1CTRL	66h
SM2CTRL	C6h

Function

Includes control settings for timing, loading, and buffering.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LDFQ				HALF	FULL	DT		COMP MODE	PRSC			SPLIT	LDMOD	DBLX	DBLEN
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 LDFQ	Load Frequency These buffered bits select the PWM load frequency. Reset clears LDFQ, selecting loading every PWM opportunity. A PWM opportunity is determined by HALF and FULL. The register bits are write-protected by MCTRL2[WRPROT] bits. **NOTE** LDFQ takes effect when the current load cycle is complete, regardless of the state of MCTRL[LDOK]. Reading LDFQ reads the buffered values and not necessarily the values currently in effect. 0000b - Every PWM opportunity 0001b - Every 2 PWM opportunities 0010b - Every 3 PWM opportunities 0011b - Every 4 PWM opportunities 0100b - Every 5 PWM opportunities 0101b - Every 6 PWM opportunities 0110b - Every 7 PWM opportunities 0111b - Every 8 PWM opportunities 1000b - Every 9 PWM opportunities

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1001b - Every 10 PWM opportunities 1010b - Every 11 PWM opportunities 1011b - Every 12 PWM opportunities 1100b - Every 13 PWM opportunities 1101b - Every 14 PWM opportunities 1110b - Every 15 PWM opportunities 1111b - Every 16 PWM opportunities
11 HALF	Half Cycle Reload This bit enables half-cycle reloads. A half cycle is defined by when the submodule counter matches the VAL0 register and does not have to be halfway through the PWM cycle. This bit is write-protected by MCTRL2[WRPROT] bits. 0b - Half-cycle reloads disabled. 1b - Half-cycle reloads enabled.
10 FULL	Full Cycle Reload This bit enables full-cycle reloads. A full cycle is defined by when the submodule counter matches the VAL1 register. Either CTRL[HALF] or CTRL[FULL] must be set to move the buffered data into the registers used by the PWM generators or CTRL[LDMOD] must be set. If both CTRL[HALF] and CTRL[FULL] are set, then reloads can occur twice per cycle. This bit is write-protected by MCTRL2[WRPROT] bits. 0b - Full-cycle reloads disabled. 1b - Full-cycle reloads enabled.
9-8 DT	Deadtime These bits reflect the sampled values of the PWM_X input at the end of each deadtime. Sampling occurs at the end of deadtime 0 for DT[0] and the end of deadtime 1 for DT[1]. Reset clears these bits. The register bits are write-protected by MCTRL2[WRPROT] bits.
7 COMPMODE	Compare Mode This bit controls how comparisons are made between the VAL* registers and the PWM submodule counter. This bit can be written one time after which it requires a reset to release the bit for writing again. 0b - The VAL* registers and the PWM counter are compared using an "equal to" method. This means that PWM edges are only produced when the counter is equal to one of the VAL* register values. This implies that a PWM_A output that is high at the end of a period maintains this state until a match with VAL3 clears the output in the following period. 1b - The VAL* registers and the PWM counter are compared using an "equal to or greater than" method. This means that PWM edges are produced when the counter is equal to or greater than one of the VAL* register values. This implies that a PWM_A output that is high at the end of a period could go low at the start of the next period if the starting counter value is greater than (but not necessarily equal to) the new VAL3 value.
6-4	Prescaler

Table continues on the next page...

Table continued from the previous page...

Field	Function
PRSC	These buffered bits select the divide ratio of the PWM clock frequency selected by CTRL2[CLK_SEL]. **NOTE** Reading CTRL[PRSC] reads the buffered values and not necessarily the values currently in effect. CTRL[PRSC] takes effect at the beginning of the next PWM cycle and only when the load okay bit MCTRL[LDOK] is set, or CTRL[LDMOD] is set. This field cannot be written when MCTRL[LDOK] is set. 000b - Prescaler 1. PWM clock frequency = f_{clk} 001b - Prescaler 2. PWM clock frequency = f_{clk} /2 010b - Prescaler 4. PWM clock frequency = f_{clk} /4 011b - Prescaler 8. PWM clock frequency = f_{clk} /8 100b - Prescaler 16. PWM clock frequency = f_{clk} /16 101b - Prescaler 32. PWM clock frequency = f_{clk} /32 110b - Prescaler 64. PWM clock frequency = f_{clk} /64 111b - Prescaler 128. PWM clock frequency = f_{clk} /128
3 SPLIT	Split the DBLPWM signal to PWM_A and PWM_B This bit is used in independent mode when DBLEN is set. This bit allows the two PWM pulses generated by DBLEN to be split with one pulse on PWM_A and one on PWM_B. The two pulses within the same PWM period are created by an XOR function of the PWM_A and PWM_B sources. The splitting function causes PWM_A to output the pulse that occurs when the PWM_A source is 1 and the PWM_B source is 0. The PWM_B output occurs when the PWM_B source is 1 and the PWM_A source is 0. (See Double switching PWMs .) This bit is write-protected by MCTRL2[WRPROT] bits. 0b - DBLPWM is not split. PWM_A and PWM_B each have double pulses. 1b - DBLPWM is split to PWM_A and PWM_B.
2 LDMOD	Load Mode Select This bit selects the timing of loading the buffered registers for this submodule. This bit is write-protected by MCTRL2[WRPROT] bits. 0b - Buffered registers of this submodule are loaded and take effect at the next PWM reload if MCTRL[LDOK] is set. 1b - Buffered registers of this submodule are loaded and take effect immediately upon MCTRL[LDOK] being set. In this case, it is not necessary to set CTRL[FULL] or CTRL[HALF].
1 DBLX	PWM_X Double Switching Enable This bit enables the double switching behavior on PWM_X. When this bit is set, the PWM_X output shall be the exclusive OR combination of PWM_A and PWM_B prior to polarity and masking considerations. This bit is write-protected by MCTRL2[WRPROT] bits. 0b - PWM_X double pulse disabled. 1b - PWM_X double pulse enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 DBLEN	Double Switching Enable This bit enables the double switching PWM behavior (See Double switching PWMs). This bit is write-protected by MCTRL2[WRPROT] bits. 0b - Double switching disabled. 1b - Double switching enabled.

31.5.6 Value Register 0 (SM0VAL0 - SM2VAL0)

Offset

Register	Offset
SM0VAL0	Ah
SM1VAL0	6Ah
SM2VAL0	CAh

Function

NOTE

The VAL0 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL0 cannot be written when MCTRL[LDOK] is set. Reading VAL0 reads the value in a buffer. It is not necessarily the value that the PWM generator is currently using.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VAL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 VAL0	Value 0 The 16-bit signed value in this buffered register defines the mid-cycle reload point for the PWM in PWM clock periods. This value also defines when the PWM_X signal is set and the local sync signal is reset. This register is not byte accessible. **NOTE** The actual behavior takes effect when counter equal VAL0+1.

31.5.7 Value Register 1 (SM0VAL1 - SM2VAL1)

Offset

Register	Offset
SM0VAL1	Eh
SM1VAL1	6Eh
SM2VAL1	CEh

Function

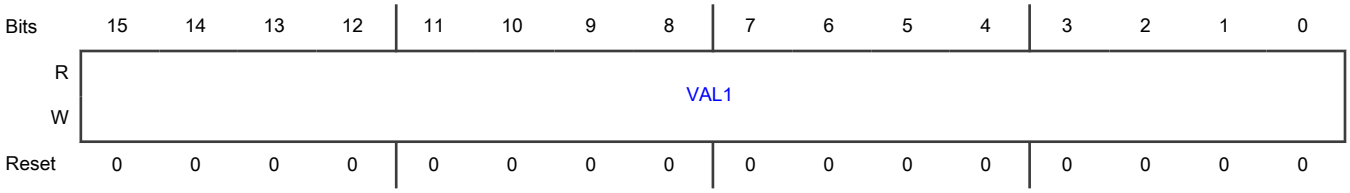
NOTE

The VAL1 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL1 cannot be written when MCTRL[LDOK] is set. Reading VAL1 reads the value in a buffer. It is not necessarily the value that the PWM generator is currently using.

When using FRACVAL1, limit the maximum value of VAL1 to 0xFFFE for unsigned applications or to 0x7FFE for signed applications, to avoid counter rollovers caused by accumulating the fractional period defined by FRACVAL1.

If the VAL1 register defines the timer period (Local Sync is selected as the counter initialization signal), a 100% duty cycle cannot be achieved on the PWM_X output. After the count reaches VAL1, the PWM_X output is low for a minimum of one count every cycle. When the Master Sync signal (only originated by the Local Sync from submodule 0) is used to control the timer period, the VAL1 register can be free for other functions such as PWM generation without the duty cycle limitation.

Diagram



Fields

Field	Function
15-0 VAL1	Value 1 The 16-bit signed value written to this buffered register defines the modulo count value (maximum count) for the submodule counter. When reaching this count value, the counter reloads itself with the contents of the INIT register and asserts the local sync signal while resetting PWM_X. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL1+1.

31.5.8 Value Register 2 (SM0VAL2 - SM2VAL2)

Offset

Register	Offset
SM0VAL2	12h
SM1VAL2	72h
SM2VAL2	D2h

Function

NOTE

The VAL2 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL2 cannot be written when MCTRL[LDOK] is set. Reading VAL2 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VAL2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 VAL2	Value 2 The 16-bit signed value in this buffered register defines the count value to set PWM23 high. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL2+1.

31.5.9 Value Register 3 (SM0VAL3 - SM2VAL3)

Offset

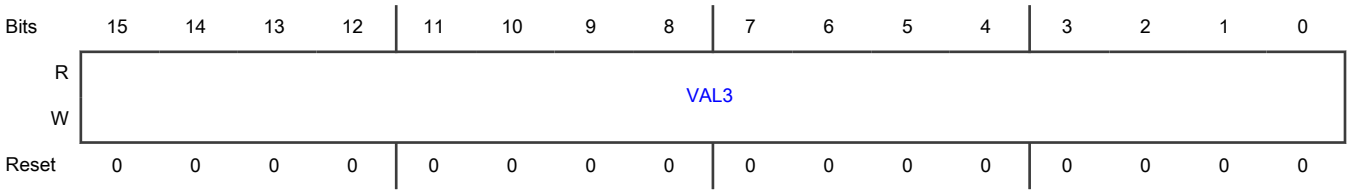
Register	Offset
SM0VAL3	16h
SM1VAL3	76h
SM2VAL3	D6h

Function

NOTE

The VAL3 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL3 cannot be written when MCTRL[LDOK] is set. Reading VAL3 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL3	Value 3 The 16-bit signed value in this buffered register defines the count value to set PWM23 low. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL3+1.

31.5.10 Value Register 4 (SM0VAL4 - SM2VAL4)

Offset

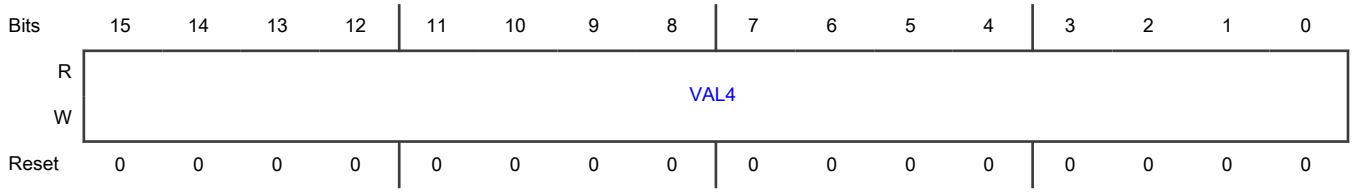
Register	Offset
SM0VAL4	1Ah
SM1VAL4	7Ah
SM2VAL4	DAh

Function

NOTE

The VAL4 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL4 cannot be written when MCTRL[LDOK] is set. Reading VAL4 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL4	Value 4 The 16-bit signed value in this buffered register defines the count value to set PWM45 high. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL4+1.

31.5.11 Value Register 5 (SM0VAL5 - SM2VAL5)

Offset

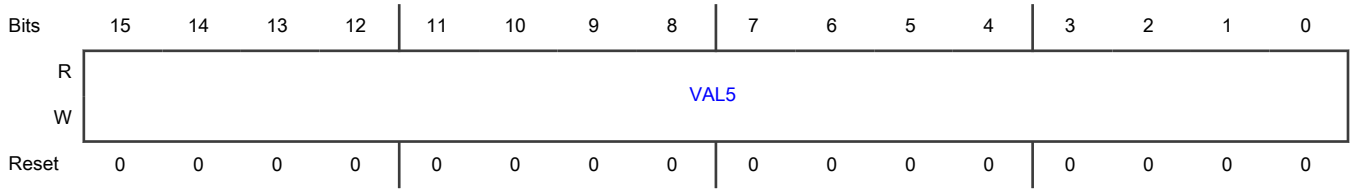
Register	Offset
SM0VAL5	1Eh
SM1VAL5	7Eh
SM2VAL5	DEh

Function

NOTE

The VAL5 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL5 cannot be written when MCTRL[LDOK] is set. Reading VAL5 reads the value in a buffer and not necessarily the value that the PWM generator is currently using.

Diagram



Fields

Field	Function
15-0 VAL5	Value 5 The 16-bit signed value in this buffered register defines the count value to set PWM45 low. This register is not byte accessible. **NOTE** The actual behavior takes effect when the counter equals VAL5+1.

31.5.12 Output Control Register (SM0OCTRL - SM2OCTRL)

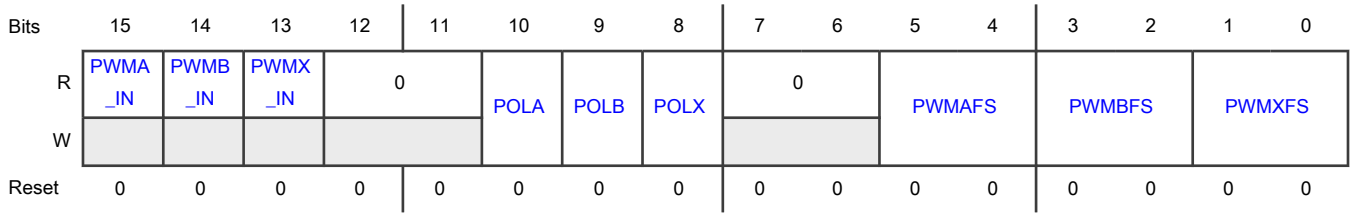
Offset

Register	Offset
SM0OCTRL	22h
SM1OCTRL	82h
SM2OCTRL	E2h

Function

Contains output controls for fault states. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15 PWMA_IN	PWM_A Input This bit shows the logic value currently being driven into the PWM_A input. The reset state is undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 PWMB_IN	PWM_B Input This bit shows the logic value currently being driven into the PWM_B input. The reset state is undefined.
13 PWMX_IN	PWM_X Input This bit shows the logic value currently being driven into the PWM_X input. The reset state is undefined.
12-11 —	Reserved
10 POLA	PWM_A Output Polarity This bit inverts the PWM_A output polarity. 0b - PWM_A output not inverted. A high level on the PWM_A pin represents the "on" or "active" state. 1b - PWM_A output inverted. A low level on the PWM_A pin represents the "on" or "active" state.
9 POLB	PWM_B Output Polarity This bit inverts the PWM_B output polarity. 0b - PWM_B output not inverted. A high level on the PWM_B pin represents the "on" or "active" state. 1b - PWM_B output inverted. A low level on the PWM_B pin represents the "on" or "active" state.
8 POLX	PWM_X Output Polarity This bit inverts the PWM_X output polarity. 0b - PWM_X output not inverted. A high level on the PWM_X pin represents the "on" or "active" state. 1b - PWM_X output inverted. A low level on the PWM_X pin represents the "on" or "active" state.
7-6 —	Reserved
5-4 PWMAFS	PWM_A Fault State These bits determine the fault state for the PWM_A output during fault conditions and Deep Sleep mode. It may also define the output state during Debug mode depending on the setting of CTRL2[DBGEN]. 00b - Output is forced to logic 0 state prior to consideration of output polarity control. 01b - Output is forced to logic 1 state prior to consideration of output polarity control. 10b,11b - Output is put in a high-impedance state.
3-2 PWMBFS	PWM_B Fault State These bits determine the fault state for the PWM_B output during fault conditions and Deep Sleep mode. It may also define the output state during Debug mode depending on the setting of CTRL2[DBGEN].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Output is forced to logic 0 state prior to consideration of output polarity control. 01b - Output is forced to logic 1 state prior to consideration of output polarity control. 10b,11b - Output is put in a high-impedance state.
1-0 PWMXFS	PWM_X Fault State These bits determine the fault state for the PWM_X output during fault conditions and Deep Sleep mode. It may also define the output state during Debug mode depending on the setting of CTRL2[DBGEN]. 00b - Output is forced to logic 0 state prior to consideration of output polarity control. 01b - Output is forced to logic 1 state prior to consideration of output polarity control. 10b,11b - Output is put in a high-impedance state.

31.5.13 Status Register (SM0STS - SM2STS)

Offset

Register	Offset
SM0STS	24h
SM1STS	84h
SM2STS	E4h

Function

Contains Compare and Capture flag status.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RUF	REF	RF	0				CFX1	CFX0	CMPF					
W			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
15 —	Reserved
14 RUF	Registers Updated Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit is set when one of the INIT, VALx, FRACVALx, or CTRL[PRSC] registers has been written, which indicates potentially non-coherent data in the set of double buffered registers. Clear this bit by a proper reload sequence consisting of a reload signal while MCTRL[LDOK] = 1. Reset clears this bit. 0b - No register update has occurred since last reload. 1b - At least one of the double buffered registers has been updated since the last reload.
13 REF	Reload Error Flag This bit is set when a reload cycle occurs while MCTRL[LDOK] is 0 and the double buffered registers are in a non-coherent state (STS[RUF] = 1). Clear this bit by writing a logic one to this location. Reset clears this bit. 0b - No reload error occurred. 1b - Reload signal occurred with non-coherent data and MCTRL[LDOK] = 0.
12 RF	Reload Flag This bit is set at the beginning of every reload cycle regardless of the state of MCTRL[LDOK]. Clear this bit by writing a logic one to this location when DMAEN[VALDE] is clear (non-DMA mode). This bit can also be cleared by the DMA done signal when DMAEN[VALDE] is set (DMA mode) . Reset clears this bit. 0b - No new reload cycle since last STS[RF] clearing 1b - New reload cycle since last STS[RF] clearing
11-8 —	Reserved
7 CFX1	Capture Flag X1 This bit is set when a capture event occurs on the Capture X1 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CX1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CX1DE] is set (DMA mode) . Reset clears this bit.
6 CFX0	Capture Flag X0 This bit is set when a capture event occurs on the Capture X0 circuit . This bit is cleared by writing a one to this bit position if DMAEN[CX0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CX0DE] is set (DMA mode) . Reset clears this bit.
5-0 CMPF	Compare Flags These bits are set when the submodule counter value matches the value of one of the VALx registers. Clear these bits by writing a 1 to a bit position. 00_0000b - No compare event has occurred for a particular VALx value. 00_0001b - A compare event has occurred for a particular VALx value.

31.5.14 Interrupt Enable Register (SM0INTEN - SM2INTEN)

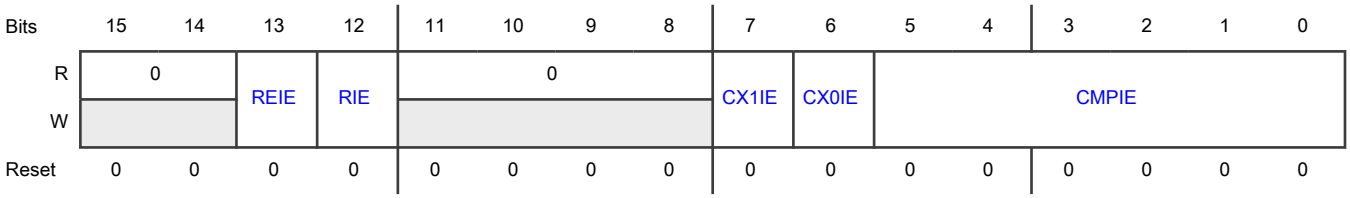
Offset

Register	Offset
SM0INTEN	26h
SM1INTEN	86h
SM2INTEN	E6h

Function

Contains Compare and Capture interrupt enables.

Diagram



Fields

Field	Function
15-14 —	Reserved
13 REIE	Reload Error Interrupt Enable This bit enables the reload error flag, STS[REF], to generate CPU interrupt requests. Reset clears this bit. 0b - STS[REF] CPU interrupt requests disabled 1b - STS[REF] CPU interrupt requests enabled
12 RIE	Reload Interrupt Enable This bit enables the reload flag, STS[RF], to generate CPU interrupt requests. Reset clears this bit. 0b - STS[RF] CPU interrupt requests disabled 1b - STS[RF] CPU interrupt requests enabled
11-8 —	Reserved
7 CX1IE	Capture X 1 Interrupt Enable This bit allows the STS[CFX1] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CX1DE].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt request disabled for STS[CFX1]. 1b - Interrupt request enabled for STS[CFX1].
6 CX0IE	Capture X 0 Interrupt Enable This bit allows the STS[CFX0] flag to create an interrupt request to the CPU. Do not set this bit and DMAEN[CX0DE]. 0b - Interrupt request disabled for STS[CFX0]. 1b - Interrupt request enabled for STS[CFX0].
5-0 CMPIE	Compare Interrupt Enables These bits enable the STS[CMPF] flags to cause a compare interrupt request to the CPU. 00_0000b - The corresponding STS[CMPF] bit will not cause an interrupt request. 00_0001b - The corresponding STS[CMPF] bit will cause an interrupt request.

31.5.15 DMA Enable Register (SM0DMAEN - SM2DMAEN)

Offset

Register	Offset
SM0DMAEN	28h
SM1DMAEN	88h
SM2DMAEN	E8h

Function

Contains controls for DMA. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						VALDE	FAND	CAPTDE		0				CX1DE	CX0DE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 VALDE	Value Registers DMA Enable This bit enables DMA write requests for the VALx and FRACVALx registers when STS[RF] is set. Reset clears this bit. 0b - DMA write requests disabled 1b - Enabled. DMA write requests for the VALx and FRACVALx registers enabled
8 FAND	FIFO Watermark AND Control This bit works in conjunction with the DMAEN[CAPTDE] field when it is set to watermark mode (DMAEN[CAPTDE] = 01). While DMAEN[CAxDE], DMAEN[CBxDE], and DMAEN[CXxDE] determine which FIFO watermarks the DMA read request is sensitive to. This bit determines if the selected watermarks are AND'ed together or OR'ed together to create the request. 0b - Selected FIFO watermarks are OR'ed together. 1b - Selected FIFO watermarks are AND'ed together.
7-6 CAPTDE	Capture DMA Enable Source Select These bits select the source of enabling the DMA read requests for the capture FIFOs. Reset clears these bits. 00b - Read DMA requests disabled. 01b - Exceeding a FIFO watermark sets the DMA read request. This requires at least one of DMAEN[CA1DE], DMAEN[CA0DE], DMAEN[CB1DE], DMAEN[CB0DE], DMAEN[CX1DE], or DMAEN[CX0DE] to be set to determine which watermark(s) the DMA request is sensitive. 10b - A local synchronization (VAL1 matches counter) sets the read DMA request. 11b - A local reload (STS[RF] being set) sets the read DMA request.
5-2 —	Reserved
1 CX1DE	Capture X1 FIFO DMA Enable This bit enables DMA read requests for the Capture X1 FIFO data when STS[CFX1] is set. Reset clears this bit. Do not set both this bit and INTEN[CX1IE].
0 CX0DE	Capture X0 FIFO DMA Enable This bit enables DMA read requests for the Capture X0 FIFO data when STS[CFX0] is set. Reset clears this bit. Do not set both this bit and INTEN[CX0IE].

31.5.16 Output Trigger Control Register (SM0TCTRL - SM2TCTRL)

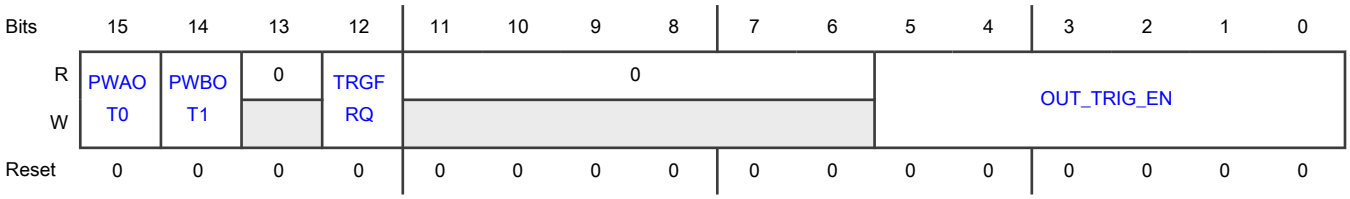
Offset

Register	Offset
SM0TCTRL	2Ah
SM1TCTRL	8Ah
SM2TCTRL	EAh

Function

Contains trigger controls.This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15 PWAOT0	Mux Output Trigger 0 Source Select This bit selects which signal to bring out on the PWM's PWM_MUX_TRIG0 port. 0b - Route the PWM_OUT_TRIG0 signal to PWM_MUX_TRIG0 port. 1b - Route the PWM_A output to the PWM_MUX_TRIG0 port.
14 PWBOT1	Mux Output Trigger 1 Source Select This bit selects which signal to bring out on the PWM's PWM_MUX_TRIG1 port. 0b - Route the PWM_OUT_TRIG1 signal to PWM_MUX_TRIG1 port. 1b - Route the PWM_B output to the PWM_MUX_TRIG1 port.
13 —	Reserved
12 TRGFRQ	Trigger Frequency This bit allows control over the frequency of the trigger outputs when using non-zero values of CTRL[LDFQ]. 0b - Trigger outputs are generated during every PWM period even if the PWM is not reloaded every period due to CTRL[LDFQ] being non-zero. 1b - Trigger outputs are generated only during the final PWM period prior to a reload opportunity when the PWM is not reloaded every period due to CTRL[LDFQ] being non-zero.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-6 —	Reserved
5-0 OUT_TRIG_EN	Output Trigger Enables These bits enable the generation of PWM_OUT_TRIG0 and PWM_OUT_TRIG1 outputs based on the counter value matching the value in one or more of the VAL0-5 registers. **NOTE** Due to delays in creating the PWM outputs, the output trigger signals lead the PWM output edges by 2-3 clock cycles depending on the fractional cycle value being used. 1x_xxxxb - PWM_OUT_TRIG1 will set when the counter value matches the VAL5 value. x1_xxxxb - PWM_OUT_TRIG0 will set when the counter value matches the VAL4 value. xx_1xxxb - PWM_OUT_TRIG1 will set when the counter value matches the VAL3 value. xx_x1xxb - PWM_OUT_TRIG0 will set when the counter value matches the VAL2 value. xx_xx1xb - PWM_OUT_TRIG1 will set when the counter value matches the VAL1 value. xx_xxx1b - PWM_OUT_TRIG0 will set when the counter value matches the VAL0 value.

31.5.17 Fault Disable Mapping Register 0 (SM0DISMAP0 - SM2DISMAP0)

Offset

Register	Offset
SM0DISMAP0	2Ch
SM1DISMAP0	8Ch
SM2DISMAP0	ECh

Function

This register determines which PWM pins are disabled by the fault protection inputs. Reset sets all of the bits in the fault disable mapping register. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	1				DIS0X				DIS0B				DIS0A			
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-12 —	Reserved
11-8 DIS0X	PWM_X Fault Disable Mask 0 Each of the four bits of this field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_X output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. DIS0X[0] is associated with FAULT0. DIS0X[1] is associated with FAULT1. DIS0X[2] is associated with FAULT2. DIS0X[3] is associated with FAULT3. A reset sets all bits in this field.
7-4 DIS0B	PWM_B Fault Disable Mask 0 Each of the four bits of this field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_B output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. DIS0B[0] is associated with FAULT0. DIS0B[1] is associated with FAULT1. DIS0B[2] is associated with FAULT2. DIS0B[3] is associated with FAULT3. A reset sets all bits in this field.
3-0 DIS0A	PWM_A Fault Disable Mask 0 Each of the four bits of this field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_A output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. DIS0A[0] is associated with FAULT0. DIS0A[1] is associated with FAULT1. DIS0A[2] is associated with FAULT2. DIS0A[3] is associated with FAULT3. A reset sets all bits in this field.

31.5.18 Deadtime Count Register 0 (SM0DTCNT0 - SM2DTCNT0)

Offset

Register	Offset
SM0DTCNT0	30h
SM1DTCNT0	90h
SM2DTCNT0	F0h

Function

Deadtime operation applies only to complementary channel operation. The values written to the DTCNTx registers are in terms of IPBus clock cycles regardless of the setting of CTRL[PRSC] and/or CTRL2[CLK_SEL]. Reset sets the deadtime count registers to a default value of 0x07FF, selecting a deadtime of 2047 IPBus clock cycles, when fractional delay is not enabled. The DTCNTx registers are not byte accessible.

Diagram

Fields

Field	Function
15-11 —	Reserved
10-0 DTCNT0	Deadtime Count Register 0 This field is used to control the deadtime during 0 to 1 transitions of the PWM_A output (assuming normal polarity).

31.5.19 Deadtime Count Register 1 (SM0DTCNT1 - SM2DTCNT1)

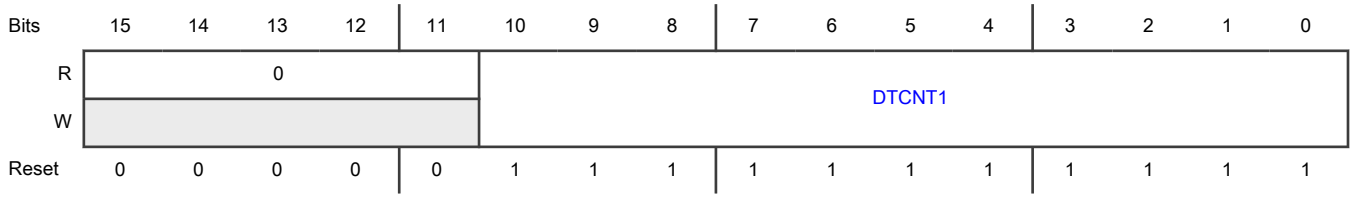
Offset

Register	Offset
SM0DTCNT1	32h
SM1DTCNT1	92h
SM2DTCNT1	F2h

Function

Deadtime operation applies only to complementary channel operation. The values written to the DTCNTx registers are in terms of IPBus clock cycles regardless of the setting of CTRL[PRSC] and/or CTRL2[CLK_SEL]. Reset sets the deadtime count registers to a default value of 0x07FF, selecting a deadtime of 2047 IPBus clock cycles, when fractional delay is not enabled. The DTCNTx registers are not byte accessible.

Diagram



Fields

Field	Function
15-11 —	Reserved
10-0 DTCNT1	Deadtime Count Register 1 This field is used to control the deadtime during 0 to 1 transitions of the complementary PWM_B output.

31.5.20 Capture Control X Register (SM0CAPTCTRLX)

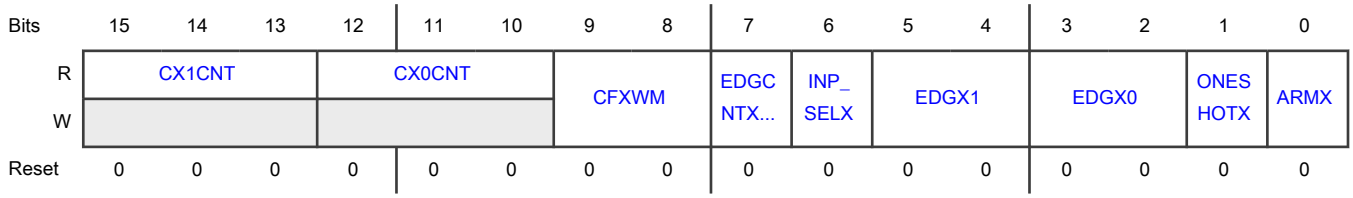
Offset

Register	Offset
SM0CAPTCTRLX	3Ch

Function

Contains capture controls for mode X. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2 EDGX0	Edge X 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

31.5.21 Capture Compare X Register (SM0CAPTCOMPX)

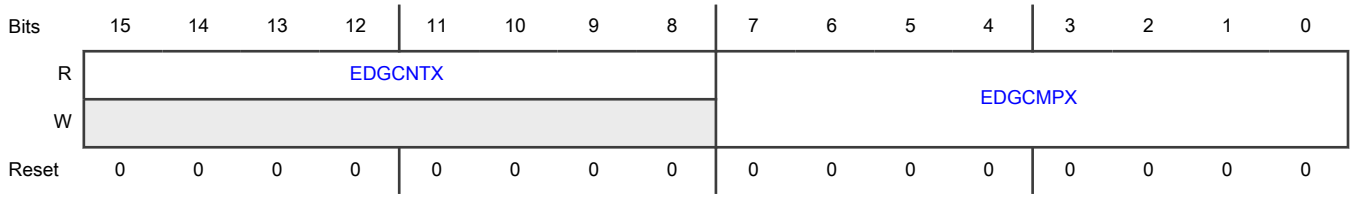
Offset

Register	Offset
SM0CAPTCOMPX	3Eh

Function

Contains capture and control values for mode X. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

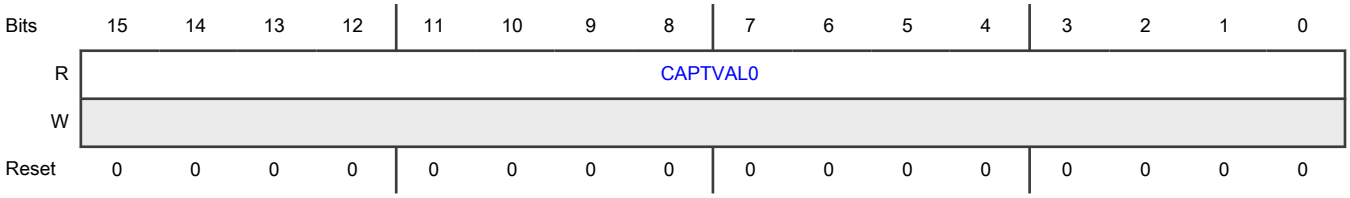
31.5.22 Capture Value 0 Register (SM0CVAL0)

Offset

Register	Offset
SM0CVAL0	40h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

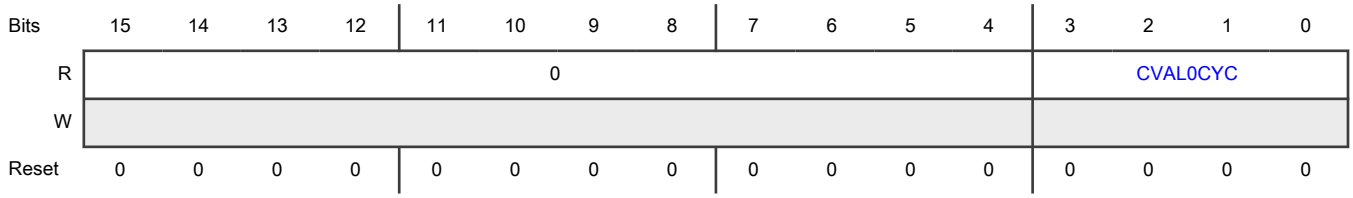
31.5.23 Capture Value 0 Cycle Register (SM0CVAL0CYC)

Offset

Register	Offset
SM0CVAL0CYC	42h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

31.5.24 Capture Value 1 Register (SM0CVAL1)

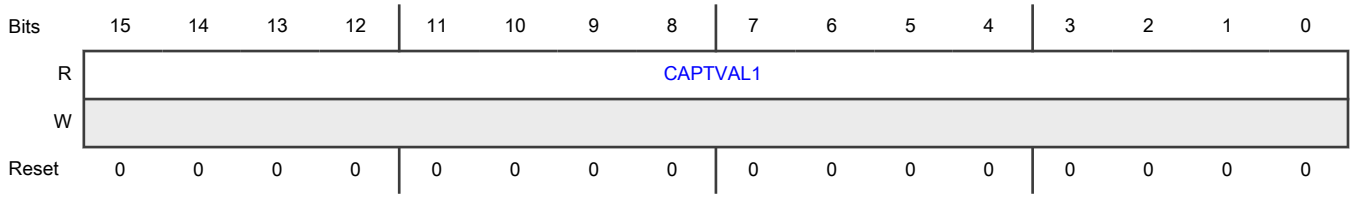
Offset

Register	Offset
SM0CVAL1	44h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

31.5.25 Capture Value 1 Cycle Register (SM0CVAL1CYC)

Offset

Register	Offset
SM0CVAL1CYC	46h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL1CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

31.5.26 Capture PWM_X Input Filter Register (SM0CAPTFILTX - SM2CAPTFILTX)

Offset

Register	Offset
SM0CAPTFILTX	5Eh
SM1CAPTFILTX	BEh
SM2CAPTFILTX	11Eh

Function

Input filter considerations include:

- The CAPTX_FILT_PER value should be set such that the sampling period is larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CAPTX_FILT_CNT value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the CAPTX_FILT_CNT+3 power.

- The values of FILT_PER and CAPTX_FILT_CNT must also be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting CAPTX_FILT_PER to a non-zero value) introduces a latency of ((CAPTX_FILT_CNT+4) x CAPTX_FILT_PER x IPBus clock period).

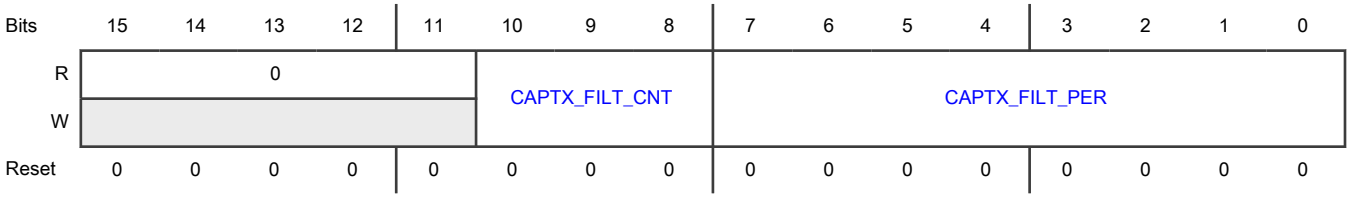
NOTE

When the filter is enabled, there is a combinational path to disable the PWM outputs to ensure rapid response to input capture conditions if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

This register is not byte accessible

This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-11 —	Reserved
10-8 CAPTX_FILT_CNT	Input Capture Filter Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bit field value of 0-7 represents 3-10 samples, respectively. The value of CAPTX_FILT_CNT affects the input latency.
7-0 CAPTX_FILT_PER	Input Capture Filter Period This field applies universally to all capture inputs. These bits represent the sampling period (in IPBus clock cycles) of the input capture pin input filter. Each input is sampled multiple times at the rate specified by this field. If CAPTX_FILT_PER is 0x00 (default), then the input filter is bypassed. The value of CAPTX_FILT_PER affects the input latency. **NOTE** When changing values for CAPTX_FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

31.5.27 Capture Control X Register (SM1CAPTCTRLX)

Offset

Register	Offset
SM1CAPTCTRLX	9Ch

Function

Contains capture controls for mode X. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CX1CNT			CX0CNT			CFXWM		EDGCNTX...	INP_SELX	EDGX1		EDGX0		ONESHOTX	ARMX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Capture rising edges 11b - Capture any edge
3-2 EDGX0	Edge X 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

31.5.28 Capture Compare X Register (SM1CAPTCOMPX)

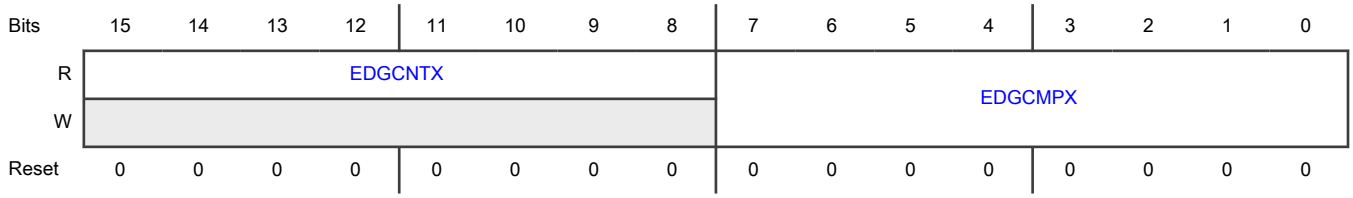
Offset

Register	Offset
SM1CAPTCOMPX	9Eh

Function

Contains capture and control values for mode X. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

31.5.29 Capture Value 0 Register (SM1CVAL0)

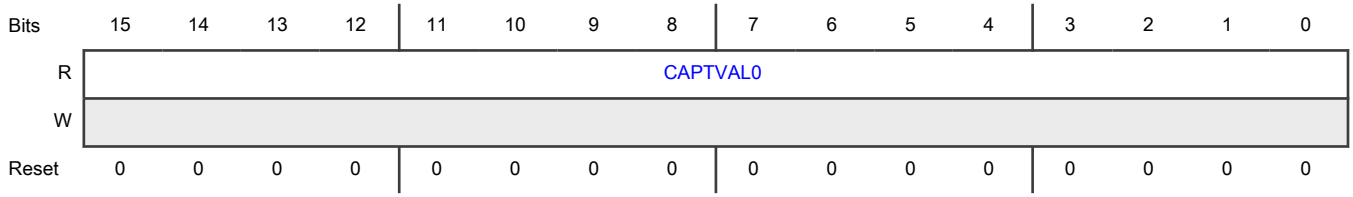
Offset

Register	Offset
SM1CVAL0	A0h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

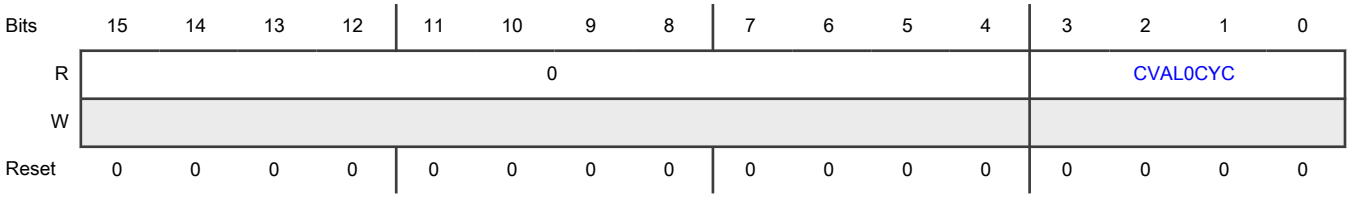
31.5.30 Capture Value 0 Cycle Register (SM1CVAL0CYC)

Offset

Register	Offset
SM1CVAL0CYC	A2h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

31.5.31 Capture Value 1 Register (SM1CVAL1)

Offset

Register	Offset
SM1CVAL1	A4h

Function
Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPTVAL1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

31.5.32 Capture Value 1 Cycle Register (SM1CVAL1CYC)

Offset

Register	Offset
SM1CVAL1CYC	A6h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL1CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

31.5.33 Phase Delay Register (SM1PHASEDLY - SM2PHASEDLY)

Offset

Register	Offset
SM1PHASEDLY	B8h
SM2PHASEDLY	118h

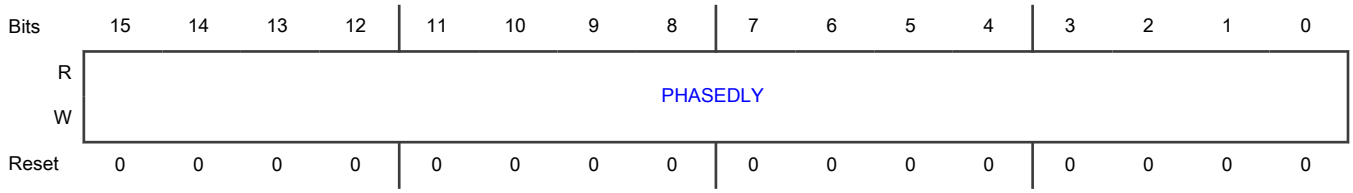
Function

The 16-bit unsigned value in this buffered register defines the delay from the master sync signal of submodule 0 to the time that this submodule recognizes the master sync in PWM clock periods. CTRL2[INIT_SEL] must be set to 10b to select the master sync signal as the source for initialization when using this register. Setting this register with a non-zero value and using the master sync signal as the initialization source, allows the output of this submodule to be a fixed number of cycles delayed from submodule 0. For PWM operation, the buffered contents of this register are updated at the start of every PWM cycle. This field is not byte accessible.

NOTE

The PHASEDLY register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. This register cannot be written when MCTRL[LDOK] is set. Reading PHASEDLY reads the value in a buffer and not necessarily the value that the PWM generator is currently using. Also, the value of this register should not be set to a value larger than the period defined in submodule 0.

Diagram



Fields

Field	Function
15-0 PHASEDLY	Initial Count Register Bits

31.5.34 Capture Control X Register (SM2CAPTCTRLX)

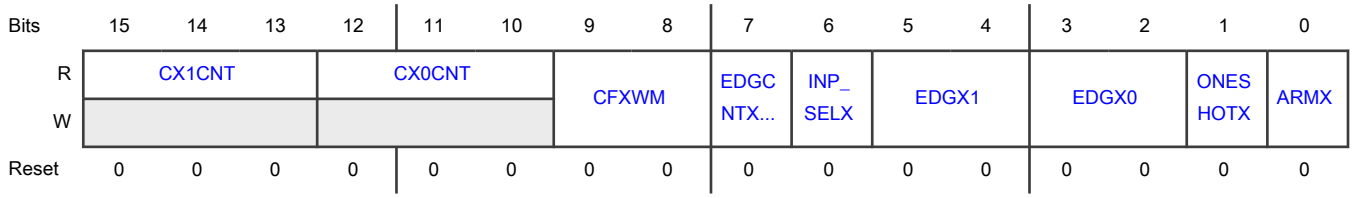
Offset

Register	Offset
SM2CAPTCTRLX	FCh

Function

Contains capture controls for mode X. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1.)
12-10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1.)
9-8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], will not be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1.)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0b - Edge counter disabled and held in reset 1b - Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0b - Raw PWM_X input signal selected as source. 1b - Edge Counter. Output of edge counter/compare selected as source. When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields to enable one or both of the capture registers.
5-4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
3-2	Edge X 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
EDGX0	These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00b - Disabled 01b - Capture falling edges 10b - Capture rising edges 11b - Capture any edge
1 ONESHOTX	One Shot Mode Aux This field selects between free running and one shot mode for the input capture circuitry. 0b - Free Running. Free running mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1b - One Shot. One shot mode is selected if both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed, and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed, and the ARMX bit is cleared. No further captures are performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture occurs on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0b - Input capture operation is disabled. 1b - Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

31.5.35 Capture Compare X Register (SM2CAPTCOMPX)

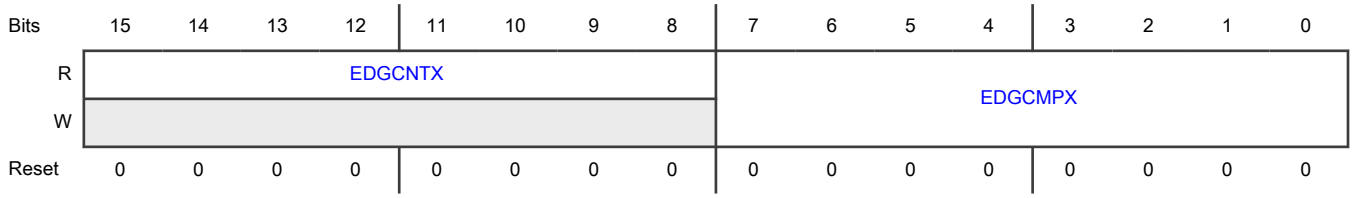
Offset

Register	Offset
SM2CAPTCOMPX	FEh

Function

Contains capture and control values for mode X. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-8 EDGCNTX	Edge Counter X This field contains the edge counter value for the PWM_X input capture circuitry.
7-0 EDGCMPX	Edge Compare X This field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

31.5.36 Capture Value 0 Register (SM2CVAL0)

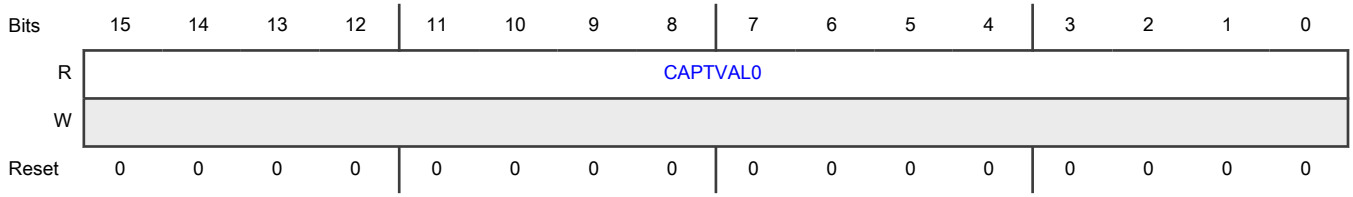
Offset

Register	Offset
SM2CVAL0	100h

Function

Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-0 CAPTVAL0	Capture Value 0 This field stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture increases the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This field is not byte accessible.

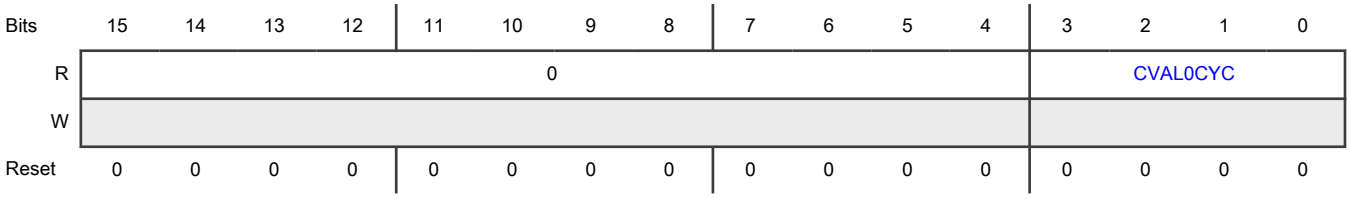
31.5.37 Capture Value 0 Cycle Register (SM2CVAL0CYC)

Offset

Register	Offset
SM2CVAL0CYC	102h

Function
Writing this register generates bus transfer error.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 CVAL0CYC	Capture Value 0 Cycle This field stores the cycle number corresponding to the value captured in CVAL0. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

31.5.38 Capture Value 1 Register (SM2CVAL1)

Offset

Register	Offset
SM2CVAL1	104h

Function
Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPTVAL1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CAPTVAL1	Capture Value 1 This field stores the value captured from the submodule counter when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this field decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This field is not byte accessible.

31.5.39 Capture Value 1 Cycle Register (SM2CVAL1CYC)

Offset

Register	Offset
SM2CVAL1CYC	106h

Function

Writing this register generates bus transfer error.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CVAL1CYC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-4 —	Reserved
3-0 CVAL1CYC	Capture Value 1 Cycle This field stores the cycle number corresponding to the value captured in CVAL1. This field is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle. Resets to 0 at POR or hard reset.

31.5.40 Output Enable Register (OUTEN)

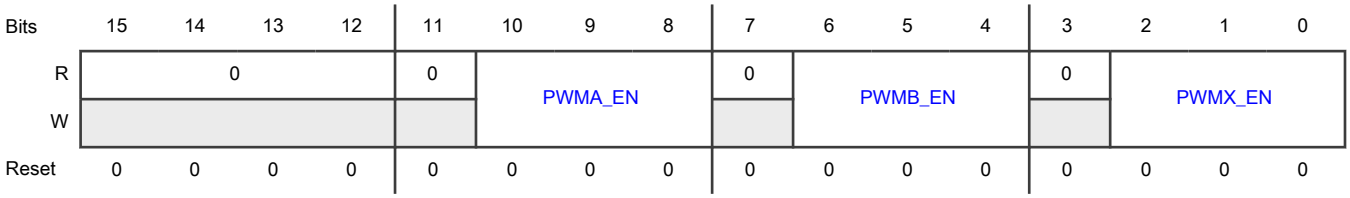
Offset

Register	Offset
OUTEN	180h

Function

Contains PWM output enables. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-12 —	Reserved
11 —	Reserved
10-8 PWMA_EN	PWM_A Output Enables This field enables the PWM_A outputs of submodules 2 - 0, respectively. Set these bits to 0 (output disabled) when a PWM_A pin is being used for input capture. • 0b0 PWM_A output disabled. • 0b1 PWM_A output enabled.
7 —	Reserved
6-4 PWMB_EN	PWM_B Output Enables This field enables the PWM_B outputs of submodules 2 - 0, respectively. Set these bits to 0 (output disabled) when a PWM_B pin is being used for input capture. • 0b0 PWM_B output disabled. • 0b1 PWM_B output enabled.
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 PWMX_EN	PWM_X Output Enables This field enables the PWM_X outputs of submodules 2 - 0, respectively. Set these bits to 0 (output disabled) when a PWM_X pin is being used for input capture or deadtime correction. • 0b0 PWM_X output disabled. • 0b1 PWM_X output enabled.

31.5.41 Mask Register (MASK)

Offset

Register	Offset
MASK	182h

Function

MASK is double buffered and does not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading MASK reads the buffered values and not necessarily the values currently in effect. This double buffering can be overridden by setting the UPDATE_MASK bits.

Diagram

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

Fields

Field	Function
15 —	Reserved
14-12 UPDATE_MASK	Update Mask Bits Immediately This field masks or unmasks the PWM_X (X = A, B, or X) outputs of submodules 2 - 0 respectively by forcing the MASK* (* = A, B or X) bits to be immediately updated within submodules 2 - 0, respectively without waiting for a FORCE_OUT event. These self-clearing bits always read as zero. Software may write to any or all of these bits and may set these bits in the same write operation that updates the MASKA, MASKB, and MASKX fields of this register. • 0b0 Normal operation. MASK* bits within the corresponding submodule are not updated until a FORCE_OUT event occurs within the submodule.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0b1 Immediate operation. MASK* bits within the corresponding submodule are updated on the following clock edge after setting this bit. For example: • UPDATE_MASK[3]=1 updates the mask bits MASKA[3], MASKB[3], and MASKX[3]. • UPDATE_MASK[3:0]=0101 updates the mask bits MASKA[2], MASKA[0], MASKB[2], MASKB[0], MASKX[2], and MASKX[0].
11 —	Reserved
10-8 MASKA	PWM_A Masks This field masks the PWM_A outputs of submodules 2 - 0, respectively by forcing the output to logic 0 prior to consideration of the output polarity. • 0b0 PWM_A output normal. • 0b1 PWM_A output masked. MASKA[3:0] masks PWM_A_3 through PWM_A_0
7 —	Reserved
6-4 MASKB	PWM_B Masks This field masks the PWM_B outputs of submodules 2 - 0, respectively by forcing the output to logic 0 prior to consideration of the output polarity. • 0b0 PWM_B output normal. • 0b1 PWM_B output masked. MASKB[3:0] masks PWM_B_3 through PWM_B_0
3 —	Reserved
2-0 MASKX	PWM_X Masks This field masks the PWM_X outputs of submodules 2 - 0, respectively by forcing the output to logic 0 prior to consideration of the output polarity. • 0b0 PWM_X output normal. • 0b1 PWM_X output masked. MASKX[3:0] masks PWM_X_3 through PWM_X_0

31.5.42 Software Controlled Output Register (SWCOUT)

Offset

Register	Offset
SWCOUT	184h

Function

These bits are double buffered and do not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading these bits reads the buffered value and not necessarily the value currently in effect.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0	SM2OUT23	SM2OUT45	SM1OUT23	SM1OUT45	SM0OUT23	SM0OUT45
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 SM2OUT23	Submodule 2 Software Controlled Output 23 This field is used when DTSRCSEL[SM2SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 2 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 2 instead of PWM23.
4 SM2OUT45	Submodule 2 Software Controlled Output 45 This field is used when DTSRCSEL[SM2SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 2 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 2 instead of PWM45.
3	Submodule 1 Software Controlled Output 23

Table continues on the next page...

Table continued from the previous page...

Field	Function
SM1OUT23	This field is used when DTSRCSEL[SM1SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 1 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 1 instead of PWM23.
2 SM1OUT45	Submodule 1 Software Controlled Output 45 This field is used when DTSRCSEL[SM1SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 1 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 1 instead of PWM45.
1 SM0OUT23	Submodule 0 Software Controlled Output 23 This field is used when DTSRCSEL[SM0SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 0 instead of PWM23. 1b - A logic 1 is supplied to the deadtime generator of submodule 0 instead of PWM23.
0 SM0OUT45	Submodule 0 Software Controlled Output 45 This field is used when DTSRCSEL[SM0SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0b - A logic 0 is supplied to the deadtime generator of submodule 0 instead of PWM45. 1b - A logic 1 is supplied to the deadtime generator of submodule 0 instead of PWM45.

31.5.43 PWM Source Select Register (DTSRCSEL)

Offset

Register	Offset
DTSRCSEL	186h

Function

The PWM source select bits are double buffered and do not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading these bits reads the buffered value and not necessarily the value currently in effect. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		SM2SEL23		SM2SEL45		SM1SEL23		SM1SEL45		SM0SEL23		SM0SEL45	
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 —	Reserved
11-10 SM2SEL23	Submodule 2 PWM23 Control Select This field selects possible overrides to the generated SM2PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM2PWM23 signal used by the deadtime logic. 01b - Inverted generated SM2PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM2OUT23] used by the deadtime logic. 11b - PWM2_EXTA signal used by the deadtime logic.
9-8 SM2SEL45	Submodule 2 PWM45 Control Select This field selects possible overrides to the generated SM2PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM2PWM45 signal used by the deadtime logic. 01b - Inverted generated SM2PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM2OUT45] used by the deadtime logic. 11b - Reserved
7-6 SM1SEL23	Submodule 1 PWM23 Control Select This field selects possible overrides to the generated SM1PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM1PWM23 signal used by the deadtime logic. 01b - Inverted generated SM1PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM1OUT23] used by the deadtime logic. 11b - PWM1_EXTA signal used by the deadtime logic.
5-4	Submodule 1 PWM45 Control Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
SM1SEL45	This field selects possible overrides to the generated SM1PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM1PWM45 signal used by the deadtime logic. 01b - Inverted generated SM1PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM1OUT45] used by the deadtime logic. 11b - Reserved
3-2 SM0SEL23	Submodule 0 PWM23 Control Select This field selects possible overrides to the generated SM0PWM23 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM0PWM23 signal used by the deadtime logic. 01b - Inverted generated SM0PWM23 signal used by the deadtime logic. 10b - SWCOUT[SM0OUT23] used by the deadtime logic. 11b - PWM0_EXTA signal used by the deadtime logic.
1-0 SM0SEL45	Submodule 0 PWM45 Control Select This field selects possible overrides to the generated SM0PWM45 signal that passes to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00b - Generated SM0PWM45 signal used by the deadtime logic. 01b - Inverted generated SM0PWM45 signal used by the deadtime logic. 10b - SWCOUT[SM0OUT45] used by the deadtime logic. 11b - Reserved

31.5.44 Master Control Register (MCTRL)

Offset

Register	Offset
MCTRL	188h

Function

In every 4-bit field in this register, each bit acts on a separate submodule. Accordingly, the description of every bit field refers to the effect of an individual bit.

Diagram

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

Fields

Field	Function
15 —	Reserved
14-12 IPOL	Current Polarity This field corresponds to submodules 2 - 0, respectively. Each bit selects between PWM23 and PWM45 as the source for the generation of the complementary PWM pair output for the corresponding submodule. MCTRL[IPOL] is ignored in independent mode. MCTRL[IPOL] does not take effect until a FORCE_OUT event takes place in the appropriate submodule. Reading MCTRL[IPOL] reads the buffered value and not necessarily the value currently in effect. 000b - PWM23 is used to generate complementary PWM pair in the corresponding submodule. 001b - PWM45 is used to generate complementary PWM pair in the corresponding submodule.
11 —	Reserved
10-8 RUN	Run This field enables the clocks to the PWM generator of submodules 2 - 0, respectively. The corresponding MCTRL[RUN] bit must be set for each submodule that is using its input capture functions or is using the local reload as its reload source. When this bit equals zero, the submodule counter is reset, and PWM outputs are held. A reset clears this field. 000b - PWM counter is stopped, but PWM outputs hold the current state. 001b - PWM counter is started in the corresponding submodule.
7 —	Reserved
6-4 CLDOK	Clear Load Okay This field corresponds to submodules 2 - 0, respectively. Each write-only bit is used to clear the corresponding bit of MCTRL[LDOK]. Write a 1 to CLDOK to clear the corresponding MCTRL[LDOK] bit. If a reload occurs within a submodule with the corresponding MCTRL[LDOK] bit set at the same time that MCTRL[CLDOK] is written, then the reload in that submodule will not be performed and MCTRL[LDOK] will be cleared. CLDOK bit is self-clearing and always reads as a 0.
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 LDOK	Load Okay This field corresponds to submodules 2 - 0, respectively. Each read/set bit loads CTRL[PRSC] and the INIT, FRACVALx, and VALx registers of the corresponding submodule into a set of buffers. The buffered prescaler divisor, submodule counter modulus value, and PWM pulse width take effect at the next PWM reload if CTRL[LDMOD] is clear or if CTRL[LDMOD] is set. Set the corresponding MCTRL[LDOK] bit by reading it when it is logic zero and then writing a logic one to it. The VALx, FRACVALx, INIT, and CTRL[PRSC] registers of the corresponding submodule cannot be written while the corresponding MCTRL[LDOK] bit is set. In Master Reload Mode (CTRL2[RELOAD_SEL]=1), it is necessary to set the LDOK bit corresponding to submodule0; however, it is recommended to also set the LDOK bit of the slave submodules, to prevent unwanted writes to the registers in the slave submodules. The MCTRL[LDOK] bit is automatically cleared after the new values are loaded, or it can be manually cleared before a reload by writing a logic 1 to the appropriate MCTRL[CLDOK] bit. LDOK bits cannot be written with a zero. MCTRL[LDOK] can be set in DMA mode when the DMA indicates that it has completed the update of all CTRL[PRSC], INIT, FRACVALx, and VALx registers in the corresponding submodule. Reset clears LDOK field. 000b - Do not load new values. 001b - Load prescaler, modulus, and PWM values of the corresponding submodule.

31.5.45 Master Control 2 Register (MCTRL2)

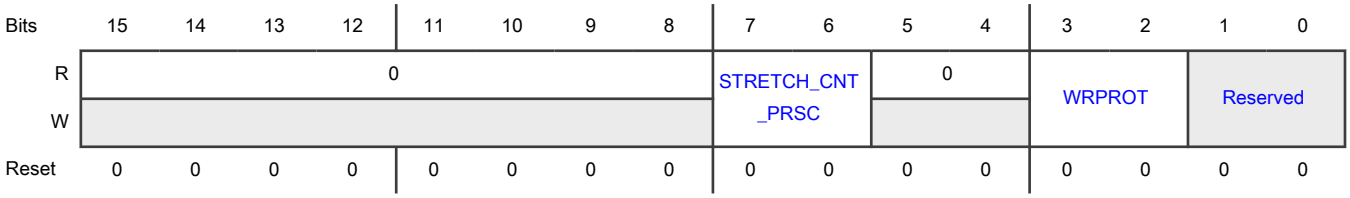
Offset

Register	Offset
MCTRL2	18Ah

Function

Includes control for monitoring the PLL state and write protection of some configuration registers.

Diagram



Fields

Field	Function
15-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-6 STRETCH_CNT_PRSC	Stretch IPBus clock count prescaler for mux0_trig/mux1_trig/out0_trig/out1_trig/pwma_trig/pwmb_trig If user config eFlexPWM work in fast clk mode(use SoC level register, eFlexPWM input signal fast_clk_mode is high), then user can use these bits to stretch output signals mux0_trig/mux1_trig/out0_trig/out1_trig/pwma_trig/pwmb_trig. 00b - Stretch count is zero, no stretch. 01b - Stretch mux0_trig/mux1_trig/out0_trig/out1_trig/pwma_trig/pwmb_trig for 2 IPBus clock period. 10b - Stretch mux0_trig/mux1_trig/out0_trig/out1_trig/pwma_trig/pwmb_trig for 4 IPBus clock period. 11b - Stretch mux0_trig/mux1_trig/out0_trig/out1_trig/pwma_trig/pwmb_trig for 8 IPBus clock period.
5-4 —	Reserved
3-2 WRPROT	Write protect Enable write protection of some configuration registers of eFlexPWM. 00b - Write protection off (default). 01b - Write protection on. 10b - Write protection off and locked until chip reset. 11b - Write protection on and locked until chip reset.
1-0 —	Reserved

31.5.46 Fault Control Register (FCTRL0)

Offset

Register	Offset
FCTRL0	18Ch

Function

For every 4-bit field in this register, the bits act on the fault inputs in order. For example, FLVL bits 15-12 act on faults 3-0, respectively. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram

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

Fields

Field	Function
15-12 FLVL	Fault Level This field selects the active logic level of the individual fault inputs 3-0, respectively. A reset clears this field. 0000b - A logic 0 on the fault input indicates a fault condition. 0001b - A logic 1 on the fault input indicates a fault condition.
11-8 FAUTO	Automatic Fault Clearing This field selects automatic or manual clearing of faults 3-0, respectively. A reset clears this field. 0000b - Manual fault clearing. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL]. If neither FFULL nor FHALF is set, then the fault condition cannot be cleared. This is further controlled by FCTRL[FSAFE]. 0001b - Automatic fault clearing. PWM outputs disabled by this fault are enabled when FSTS[FFPINx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL] without regard to the state of FSTS[FFLAGx]. If neither FFULL nor FHALF is set, then the fault condition cannot be cleared.
7-4 FSAFE	Fault Safety Mode This field selects the safety mode during manual fault clearing. A reset clears this field. FSTS[FFPINx] may indicate that a fault condition still exists even though the actual fault signal at the FAULTx pin is clear due to the fault filter latency. 0000b - Normal mode. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL] without regard to the state of FSTS[FFPINx]. If neither FHALF nor FFULL is set, then the fault condition cannot be cleared. The PWM outputs disabled by this fault input will not be re-enabled until the actual FAULTx input signal de-asserts since the fault input will combinationally disable the PWM outputs (as programmed in DISMAPn). 0001b - Safe mode. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear and FSTS[FFPINx] is clear at the start of a half cycle or full cycle depending on the states of FSTS[FHALF] and FSTS[FFULL]. If neither FHLAF nor FFULL is set, then the fault condition cannot be cleared.
3-0 FIE	Fault Interrupt Enables This field enables CPU interrupt requests generated by the FAULTx pins. A reset clears this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The fault protection circuit is independent of the FIEx bit and is always active. If a fault is detected, the PWM outputs are disabled according to the disable mapping register. 0000b - FAULTx CPU interrupt requests disabled. 0001b - FAULTx CPU interrupt requests enabled.

31.5.47 Fault Status Register (FSTS0)

Offset

Register	Offset
FSTS0	18Eh

Function

Includes controls related to fault conditions.

Diagram

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

Fields

Field	Function
15-12 FHALF	Half Cycle Fault Recovery This field is used to control the timing for re-enabling the PWM outputs after a fault condition. These bits apply to both automatic and manual clearing of a fault condition. These register bits are write-protected by MCTRL2[WRPROT] bits. **NOTE** Both FHALF and FFULL can be set so that the fault recovery occurs at the start of a full cycle and at the start of a half cycle (as defined by VAL0). If neither FHALF nor FFULL is set, then no fault recovery is possible. 0000b - PWM outputs are not re-enabled at the start of a half cycle. 0001b - PWM outputs are re-enabled at the start of a half cycle (as defined by VAL0).
11-8	Filtered Fault Pins

Table continues on the next page...

Table continued from the previous page...

Field	Function
FFPIN	These read-only bits reflect the current state of the filtered FAULTx pins converted to high polarity. A logic 1 indicates that a fault condition exists on the filtered FAULTx pin. A reset has no effect on this field. After the system reset de-asserts, these are the possible values of FFPIN: If FCTRL[FLVL] = 0 and FAULTx = 0, then FFPIN is set to 1. If FCTRL[FLVL] = 0 and FAULTx = 1, then FFPIN is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 0, then FFPIN is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 1, then FFPIN is set to 1.
7-4 FFULL	Full Cycle This field is used to control the timing for re-enabling the PWM outputs after a fault condition. These bits apply to both automatic and manual clearing of a fault condition. These register bits are write-protected by MCTRL2[WRPROT] bits. **NOTE** Both FHALF and FFULL can be set so that the fault recovery occurs at the start of a full cycle and at the start of a half cycle (as defined by VAL0). If neither FHALF nor FFULL is set, then no fault recovery is possible. 0000b - PWM outputs are not re-enabled at the start of a full cycle 0001b - PWM outputs are re-enabled at the start of a full cycle
3-0 FFLAG	Fault Flags These read-only flags are set within two CPU cycles after a transition to active on the FAULTx pin. Clear this bit by writing a logic one to it. A reset clears this field. While the reset value is 0, these bits may be set to 1 by the time they can be read depending on the state of the fault input signals. After the system reset de-asserts, these are the possible values of FFLAG: If FCTRL[FLVL] = 0 and FAULTx = 0, then FFLAG is set to 1. If FCTRL[FLVL] = 0 and FAULTx = 1, then FFLAG is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 0, then FFLAG is kept as 0. If FCTRL[FLVL] = 1 and FAULTx = 1, then FFLAG is set to 1. 0000b - No fault on the FAULTx pin. 0001b - Fault on the FAULTx pin.

31.5.48 Fault Filter Register (FFILT0)

Offset

Register	Offset
FFILT0	190h

Function

The settings in this register are shared among each of the fault input filters within the fault channel.

Input filter considerations include:

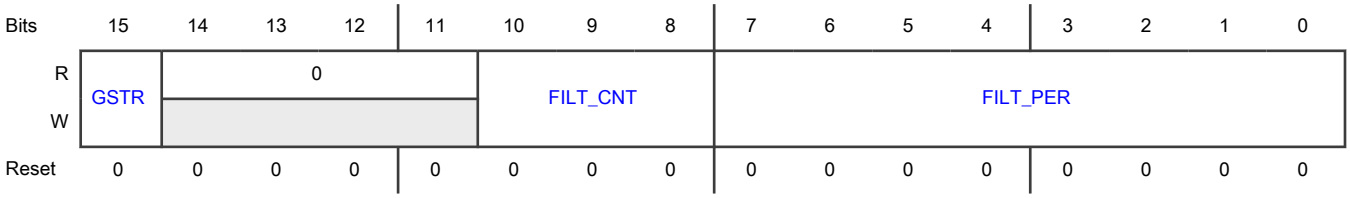
- Set the FILT_PER value such that the sampling period is larger than the period of the expected noise. This way a noise spike corrupts one sample. Select the FILT_CNT value to reduce and recognize the probability of noisy samples causing an incorrect transition. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the FILT_CNT+3 power.
- The values of FILT_PER and FILT_CNT must be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting FILT_PER to a non-zero value) introduces a latency of ((FILT_CNT+4) x FILT_PER x IPBus clock period).

NOTE

When the filter is enabled, there is a combinational path to disable the PWM outputs to ensure rapid response to fault conditions, and fault response if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15 GSTR	Fault Glitch Stretch Enable This field is used to enable the fault glitch-stretching logic. This logic ensures that narrow fault glitches are stretched to be at least 2 IPBus clock cycles wide. In some cases, a narrow fault input can cause problems due to the short PWM output shutdown/re-activation time. The stretching logic ensures that a glitch on the fault input, when the fault filter is disabled, will be registered in the fault flags. 0b - Fault input glitch stretching is disabled. 1b - Input fault signals are stretched to at least 2 IPBus clock cycles.
14-11 —	Reserved
10-8 FILT_CNT	Fault Filter Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bit field value of 0-7 represents 3-10 samples, respectively. The value of FILT_CNT affects the input latency.
7-0 FILT_PER	Fault Filter Period This 8-bit field applies universally to all fault inputs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	These bits represent the sampling period (in IPBus clock cycles) of the fault pin input filter. Each input is sampled multiple times at the rate specified by this field. If FILT_PER is 0x00 (default), then the input filter is bypassed. The value of FILT_PER affects the input latency. **NOTE** When changing values for FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

31.5.49 Fault Test Register (FTST0)

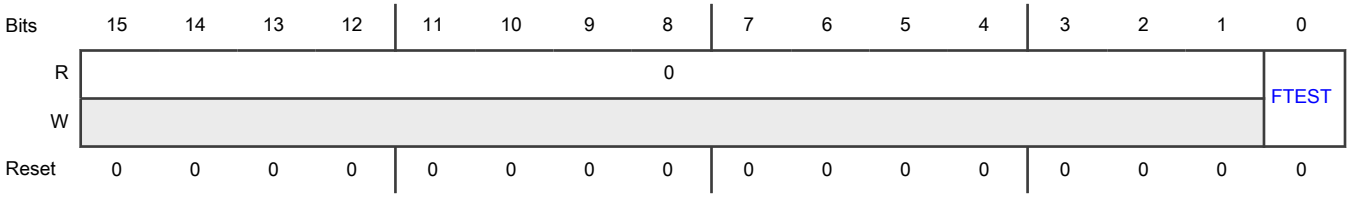
Offset

Register	Offset
FTST0	192h

Function

Contains FTEST field for fault simulation.

Diagram



Fields

Field	Function
15-1 —	Reserved
0 FTEST	Fault Test This field is used to simulate a fault condition. Setting this bit causes a simulated fault to be sent into all of the fault filters. The condition propagates to the fault flags and possibly the PWM outputs depending on the DISMAPn settings. Clearing this bit removes the simulated fault condition. This register bit is write-protected by MCTRL2[WRPROT] bits. 0b - No fault 1b - Cause a simulated fault

31.5.50 Fault Control 2 Register (FCTRL20)

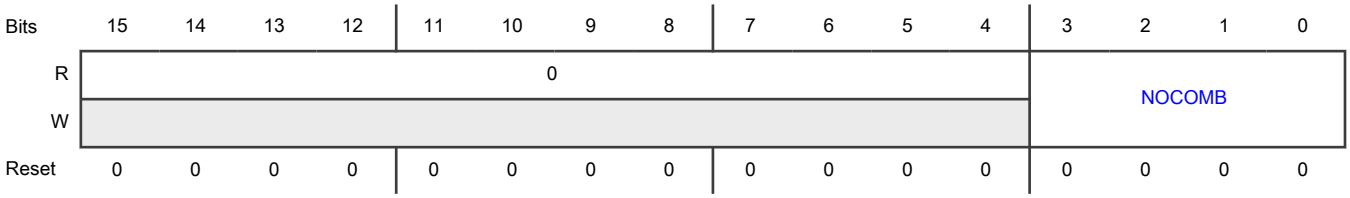
Offset

Register	Offset
FCTRL20	194h

Function

Controls combinational link from fault inputs to PWM outputs. This register is write-protected by MCTRL2[WRPROT] bits.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 NOCOMB	No Combinational Path From Fault Input To PWM Output This field is used to control the combinational path from the fault inputs to the PWM outputs. When these bits are low (default), the corresponding fault inputs have a combinational path to the PWM outputs that are sensitive to these fault inputs (as defined by DISMAP0). This combinational path is a safety feature that ensures the output is disabled even if the SOC has a failure of its clocking system. The combinational path also means that a pulse on the fault input can cause a brief disable of the PWM output even if the fault pulse is not wide enough to get through the input filter and be latched in the fault logic. Setting these bits removes the combinational path and uses the filtered and latched fault signals as the fault source to disable the PWM outputs. This eliminates fault glitches from creating PWM output glitches but also increases the latency to respond to a real fault. 0000b - There is a combinational link from the fault inputs to the PWM outputs. The fault inputs are combined with the filtered and latched fault signals to disable the PWM outputs. 0001b - The direct combinational path from the fault inputs to the PWM outputs is disabled and the filtered and latched fault signals are used to disable the PWM outputs.

Chapter 32
Quadrature Decoder (eQDC)

32.1 Chip-specific QDC information

Table 175. Reference links to related information

Topic	Related module	Reference
Full description	QDC	QDC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

32.1.1 Module instances

This device has two instances of the QDC module: QDC0.QDC1.

32.1.2 Peripheral interconnections

This module provides interfacing capability to position/speed sensors used in industrial motor control applications. It has five input signals: PHASEA, PHASEB, INDEX, TRIGGER, and HOME. This module is used to decode shaft position, revolution count and speed.

See the attached Peripheral Mux Assignments spreadsheet for details.

32.2 Overview

eQDC does the following:

- Interfaces to position or speed sensors that are used in industrial motor control applications.
- Decodes shaft position, revolution count, and speed.

eQDC receives the following 8 input signals from position or speed sensors:

- PHASEA
- PHASEB
- INDEX/PRESET
- TRIGGER
- HOME/ENABLE
- ICAP[3:1]

eQDC outputs the following 11 signals for system use:

- POS_MATCH[3:0]
- COMP_FLG[3:0]
- DIR
- CNT_DN

- CNT_UP

NOTE

For eQDC, "asserted" indicates an output of 1 and "de-asserted" indicates an output of 0.

32.2.1 Block diagram

This is the block diagram of eQDC.

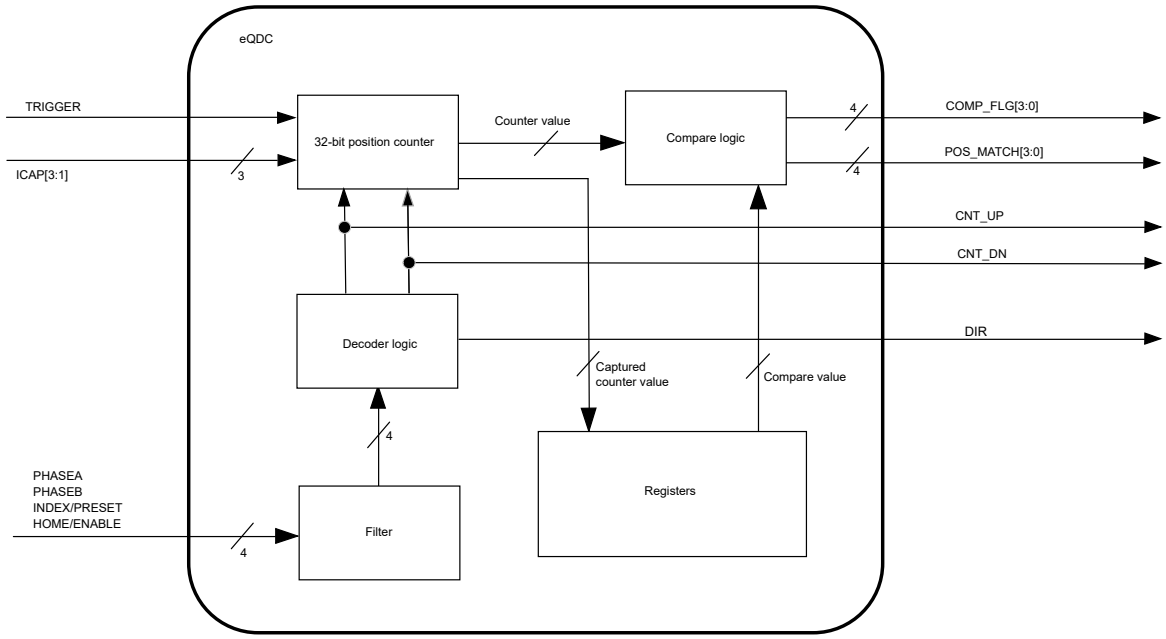


Figure 113. Block diagram

32.2.2 Features

- Logic to decode quadrature signals
- Configurable digital filters for inputs (these filters can be bypassed)
- 32-bit position counter capable of modulo counting
- Position counter that can be initialized by software or external events
- 16-bit position difference register
- Compare function that can indicate when shaft has reached a defined position
- A watchdog timer that can detect a non-rotating shaft condition
- Preloadable 16-bit revolution counter
- Maximum count frequency equals the peripheral clock rate
- Configurable interrupt when both PHASEA and PHASEB inputs change in the same cycle

32.3 Functional description

The following timing diagram shows the basic operation of an incremental position quadrature decoder.

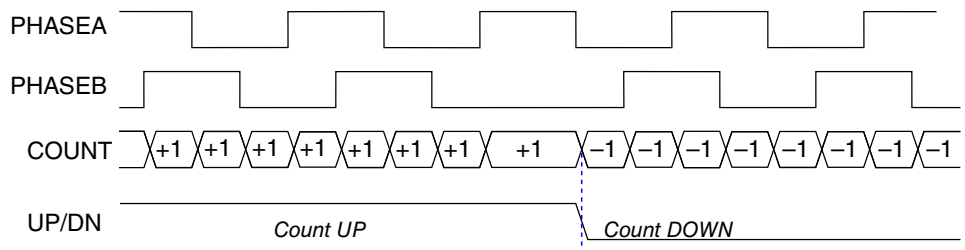


Figure 114. Quadrature decoder signals

32.3.1 Positive versus negative direction

A typical quadrature encoder has 3 outputs: PHASEA signal, PHASEB signal, INDEX pulse (not shown).

- If PHASEA leads PHASEB, then motion is in the positive direction.
- If PHASEA trails PHASEB, then motion is in the negative direction.

Transitions on these phases can be integrated to yield position or differentiated to yield velocity. The quadrature decoder is designed to perform these functions in hardware.

32.3.2 Speed measurement

For applications with a fast-moving shaft encoder, the speed can be measured using either of the following methods:

- calculating the change in the position counter per unit time
- reading the position difference counter register

For applications with low motor speeds and low-line count quadrature encoders, the timer module enables high-resolution speed measurement by calculating the time intervals between quadrature phases.

- The timer module uses a 16-bit free running counter operated from a prescaled version of the peripheral clock.
- The prescaler FILT[PRSC] divides the peripheral clock by values ranging from 1 to 32768. A 100 MHz peripheral clock frequency would yield a resolution of from 10 ns to 327.68 µs and a maximum count period of from 0.65535 ms to 21474.5 ms. For example, with a 1000-tooth decoder, speeds could be calculated down to 0.0007 rpm using a prescaler.

32.3.3 Glitch filter

Because the quadrature decoder logic must sense signal transitions, the signal inputs are first run through a glitch filter. This glitch filter has a digital delay line, which samples multiple time points on the signal and verifies a stable new signal state before outputting this new signal state to the internal quadrature decoder logic. To adapt to a variety of signal bandwidths, the sample rate of this delay line is programmable.

32.3.4 Edge detect state machine

The Edge detect state machine looks for changes in the 4 possible states of the filtered PHASEA and PHASEB input signals. Direction of motion is calculated using these changes. Direction of motion is formatted as Count_Up and Count_Down signals. These signals are routed into up to 3 up/down counters:

- Position counter
- Revolution counter
- Position difference counter

32.3.5 Counter registers, and hold registers and how to initialize

When any of the counter registers is read, the contents of this counter register is written to the corresponding hold register. Taking a snapshot of the counters' values allows a consistent view of a system's position and the speed to be attained. If CTRL2[PMEN] is set, the position difference hold register is only updated when the position difference counter register is read. To capture a time stamp of when these registers are read, use the POS_MATCH output with a timer channel.

32.3.5.1 Position counter (POS)

POS counts up (upper position counter, UPOS) or down (lower position counter, LPOS) on every count pulse generated by the position difference of PHASEA and PHASEB input signals. POS acts as an integration_info, whose count value is proportional to position. De-assertion of ENABLE pin stops the operation of POS. POS is initialized when any of the following conditions is met:

- If CTRL2[INITPOS] is set, a positive edge of TRIGGER input occurs.
- A write to CTRL[SWIP]
- If CTRL2[OPMODE] is cleared and CTRL[XIP] is set to 1, INDEX signal transition occurs
- If CTRL2[OPMODE] is cleared and CTRL[HIP] is set to 1, HOME signal transition occurs
- If CTRL2[OPMODE] is set, a positive edge of PRESET signal occurs.
- Position counter roll-over and roll-under

POS can be initialized to initial register if CTRL[REV]=0 and initialized to modulus register if CTRL[REV]=1, but if CTRL2[EMIP]=1 and INDEX signal transition initiates POS, POS is initialized under the circumstances listed in the following table:

	CTRL[XNE] = 0	CTRL[XNE] = 1	CTRL[REV] = 0	CTRL[REV] = 1
PHASEA leads PHASEB (Clockwise)	positive edge of INDEX resets POS	negative edge of INDEX resets POS	POS is reset to initial value	POS is reset to modulus value
PHASEA lags PASEHB (Counterclockwise)	negative edge of INDEX resets POS	positive edge of INDEX resets POS	POS is reset to modulus value	POS is reset to initial value

The INDEX and HOME signals can be programmed to interrupt the processor. When UPOS or LPOS is read, a snapshot of each of the following counter registers is placed into the corresponding hold register:

- position counter register
- If CTRL2[PMEN] is cleared, position difference counter register
- revolution counter register

32.3.5.2 Position counter hold (POSH)

POSH stores a copy of POS when POS is read.

When POS is initialized by any event, the contents of POS are also copied into POSH. Meanwhile, count direction flag IOMR[DIR] is copied into count direction flag hold IOMR[DIRH]. When CTRL2[UPDHLD] is set to1, the rising edge of TRIGGER input takes a snapshot of POS and stores it in UPOSH or LPOS.

Beside taking snapshot of POS counter by positive transient of TRIGGER input, 3 more signals are added to take snapshot of POS counter:

- Positive transition of ICAP[3] input can take snapshot of POS counter into UPOSH3/LPOS3
- Positive transition of ICAP[2] input can take snapshot of POS counter into UPOSH2/LPOS2
- Positive transition of ICAP[1] input can take snapshot of POS counter into UPOSH1/LPOS1

Note: This is used for an error checking mechanism to check of the position counter accumulated the correct number of counts between the same event. As example, the 1024-line incremental encoder must count 4096 counts when moving in the same direction between the index.

32.3.5.3 Position difference counter (POSD)

POSD contains the position difference value occurring between each read of the position register. The position register counts up or down on every count pulse. The position difference counter acts as a differentiator, whose count value is proportional to the change in position since the last time the position counter was read.

POSD is cleared when any of the following conditions is met:

- If CTRL2[PMEN] is cleared, when POS, POSD, or REV register is read, POSD is cleared and its contents are copied into POSDH.
- If CTRL2[PMEN] is cleared, positive edge of TRIGGER input occurs when CTRL2[UPDPOS] is set
- If CTRL2[PMEN] is set, when POSD is read.
- If CTRL2[PMEN] is set, positive edge of TRIGGER input occurs when CTRL2[UPDHLD] is set.

32.3.5.4 Position difference counter hold (POSDH)

POSDH takes and stores a snapshot of POSD. The snapshot is taken when any of the following conditions is met:

- If CTRL2[PMEN] is cleared, when POS, POSD, or REV register is read.
- When POSD is read (no matter CTRL2[PMEN] is clear or set).
- If CTRL2[UPDHLD] is set, a positive edge of the TRIGGER input occurs (no matter CTRL2[PMEN] is cleared or set).

32.3.5.5 Revolution counter (REV)

REV counts or integrates revolutions by counting index pulses. The direction of the count is determined by PHASEA and PHASEB input signals. A different count direction on the rising and falling edges of the index pulse indicates that eQDC changed direction on the index pulse. If CTRL2[OPMODE] is set, de-assertion of ENABLE input signal stops the operation of REV and PRESET signal transition initializes revolution counter to zero.

32.3.5.6 Revolution counter hold (REVH)

REVH takes a snapshot of REV when either of the following conditions is met:

- When POS, POSD, or REV register is read.
- If CTRL2[UPDHLD] is set, when a positive edge of the TRIGGER input signal occurs.

32.3.5.7 Watchdog timer (WDG)

WDG ensures that the algorithm is indicating motion in the shaft; 2 successive counts indicate proper operation and resets the timer. If CTRL2[OPMODE] is set, PRESET signal transition initializes WDG r to zero, and de-assertion of ENABLE input stops the operation of the WDG.

- The timeout value is programmable.
- When a timeout occurs, an interrupt to the processor can be generated.

32.3.5.8 Last edge time counter (LASTEDGE)

LASTEDGE contains the time since the last edge on PHASEA or PHASEB. The last edge time counter counts up on every prescaled clock pulse.

When POSD is read, contents in LASTEDGE are copied into LASTEDGEH.

32.3.5.9 Position difference period counter (POSDPER)

POSDPER contains the accumulated time from the last time the position difference counter was read. It counts up on every prescaled clock pulse and is loaded from the last edge time counter when POSD is read.

32.3.6 Double-set registers loading operation

The following registers are double-set (namely outer or inner-set), which enables them to be re-configured when eQDC is operating:

- Compare registers (UCOMP0/LCOMP0,UCOMP1/LCOMP1,UCOMP2/LCOMP2,UCOMP3/LCOMP3)
- Initial registers (UINIT/LINIT)
- Modulus registers (UMOD/LMOD)

After a value is written to one of these registers, the value is "buffered" into outer-set registers temporarily. Values will be loaded into inner-set registers and take effect using the following two methods:

- If CTRL2[LDMODE] is set to 1, "buffered" values are loaded into inner-set and take effect at the next roll-over or roll-under if CTRL[LDOK] is set.
- If CTRL2[LDMODE] is set to 0, "buffered" values are loaded into inner-set and take effect immediately when CTRL[LDOK] is set.

The double-set registers can be configured manually or automatically.

To configure the double-set registers manually:

1. Initialize the double-set registers:
    a. Set CTRL2[LDMODE] to 0.
    b. Clear CTRL[LDOK] if it is not 0.
    c. Write values into double-set registers, so the values are "buffered" into outer-set registers.
    d. Set CTRL[LDOK] to 1.
    e. Wait for CTRL[LDOK] to become 0, which means the values have been loaded into inner-set registers.
    f. Change CTRL2[LDMODE] value as desired.
2. Update the double-set registers:
    a. Clear CTRL[LDOK] if it is not 0.
    b. Write values into double-set registers, so the values are "buffered" into outer-set registers.
    c. Set CTRL[LDOK] to 1. When the values are loaded into inner-set registers, CTRL[LDOK] is cleared automatically.
    d. Go back to step a for iteration.

To configure the double-set registers automatically:

1. Initialize the double-set registers:
    a. Set CTRL2[LDMODE] to 0.
    b. Clear CTRL[LDOK] if it is not 0.
    c. Write values into double-set registers, so the values are "buffered" into outer-set registers.
    d. Set CTRL[LDOK] to 1.
    e. Wait for CTRL[LDOK] to become 0, which means the values have been loaded into inner-set registers.
    f. Set CTRL2[LDMODE] to 1.
    g. Set CTRL[DMAEN] to 1 to enable DMA function.
    h. Set CTRL[LDOK] to 1. This step is to ensure the DMA automatic operation occurs properly. DMA is triggered by loading values from outer-set to inner-set.
2. After initialization, the DMA is triggered in the following manner:
    a. Values are loaded into inner-set registers when the counter rolls over or rolls under, then CTRL[LDOK] is cleared automatically and a DMA request is sent out for next outer-set registers value update.

b. After DMA updates the outer-set registers, CTRL[LDOK] is set to 1 automatically. It then goes back to step a for iteration.

32.3.7 Modes of operation

eQDC operates in the following modes:

- Quadrature Decode Operation mode (CTRL[PH1] = 0,CTRL2[OPMODE] = 0)
- Quadrature Count Operation mode (CTRL[PH1] = 0,CTRL2[OPMODE] = 1)
- Single Phase Decode Operation mode (CTRL[PH1] = 1,CTRL2[OPMODE] = 0)
- Single Phase Count Operation mode (CTRL[PH1] = 1,CTRL2[OPMODE] = 1)

These operation modes support the following applications:

- Pulse accumulator
- Frequency Meter
- Period Meter
- PulseTrain Output with optional Quadrature and Direction

32.3.7.1 Quadrature decode (QDC) operation mode (CTRL[PH1] = 0, CTRL2[OPMODE] = 0)

In this mode, PHASEA and PHASEB input signals from speed or position sensors are decoded by the following signals in eQDC:

1. PHASEA/PHASEB: Two 90 degree out of phase pulse trains.
2. INDEX: Can be configured to cause a change of state on REV, initialize POS, and reset POSD.
3. HOME: Can be configured to initialize POS.
4. TRIGGER: Can be configured to take a snapshot of POS, POSD, and REV and to reinitialize POS.
5. ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: Position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare register or the time at which the position information is read:

  — If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare register (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).

  — If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.

- COMP_FLG[3:0]: Position Counter Compare Output:

  — COMP_FLG[x](x range is 0-3) is asserted when the value of Position Counter is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).

  — COMP_FLG[x](x range is 0-3) is de-asserted when the value of Position Counter is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).

- DIR: Direction of position counting:

  — Direction is up when DIR is asserted

  — Direction is down when DIR is de-asserted.

- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.

- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes(if CTRL[REV]=0,it is normal operation; if CTRL[REV]=1,it is reverse operation):

- Mode 0[CM0](CTRL2[CMODE]=00): Normal Quadrature X4 / Reverse Quadrature X4
- Mode 1[CM1](CTRL2[CMODE]=01): Normal Quadrature X2 / Reverse Quadrature X2
- Mode 2[CM2](CTRL2[CMODE]=10): Normal Quadrature X1 / Reverse Quadrature X1

32.3.7.2 Quadrature count (QCT) operation mode (CTRL[PH1] = 0, CTRL2[OPMODE] = 1)

In this mode, PHASEA and PHASEB input signals from speed or position sensors are counted by the following signals in eQDC:

- PHASEA/PHASEB: Two 90 degree out of phase pulse trains.
- PRESET: Initializes POS and resets POSD:
  - If CTRL[REV]=0, a positive edge of PRESET initializes POS to INIT value (UNIT/LINIT).
  - If CTRL[REV]=1, a positive edge of PRESET initializes POS to modulus value (UMOD/LMOD).
- ENABLE: All counters start counting when ENABLE is asserted; all counters stop counting when ENABLE is de-asserted.
- TRIGGER: Can be configured to take a snapshot of POS, POSD, and REV and to reinitialize POS.
- ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare register or the time at which the position information is read:
  - If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare register (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).
  - If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.
- COMP_FLG[3:0]: Position Counter Compare Output:
  - COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).
  - COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).
- DIR: Direction of position counting:
  - Direction is up when DIR is asserted
  - Direction is down when DIR is de-asserted.
- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes(if CTRL[REV]=0,it is normal operation; if CTRL[REV]=1,it is reverse operation):

- Mode 0[CM0](CTRL2[CMODE]=00): Normal Quadrature X4 / Reverse Quadrature X4
- Mode 1[CM1](CTRL2[CMODE]=01): Normal Quadrature X2 / Reverse Quadrature X2
- Mode 2[CM2](CTRL2[CMODE]=10): Normal Quadrature X1 / Reverse Quadrature X1

32.3.7.3 Single phase decode (PH1DC) operation mode (CTRL[PH1] = 1, CTRL2[OPMODE] = 0)

In this mode, PHASEA and PAHSEB input signals from speed or position sensors are decoded by the following signals in eQDC:

- PHASEA: Pulse trains.
- PHASEB: Pulse trains or direction.
- INDEX: Can be configured to cause a change of state on REV and to initialize POS.
- HOME: Can be configured to initialize POS.
- TRIGGER: Can be configured to take a snapshot of POS, POSD, and REV and to reinitialize POS.
- ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare register or the time at which the position information is read:

  — If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare register (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).

  — If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.

- COMP_FLG[3:0]: Position Counter Compare Output:

  — COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).

  — COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).

- DIR: Direction of position counting:

  — Direction is up when DIR is asserted

  — Direction is down when DIR is de-asserted.

- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes:

- Mode 0[CM0](CTRL2[CMODE]=00): UP/DOWN Pulse Count mode. Both position counter (POS) and position difference counter (POSD) count in the up direction when input PHASEA rising edge occurs. Both counters count in the down direction when input PHASEB rising edge occurs. If CTRL[REV] is 1, then the position counter will count in the opposite direction.
- Mode 1[CM1](CTRL2[CMODE]=01): Signed Count mode (Double Edge). Both position counter (POS) and position difference counter (POSD) count the input PHASEA on both rising edge and falling edge while the input PHASEB provides the selected position counter direction (up/down). If CTRL[REV] is 1, then the position counter will count in the opposite direction.
- Mode 2[CM2](CTRL2[CMODE]=10): Signed Count mode (Single Edge). Both position counter (POS) and position difference counter (POSD) count on the input PHASEA rising edge while the input PHASEB provides the selected position counter direction (up/down). If CTRL[REV] is 1, then the position counter will count in the opposite direction.

NOTE

If the positive edge of both PHASEA and PHASEB occurs simultaneously, an error interrupt is generated CTRL2[SABIRQ] and counter is unchanged.

32.3.7.4 Single phase count (PH1CT) operation mode (CTRL[PH1] =1, CTRL2[OPMODE] = 1)

In this mode, PHASEA and PHASEB input signals from speed or position sensors are counted by the following signals in eQDC:

- PHASEA: Pulse trains.
- PHASEB: Pulse trains or direction.
- PRESET: Initializes POS and resets POSD:
  - If CTRL[REV]=0, a positive edge of PRESET initializes POS to initial value (UNIT/LINIT).
  - If CTRL[REV]=1, a positive edge of PRESET initializes POS to modulus value (UMOD/LMOD).
- ENABLE: All counters start counting when ENABLE is asserted; all counters stop counting when ENABLE is de-asserted.
- TRIGGER: Can be configured to take a snapshot of POS, POSD, and REV, and to reinitialize POS.
- ICAP[3:1]: Takes a snapshot of POS.

eQDC outputs the following signals for system use:

- POS_MATCH[3:0]: position match:

  Records the time at which the position of the shaft matches a user-defined value of the compare register or the time at which the position information is read:
  - If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare register (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx).
  - If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.
- COMP_FLG[3:0]: Position Counter Compare Output:
  - COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx).
  - COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).
- DIR: Direction of position counting:
  - Direction is up when DIR is asserted
  - Direction is down when DIR is de-asserted.
- CNT_UP: Position counter count up: when eQDC decodes a count-up event, CNT_UP outputs a pulse which width is 1 peripheral clock cycle.
- CNT_DN: Position counter count down: when eQDC decodes a count-down event, CNT_DN outputs a pulse which width is 1 peripheral clock cycle.

There are three count modes:

- Mode 0[CM0](CTRL2[CMODE]=00): UP/DOWN Pulse Count mode. Both position counter (POS) and position difference counter (POSD) count in the up direction when input PHASEA rising edge occurs, both counters count in the down direction when input PHASEB rising edge occurs.
- Mode 1[CM1](CTRL2[CMODE]=01): Signed Count mode (Double Edge). Both position counter (POS) and position difference counter (POSD) count the input PHASEA on both rising edge and falling edge while the input PHASEB provides the selected position counter direction (up/down) .

- Mode 2[CM2](CTRL2[CMODE]=10): Signed Count mode (Single Edge). Both position counter (POS) and position difference counter (POSD) count on the input PHASEA rising edge while the input PHASEB provides the selected position counter direction (up/down) .

NOTE

If the positive edge of both PHASEA and PHASEB occurs simultaneously, an error interrupt is generated CTRL2[SABIRQ] and counter is unchanged.

32.3.7.5 Normal quadrature X1

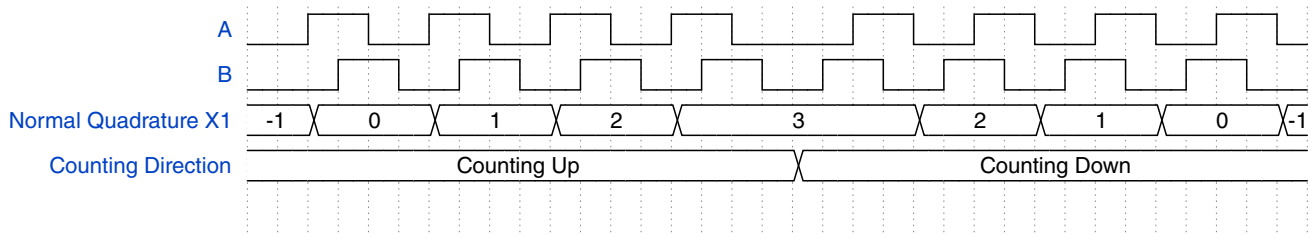


Figure 115. Normal quadrature X1

32.3.7.6 Normal quadrature X2

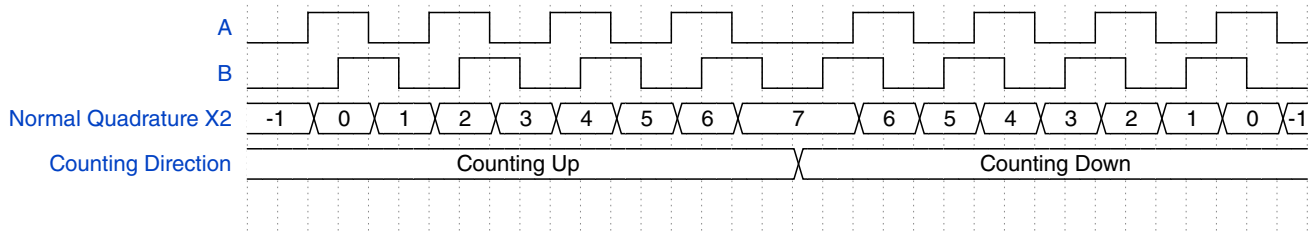


Figure 116. Normal quadrature X2

32.3.7.7 Normal quadrature X4

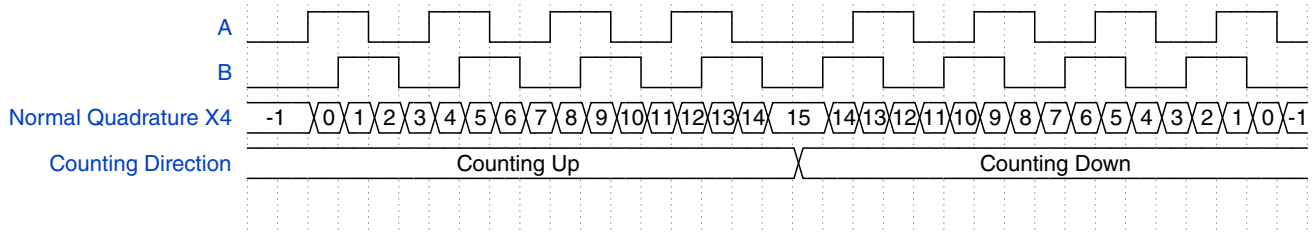


Figure 117. Normal quadrature X4

32.3.7.8 Reverse quadrature X1

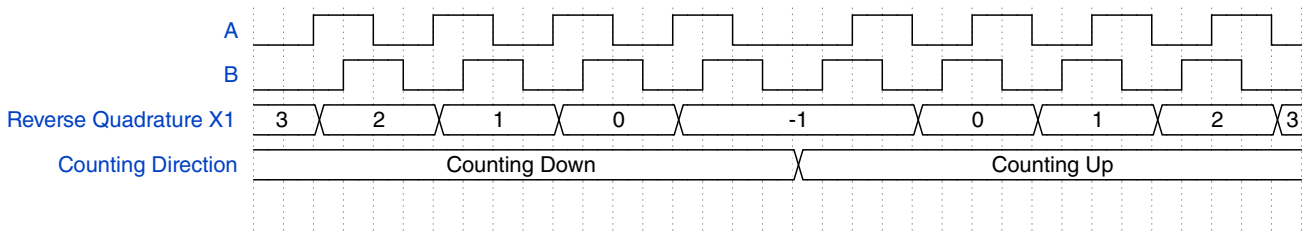


Figure 118. Reverse quadrature X1

NOTE

Set CTRL[REV] to 1 for the position counter to count in the opposite direction.

32.3.7.9 Reverse quadrature X2

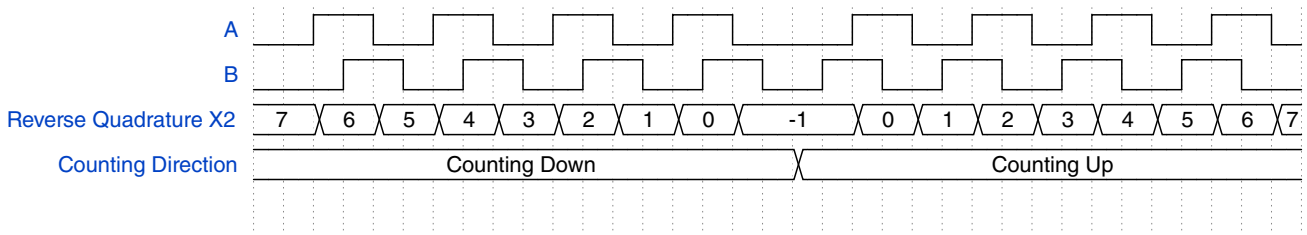


Figure 119. Reverse quadrature X2

NOTE

Set CTRL[REV] to 1 for the position counter to count in the opposite direction.

32.3.7.10 Reverse quadrature X4

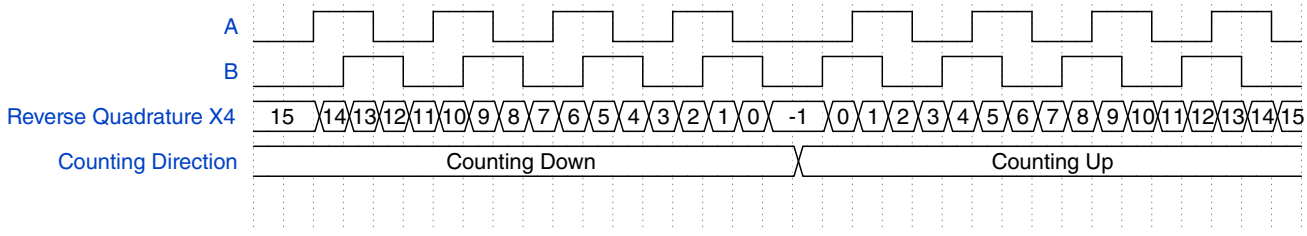


Figure 120. Reverse quadrature X4

NOTE

Set CTRL[REV] to 1 for the position counter to count in the opposite direction.

32.3.7.11 UP/DOWN pulse count mode

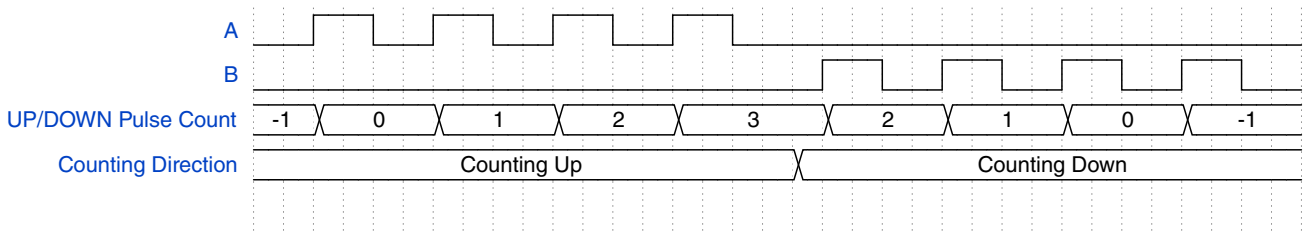


Figure 121. UP/DOWN pulse count mode

32.3.7.12 Signed count mode (double edge)

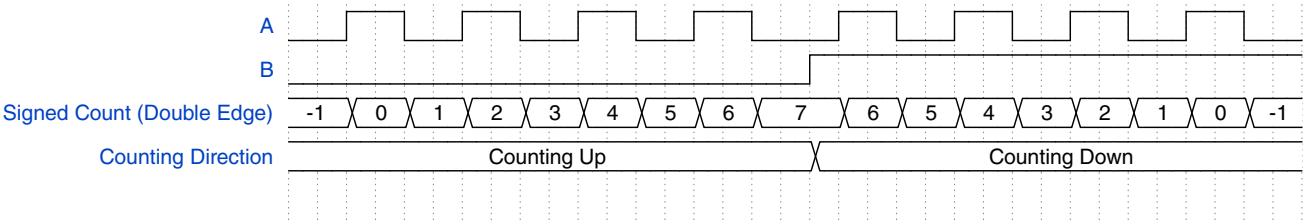


Figure 122. Signed count mode (double edge)

32.3.7.13 Signed count mode (single edge)

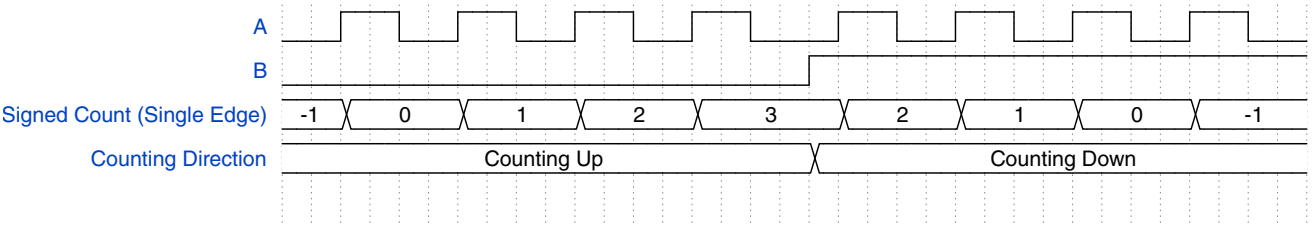


Figure 123. Signed count mode (single edge)

32.3.8 Speed measurement method

This section explains the speed measurement method, including the speed measurement algorithm.

The following figure shows the module working mechanism.

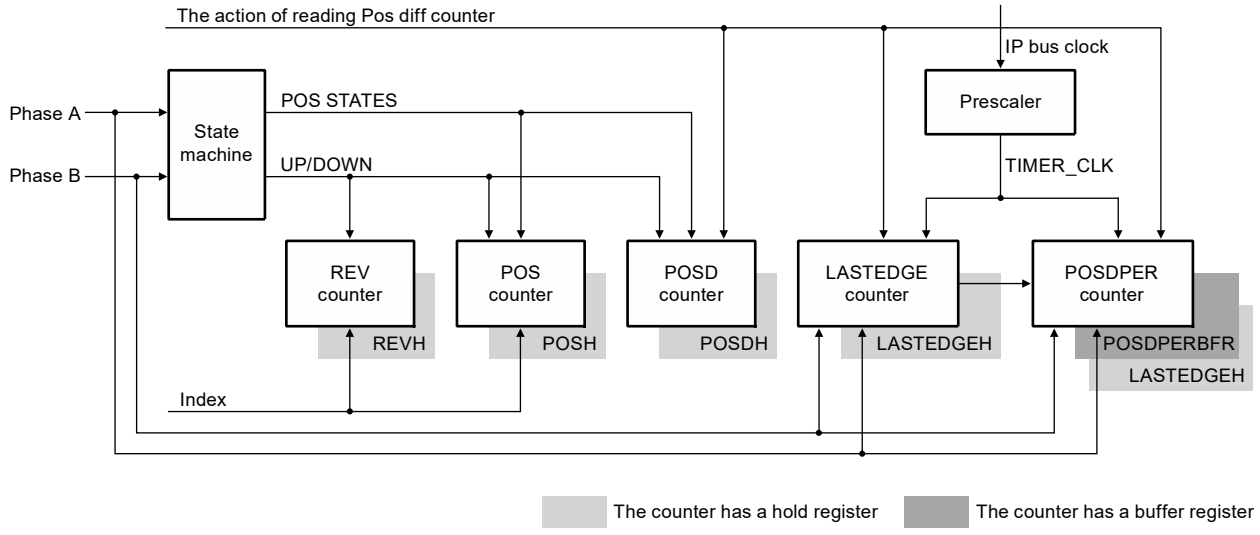


Figure 124. Quadrature decoder working mechanism

The following figure shows counters behavior and updating mechanism in the speed measurement method.

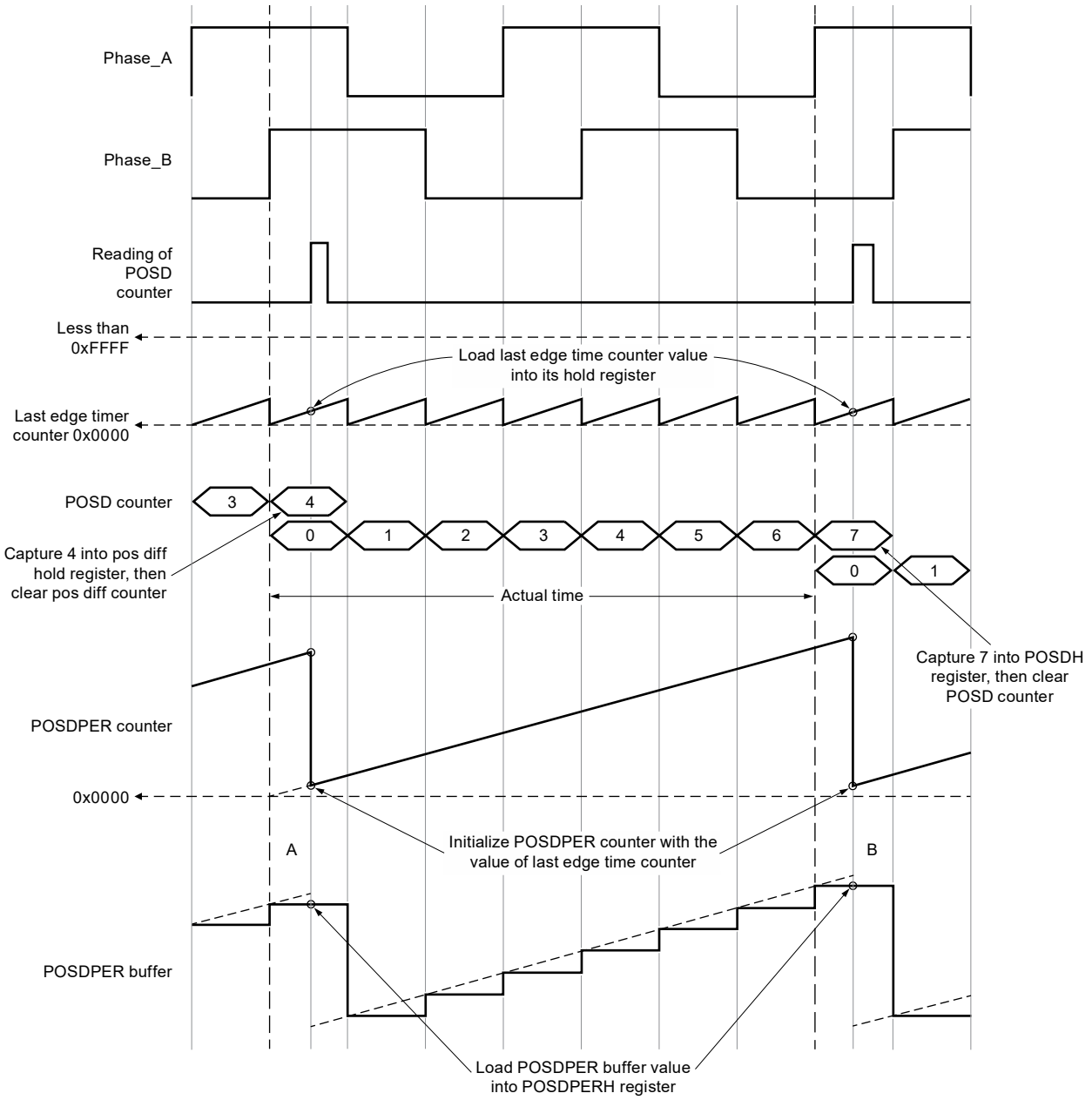


Figure 125. Counters behavior and updating mechanism

A reading of POSD occurs at time point A and it also occurs at time point B. “Actual time” represents the time duration between the first PhaseA/B edges right before time points A and B. At time point B, after reading POSD, POSDPERH contains the time length of “Actual time” and POSDH contains the value of PhaseA/B pulses within that “Actual time”. Speed can be calculated based on POSDPERH and POSDH. A visualized explanation of speed measurement with this method is shown in the figure below.

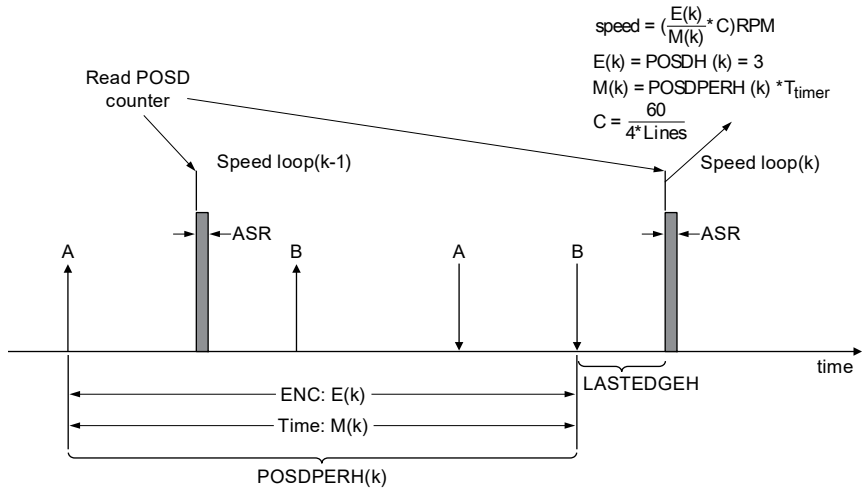


Figure 126. High speed measurement with method

Speed calculation algorithm

In order to cover all the cases in real applications, a simple speed measurement algorithm is necessary. See the algorithm shown below.

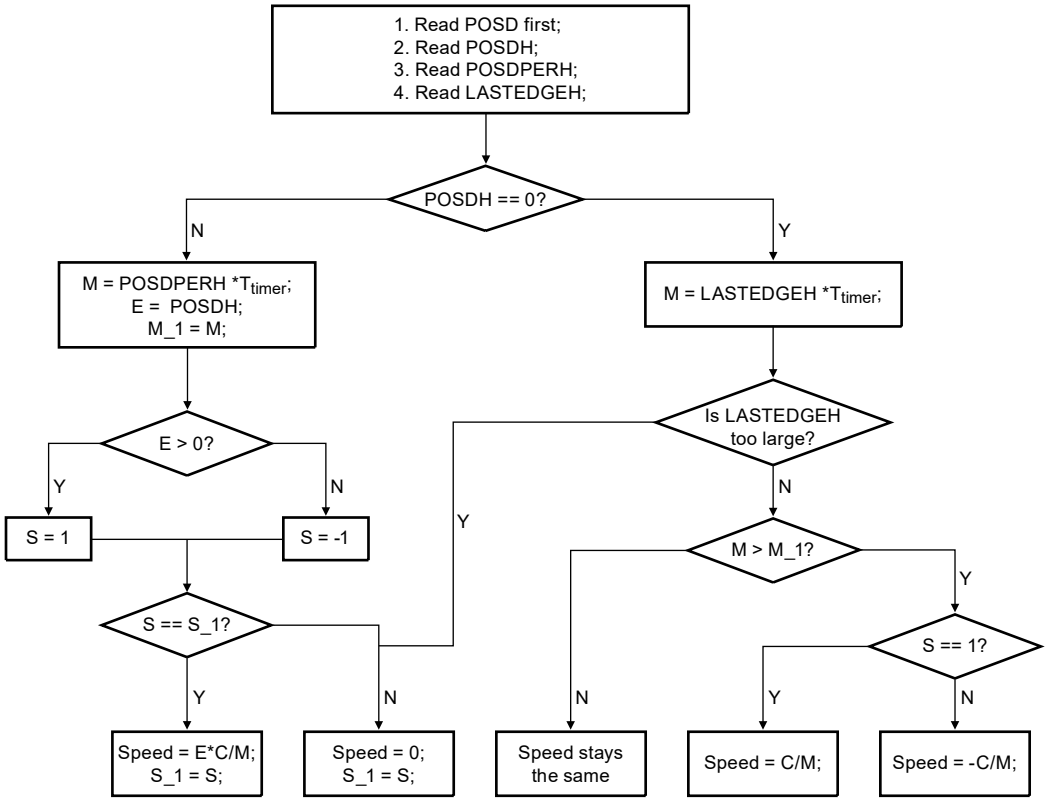


Figure 127. Flowchart of speed calculation with method

Speed calculation of motor's starting from standstill (chip is out of reset)

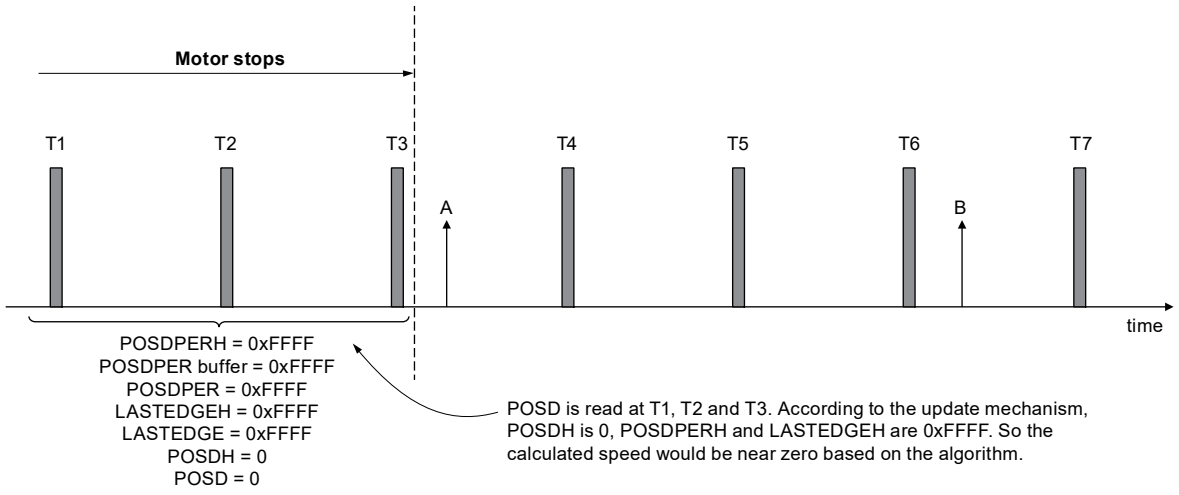


Figure 128. Speed measurement at time point T1~T3

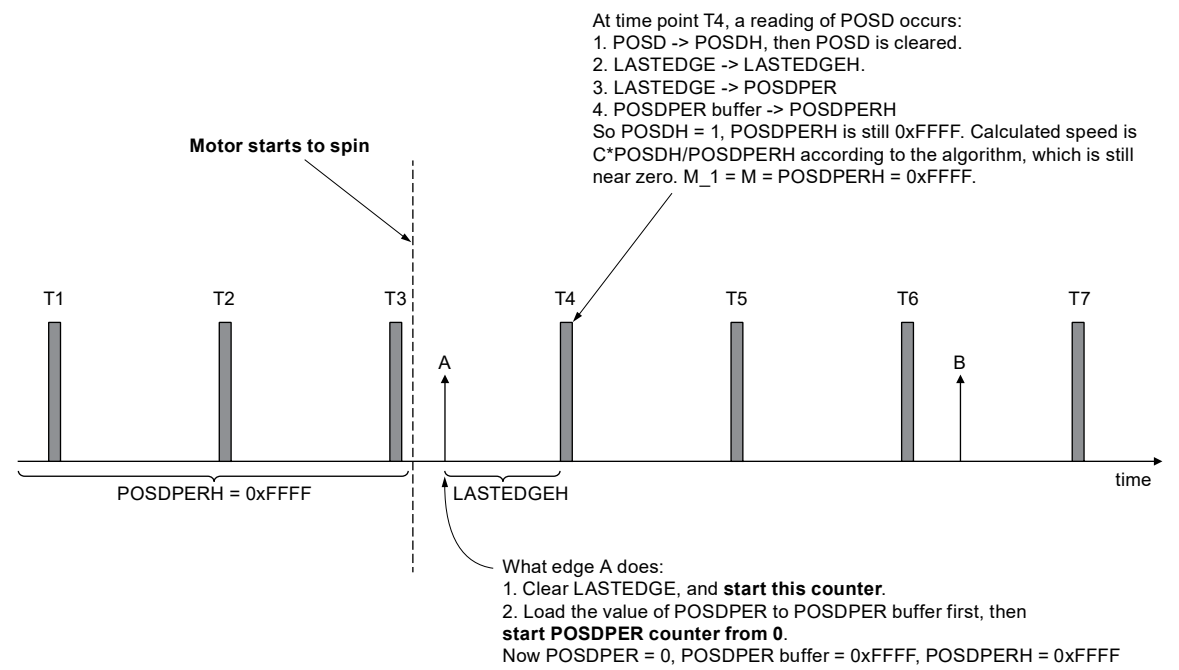


Figure 129. Speed measurement at time point T4

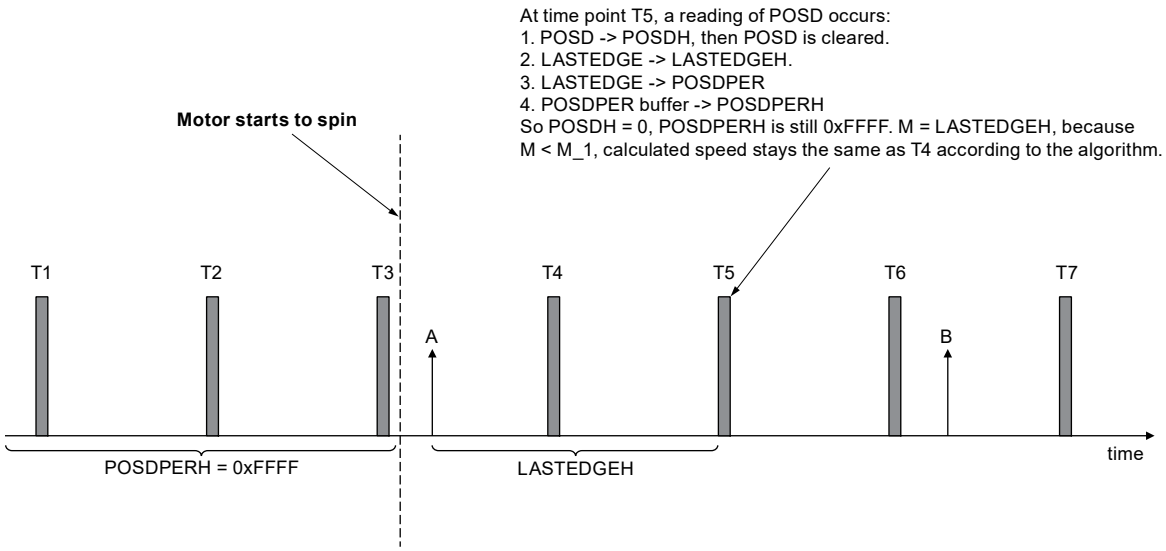


Figure 130. Speed measurement at time point T5

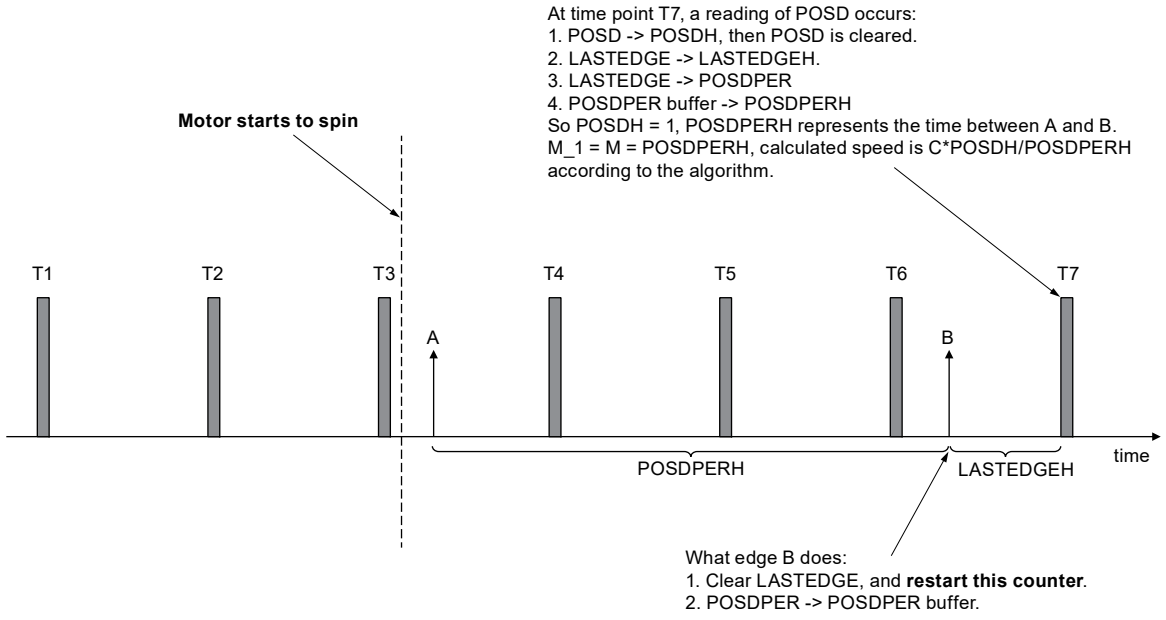


Figure 131. Speed measurement at time point T7

Speed calculation when motor stops

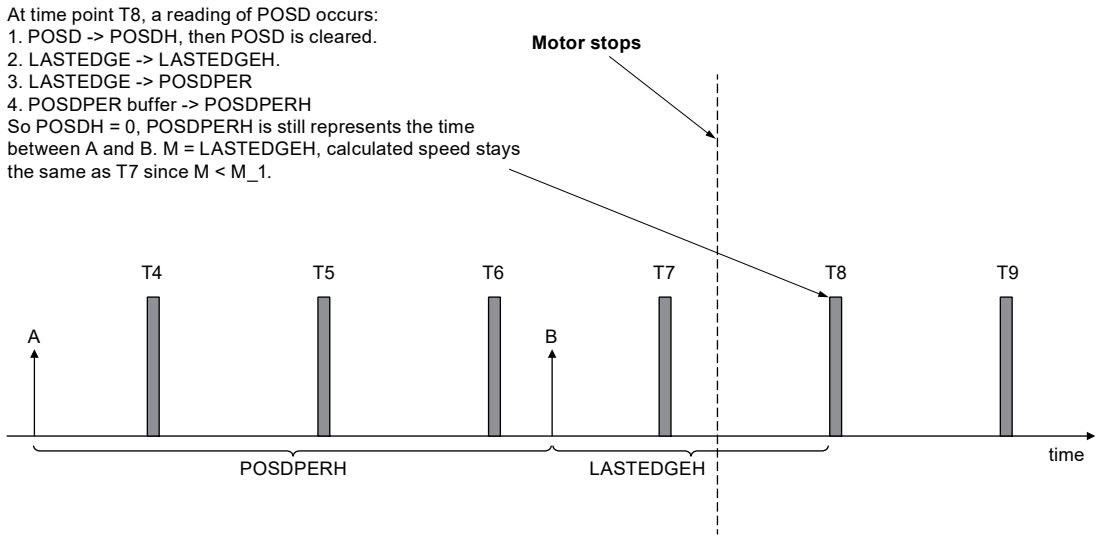


Figure 132. Speed measurement at time point T8

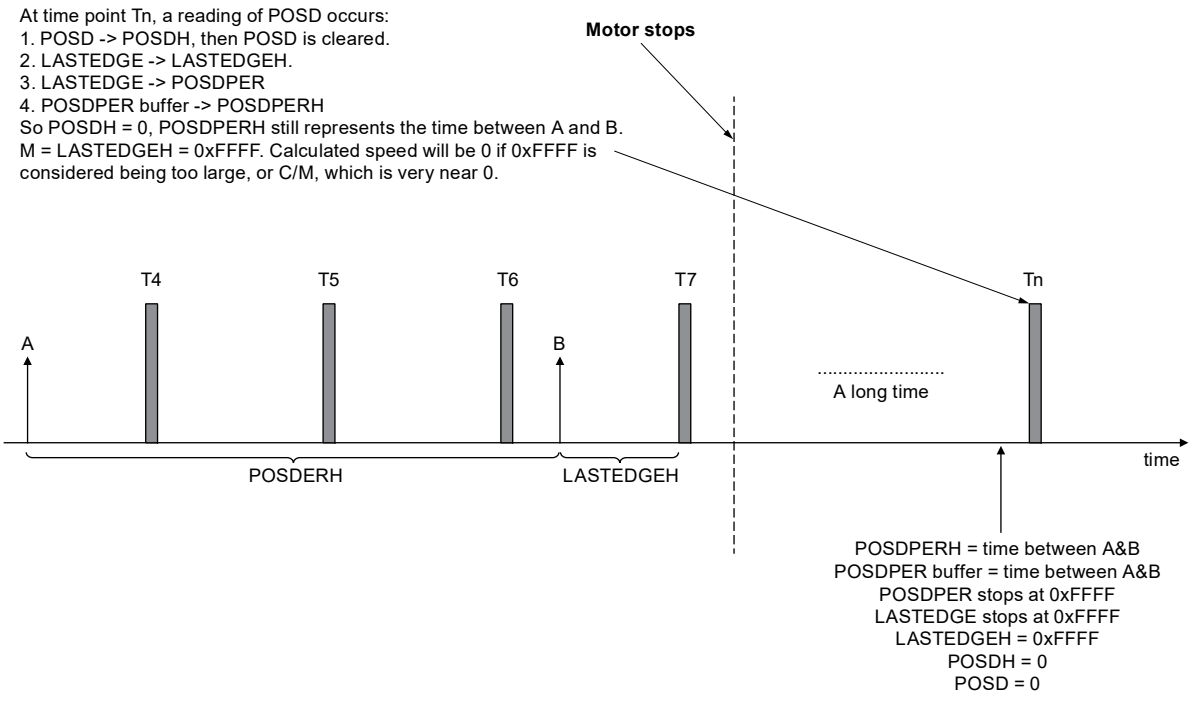


Figure 133. Speed measurement at time point Tn

Speed calculation when motor starts again

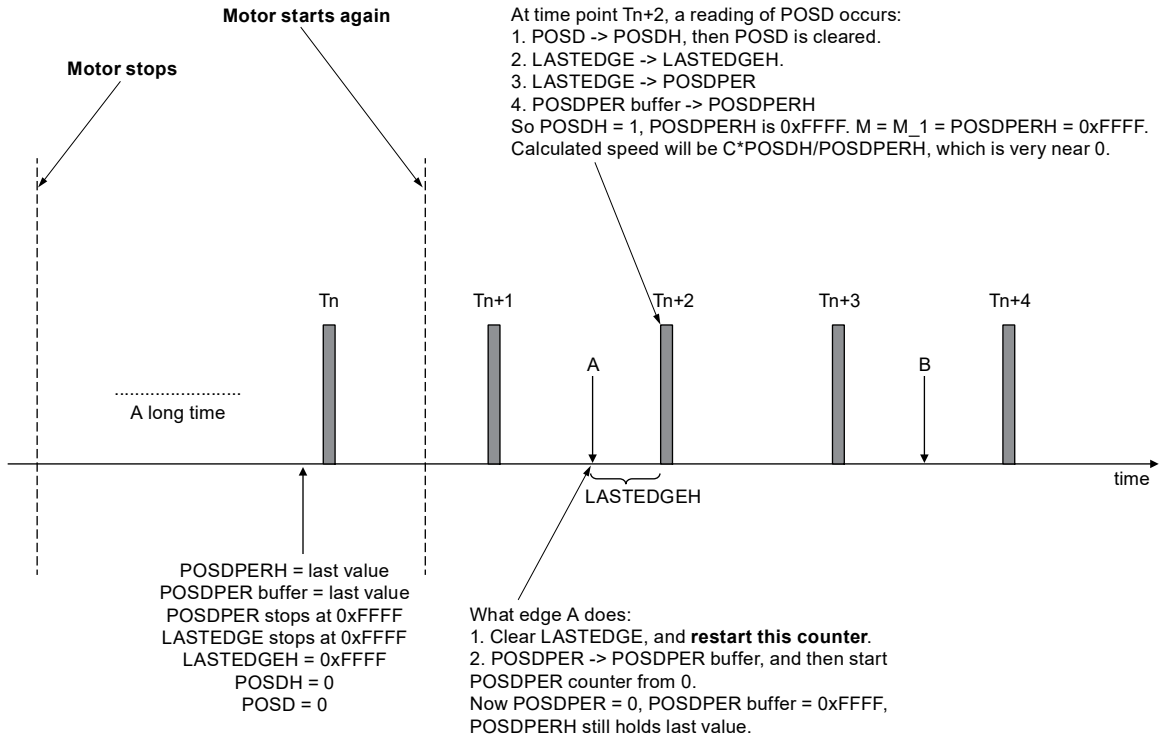


Figure 134. Speed measurement at time point Tn+2

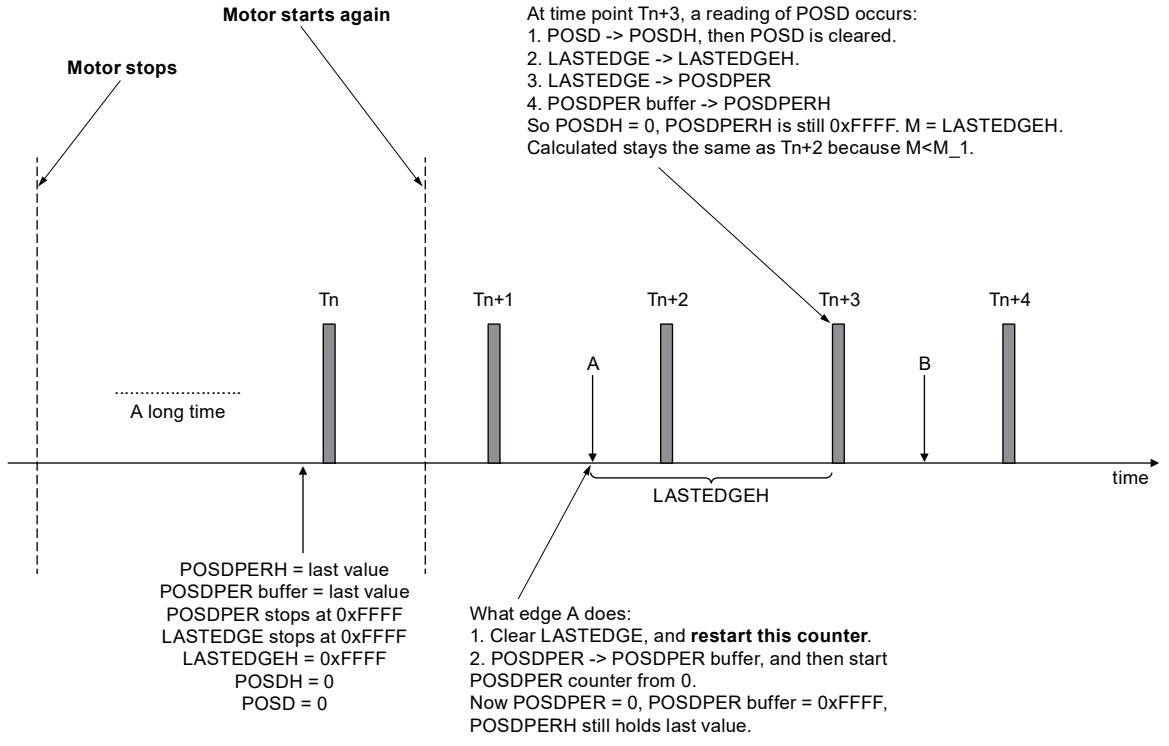


Figure 135. Speed measurement at time point Tn+3

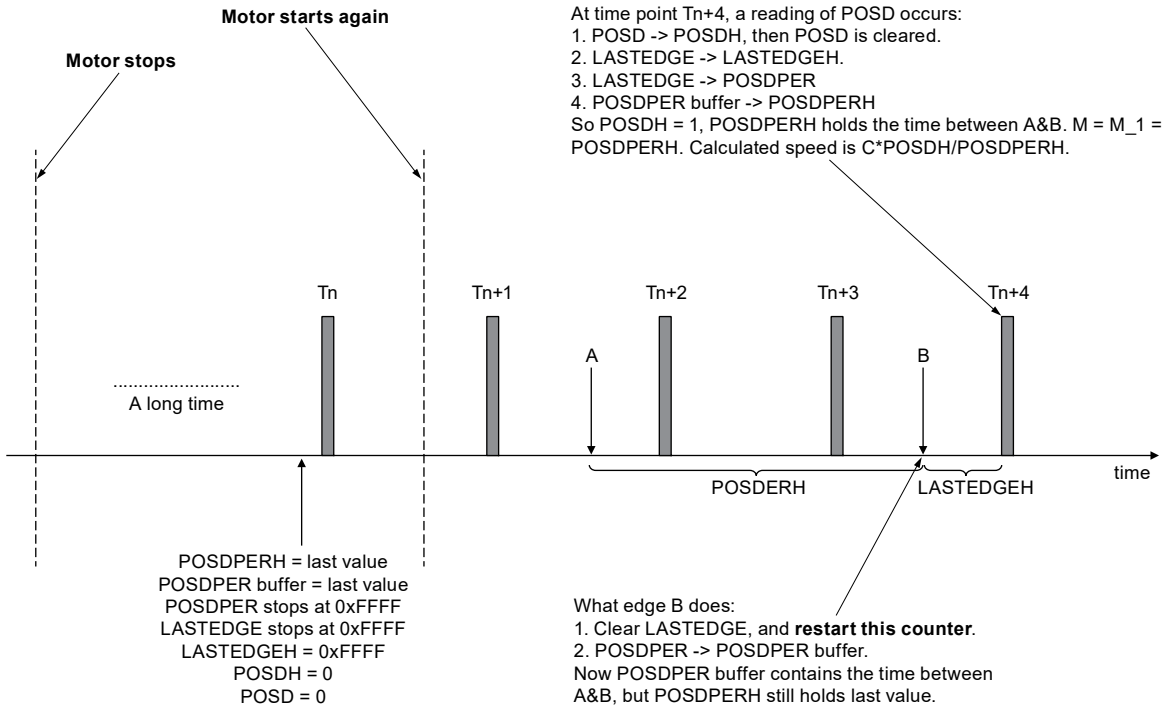


Figure 136. Speed measurement at time point Tn+4

32.3.9 Reset

eQDC is reset by any system reset. Make sure you clear all inputs (of PhaseA, PhaseB, and etc.) before resetting eQDC.

32.3.10 Clocking

The peripheral clock is the only clock required when eQDC operates.

32.3.11 Interrupts

The following table lists the module interrupts.

Table 176. Interrupt Summary

Core Interrupt	Interrupt Flag	Interrupt Enable	Description
ipi_int_home	CTRL[HIRQ]	CTRL[HIE]	HOME/ENABLE signal transition interrupt
ipi_int_index	CTRL[XIRQ]	CTRL[XIE]	INDEX/PRESET signal transition interrupt
ipi_int_ro	INTCTRL[ROIRQ]	INTCTRL[ROIE]	roll-over interrupt
ipi_int_ru	INTCTRL[RUIRQ]	INTCTRL[RUIE]	roll-under interrupt
ipi_int_wdog	CTRL[WDIRQ]	CTRL[WDIE]	Watchdog timeout interrupt
ipi_int_comp	INTCTRL[CMP0IRQ] ~ INTCTRL[CMP3IRQ]	INTCTRL[CMP0IE] ~ INTCTRL[CMP3IE]	Compare match interrupt

Table continues on the next page...

Table 176. Interrupt Summary (continued)

Core Interrupt	Interrupt Flag	Interrupt Enable	Description
ipi_int_dir	INTCTRL[DIRIRQ]	INTCTRL[DIRIE]	Count direction change interrupt
ipi_int_sab	INTCTRL[SABIRQ]	INTCTRL[SABIE]	Simultaneous PHASEA and PHASEB change interrupt

32.4 External signals

eQDC receives the following 8 input signals from position or speed sensors:

- PHASEA
- PHASEB
- INDEX/PRESET
- TRIGGER
- HOME/ENABLE
- ICAP[3:1]

eQDC outputs the following 11 signals for system use:

- POS_MATCH[3:0]
- COMP_FLG[3:0]
- DIR
- CNT_DN
- CNT_UP

Table 177. Signals

Signal			Description
PHASEA	Phase A	Input	The PHASEA input can be connected to one of the phases from a two-phase shaft quadrature encoder output. PHASEA and PHASEB are used by eQDC to indicate a decoder increment has passed, and to calculate its direction. • When eQDC is used as a single phase decode/count, PHASEA can also be used as the single input. Direction for two-phase shaft quadrature encoder output when CTRL[REV] = 0: • PHASEA is the leading phase for a shaft rotating in the positive direction. • PHASEA is the trailing phase for a shaft rotating in the negative direction.
PHASEB	Phase B	Input	The PHASEB input can be connected to the other phase from a two-phase shaft quadrature encoder output. • When eQDC is used as a single phase decode/count mode, PHASEB and CTRL[REV] control counter direction. Direction for two-phase shaft quadrature encoder output when CTRL[REV] = 0: • PHASEB is the trailing phase for a shaft rotating in the positive direction. • PHASEB is the leading phase for a shaft rotating in the negative direction.

Table continues on the next page...

Table 177. Signals (continued)

Signal			Description
INDEX/ PRESET	INDEX/ PRESET	Input	• This input can work as INDEX or PRESET, selection bit is CTRL2[OPMODE] • Normally connected to the index pulse output of a quadrature encoder, the INDEX pulse can optionally reset the position counter of eQDC. The INDEX pulse also causes a change of state on the revolution counter. The direction of this state change (increment or decrement) is calculated from the PHASEA and PHASEB inputs. In Quadrature Count mode or Single Phase Count mode, this signal can present the counter PRESET function. • The positive edge of PRESET input can initialize position/revolution/ watchdog counters.
HOME/ENABLE	HOME/ENABLE	Input	The HOME input can be used by eQDC and the timer module. • This input can work as HOME or ENABLE, selection bit is CTRL2[OPMODE] • The HOME input can be used to trigger the initialization of the position counter (UPOS/LPOS). Often the HOME signal is connected to a sensor on the motor or machine, sending notification that it has reached a defined home position. In Quadrature Count mode or Single Phase Count mode, this signal can present the counter ENABLE function. • All counters start counting when ENABLE is asserted,all counters stop counting when ENABLE is de-asserted.
TRIGGER	TRIGGER	Input	• Initializes UPOS or LPOS. • Takes a snapshot of POS, REV, and POSD.
ICAP[3:1]	Input capture	Input	Takes snapshots of POS and stores them into the corresponding UPOSH/LPOSH: • Positive edge of ICAP[3] input takes snapshots of POS and stores it into UPOSH3/LPOSH3. • Positive edge of ICAP[2] input takes snapshots of POS and stores it into UPOSH2/LPOSH2. • Positive edge of ICAP[1] input takes snapshots of POS and stores it into UPOSH1/LPOSH1.
POS_MATCH[3 :0]	Position Match	Output	• Records the time at which the position of the shaft matches a user-defined compare value (UCOMPx/LCOMPx), the POS_MATCH[x](x range is 0-3) output can be used to trigger a timer channel. • Alternatively, records the time at which the position information was read, (when the position counters (UPOS and LPOS), revolution counter (REV), or position difference counter (POSD) registers are read) the POS_MATCH[x](x range is 0-3) output can be used to trigger a timer channel.

Table continues on the next page...

Table 177. Signals (continued)

Signal			Description
			Note: • If CTRL2[OUTCTL] is cleared, POS_MATCH[x](x range is 0-3) is asserted when the value of POS equals the value of the corresponding compare register (UCOMPx/LCOMPx), and de-asserted when the value of POS does not equal the value of the corresponding compare register (UCOMPx/LCOMPx). • If CTRL2[OUTCTL] is set, POS_MATCH[x](x range is 0-3) is asserted a pulse with 1 peripheral clock cycle width when UPOS or LPOS, REV, or POSD is read.
COMP_FLG[3:0]	Position Counter Compare Output	Output	• COMP_FLG[x](x range is 0-3) is asserted when the value of POS is equal to or greater than the value of the corresponding compare register (UCOMPx/LCOMPx). • COMP_FLG[x](x range is 0-3) is de-asserted when the value of POS is smaller than the value of the corresponding compare register (UCOMPx/LCOMPx).
DIR	Direction of position counting	Output	Indicates the direction of position counting: • 1 indicates up. • 0 indicates down.
CNT_UP	Position counter count up	Output	When eQDC decodes a count-up event, CNT_UP outputs a pulse with a width of 1 peripheral clock cycle.
CNT_DN	position counter count down	Output	When eQDC decodes a count-down event, CNT_DN outputs a pulse with a width of 1 peripheral clock cycle.

32.5 Initialization

This module does not require initialization.

32.6 Register descriptions

This section describes the memory map and registers of eQDC.

NOTE

Writing to a read-only (RO) register will cause bus errors.

32.6.1 Quadrature_Decoder register descriptions

The address of a register is the sum of a base address and an address offset.

32.6.1.1 Quadrature_Decoder memory map

eQDC0 base address: 400A_7000h

eQDC1 base address: 400A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register (CTRL)	16	RW	0000h
2h	Control 2 Register (CTRL2)	16	RW	0000h
4h	Input Filter Register (FILT)	16	RW	0000h
6h	Last Edge Time Register (LASTEDGE)	16	R	FFFFh
8h	Position Difference Period Counter Register (POSDPER)	16	R	FFFFh
Ah	Position Difference Period Buffer Register (POSDPERBFR)	16	R	FFFFh
Ch	Upper Position Counter Register (UPOS)	16	RW	0000h
Eh	Lower Position Counter Register (LPOS)	16	RW	0000h
10h	Position Difference Counter Register (POSD)	16	RW	0000h
12h	Position Difference Hold Register (POSDH)	16	R	0000h
14h	Upper Position Hold Register (UPOSH)	16	R	0000h
16h	Lower Position Hold Register (LPOSH)	16	R	0000h
18h	Last Edge Time Hold Register (LASTEDGEH)	16	R	FFFFh
1Ah	Position Difference Period Hold Register (POSDPERH)	16	R	FFFFh
1Ch	Revolution Hold Register (REVH)	16	R	0000h
1Eh	Revolution Counter Register (REV)	16	RW	0000h
20h	Upper Initialization Register (UINIT)	16	RW	0000h
22h	Lower Initialization Register (LINIT)	16	RW	0000h
24h	Upper Modulus Register (UMOD)	16	RW	0000h
26h	Lower Modulus Register (LMOD)	16	RW	0000h
28h	Upper Position Compare Register 0 (UCOMP0)	16	RW	8000h
2Ah	Lower Position Compare Register 0 (LCOMP0)	16	RW	0000h
2Ch	Upper Position Compare 1 (UCOMP1)	16	W	8000h
2Ch	Upper Position Holder Register 1 (UPOSH1)	16	R	0000h
2Eh	Lower Position Compare 1 (LCOMP1)	16	W	0000h
2Eh	Lower Position Holder Register 1 (LPOSH1)	16	R	0000h
30h	Upper Position Compare 2 (UCOMP2)	16	W	8000h
30h	Upper Position Holder Register 3 (UPOSH2)	16	R	0000h
32h	Lower Position Compare 2 (LCOMP2)	16	W	0000h
32h	Lower Position Holder Register 2 (LPOSH2)	16	R	0000h
34h	Upper Position Compare 3 (UCOMP3)	16	W	8000h
34h	Upper Position Holder Register 3 (UPOSH3)	16	R	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
36h	Lower Position Compare 3 (LCOMP3)	16	W	0000h
36h	Lower Position Holder Register 3 (LPOSH3)	16	R	0000h
38h	Interrupt Control Register (INTCTRL)	16	RW	0000h
3Ah	Watchdog Timeout Register (WTR)	16	RW	0000h
3Ch	Input Monitor Register (IMR)	16	RW	0000h
3Eh	Test Register (TST)	16	RW	0000h
50h	Upper VERID (UVERID)	16	R	0001h
52h	Lower VERID (LVERID)	16	R	0001h

32.6.1.2 Control Register (CTRL)

Offset

Register	Offset
CTRL	0h

Function

Controls various features of eQDC.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIRQ	HIE	HIP	HNE	0	REV	PH1	XIRQ	XIE	XIP	XNE	WDIRQ	WDIE	WDE	DMAEN	LDOK
W	W1C				SWIP			W1C				W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 HIRQ	HOME/ENABLE Signal Transition Interrupt Request HOME/ENABLE Signal Transition Interrupt Request is set when a positive or negative transition on the HOME/ENABLE signal occurs, according to the Use Negative Edge of HOME/ENABLE Input bit (CTRL[HNE]). If HOME/ENABLE Signal Transition Interrupt Request bit is set and HOME/ENABLE Interrupt Enable bit (CTRL[HIE]) is set, then a HOME/ENABLE interrupt occurs. • HOME/ENABLE Signal Transition Interrupt Request bit remains set until it is cleared by software

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Write a one to this bit (HIRQ) to clear it 0b - No transition on the HOME/ENABLE signal has occurred 1b - A transition on the HOME/ENABLE signal has occurred
14 HIE	HOME/ENABLE Interrupt Enable Enables/disables HOME/ENABLE signal interrupts. 0b - Disabled 1b - Enabled
13 HIP	Enable HOME to Initialize Position Counter UPOS/LPOS Enables the position counter to be initialized by the HOME signal. 0b - No action 1b - HOME signal initializes the position counter
12 HNE	Use Negative Edge of HOME/ENABLE Input Selects using the positive/negative edge of the HOME/ENABLE input. 0b - When CTRL[OPMODE] = 0,use HOME positive edge to trigger initialization of position counters. When CTRL[OPMODE] = 1,use ENABLE high level to enable POS/POSD/WDG/REV counters 1b - When CTRL[OPMODE] = 0,use HOME negative edge to trigger initialization of position counters. When CTRL[OPMODE] = 1,use ENABLE low level to enable POS/POSD/WDG/REV counters
11 SWIP	Software-Triggered Initialization of Position Counters UPOS and LPOS Writing 1 to Software-Triggered Initialization of Position Counters bit initializes the position counter. • Software-Triggered Initialization of Position Counters bit is always read as 0. 0b - No action 1b - Initialize position counter
10 REV	Enable Reverse Direction Counting Selects the direction of the count and position counter initial value. Note: if CTRL2[EMIP]=1 and INDEX input transition initialize position counter, this REV bit affects position counter initial value in a different manner, please refer to chapter "Position Counter(POS)" in Functional Description. Otherwise, REV bit takes affect as following: 0b - Count normally and the position counter initialization uses upper/lower initialization register UINIT/LINIT 1b - Count in the reverse direction and the position counter initialization uses upper/lower modulus register UMOD/LMOD
9	Enable Single Phase Mode

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>PH1</td><td>Bypass/use the quadrature decoder logic.
0b - Standard quadrature decoder, where PHASEA and PHASEB represent a two-phase quadrature signal.
1b - Single phase mode, bypass the quadrature decoder, refer to CTRL2[CMODE] description</td></tr>
<tr><td>8
XIRQ</td><td>INDEX/PRESET Pulse Interrupt Request
INDEX/PRESET Pulse Interrupt Request is set when a transition on the INDEX/PRESET signal occurs, according to the Use Negative Edge of INDEX/PRESET Pulse bit (CTRL[XNE]). If INDEX/PRESET Pulse Interrupt Request bit is set and INDEX/PRESET Pulse Interrupt Enable bit (CTRL[XIE]) is set, then an INDEX/PRESET interrupt occurs.
• INDEX/PRESET Pulse Interrupt Request remains set until cleared by software
• Write a one to this bit (XIRQ) to clear it
0b - INDEX/PRESET pulse has not occurred
1b - INDEX/PRESET pulse has occurred</td></tr>
<tr><td>7
XIE</td><td>INDEX/PRESET Pulse Interrupt Enable
Enables/disables INDEX/PRESET pulse interrupts.
0b - Disabled
1b - Enabled</td></tr>
<tr><td>6
XIP</td><td>INDEX Triggered Initialization of Position Counters UPOS and LPOS
Enables/disables the position counter to be initialized by the INDEX pulse.
0b - INDEX pulse does not initialize the position counter
1b - INDEX pulse initializes the position counter</td></tr>
<tr><td>5
XNE</td><td>Select Positive/Negative Edge of INDEX/PRESET Pulse
Selects the positive/negative edge of the INDEX/PRESET pulse.
Note: if CTRL2[EMIP]=1 and CTRL2[OPMODE]=0, XNE bit affects INDEX pulse in a different manner as following table:
<table>
<tr><th></th><th>CTRL[XNE] = 0</th><th>CTRL[XNE] = 1</th></tr>
<tr><td>PHA leads PHB (Clockwise)</td><td>Use rising edge of Index</td><td>Use falling edge of Index</td></tr>
<tr><td>PHA lags PHB
(Counter Clockwise)</td><td>Use falling edge of Index</td><td>Use rising edge of Index</td></tr>
</table>
otherwise, XNE bit affects INDEX pulse as following:
0b - Use positive edge of INDEX/PRESET pulse
1b - Use negative edge of INDEX/PRESET pulse</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 WDIRQ	Watchdog Timeout Interrupt Request Watchdog Timeout Interrupt Request is set when a watchdog timeout interrupt occurs. • Watchdog Timeout Interrupt Request remains set until cleared by software • Write a one to this bit (WDIRQ) to clear it • Watchdog Timeout Interrupt Request bit is also cleared when Watchdog Enable (CTRL[WDE]) is disabled (=0) 0b - No Watchdog timeout interrupt has occurred 1b - Watchdog timeout interrupt has occurred
3 WDIE	Watchdog Timeout Interrupt Enable Enables/disables watchdog timeout interrupts. 0b - Disabled 1b - Enabled
2 WDE	Watchdog Enable Enables/disables the watchdog timer that is monitoring the PHASEA and PHASEB inputs for motor movement. 0b - Disabled 1b - Enabled
1 DMAEN	DMA Enable This bit enables DMA for new written buffer values of COMPx/INIT/MOD(x range is 0-3) Please note: When set to 1, this bit can't be cleared by writing 0 to it. 0b - DMA is disabled 1b - DMA is enabled. DMA request asserts automatically when the values in the outer-set of buffered compare registers (UCOMP0/LCOMP0;UCOMP1/LCOMP1;UCOMP2/LCOMP2;UCOMP3/LCOMP3), initial registers(UINIT/LINIT) and modulus registers (UMOD/LMOD) are loaded into the inner-set of buffer and then LDOK is cleared automatically. After the completion of this DMA transfer, LDOK is set automatically, it ensures outer-set values can be loaded into inner-set which in turn triggers DMA again.
0 LDOK	Load Okay This bit enables that the outer-set values of buffered compare registers (UCOMPx/LCOMPx, x=0~3), initial register(UINIT/LINIT) and modulus register(UMOD/LMOD) can be loaded into their inner-sets and take effect. When LDOK is set, this loading action occurs at the next position counter roll-over or roll-under if CTRL2[LDMOD] is set, or it occurs immediately if CTRL2[LDMOD] is cleared. LDOK is automatically cleared after the values in outer-set is loaded into the inner-set. Set LDOK bit by reading it, then write a logic 1 to it. LDOK can be manually cleared before a loading action occurs by writing a logic 0 to it. The outer-set of the registers cannot be written while LDOK is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No loading action taken. Users can write new values to buffered registers (writing into outer-set of these buffered registers) 1b - Outer-set values are ready to be loaded into inner-set and take effect. The loading time point depends on CTRL2[LDMOD].

32.6.1.3 Control 2 Register (CTRL2)

Offset

Register	Offset
CTRL2	2h

Function

Controls various features of eQDC.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMODE		ONCE	INITPOS	EMIP	PMEN	OUTCTL	REVMOD	0				LDMOD	OPMODE	UPDPOS	UPDHLD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 CMODE	Counting Mode These bits control the basic counting and behavior of Position Counter and Position Difference Counter. Setting CTRL[REV] to 1 can reverse the counting direction. In quadrature Mode (CTRL[PH1] = 0): 00b - CM0: Normal/Reverse Quadrature X4 01b - CM1: Normal/Reverse Quadrature X2 10b - CM2: Normal/Reverse Quadrature X1 11b - CM3: Reserved In Single Phase Mode (CTRL[PH1] = 1): 00b - CM0: UP/DOWN Pulse Count Mode 01b - CM1: Signed Mode, count PHASEA rising/falling edge, position counter counts up when PHASEB is low and counts down when PHASEB is high

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>10b - CM2: Signed Count Mode,count PHASEA rising edge only, position counter counts up when PHASEB is low and counts down when PHASEB is high
11b - CM3: Reserved</td></tr>
<tr><td>13
ONCE</td><td>Count Once
This bit selects modulo loop or one shot counting mode.
NOTE
Any of the following event initializes and restarts the position counter:
• Write to CTRL[SWIP].
• INDEX signal transition if CTRL2[OPMODE] bit is clear and CTRL[XIP] is set.
• HOME signal transition if CTRL2[OPMODE] bit is clear and CTRL[HIP] is set.
• PRESET signal transition if CTRL2[OPMODE] bit is set and ENABLE signal is 1.
• Positive edge transition of TRIGGER input if CTRL2[INITPOS] is set.
0b - Position counter counts repeatedly
1b - Position counter counts until roll-over or roll-under, then stop.</td></tr>
<tr><td>12
INITPOS</td><td>Initial Position Register
• When Initial Position Register is set (=1), it allows the rising edge of TRIGGER input to initialize the position counter (UPOS and LPOS registers) to UINIT/LINIT values (if CTRL[REV]=0) or UMOD/LMOD values (if CTRL[REV]=1).
• When Initial Position Register is clear (=0), the position counter (UPOS/LPOS) is not initialized on rising edge of TRIGGER input
0b - Don't initialize position counter on rising edge of TRIGGER
1b - Initialize position counter on rising edge of TRIGGER</td></tr>
<tr><td>11
EMIP</td><td>Enables/disables the position counter to be initialized by Index Event Edge Mark
When CTRL[XIP] =1, this bit is used to enable the position counter to be initialized by Index Event Edge Mark cooperated with CTRL[REV] and CTRL[XNE] as following table:
<table>
<tr><th></th><th>CTRL[XNE] = 0</th><th>CTRL[XNE] = 1</th><th>CTRL[REV] = 0</th><th>CTRL[REV] = 1</th></tr>
<tr><td>PHA leads PHB (Clockwise)</td><td>INDEX rising edge reset position counter</td><td>INDEX falling edge reset position counter</td><td>Reset position counter to initial value</td><td>Reset position counter to modulus value</td></tr>
<tr><td>PHA lags PHB (Counter Clockwise)</td><td>INDEX falling edge reset position counter</td><td>INDEX rising edge reset position counter</td><td>Reset position counter to modulus value</td><td>Reset position counter to initial value</td></tr>
</table>
0b - disables the position counter to be initialized by Index Event Edge Mark
1b - enables the position counter to be initialized by Index Event Edge Mark.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 PMEN	Period measurement function enable This bit is used to indicate the period measurement functions of LASTEDGE, LASTEDGEH, POSDPER, POSDPERBFR, and POSDPERH are being used. 0b - Period measurement functions are not used. POSD is loaded to POSDH and then cleared whenever POSD, UPOS, LPOS or REV is read. 1b - Period measurement functions are used. POSD is loaded into POSDH and then cleared only when POSD is read.
9 OUTCTL	Output Control Output Control controls the pulsing of the POS_MATCH output signal. 0b - POS_MATCH[x](x range is 0-3) is asserted when the Position Counter is equal to according compare value (UCOMPx/LCOMPx)(x range is 0-3), and de-asserted when the Position Counter not equal to the compare value (UCOMPx/LCOMPx)(x range is 0-3) 1b - All POS_MATCH[x](x range is 0-3) are asserted a pulse, when the UPOS, LPOS, REV, or POSD registers are read
8 REVMOD	Revolution Counter Modulus Enable Revolution Counter Modulus Enable selects how the revolution counter (REV) is incremented/ decremented. By default, the revolution counter is controlled based on the count direction and the INDEX pulse. As an option, the revolution counter can be controlled using the roll-over/under detection during modulo counting. 0b - Use INDEX pulse to increment/decrement revolution counter (REV) 1b - Use modulus counting roll-over/under to increment/decrement revolution counter (REV)
7-4 —	Reserved
3 LDMOD	Buffered Register Load (Update) Mode Select This bit selects the loading time point of the buffered compare registers UCOMPx/LCOMPx, x=0~3, initial register (UINIT/LINIT), and modulus register (UMOD/LMOD) 0b - Buffered registers are loaded and take effect immediately upon CTRL[LDOK] is set. 1b - Buffered registers are loaded and take effect at the next roll-over or roll-under if CTRL[LDOK] is set.
2 OPMODE	Operation Mode Select 0b - Decode Mode: Input nodes INDEX/PRESET and HOME/ENABLE are assigned to function of INDEX and HOME. 1b - Count Mode: Input nodes INDEX/PRESET and HOME/ENABLE are assigned to functions of PRESET and ENABLE. In this mode: (1)only when ENABLE=1, all counters (position/ position difference/revolution/watchdog) can run, when ENABLE=0, all counters (position/position difference/revolution/watchdog) can't run. (2) the rising edge of PRESET input can initialize

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>position/revolution/watchdog counters (position counter initialization also need referring to bit CTRL[REV]).</td></tr>
<tr><td>1
UPDPOS</td><td>Update Position Registers
When UPDPOS bit is set (=1), it allows the rising edge of TRIGGER input to clear the POSD, REV, UPOS and LPOS registers to zero.
When UPDPOS bit is clear (=0), the rising edge of TRIGGER input does not clear POSD, REV, UPOS and LPOS registers to zero.
If both UPDPOS bit and INITPOS bit are set, then when rising edge of TRIGGER input occurs, the UPOS/LPOS registers are not cleared to zero, but be initialized to value of UINIT/LINIT registers(if CTRL[REV]=0) or value of UMOD/LMOD registers(if CTRL[REV]=1).
Do not set this bit when using the LASTEDGE and POSDPER registers to measure speed.</td></tr>
<tr><td>0
UPDHLD</td><td>Update Hold Registers
When UPDHLD bit is set (=1), it allows the rising edge of TRIGGER input to cause an update of the POSDH, REVH, UPOSH, and LPOSH registers
When UPDHLD bit is clear (=0), the hold registers (POSDH, REVH) are not updated by the rising edge of TRIGGER input, and hold registers UPOSH/LPOSH are not updated by the rising edge of TRIGGER input if INITPOS bit is also clear.
Do not set this bit when using the LASTEDGE and POSDPER registers to measure speed.
UPOSH/LPOSH has little difference with POSDH and REVH on UPDHLD bit, both UPDHLD bit and INITPOS bit allow the rising edge of TRIGGER input to take effect on UPOSH/LPOSH, so make a list here:
<table>
<tr><th>INITPOS</th><th>UPDHLD</th><th>State of UPOSH/LPOSH</th></tr>
<tr><td>0</td><td>0</td><td>Rising edge of TRIGGER input does not take effect on UPOSH/LPOSH</td></tr>
<tr><td>0</td><td>1</td><td>Rising edge of TRIGGER input updates UPOSH/LPOSH with the value of POS counter</td></tr>
<tr><td>1</td><td>0</td><td>Rising edge of TRIGGER input updates UPOSH/LPOSH with the value of POS counter, and then POS counter is initialized</td></tr>
<tr><td>1</td><td>1</td><td>Rising edge of TRIGGER input updates UPOSH/LPOSH with the value of POS counter, and then POS counter is initialized</td></tr>
</table>
Note:Updating the Position Difference Hold Register (POSDH) also causes the Position Difference Counter Register (POSD) to be cleared.</td></tr>
</table>

32.6.1.4 Input Filter Register (FILT)

Offset

Register	Offset
FILT	4h

Function

The Input Filter Register sets the values of the input filter sample period (FILT_PER) and the input filter sample count (FILT_CNT).

- The Input Filter Sample Period (FILT_PER) should be set so that the sampling period is larger than the period of the expected noise. Doing this means that a noise spike only corrupts one sample.
- The Input Filter Sample Count (FILT_CNT) should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of FILT_CNT+3.

The values of the Input Filter Sample Period (FILT_PER) and the Input Filter Sample Count (FILT_CNT) must also be traded off against the desire for minimal latency in recognizing valid input transitions. Turning on the input filter (setting the Input Filter Sample Period (FILT_PER) to a non-zero value) introduces a latency of ((FILT_CNT+3)*FILT_PER+2) IP Bus clock periods.

The filter latency can be measured:

- Drive the quadrature decoder inputs (PHASEA, PHASEB, INDEX, HOME)
- Monitor the filtered output in the Input Monitor Register (IMR)
- Determine how many IP Bus clock cycles that it takes before the output shows up, by using the following equations, where f is FILT_PER and s is FILT_CNT.
  - DELAY (IP Bus clock cycles) = f * (s+3) +1 (to read the filtered output)
  - DELAY (IP Bus clock cycles) = f * (s+3) +2 (to monitor the output in the IMR)

One more additional IP Bus clock cycle is needed to read the filtered output, and 2 more IP Bus clock cycles are needed to monitor the filtered output in the IMR. The sample rate is set when it reaches the number f. The following examples use the preceding equations:

- Example: when f = 0, the filter is bypassed: DELAY = 1 or 2 clock cycles.
- Example: when f = 5 and s = 2: DELAY = 5 * (2+3) + (1 or 2) = 26 or 27 clock cycles.

Diagram

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

Fields

Field	Function
15-12	Prescaler

Table continues on the next page...

Table continued from the previous page...

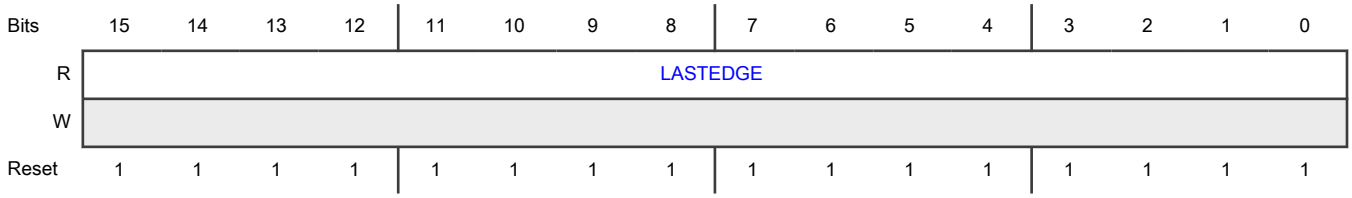
Field	Function
PRSC	The Prescaler field is used to prescale the peripheral clock that is used by the LASTEDGE and POSDPER counters as well as watchdog timer and Input Filter. The clock is prescaled by a value of 2^PRSC which means that the prescaler logic can divide the clock by a minimum of 1 and a maximum of 32,768.
11 FILT_CS	Filter Clock Source selection 0b - Peripheral Clock 1b - Prescaled peripheral clock by PRSC
10-8 FILT_CNT	Input Filter Sample Count The Input Filter Sample Count represents the number of consecutive samples that must agree, before the input filter accepts an input transition. • A value of 0x0 represents 3 samples • A value of 0x7 represents 10 samples The value of the Input Filter Sample Count (FILT_CNT) affects the input latency.
7-0 FILT_PER	Input Filter Sample Period The Input Filter Sample Period represents the sampling period (in IP Bus clock cycles) of the decoder input signals. Each input is sampled multiple times at the rate specified by the Input Filter Sample Period. • If FILT_PER is 0x00 (default), then the input filter is bypassed. Bypassing the digital filter enables the position/position difference counters to operate with count rates up to the IP Bus frequency. • The value of the Input Filter Sample Period (FILT_PER) affects the input latency. • When changing the Input Filter Sample Period (FILT_PER) from a non-zero value to another non-zero value, write a value of 0 first, to clear the filter.

32.6.1.5 Last Edge Time Register (LASTEDGE)

Offset

Register	Offset
LASTEDGE	6h

Diagram



Fields

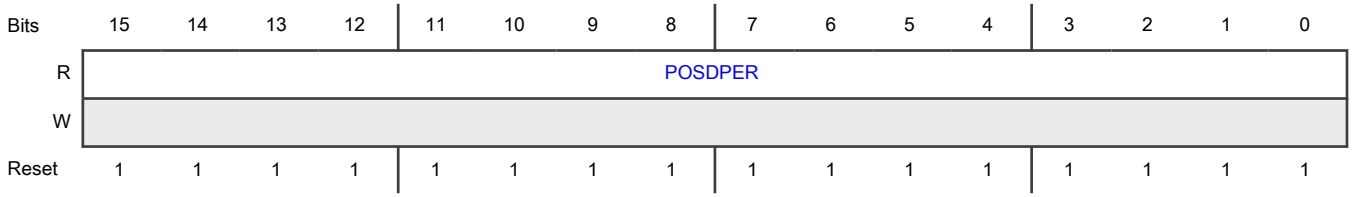
Field	Function
15-0 LASTEDGE	Last Edge Time Counter The Last Edge Time counter represents the time since the last edge occurred on PHASEA or PHASEB. LASTEDGE counts up using the peripheral clock that is prescaled by FILT[PRSC]. Any edge on PHASEA or PHASEB resets this register to 0 and starts counting. If the LASTEDGE count reaches 0xffff, the counting stops in order to prevent an overflow. Counting continues when an edge occurs on PHASEA or PHASEB.

32.6.1.6 Position Difference Period Counter Register (POSDPER)

Offset

Register	Offset
POSDPER	8h

Diagram



Fields

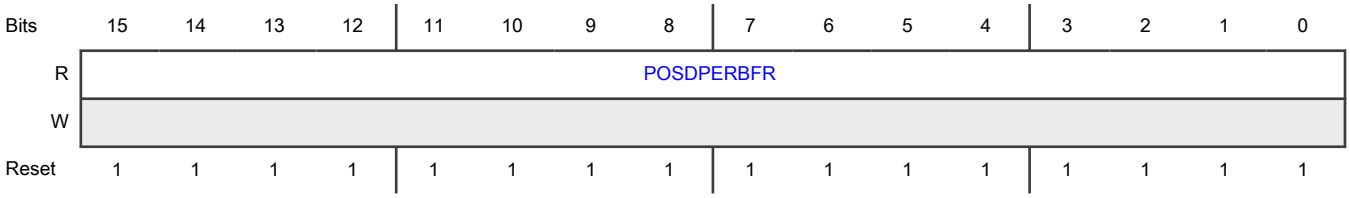
Field	Function
15-0 POSDPER	Position difference period The Position Difference Period counter counts up using the peripheral clock that is prescaled by FILT[PRSC]. Reading the POSD register also causes the value of the LASTEDGE register to be loaded into POSDPER. If the POSDPER count reaches 0xffff, the counting stops in order to prevent an overflow. Counting continues when an edge occurs on PHASEA or PHASEB.

32.6.1.7 Position Difference Period Buffer Register (POSDPERBFR)

Offset

Register	Offset
POSDPERBFR	Ah

Diagram



Fields

Field	Function
15-0 POSDPERBFR	Position difference period buffer The Position Difference Period buffer register is a buffer register for the POSDPER value. POSDPERBFR is updated with the value of POSDPER when any edge occurs on PHASEA or PHASEB.

32.6.1.8 Upper Position Counter Register (UPOS)

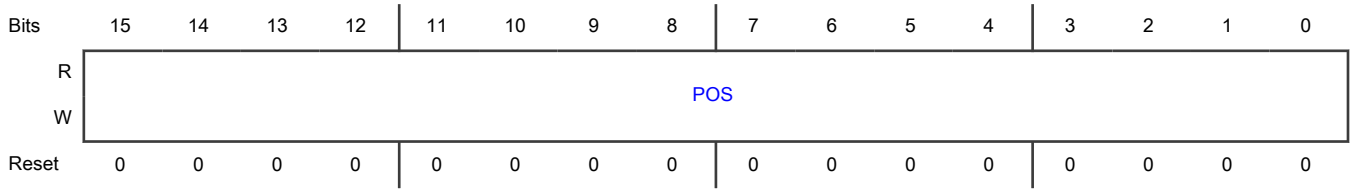
Offset

Register	Offset
UPOS	Ch

Function

Contains the upper (most significant) half of the Position Counter. This is the binary count from the Position Counter.

Diagram



Fields

Field	Function
15-0 POS	POS The upper (most significant) half of the Position Counter

32.6.1.9 Lower Position Counter Register (LPOS)

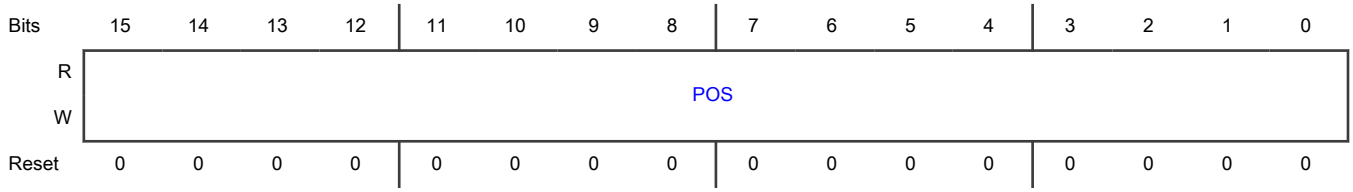
Offset

Register	Offset
LPOS	Eh

Function

Contains the lower (least significant) half of the Position Counter. This is the binary count from the Position Counter.

Diagram



Fields

Field	Function
15-0 POS	POS The lower (least significant) half of the Position Counter

32.6.1.10 Position Difference Counter Register (POSD)

Offset

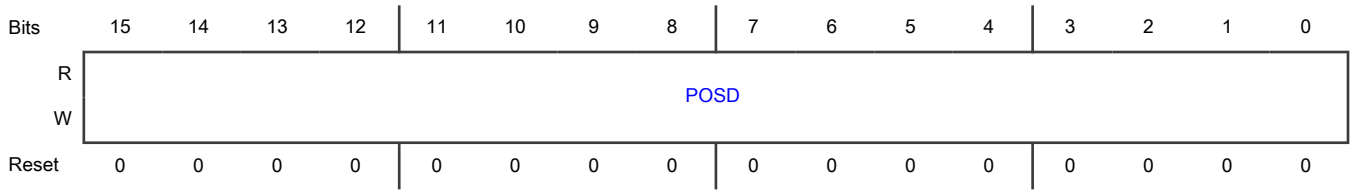
Register	Offset
POSD	10h

Function

The Position Difference Counter Register contains the position change in value occurring between each read of the Position Register. The value of the Position Difference Counter Register can be used to calculate velocity.

- The 16-bit Position Difference Counter computes up or down on every count pulse.
- This Position Difference Counter acts as a differentiator, whose count value is proportional to the change in position since the last time that the Position Counter was read.
- When the Position Difference Counter (POSD) is read, its contents are copied into the Position Difference Hold Register (POSDH) and the Position Difference Counter is cleared.

Diagram



Fields

Field	Function
15-0 POSD	POSD The position change in value between each read of the Position Register.

32.6.1.11 Position Difference Hold Register (POSDH)

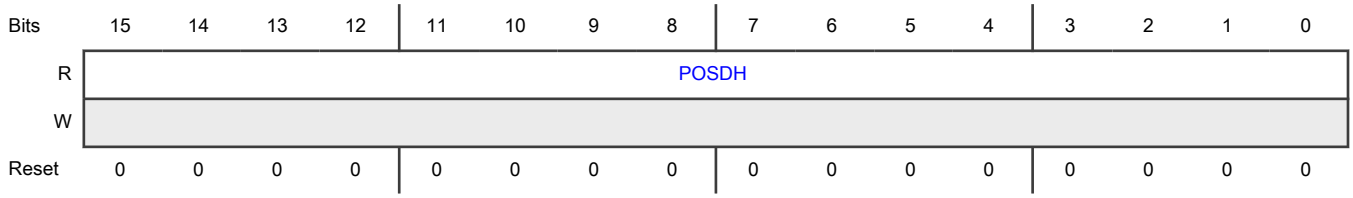
Offset

Register	Offset
POSDH	12h

Function

The Position Difference Hold Register register contains a snapshot of the value of the Position Difference Counter Register (POSD). The value of the Position Difference Hold Register (POSDH) can be used to calculate velocity.

Diagram



Fields

Field	Function
15-0 POSDH	POSDH The value of the POSD register

32.6.1.12 Upper Position Hold Register (UPOSH)

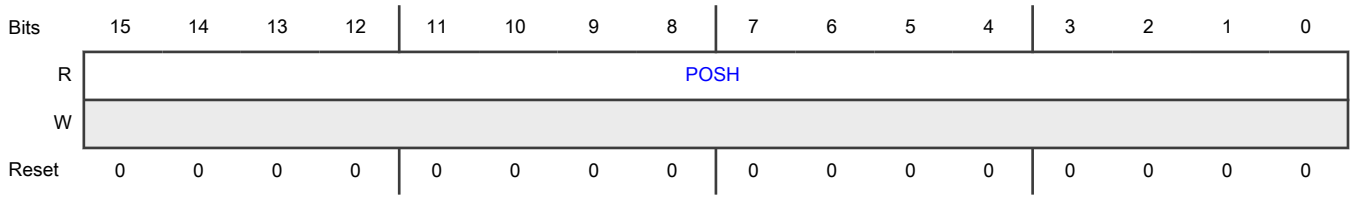
Offset

Register	Offset
UPOSH	14h

Function

Contains a snapshot of the Upper Position Counter Register (UPOS register).

Diagram



Fields

Field	Function
15-0 POSH	POSH The value of the Upper Position Counter Hold Register (UPOSH)

32.6.1.13 Lower Position Hold Register (LPOSH)

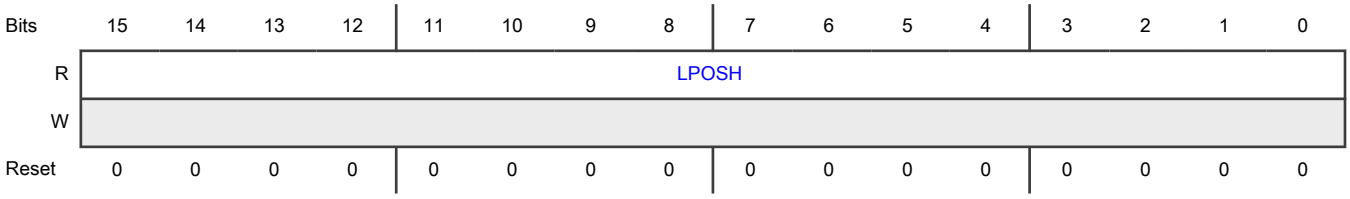
Offset

Register	Offset
LPOSH	16h

Function

Contains a snapshot of the Lower Position Counter Register (LPOS).

Diagram



Fields

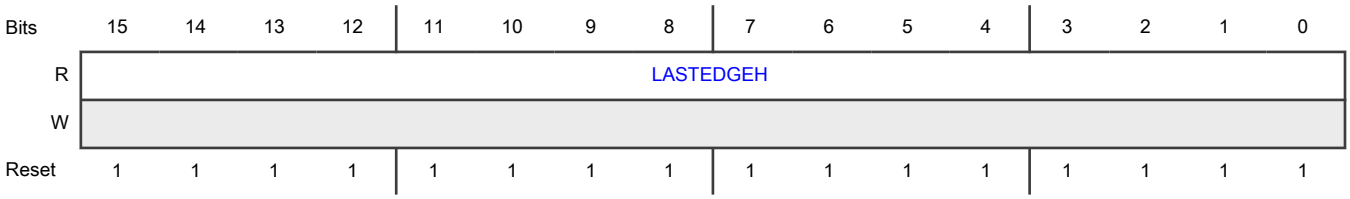
Field	Function
15-0 LPOSH	POSH The value of the Lower Position Counter Hold Register (LPOSH)

32.6.1.14 Last Edge Time Hold Register (LASTEDGEH)

Offset

Register	Offset
LASTEDGEH	18h

Diagram



Fields

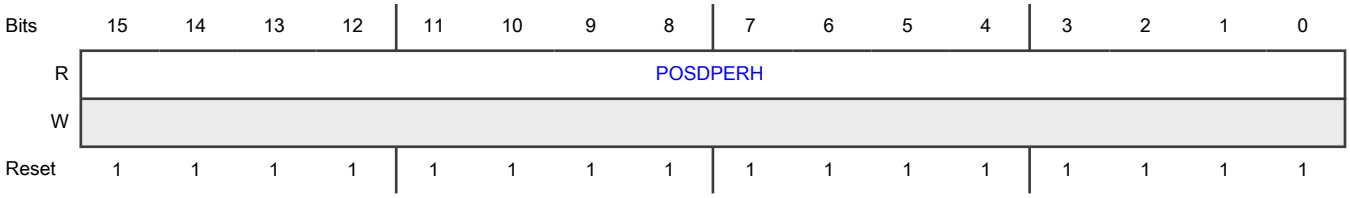
Field	Function
15-0 LASTEDGEH	Last Edge Time Hold The Last Edge Time Hold register is a hold register for the LASTEDGE value. LASTEDGEH is updated with the value of LASTEDGE when the POSD register is read.

32.6.1.15 Position Difference Period Hold Register (POSDPERH)

Offset

Register	Offset
POSDPERH	1Ah

Diagram



Fields

Field	Function
15-0 POSDPERH	Position difference period hold The Position Difference Period Hold register is a hold register for the POSDPER value. POSDPERH is updated with the value of POSDPERBFR when the POSD register is read.

32.6.1.16 Revolution Hold Register (REVH)

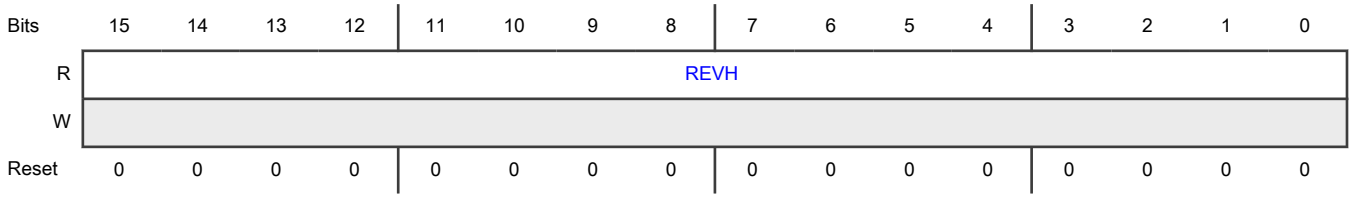
Offset

Register	Offset
REVH	1Ch

Function

Contains a snapshot of the value of the Revolution Counter Register (REV).

Diagram



Fields

Field	Function
15-0 REVH	REVH The value of the Revolution Counter Register (REV)

32.6.1.17 Revolution Counter Register (REV)

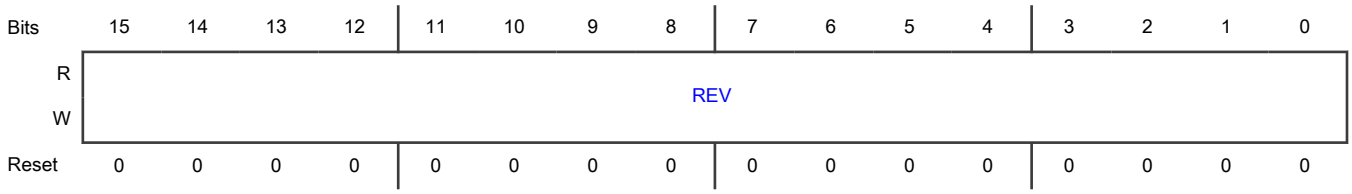
Offset

Register	Offset
REV	1Eh

Function

Contains the current value of the Revolution Counter.

Diagram



Fields

Field	Function
15-0 REV	REV The current value of the Revolution Counter

32.6.1.18 Upper Initialization Register (UINIT)

Offset

Register	Offset
UINIT	20h

Function

Contains the value to be used to initialize the upper half of the position counter (UPOS). It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram

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

Fields

Field	Function
15-0 INIT	INIT The value to be used to initialize the upper half of the position counter (UPOS).

32.6.1.19 Lower Initialization Register (LINIT)

Offset

Register	Offset
LINIT	22h

Function

Contains the value to be used to initialize the lower half of the position counter (LPOS). It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram

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

Fields

Field	Function
15-0 INIT	INIT The value to be used to initialize the lower half of the position counter (LPOS)

32.6.1.20 Upper Modulus Register (UMOD)

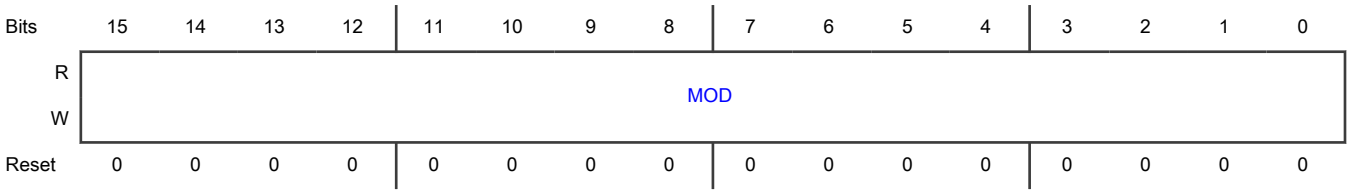
Offset

Register	Offset
UMOD	24h

Function

The Upper Modulus Register contains the upper (most significant) half of the Modulus register. MOD acts as the upper bound during modulo counting, and as the upper reload value when rolling over from the lower bound. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 MOD	MOD Upper (most significant) half of the Modulus register

32.6.1.21 Lower Modulus Register (LMOD)

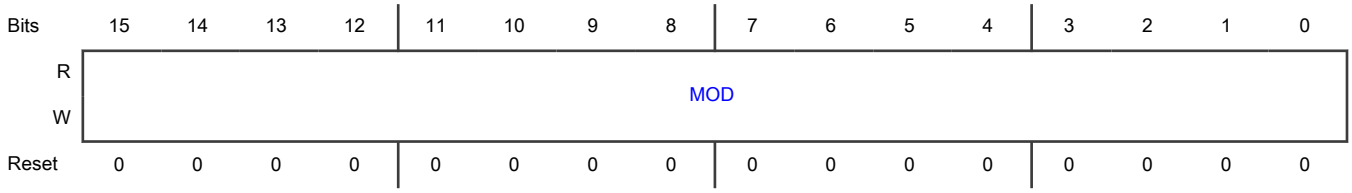
Offset

Register	Offset
LMOD	26h

Function

The Lower Modulus Register contains the lower (least significant) half of the Modulus register. MOD acts as the upper bound during modulo counting, and as the upper reload value when rolling over from the lower bound. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 MOD	MOD Lower (least significant) half of the Modulus register

32.6.1.22 Upper Position Compare Register 0 (UCOMP0)

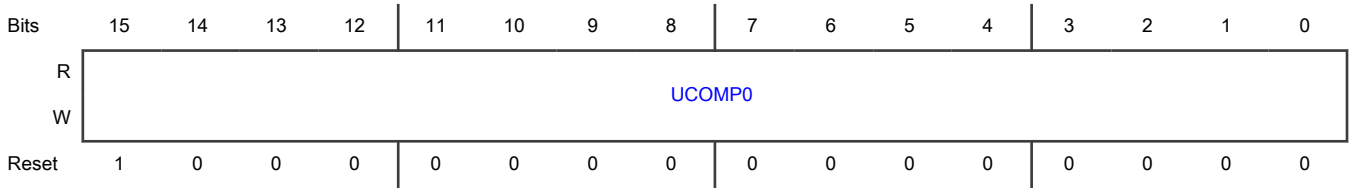
Offset

Register	Offset
UCOMP0	28h

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 0. When the value of the Position counter (POS) matches the value of the Position Compare register 0(COMP0), the Compare Interrupt Request flag (INTCTRL[CMP0IRQ]) is set and the POS_MATCH[0] output is asserted. COMP_FLG[0] is asserted when the Position Counter is equal or greater than COMP0. COMP_FLG[0] is de-asserted when the Position Counter is less than COMP0. Note: If COMP0 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[0] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 UCOMP0	UCOMP0 Upper (most significant) half of the Position Compare register 0

32.6.1.23 Lower Position Compare Register 0 (LCOMP0)

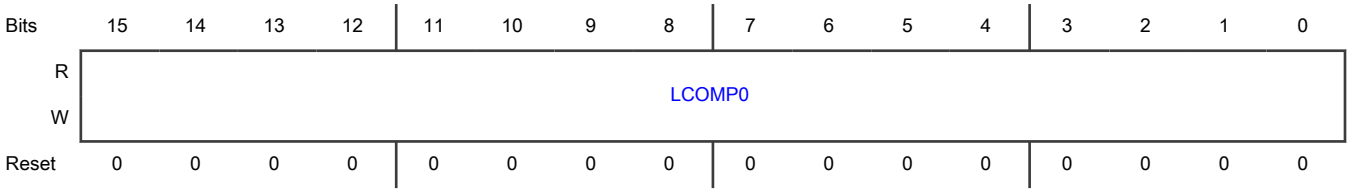
Offset

Register	Offset
LCOMP0	2Ah

Function

The Lower Position Compare Register contains the lower (least significant) half of the Position Compare register 0. When the value of the Position counter (POS) matches the value of the Position Compare register 0(COMP0), the Compare Interrupt Request flag (INTCTRL[CMP0IRQ]) is set and the POS_MATCH[0] output is asserted. COMP_FLG[0] is asserted when the Position Counter is equal or greater than COMP0. COMP_FLG[0] is de-asserted when the Position Counter is less than COMP0. Note: If COMP0 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[0] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set, and reading this register returns inner-set value which is the value that is taking effect.

Diagram



Fields

Field	Function
15-0 LCOMP0	LCOMP0 Lower (least significant) half of the Position Compare register 0

32.6.1.24 Upper Position Compare 1 (UCOMP1)

Offset

Register	Offset
UCOMP1	2Ch

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 1. When the value of the Position counter (POS) matches the value of the Position Compare register 1 (COMP1), the Compare Interrupt Request flag (INTCTRL[CMP1IRQ]) is set and the POS_MATCH[1] output is asserted. COMP_FLG[1] is asserted when the Position Counter is equal or greater than COMP1. COMP_FLG[1] is de-asserted when the Position Counter is less than COMP1. Note: If COMP1 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[1] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	UCOMP1															
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 UCOMP1	UCOMP1 When write, it writes Upper (most significant) half of the Position Compare register 1.

32.6.1.25 Upper Position Holder Register 1 (UPOSH1)

Offset

Register	Offset
UPOSH1	2Ch

Function

UPOSH1 share the same address with UCOMP1.When read,this register means the value of the Upper Position Counter Hold Register 1(UPOSH1),which is the upper 16 bits of POSH1.Position counter is captured into POSH1 on the rising edge of ICAP[1].

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	UPOSH1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 UPOSH1	UPOSH1 When read, it means the value of the Upper Position Hold Counter Register 1(UPOSH1)

32.6.1.26 Lower Position Compare 1 (LCOMP1)

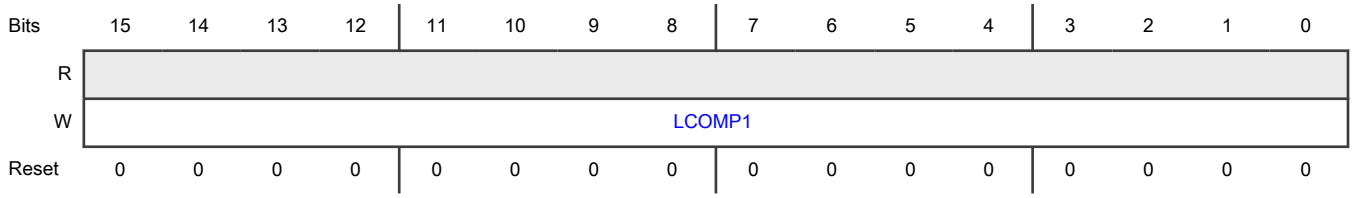
Offset

Register	Offset
LCOMP1	2Eh

Function

The Lower Position Compare Register contains the upper (most significant) half of the Position Compare register 1. When the value of the Position counter (POS) matches the value of the Position Compare register 1 (COMP1), the Compare Interrupt Request flag (INTCTRL[CMP1IRQ]) is set and the POS_MATCH[1] output is asserted. COMP_FLG[1] is asserted when the Position Counter is equal or greater than COMP1. COMP_FLG[1] is de-asserted when the Position Counter is less than COMP1. Note: If COMP1 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[1] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 LCOMP1	LCOMP1 When write, it writes Lower (most significant) half of the Position Compare register 1.

32.6.1.27 Lower Position Holder Register 1 (LPOSH1)

Offset

Register	Offset
LPOSH1	2Eh

Function

LPOSH1 share the same address with LCOMP1.When read, this register means the value of the Lower Position Counter Hold Register 1(LPOSH1),which is the lower 16 bits of POSH1.Position counter is captured into POSH1 on the rising edge of ICAP[1].

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LPOSH1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 LPOSH1	LPOSH1 When read, it means the value of the Lower Position Counter Hold Register 1(LPOSH1)

32.6.1.28 Upper Position Compare 2 (UCOMP2)

Offset

Register	Offset
UCOMP2	30h

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 2. When the value of the Position counter (POS) matches the value of the Position Compare register 2 (COMP2), the Compare Interrupt Request flag (INTCTRL[CMP2IRQ]) is set and the POS_MATCH[2] output is asserted. COMP_FLG[2] is asserted when the Position Counter is equal or greater than COMP2. COMP_FLG[2] is de-asserted when the Position Counter is less than COMP2. Note: If COMP2 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[2] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	UCOMP2															
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 UCOMP2	UCOMP2 When write, it writes Upper (most significant) half of the Position Compare register 2.

32.6.1.29 Upper Position Holder Register 3 (UPOSH2)

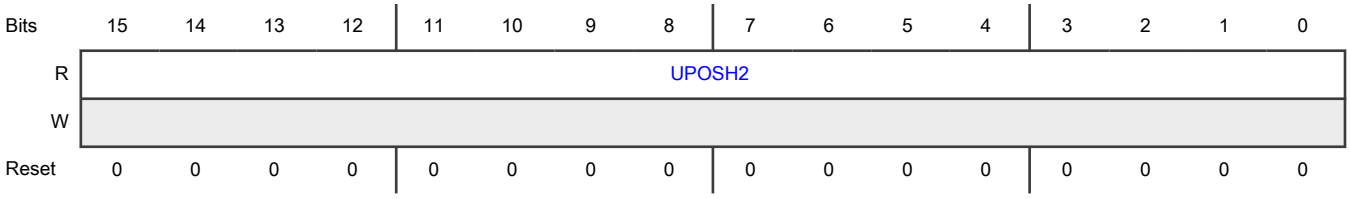
Offset

Register	Offset
UPOSH2	30h

Function

UPOSH2 share the same address with UCOMP2.When read,this register means the value of the Upper Position Counter Hold Register 2(UPOSH2),which is the upper 16 bits of POSH2.Position counter is captured into POSH2 on the rising edge of ICAP[2].

Diagram



Fields

Field	Function
15-0 UPOSH2	UPOSH2 When read, it means the value of the Upper Position Counter Hold Register 2(UPOSH2)

32.6.1.30 Lower Position Compare 2 (LCOMP2)

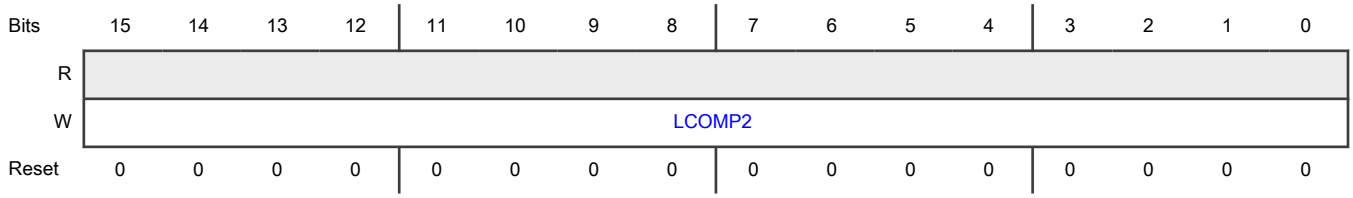
Offset

Register	Offset
LCOMP2	32h

Function

The Lower Position Compare Register contains the Lower (most significant) half of the Position Compare register 2. When the value of the Position counter (POS) matches the value of the Position Compare register 2 (COMP2), the Compare Interrupt Request flag (INTCTRL[CMP2IRQ]) is set and the POS_MATCH[2] output is asserted. COMP_FLG[2] is asserted when the Position Counter is equal or greater than COMP2. COMP_FLG[2] is de-asserted when the Position Counter is less than COMP2. Note: If COMP2 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[2] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 LCOMP2	LCOMP2 When write, it writes Lower (most significant) half of the Position Compare register 2.

32.6.1.31 Lower Position Holder Register 2 (LPOSH2)

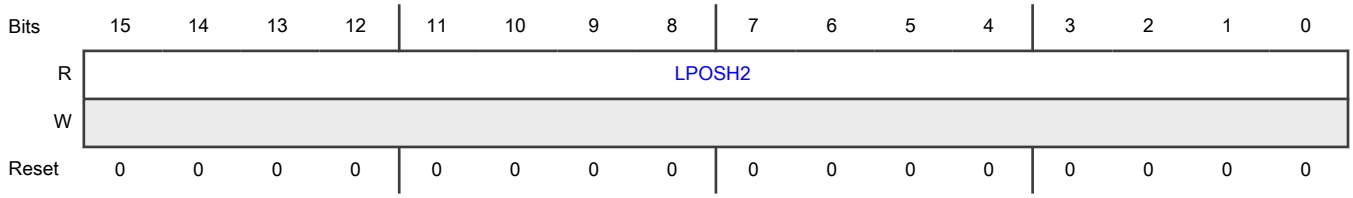
Offset

Register	Offset
LPOSH2	32h

Function

LPOSH2 share the same address with LCOMP2.When read,this register means the value of the Lower Position Counter Hold Register 2(LPOSH2),which is the lower 16 bits of POSH2.Position counter is captured into POSH2 on the rising edge of ICAP[2].

Diagram



Fields

Field	Function
15-0 LPOSH2	LPOSH2 When read, it means the value of the Lower Position Counter Hold Register 2(LPOSH2)

32.6.1.32 Upper Position Compare 3 (UCOMP3)

Offset

Register	Offset
UCOMP3	34h

Function

The Upper Position Compare Register contains the upper (most significant) half of the Position Compare register 3. When the value of the Position counter (POS) matches the value of the Position Compare register 3 (COMP3), the Compare Interrupt Request flag (INTCTRL[CMP3IRQ]) is set and the POS_MATCH[3] output is asserted. COMP_FLG[3] is asserted when the Position Counter is equal or greater than COMP3. COMP_FLG[3] is de-asserted when the Position Counter is less than COMP3. Note: If COMP3 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[3] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	UCOMP3															
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 UCOMP3	UCOMP3 When write, it writes Upper (most significant) half of the Position Compare register 3.

32.6.1.33 Upper Position Holder Register 3 (UPOSH3)

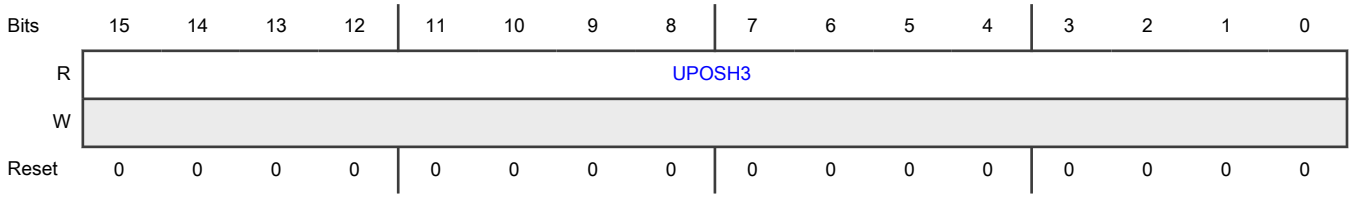
Offset

Register	Offset
UPOSH3	34h

Function

UPOSH3 share the same address with UCOMP3.When read,this register means the value of the Upper Position Counter Hold Register 3(UPOSH3),which is the upper 16 bits of POSH3.Position counter is captured into POSH3 on the rising edge of ICAP[3].

Diagram



Fields

Field	Function
15-0 UPOSH3	UPOSH3 When read, it means the value of the Upper Position Counter Hold Register 3(UPOSH3)

32.6.1.34 Lower Position Compare 3 (LCOMP3)

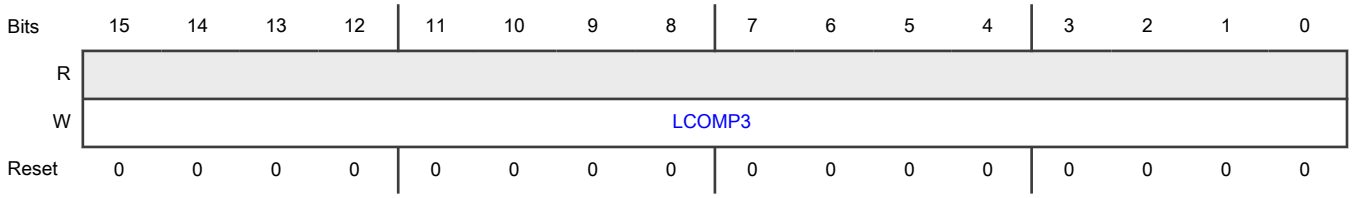
Offset

Register	Offset
LCOMP3	36h

Function

The Lower Position Compare Register contains the Lower (most significant) half of the Position Compare register 3. When the value of the Position counter (POS) matches the value of the Position Compare register 3 (COMP3), the Compare Interrupt Request flag (INTCTRL[CMP3IRQ]) is set and the POS_MATCH[3] output is asserted. COMP_FLG[3] is asserted when the Position Counter is equal or greater than COMP3. COMP_FLG[3] is de-asserted when the Position Counter is less than COMP3. Note: If COMP3 value equals initial value (INIT) or modulus value (MOD), COMP_FLG[3] does not take effect. It is a double-set register (please refer to chapter "Double-set Registers Loading Operation"). Writing this register actually writes to the out-set.

Diagram



Fields

Field	Function
15-0 LCOMP3	LCOMP3 When write, it writes Lower (most significant) half of the Position Compare register 3.

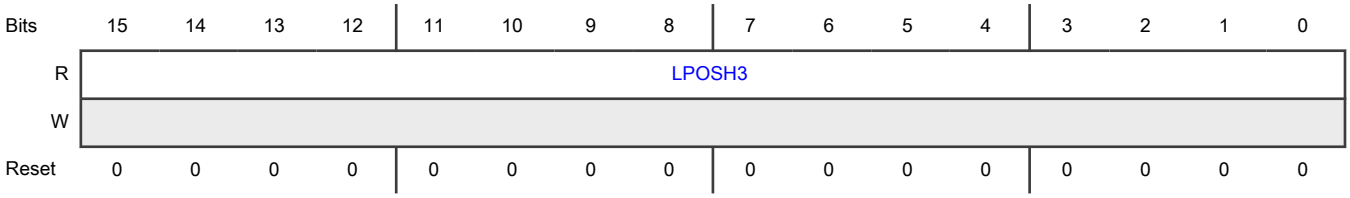
32.6.1.35 Lower Position Holder Register 3 (LPOSH3)

Offset

Register	Offset
LPOSH3	36h

Function
When read,this register means the value of the Lower Position Counter Register 3(LPOS)

Diagram



Fields

Field	Function
15-0 LPOSH3	LPOSH3 LPOSH3 share the same address with LCOMP3.When read,it means the value of the Lower Position Counter Register 3(LPOS3),which is the lower 16 bits of POSH3.Position counter is captured into POSH3 on the rising edge of ICAP[3].

32.6.1.36 Interrupt Control Register (INTCTRL)

Offset

Register	Offset
INTCTRL	38h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMP3IRQ	CMP3IE	CMP2IRQ	CMP2IE	CMP1IRQ	CMP1IE	CMP0IRQ	CMP0IE	ROIRQ	ROIE	RUIRQ	RUIE	DIRIRQ	DIRIE	SABIRQ	SABIE
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	0

Fields

Field	Function
15 CMP3IRQ	Compare3 Interrupt Request Compare3 Interrupt Request is set when the position counter matches the COMP3 value. • Compare3 Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP3IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP3 value) 1b - COMP3 match has occurred (the position counter matches the COMP3 value)
14 CMP3IE	Compare3 Interrupt Enable Enables/disables compare3 interrupt. 0b - Disabled 1b - Enabled
13 CMP2IRQ	Compare2 Interrupt Request Compare2 Interrupt Request is set when the position counter matches the COMP2 value. • Compare2 Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP2IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP2 value) 1b - COMP2 match has occurred (the position counter matches the COMP2 value)
12 CMP2IE	Compare2 Interrupt Enable Enables/disables compare2 interrupts. 0b - Disabled 1b - Enabled
11 CMP1IRQ	Compare1 Interrupt Request Compare1 Interrupt Request is set when the position counter matches the COMP1 value. • Compare Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP1IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP1 value) 1b - COMP1 match has occurred (the position counter matches the COMP1 value)

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 CMP1IE	Compare1 Interrupt Enable Enables/disables compare1 interrupt. 0b - Disabled 1b - Enabled
9 CMP0IRQ	Compare 0 Interrupt Request Compare 0 Interrupt Request is set when the position counter matches the COMP0 value. • Compare Interrupt Request remains set until cleared by software • Write 1 to this bit (CMP0IRQ) to clear it 0b - No match has occurred (the position counter does not match the COMP0 value) 1b - COMP match has occurred (the position counter matches the COMP0 value)
8 CMP0IE	Compare 0 Interrupt Enable Enables/disables compare 0 interrupts. 0b - Disabled 1b - Enabled
7 ROIRQ	Roll-over Interrupt Request Roll-over Interrupt Request bit is set (=1) when the position counter (POS) value is greater than the modulus value, which means roll-over. • Roll-over Interrupt Request remains set until cleared by software • Write a one to this bit (ROIRQ) to clear it 0b - No roll-over has occurred 1b - Roll-over has occurred
6 ROIE	Roll-over Interrupt Enable Roll-over Interrupt Enable read/write bit enables roll-over interrupts, based on Roll-under Interrupt Request (INTCTRL[ROIRQ]) being set. 0b - Disabled 1b - Enabled
5 RUIRQ	Roll-under Interrupt Request Roll-under Interrupt Request bit is set (=1) when the position counter (POS) value less than initial value, which means roll-under • Roll-under Interrupt Request remains set until cleared by software • Write a one to this bit (RUIRQ) to clear it 0b - No roll-under has occurred 1b - Roll-under has occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 RUIE	Roll-under Interrupt Enable Roll-under Interrupt Enable read/write bit enables roll-under interrupts, based on the Roll-over Interrupt Request bit (INTCTRL[RUIRQ]) being set. 0b - Disabled 1b - Enabled
3 DIRIRQ	Count direction change interrupt Count direction change interrupt bit is set(=1) when POS/POSD/REV counter count direction changes, which means IMR[DIR] changes from 0 to 1 or from 1 to 0 0b - Count direction unchanged 1b - Count direction changed
2 DIRIE	Count direction change interrupt enable Count direction change interrupt enable interrupt 0b - Disabled 1b - Enabled
1 SABIRQ	Simultaneous PHASEA and PHASEB Change Interrupt Request Simultaneous PHASEA and PHASEB Change Interrupt Request indicates that the PHASEA and PHASEB inputs changed simultaneously (within a single clock period). This event typically indicates an error condition, because quadrature coding requires only one of these inputs to change at a time. • Simultaneous PHASEA and PHASEB Change Interrupt Request bit remains set until it is cleared by software or a reset • Write 1 to this bit (SABIRQ) to clear it 0b - No simultaneous change of PHASEA and PHASEB has occurred 1b - A simultaneous change of PHASEA and PHASEB has occurred
0 SABIE	Simultaneous PHASEA and PHASEB Change Interrupt Enable Simultaneous PHASEA and PHASEB Change Interrupt Enable bit enables simultaneous PHASEA and PHASEB change interrupts, based on the Simultaneous PHASEA and PHASEB Change Interrupt Request (SABIRQ) bit being set. 0b - Disabled 1b - Enabled

32.6.1.37 Watchdog Timeout Register (WTR)

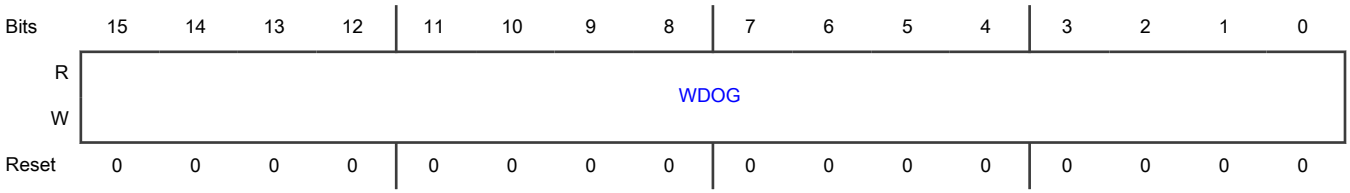
Offset

Register	Offset
WTR	3Ah

Function

The Watchdog Timeout Register stores the timeout count for the quadrature decoder module watchdog timer. This quadrature decoder module watchdog timer is separate from any other watchdog timer(s) that may also be in the device.

Diagram



Fields

Field	Function
15-0 WDOG	WDOG WDOG[15:0] is a binary representation of the number of clock cycles, using the peripheral clock prescaled by FILT[PRSC], 0xFFFF plus one count in this register determines the timing out period.

32.6.1.38 Input Monitor Register (IMR)

Offset

Register	Offset
IMR	3Ch

Function

The Input Monitor Register contains the values of the raw and filtered PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals. The reset value depends on the values of the raw and filtered values of PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals:

- The Input Monitor Register [3:0] contains the values of the raw value of PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals.
- The Input Monitor Register [7:4] contains the values of the filtered value of PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals.
- The Input Monitor Register [7:4] supply filter bypass function for PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals.
- The Input Monitor Register [11:8] contains the values of the 4 POS counter CMPs flag.
- The Input Monitor Register [15:14] contains the values of the count direction flag and count direction flag hold.
- If no pull-up or pull-down is connected to PHASEA, PHASEB, INDEX/PRESET and HOME/ENABLE input signals, then the reset value of the 4 lower bits of the Input Monitor Register (IMR) are all unknown.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIR	DIRH	0		CMP3F	CMP2F	CMP1F	CMPF0	FPHA	FPHB	FIND_PRE	FHOM_ENA	PHA	PHB	INDEX_P...	HOME_EN...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 DIR	Count Direction Flag Output The Count Direction Flag indicates the direction of the current count. 0b - Current count was in the down direction 1b - Current count was in the up direction
14 DIRH	Count Direction Flag Hold DIRH contains a snapshot of the DIR when initializing POS
13-12 —	Reserved
11 CMP3F	Position Compare3 Flag Output Flag of compare3 0b - When the position counter value is less than value of COMP3 register 1b - When the position counter is greater or equal than value of COMP3 register
10 CMP2F	Position Compare2 Flag Output Flag of compare2 0b - When the position counter is less than value of COMP2 register 1b - When the position counter is greater or equal than value of COMP2 register
9 CMP1F	Position Compare1 Flag Output Flag of compare1 0b - When the position counter is less than value of COMP1 register 1b - When the position counter is greater or equal than value of COMP1 register
8 CMPF0	Position Compare 0 Flag Output Flag of compare 0 0b - When the position counter is less than value of COMP0 register 1b - When the position counter is greater or equal than value of COMP0 register
7	filter operation on PHASEA input

Table continues on the next page...

Table continued from the previous page...

Field	Function
FPHA	When write 1, it means filter for PHASEA input is bypassed When read, it shows filtered version of PHASEA input
6 FPHB	filter operation on PHASEB input When write 1, it means filter for PHASEB input is bypassed When read, it shows filtered version of PHASEB input
5 FIND_PRE	filter operation on INDEX/PRESET input When write 1, it means filter for INDEX/PRESET input is bypassed When read, it shows filtered version of INDEX/PRESET input
4 FHOM_ENA	filter operation on HOME/ENABLE input When write 1, it means filter for HOME/ENABLE input is bypassed When read, it shows filtered version of HOME/ENABLE input
3 PHA	PHA Raw PHASEA input
2 PHB	PHB Raw PHASEB input
1 INDEX_PRESET	INDEX_PRESET Raw INDEX/PRESET input
0 HOME_ENABLE	HOME_ENABLE The raw HOME/ENABLE input

32.6.1.39 Test Register (TST)

Offset

Register	Offset
TST	3Eh

Function

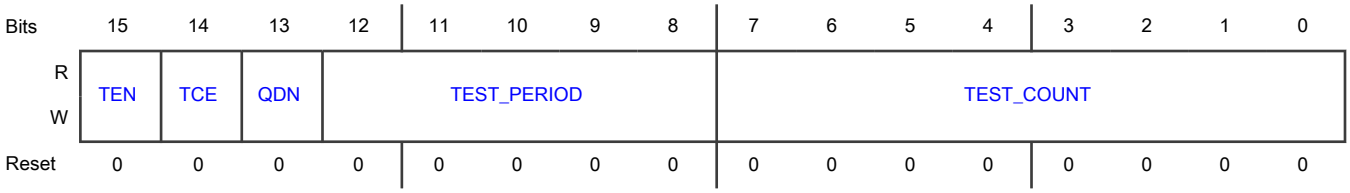
The Test Register controls and sets the frequency of a quadrature signal generator; it provides a quadrature test signal to the inputs of the quadrature decoder module.

- The TEST_COUNT value is counted down to zero when the test module is enabled (TEN = 1) and the counter is enabled (TCE = 1).

- Each count value of one represents a single quadrature cycle interpreted as a count of one by the position counter (UPOS and LPOS) if it is enabled (TEN = 1, TCE = 1).
- Repeated writing of new values to TEST_COUNT can cause an extra phase transition and therefore an extra count by the Position Counter.
- The period field determines the length (in IP Bus clock cycles) of each quadrature cycle phase.

The Test Register is a factory test feature; however, it can also be useful in customer software development and testing.

Diagram



Fields

Field	Function
15 TEN	Test Mode Enable Connects the test module to inputs of the quadrature decoder module. 0b - Disabled 1b - Enabled
14 TCE	Test Counter Enable Connects the test counter to inputs of the quadrature decoder module. 0b - Disabled 1b - Enabled
13 QDN	Quadrature Decoder Negative Signal Selects whether a negative or positive Quadrature Decoder signal is generated. 0b - Generates a positive quadrature decoder signal 1b - Generates a negative quadrature decoder signal
12-8 TEST_PERIOD	TEST_PERIOD Period of quadrature phase in IP Bus clock cycles
7-0 TEST_COUNT	TEST_COUNT The number of quadrature advances to generate

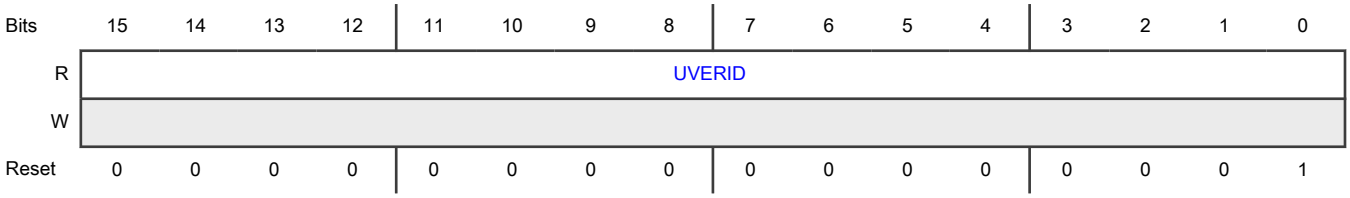
32.6.1.40 Upper VERID (UVERID)

Offset

Register	Offset
UVERID	50h

Function
Version ID Upper part

Diagram



Fields

Field	Function
15-0 UVERID	UVERID Upper (most significant) half of the VERID

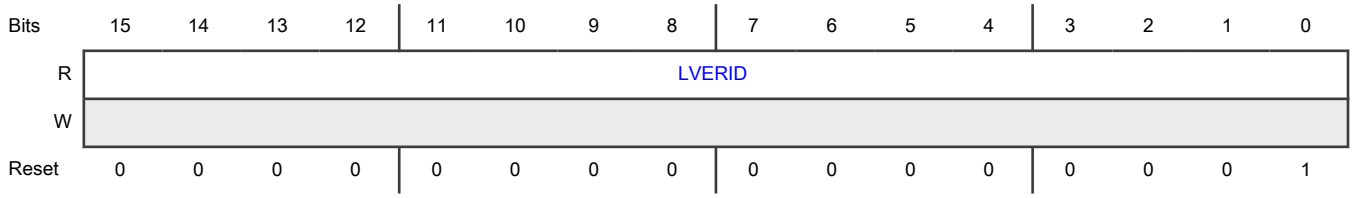
32.6.1.41 Lower VERID (LVERID)

Offset

Register	Offset
LVERID	52h

Function
Version ID Lower part

Diagram



Fields

Field	Function
15-0 LVERID	LVERID Lower (most significant) half of the VERID

Chapter 33
Frequency Measurement (FREQME)

33.1 Chip-specific Frequency Measurement information

Table 178. Reference links to related information

Topic	Related module	Reference
Full description	FREQME	Frequency Measurement
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

33.1.1 Module instances

This device has one instance of the Frequency Measurement module, FREQME0.

33.1.2 Input clock sources

Following are the input clocks for frequency measurement circuit:

- 0 Reserved
- 1 clk_in
- 2 FRO_OSC_12M
- 3 fro_hf_div
- 4 Reserved
- 5 clk_16k[1]
- 6 SLOW_CLK
- 7 FREQME_CLK_IN0
- 8 FREQME_CLK_IN1
- 9 AOI0_OUT0
- 10 AOI0_OUT1
- 11 PWM0_SM0_MUX_TRIG0
- 12 PWM0_SM0_MUX_TRIG1
- 13 PWM0_SM1_MUX_TRIG0
- 14 PWM0_SM1_MUX_TRIG1
- 15 PWM0_SM2_MUX_TRIG0
- 16 PWM0_SM2_MUX_TRIG1
- 17 Reserved
- 18 Reserved

- 32 AOI1_OUT0
- 33 AOI1_OUT1
- 34 PWM1_SM0_MUX_TRIG0
- 35 PWM1_SM0_MUX_TRIG1
- 36 PWM1_SM1_MUX_TRIG0
- 37 PWM1_SM1_MUX_TRIG1
- 38 PWM1_SM2_MUX_TRIG0
- 39 PWM1_SM2_MUX_TRIG1
- 40 Reserved
- 41 Reserved

For details, see the FREQMEAS_REF and FREQMEAS_TAR registers in Peripheral Input multiplexing chapter.

33.2 Overview

FREQME accurately measures the frequency of an on- or off-chip target clock signal using a selectable on-chip reference clock. For example, it can accurately determine the frequency of a low-power oscillator that varies depending on process and temperature.

33.2.1 Block diagram

Figure 137. Block diagram

See the chip-specific FREQME information for the target and reference clock selection options.

33.2.2 Features

- High-accuracy Frequency Measurement mode for on- and off-chip clocks
- Pulse Width Measurement (PWM) mode
- Reference and target clock inputs selectable from among various chip-specific options
- Optional measurement complete interrupt
- Result out-of-range detection with optional interrupt

33.3 Functional description

The following sections describe FREQME functional details.

33.3.1 Frequency Measurement mode

In Frequency Measurement mode (CTL_W[PULSE_MODE] = 0), FREQME counts the number of target clock cycles that occur during a specified number of cycles from a reference clock that has a known frequency. You can then calculate the target clock frequency based on the frequency of the reference clock, the number of reference clock cycles, and the number of target clock cycles (see Equation 1).

The frequency measurement circuit is based on two 31-bit counters—one clocked by the selected reference clock and one by the selected target clock. You can only read the target clock counter. FREQME synchronizes the clocks at the start and end of

each count sequence during preparation for the next count sequence. A count sequence consists of incrementing the target clock counter for each target clock pulse that occurs during the time defined by $2^{CTRL\_W[REF\_SCALE]}$ periods of the reference clock.

After selecting reference and target clocks, you initiate a measurement cycle by writing 1 to CTRL_W[MEASURE_IN_PROGRESS]. You can then poll this same field, which automatically transitions to 0 when the measurement operation completes. Alternatively, the completion of the measurement operation can generate an interrupt.

The measurement cycle terminates when the reference counter equals the value $2^{CTRLSTAT[REFSCALE]}$. If CTRLSTAT[REFSCALE] = 0, the reference counter counts to one and stops. You can use this feature to measure the frequency of a fast target clock using a slow reference clock such as the 32 kHz clock without taking much time for the measurement to complete. The penalty is reduced accuracy in the measurement.

When the counting operation completes, the state of the target counter is loaded into CTRL_R[RESULT], and the CTRL_W[MEASURE_IN_PROGRESS] field is 0. You can then read the value and calculate the target frequency as follows, with the frequencies given in MHz.

$$F_{target} = (CTRL\_R[RESULT] - 2) \times F_{reference} \div 2^{CTRLSTAT[REFSCALE]}$$

Equation 1. Calculating the target clock frequency

33.3.1.1 Accuracy

The Frequency Measurement mode can measure the frequency of any on-chip (or off-chip) clock (referred to as the target clock) with a high degree of accuracy by using an on-chip clock of known frequency as a reference clock.

Uncertainty in the reference clock (for example a ±1 % accuracy of the clock) adds to the measurement error of the target clock. In general, though, this additional error is less than the uncertainty of the reference clock.

33.3.2 Pulse Measurement mode

When you write 1 to CTRL_W[PULSE_MODE] and write 1 to CTRL_W[MEASURE_IN_PROGRESS] to start a measurement cycle, FREQME counts target clock pulses while the reference clock is in a specific state (high or low), selected by writing to CTRL_W[PULSE_POL]. See the description of these fields for more details.

33.3.3 Clocking

The clock inputs are as follows:

- Reference clock (REF_CLK_IN). This clock has a known frequency.
- Target clock (TARGET_CLK_IN). This clock frequency can be calculated based on the reference clock.
- Peripheral clock used for register access (PCLK). This clock is used to write and read memory-mapped registers of this module.

NOTE

These clocks can be all synchronous.

See the chip-specific FREQME information for the target and reference clock selection options.

33.3.4 Interrupts

The following interrupts can optionally be generated at the completion of a frequency measurement cycle.

Table 179. Interrupts

Condition	Interrupt enable field	Interrupt status field
Result is ready to read	CTRL_W[RESULT_READY_INT_EN]	CTRLSTAT[RESULT_READY_INT_EN]

Table continues on the next page...

Table 179. Interrupts (continued)

Condition	Interrupt enable field	Interrupt status field
Result is greater than MAX[MAX_VALUE]	CTRL_W[GT_MAX_INT_EN]	CTRLSTAT[GT_MAX_INT_EN]
Result is less than MIN[MIN_VALUE]	CTRL_W[LT_MIN_INT_EN]	CTRLSTAT[LT_MIN_INT_EN]

33.4 External signals

Table 180. External signals

Signal	Description	Direction
REF_CLK_IN	Reference clock. On-chip clock with known frequency chosen as reference.	Input
TARGET_CLK_IN	Target clock. On- or off-chip clock to be measured by FREQME.	Input

33.5 Initialization

Initialize FREQME by performing the following steps:

1. Enable the clock that drives FREQME in the chip-level clock control register. This step enables the register interface and the peripheral function clock.
2. Clear the FREQME peripheral reset in the chip-level reset control register.
3. If measuring an external clock, use the Input/Output pin configuration registers to connect the FREQME target clock input (TARGET_CLK_IN) to an external pin.
4. Ensure that the reference and target clocks are enabled.
5. Select the reference and target clocks.

33.6 Memory map and register definition

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

33.6.1 FREQME register descriptions

33.6.1.1 FREQME memory map

FREQME0 base address: 4000_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (in Read mode) (CTRL_R)	32	R	0000_0000h
0h	Control (in Write mode) (CTRL_W)	32	W	0000_0000h
4h	Control Status (CTRLSTAT)	32	RW	0000_0000h
8h	Minimum (MIN)	32	RW	0000_0000h
Ch	Maximum (MAX)	32	RW	7FFF_FFFFh

33.6.1.2 Control (in Read mode) (CTRL_R)

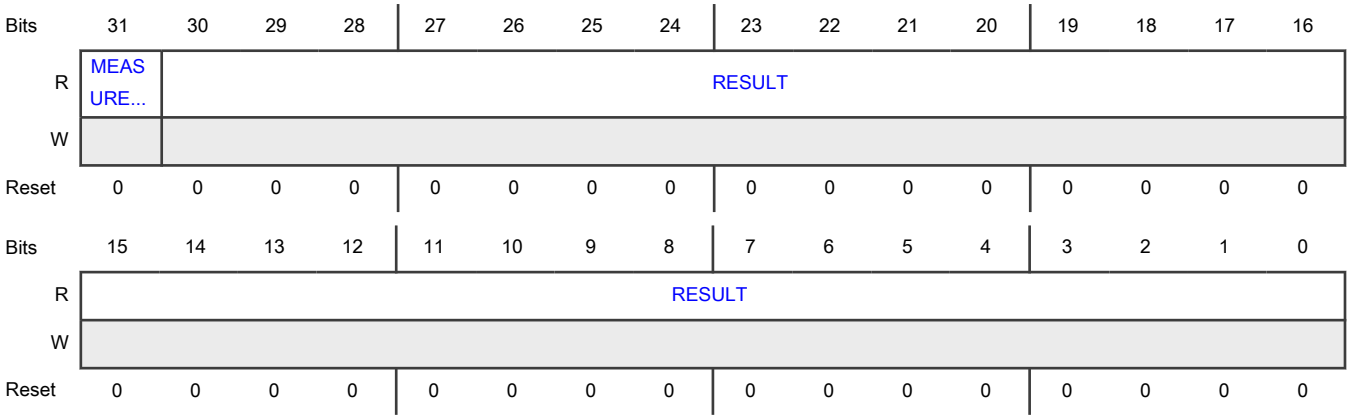
Offset

Register	Offset
CTRL_R	0h

Function

Contains different fields depending on whether you are performing a read or a write. Reading this register provides the measurement results and the status of the measurement cycle.

Diagram



Fields

Field	Function
31 MEASURE_IN_PROGRESS	Measurement In Progress Indicates whether measurement is in progress or complete. If complete, you can read the result from RESULT. 0b - Complete 1b - In progress
30-0 RESULT	Indicates the measurement result—either the target clock counter value (for Frequency Measurement mode) or pulse width measurement (for Pulse Width Measurement mode). This field is valid only when MEASURE_IN_PROGRESS = 0, In Continuous mode (CTRL_W[CONTINUOUS_MODE_EN] = 1), the value is the result from the most-recently completed measurement.

33.6.1.3 Control (in Write mode) (CTRL_W)

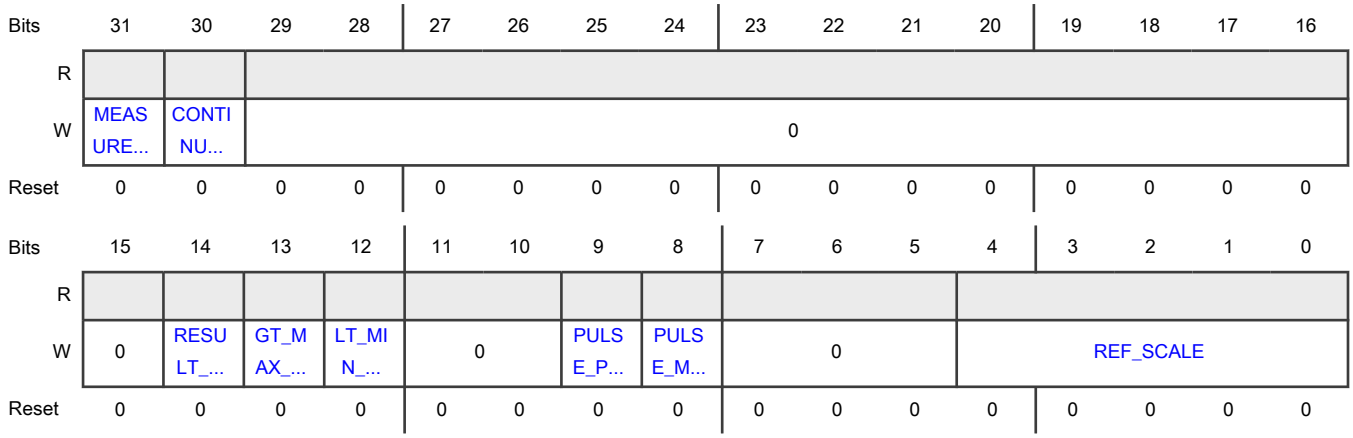
Offset

Register	Offset
CTRL_W	0h

Function

Writes to this register control and configure the measurement cycle.

Diagram



Fields

Field	Function
31 MEASURE_IN_PROGRESS	Measurement In Progress Initiates a frequency measurement cycle or terminates a measurement cycle that is in progress. Writing 1 to this field initiates a frequency or pulse width measurement process. Hardware automatically writes 0 to the MEASURE_IN_PROGRESS field when the measurement cycle completes. If there is an active measurement in progress, a new measurement starts. Writing 0 to this field forces the termination of any measurement cycle currently in progress and resets CTRL_R[RESULT] or just resets CTRL_R[RESULT] if idle. 0b - Terminates measurement 1b - Initiates measurement
30 CONTINUOUS_MODE_EN	Continuous Mode Enable When you write 1 to both MEASURE_IN_PROGRESS and this field, measurement is performed continuously. The result for the most-recently completed measurement is available in CTRL_R[RESULT]. When the result is out of range (that is, CTRLSTAT[GT_MAX_STAT] = 1 or CTRLSTAT[LT_MIN_STAT] = 1), this field automatically transitions to 0 and the continuous measurement is suspended. After clearing both status bits, you must write 1 to both MEASURE_IN_PROGRESS and this field restart measurement in continuous mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
29-15 —	Reserved
14 RESULT_READY_INT_EN	Result Ready Interrupt Enable Generates interrupt when a measurement completes and the result is ready (CTRLSTAT[RESULT_READY_STAT] = 1). 0b - Disable 1b - Enable
13 GT_MAX_INT_EN	Greater Than Maximum Interrupt Enable Generates an interrupt when the result is greater than MAX[MAX_VALUE] (CTRLSTAT[GT_MAX_STAT] = 1). 0b - Disable 1b - Enable
12 LT_MIN_INT_EN	Less Than Minimum Interrupt Enable Generates an interrupt when the result is less than MIN[MIN_VALUE] (CTRLSTAT[LT_MIN_STAT] = 1). 0b - Disable 1b - Enable
11-10 —	Reserved
9 PULSE_POL	Pulse Polarity Specifies whether a pulse width (high period or low period) of the reference clock is measured in the Pulse Width Measurement mode. **NOTE** PULSE_POL is valid only in the Pulse Width Measurement mode. • A high period measurement is triggered by the rising edge on the reference clock input. • A low period measurement is triggered by the falling edge on the reference clock input. 0b - High period 1b - Low period
8 PULSE_MODE	Pulse Width Measurement Mode Select Selects the measurement mode—either Frequency Measurement mode or Pulse Width Measurement mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In Frequency Measurement mode, measurement begins at the rising edge of the selected reference clock and continues until a count of 2^{REF_SCALE} reference clock pulses is reached. In Pulse Width Measurement mode, the counter starts incrementing when the selected trigger edge (rising edge for high period measurement or falling edge for low period) occurs, and stops at the next edge (falling edge for a high period measurement or rising edge for a low period measurement). Select high or low period measurement by writing the desired value to PULSE_POL. 0b - Frequency Measurement mode 1b - Pulse Width Measurement mode
7-5 —	Reserved
4-0 REF_SCALE	Reference Clock Scaling Factor Specifies the reference clock scaling factor in Frequency Measurement mode. The reference count cycle is 2^{REF_SCALE}. A higher number provides better accuracy but consumes more processing time. This field is valid only in Frequency Measurement mode.

33.6.1.4 Control Status (CTRLSTAT)

Offset

Register	Offset
CTRLSTAT	4h

Function

Contains the current CTRL_W register configuration fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MEASURE...	CONTINU...	0			RESULT_...	GT_MAX_...	LT_MIN_...	0							
W						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	0	RESULT_...	GT_MAX_...	LT_MIN_...	0		PULSE_P...	PULSE_M...	0			REF_SCALE				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MEASURE_IN_PROGRESS	Measurement in Progress Status Indicates the current CTRL_W[MEASURE_IN_PROGRESS] value. 0b - Not in progress 1b - In progress
30 CONTINUOUS_MODE_EN	Continuous Mode Enable Status Indicates the current CTRL_W[CONTINUOUS_MODE_EN] value. 0b - Disabled 1b - Enabled
29-27 —	Reserved
26 RESULT_READY_STAT	Result Ready Status Indicates that a measurement is complete and CTRL_R[RESULT] is ready to read. Write 1 to this field to clear. 0b - Not complete 1b - Complete
25 GT_MAX_STAT	Greater Than Maximum Result Status Indicates that a measurement is complete and the result is greater than the maximum expected value (CTRL_R[RESULT] > MAX[MAX_VALUE]). Write 1 to this field to clear. 0b - Less than MAX[MAX_VALUE] 1b - Greater than MAX[MAX_VALUE]
24 LT_MIN_STAT	Less Than Minimum Results Status Indicates that a measurement is complete and the result is less than the minimum expected value (CTRL_R[RESULT] < MIN[MIN_VALUE]). Write 1 to this field to clear. 0b - Greater than MIN[MIN_VALUE] 1b - Less than MIN[MIN_VALUE]
23-15 —	Reserved
14 RESULT_READY_INT_EN	Result Ready Interrupt Enable Indicates the current CTRL_W[RESULT_READY_INT_EN] value. 0b - Disabled 1b - Enabled
13	Greater Than Maximum Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
GT_MAX_INT_EN	Indicates the current CTRL_W[GT_MAX_INT_EN] value. 0b - Disabled 1b - Enabled
12 LT_MIN_INT_EN	Less Than Minimum Interrupt Enable Indicates the current CTRL_W[LT_MIN_INT_EN] value. 0b - Disabled 1b - Enabled
11-10 —	Reserved
9 PULSE_POL	Pulse Polarity Indicates the current CTRL_W[PULSE_POL] value. 0b - High period 1b - Low period
8 PULSE_MODE	Pulse Mode Indicates the current CTRL_W[PULSE_MODE] value. 0b - Frequency Measurement mode 1b - Pulse Width Measurement mode
7-5 —	Reserved
4-0 REF_SCALE	Reference Scale Indicates the current CTRL_W[REF_SCALE] value.

33.6.1.5 Minimum (MIN)

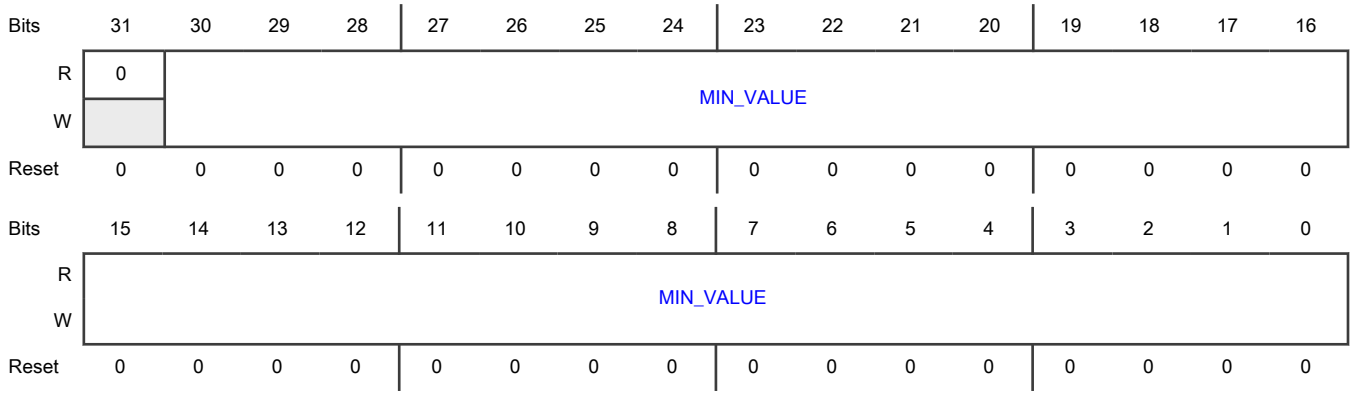
Offset

Register	Offset
MIN	8h

Function

Specifies the minimum expected value for the measurement result.

Diagram



Fields

Field	Function
31 —	Reserved
30-0 MIN_VALUE	Minimum Value Minimum expected value for the measurement result.

33.6.1.6 Maximum (MAX)

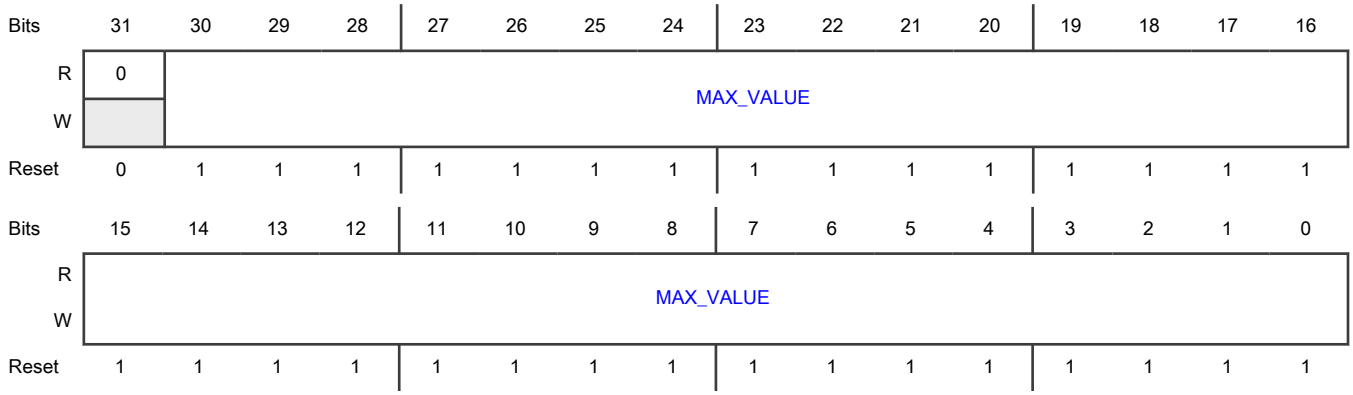
Offset

Register	Offset
MAX	Ch

Function

Specifies the maximum expected value for the measurement result.

Diagram



Fields

Field	Function
31 —	Reserved
30-0 MAX_VALUE	Maximum Value Maximum expected value for the measurement result.

Chapter 34
Low-power Timer (LPTMR)

34.1 Chip-specific LPTMR information

Table 181. Reference links to related information

Topic	Related module	Reference
Full description	LPTMR	LPTMR
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

34.1.1 Module instances

This device has one instance of the LPTMR module, LPTMR0.

34.1.2 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 25 MHz.

Table 182. LPTMRn prescaler/glitch filter clocking options

LPTMRn_PSR[PCS]	Prescaler/glitch filter clock number	Chip clock
00	0	Reserved.
01	1	FRO_16K
10	2	Reserved
11	3	Combination of clocks configured in MRCC_LPTMR0_CLKSEL[MUX] field in SYSCON module. The Clock frequency must be less than 25 MHz to be used as a clock for the Low Power Timers. See Figure 44

34.1.3 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 183. LPTMRn (n=0,1) pulse counter input options

LPTMRn_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0_OUT
01	1	CMP1_OUT
10	2	LPTMRn_Alt2
11	3	LPTMRn_Alt3

34.2 Overview

You can configure LPTMR to operate as a time counter with an optional prescaler, or as a pulse counter with an optional glitch filter, across all power modes, including low-power modes. It is reset only on POR only or Cold Reset which would include LVD/HVD sources., allowing it to be used as a time-of-day counter.

34.2.1 Block diagram

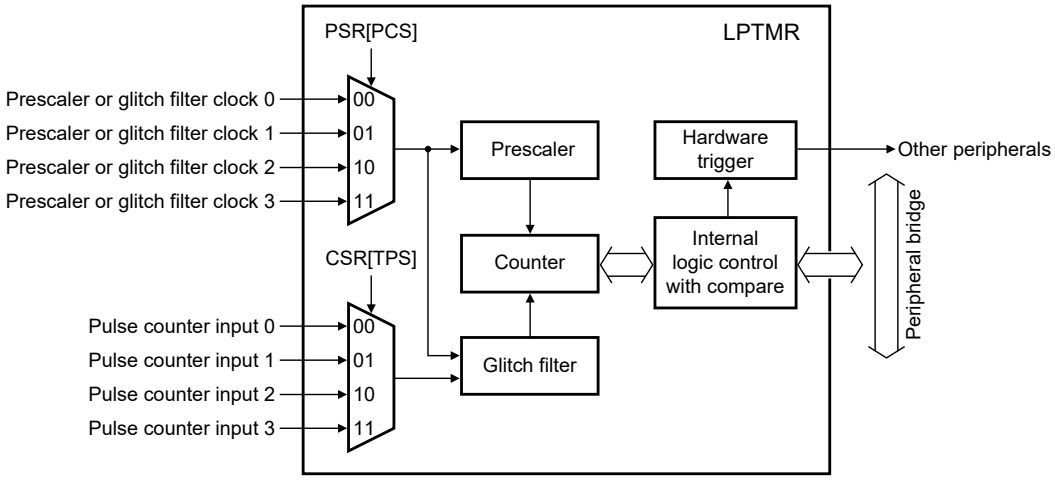


Figure 138. Block diagram

34.2.2 Features

- 32-bit time counter or pulse counter with compare:
  - Optional interrupt that can generate an asynchronous wake-up from any low-power mode
  - Hardware trigger output
  - Counter that supports a free-running mode or reset on compare
- Configurable clock source for prescaler and glitch filter
- Configurable input source for pulse counter (rising-edge or falling-edge)

34.3 Functional description

34.3.1 Low-power modes

In low-power modes, LPTMR continues to operate normally. You can configure LPTMR to exit a low-power mode by generating either an interrupt or a DMA request.

34.3.2 Clocks

The LPTMR prescaler and glitch filter can be clocked by one of the clocks that you configure by using PSR[PCS]. You must enable the clock source before you enable LPTMR.

In Pulse Counter mode, with the glitch filter bypassed, the selected input source directly clocks Counter (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

- You may need to configure the clock source that you select in PSR[PCS] for it to remain enabled in low-power modes. Otherwise, LPTMR does not operate in low-power modes.
- The clock source or pulse input source selected for LPTMR must not exceed the maximum frequency of f_{LPTMR} defined in the chip data sheet.

34.3.3 Reset

LPTMR is reset only on POR only or Cold Reset which would include LVD/HVD sources.. When configuring LPTMR registers, you must initially write to Control Status (CSR) with LPTMR disabled, before configuring Prescaler and Glitch Filter (PSR) and Compare (CMR). Then, you must write 1 to CSR[TEN] 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 POR only or Cold Reset which would include LVD/HVD sources..

34.3.4 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

You must not alter the prescaler and glitch filter configuration when LPTMR is enabled.

34.3.4.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks Counter (CNR). When LPTMR is enabled, CNR increments every 2^1 to 2^{16} prescaler clock cycles. After LPTMR is enabled, the first increment of CNR takes an additional one or two prescaler clock cycles because of the synchronization logic.

34.3.4.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments Counter (CNR) on every clock cycle. When LPTMR is enabled, the first increment takes an additional one or two prescaler clock cycles because of the synchronization logic.

34.3.4.3 Glitch filter enabled

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks Counter (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 184. 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 sampled only on the rising clock edge.

The value of CNR increments each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which CNR can increment is once every 2^2 to 2^{16} glitch filter clock edges. When first enabled, the glitch filter waits for an additional one or two glitch filter clock edges because of the synchronization logic.

34.3.4.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the value of Counter (CNR) every time it asserts. Before LPTMR is first enabled, the selected input source is forced to be asserted. This prevents CNR from incrementing if the selected input source is already asserted when LPTMR is first enabled.

34.3.5 Counter

The value of Counter (CNR) increments by 1 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.

CNR is reset when LPTMR is disabled or if the counter register overflows. If CSR[TFC] = 0, then CNR is also reset whenever CSR[TCF] = 1.

When the core is halted in Debug mode:

- CNR continues incrementing if configured for Pulse Counter mode.
- CNR stops incrementing if configured for Time Counter mode.

You cannot initialize CNR but can read it at any time. On each read of CNR, you must first write a value to it. This synchronizes and registers the current value of CNR into a temporary register. The contents of the temporary register are returned on each read of CNR.

When reading 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.

34.3.6 Compare

After the next Counter (CNR) increment (when its value is equal to that of Compare (CMR)), the following events occur:

- CSR[TCF] is read as 1b.
- LPTMR generates an interrupt if CSR[TIE] is 1 as well.
- LPTMR generates a hardware trigger.
- LPTMR writes 0 to CNR if CSR[TFC] is 0.

When LPTMR is enabled, you can modify the value of CMR only when CSR[TCF] is 1. When updating CMR, you must write to it and clear CSR[TCF] before the LPTMR counter increments past the new LPTMR compare value.

NOTE

When LPTMR is enabled in Time Counter mode, the first increment takes an additional one or two clock cycles because of the synchronization logic. This results in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster prescaler clock or larger prescaler value minimizes this impact.

34.3.7 Interrupt

LPTMR generates an interrupt whenever CSR[TIE] and CSR[TCF] are 1. CSR[TCF] is cleared by disabling LPTMR or writing a logic 1 to it.

You can modify the value of 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.

34.3.8 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 your intervention. The hardware trigger is always enabled.

Table 185. Hardware trigger

When	Then
CMR[COMPARE] and CSR[TFC] are 0	The LPTMR hardware trigger asserts on the first compare and does not deassert.
CMR[COMPARE] 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 Counter (CNR).

34.4 External signals

Table 186. 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.		Input
	State meaning	Assertion—If configured for Pulse Counter mode with an active-high input, assertion causes Counter (CNR) to increment. Deassertion—If configured for Pulse Counter mode with an active-low input, deassertion causes CNR to increment.	
	Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.	

34.5 Initialization

Perform the following procedure to initialize LPTMR:

1. Configure Control Status (CSR) for the selected mode and pin configuration, when CSR[TEN] is 0. This resets the counter and clears the flag.
2. Configure Prescaler and Glitch Filter (PSR) with the selected clock source and prescaler or glitch filter configuration.
3. Configure Compare (CMR) with the selected compare point.
4. Write 1 to CSR[TEN] to enable LPTMR.

34.6 Application information

34.6.1 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 a 32.768 kHz clock source by configuring PSR[PCS].
3. Bypass the prescaler and glitch filter by writing 1 to PSR[PBYP].
4. Assert an 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].

NOTE

This is just an example. See the chip-specific LPTMR information for the clocks supported on a given chip.

34.6.2 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 a 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] = 0Eh.
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].

NOTE

This is just an example. See the chip-specific LPTMR information for the clocks supported on a given chip.

34.7 Memory map and register definition

NOTE

The LPTMR registers are reset only on POR only or Cold Reset which would include LVD/HVD sources.. See Reset for more information.

34.7.1 LPTMR register descriptions

34.7.1.1 LPTMR memory map

LPTMR0 base address: 400A_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Status (CSR)	32	RW	0000_0000h
4h	Prescaler and Glitch Filter (PSR)	32	RW	0000_0000h
8h	Compare (CMR)	32	RW	0000_0000h
Ch	Counter (CNR)	32	RW	0000_0000h

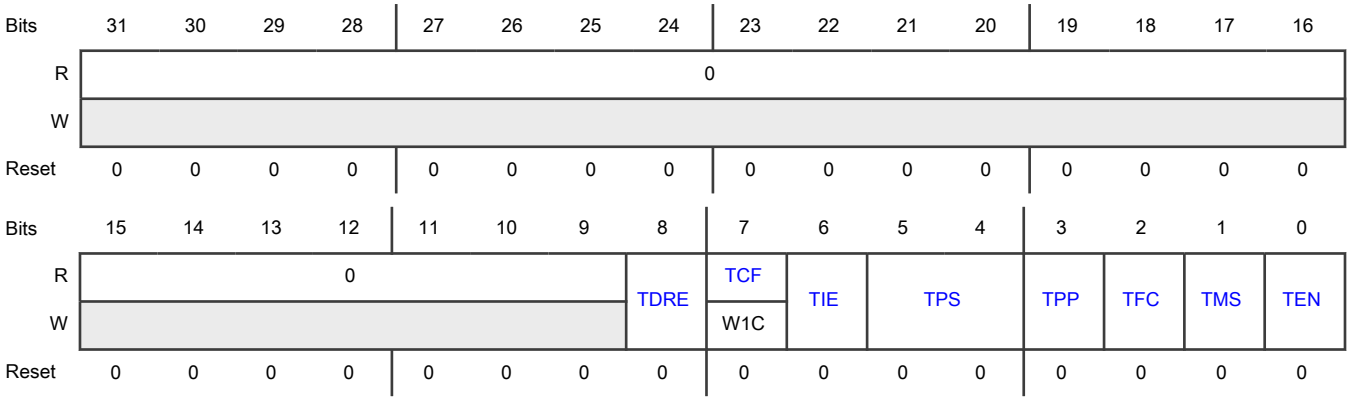
34.7.1.2 Control Status (CSR)

Offset

Register	Offset
CSR	0h

Function
Controls various features of LPTMR.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 TDRE	Timer DMA Request Enable Enables the timer DMA request. When TDRE is 1, the LPTMR DMA request is generated whenever CSR[TCF] is also set. Then, CSR[TCF] is cleared after the DMA controller completes execution. 0b - Disable 1b - Enable
7 TCF	Timer Compare Flag Compares the timer. TCF sets on the next Counter (CNR) increment when LPTMR is enabled and Counter (CNR) equals Compare (CMR). TCF is cleared when LPTMR is disabled or a logic 1 is written to it. **NOTE** You must clear this flag before enabling the timer interrupt or DMA request. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0b - CNR ≠ (CMR + 1) 1b - CNR = (CMR + 1) When writing 0b - No effect 1b - Clear the flag
6 TIE	Timer Interrupt Enable Enables the timer interrupt. If TIE is 1, then LPTMR generates an interrupt if CSR[TCF] is 1 as well. 0b - Disable 1b - Enable
5-4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. The input connections vary by chip. For details, see the chip configuration information about connections to these inputs. You must modify this field only when LPTMR is disabled. 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. If TPP is 0, then the pulse counter input source is active-high, and Counter (CNR) increments on the rising edge. If TPP is 1, then the pulse counter input source is active-low, and CNR increments on the falling edge. You must modify this field only when LPTMR is disabled. 0b - Active-high 1b - Active-low
2 TFC	Timer Free-Running Counter Specifies when the counter resets. If TFC is 0, Counter (CNR) resets on the count cycle following Counter (CNR) becoming equal to Compare (CMR). If TFC is 1, CNR resets on overflow. In both cases, CSR[TCF] sets to 1 on the cycle after CNR and CMR match. You must modify this field only when LPTMR is disabled. 0b - Reset when TCF asserts 1b - Reset on overflow
1	Timer Mode Select Configures the mode of LPTMR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
TMS	You must modify this field only when LPTMR is disabled. 0b - Time Counter 1b - Pulse Counter
0 TEN	Timer Enable Enables the LPTMR timer. If TEN is 0, it resets the LPTMR internal logic, including CNR[COUNTER] and CSR[TCF]. If TEN is 1, LPTMR is enabled. Do not alter CSR[5:1] when writing 1 to this field. 0b - Disable 1b - Enable

34.7.1.3 Prescaler and Glitch Filter (PSR)

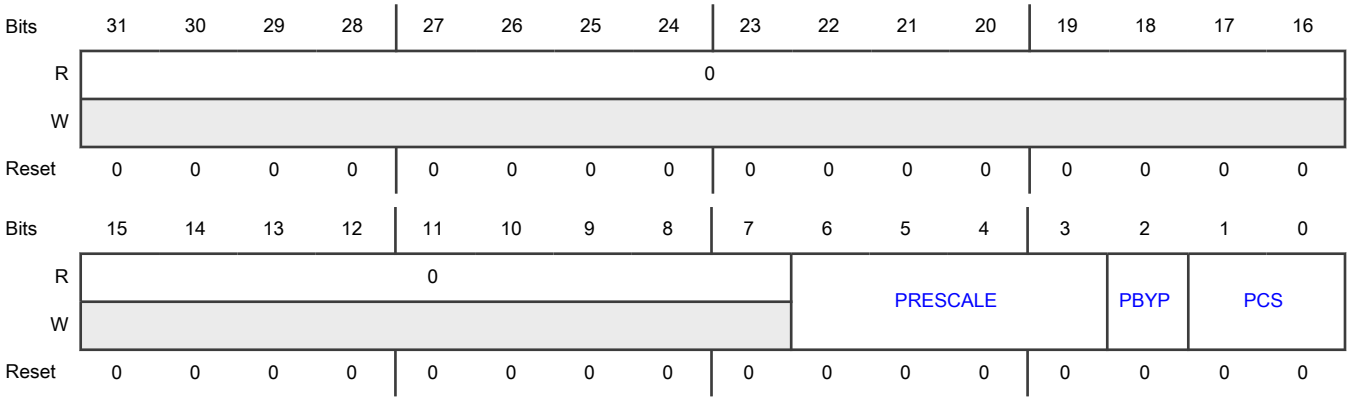
Offset

Register	Offset
PSR	4h

Function

Configures features related to the prescaler and glitch filter.

Diagram



Fields

Field	Function
31-7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-3 PRESCALE	Prescaler and Glitch Filter Value Configures the size of the prescaler in Time Counter mode and the width of the glitch filter in Pulse Counter mode. The width of the glitch filter can vary by one cycle because of the pulse counter input synchronization. You must modify this field 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 two rising clock edges 0010b - Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after four rising clock edges 0011b - Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after eight rising clock edges 0100b - Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges 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 and Glitch Filter Bypass Controls the clocking of Counter (CNR). If PBYP is 0, the output of the prescaler or glitch filter clocks CNR. If PBYP is 1, the selected prescaler clock in Time Counter mode, or else the selected input source in Pulse Counter mode, directly clocks CNR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You must modify this field only when LPTMR is disabled. 0b - Prescaler and glitch filter enable 1b - Prescaler and glitch filter bypass
1-0 PCS	Prescaler and Glitch Filter Clock Select Selects the clock to be used by the LPTMR prescaler and glitch filter. 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. See the chip configuration details for information on connections to these inputs. You must modify this field only when LPTMR is disabled. 00b - Clock 0 01b - Clock 1 10b - Clock 2 11b - Clock 3

34.7.1.4 Compare (CMR)

Offset

Register	Offset
CMR	8h

Function

Configures the compare values to Counter (CNR) (see Compare for more information).

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 Configures the compare values to Counter (CNR). On the next CNR increment, if LPTMR is enabled and Counter (CNR) equals Compare (CMR), then: 1. LPTMR writes 1 to CSR[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 modify the value of CMR only if CSR[TCF] is 1.

34.7.1.5 Counter (CNR)

Offset

Register	Offset
CNR	Ch

Function
Specifies counter values (see Counter for more information).

Diagram

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

Fields

Field	Function
31-0 COUNTER	Counter Value Contains the current value of the LPTMR counter at the time you last wrote to this register.

Chapter 35
USB Full Speed (USBFS) Device Controller

35.1 Chip-specific USBFS information

Table 187. Reference links to related information

Topic	Related module	Reference
Full description	USBFS	USBFS
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

35.1.1 Module instances

This device has one instance of the USBFS module, USBFS0 and one USB FS PHY. Only Device mode is supported on this device. Any references to the host mode should be ignored.

35.2 Overview

USBFS comprises a Device-only subsystem that provides functionality for implementing a USB 2.0 Full-Speed compliant peripheral.

In this subsystem, the USB full-speed controller interfaces to a USB FS/LS transceiver and Memory map and register definitions also includes configuration controls for that transceiver.

In Device mode, the USBFS subsystem working with the system clock module can use the FIRC to generate a 48 MHz USB controller clock tuned using the incoming host USB data for crystal-less operation. Memory map and register definitions also include the registers necessary for configuration controls and status signals of the USB portion of this crystal-less operation.

35.2.1 Block diagram

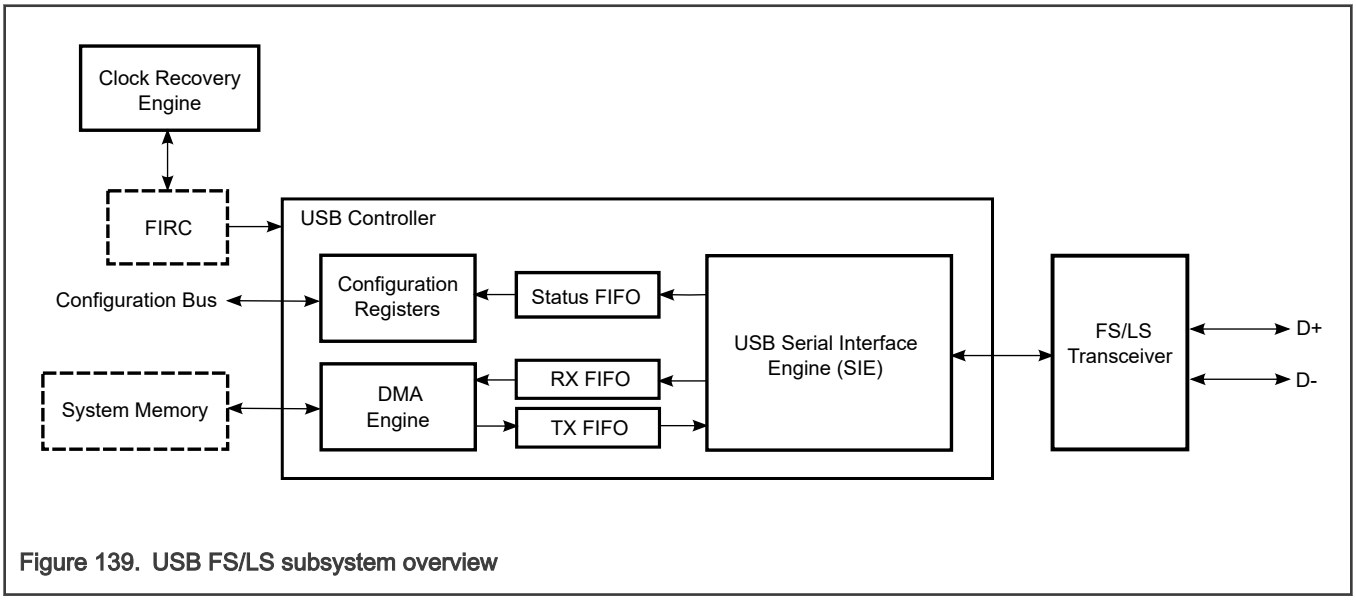


Figure 139. USB FS/LS subsystem overview

35.2.2 Features

- Includes a dual-role USB OTG-capable controller that supports Full-Speed (FS) device operation. The controller operates with DMA or FIFO data stream interfaces and supports up to 16 bidirectional endpoints. The module complies with the USB 2.0 specification.
- Supports a USB transceiver that includes a 1.5 kΩ pullup on the D+ line for Device mode functionality.
- Includes FIRC with clock recovery block to eliminate the need for a clock from a 48 MHz crystal. The USB frequency tuning feature is available only for USB device mode operation.
- A configurable connection allows any one of the LPUART transmit and receive pins to be connected to the Full Speed USB physical layer.
- Supports lane reversal feature to allow switching the roles of package USB0_DM and USB0_DP pins providing additional flexibility for PC board layout.

35.2.3 USB interface general overview

The USB is a cable bus that supports data exchange between a host computer and a wide range of simultaneously accessible peripherals. The bus allows peripherals to be attached, configured, used, and detached when the host and other peripherals operate. The attached peripherals share USB bandwidth through a host-scheduled, token-based protocol.

USB software provides a uniform system view for all application software, hiding implementation details to make application software more portable. It manages the dynamic attaching and detaching of peripherals.

There is only one host in any USB system. The USB interface to the host computer system is referred to as the host controller.

Any system may have multiple USB devices, such as human interface devices, speakers, printers, etc. USB devices present a standard USB interface regarding comprehension, response, and standard capability.

The host initiates transactions to specific peripherals, whereas the device responds to control transactions. The device sends and receives data to and from the host using a standard USB data format. USB 2.0 full-speed/low-speed peripherals operate at 12 Mbit/s or 1.5 Mbit/s.

See the USB 2.0 specification for more information.

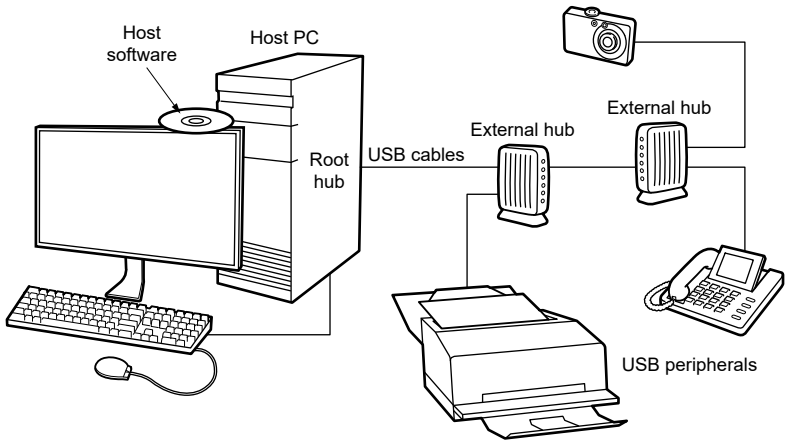


Figure 140. Example USB 2.0 system configuration

35.2.4 References

The following publications are referenced in this document. For copies or updates to these specifications, see the USB Implementers Forum, Inc. website at http://www.usb.org.

- *Universal Serial Bus Specification, Revision 2.0*, 2000, with amendments including those listed below

- *Errata for "USB Revision 2.0 April 27, 2000" as of 12/7/2000*
- *Errata for "USB Revision 2.0 April 27, 2000" as of May 28, 2002*
- *Pullup / Pulldown Resistors* (USB Engineering Change Notice)
- *Suspend Current Limit Changes* (USB Engineering Change Notice)
- *Device Capacitance* (USB Engineering Change Notice)
- *USB 2.0 Connect Timing Update* (USB Engineering Change Notice as of April 4, 2013)
- *USB 2.0 VBUS Max Limit* (USB Engineering Change Notice)
- *On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification*, Revision 2.0 version 1.1a, July 27, 2012
- *Maximum VBUS Voltage* (USB OTGEH Engineering Change Notice)
- *Universal Serial Bus Micro-USB Cables and Connectors Specification*, Revision 1.01, 2007

35.3 Functional description

35.3.1 On-chip transceiver required external components

USB system operation requires external components to ensure that you have met the driver output impedance, eye diagram, and VBUS cable fault tolerance requirements. DM and DP I/O pads must connect through series resistors (approximately 33 Ω each) to the USB connector on the application printed circuit board (PCB). Additionally, signal quality optimizes when you mount these 33 Ω resistors closer to the processor than the USB board-level connector. The USB transceiver includes the following:

- An internal 1.5 kΩ pullup resistor on the USB_DP line for the full-speed device (CONTROL[DPPULLUPNONOTG] controls).

The following diagram provides an overview of Device-only connections. See *Kinetis Peripheral Module Quick Reference (KQRUG)*, available on www.nxp.com, for more information.

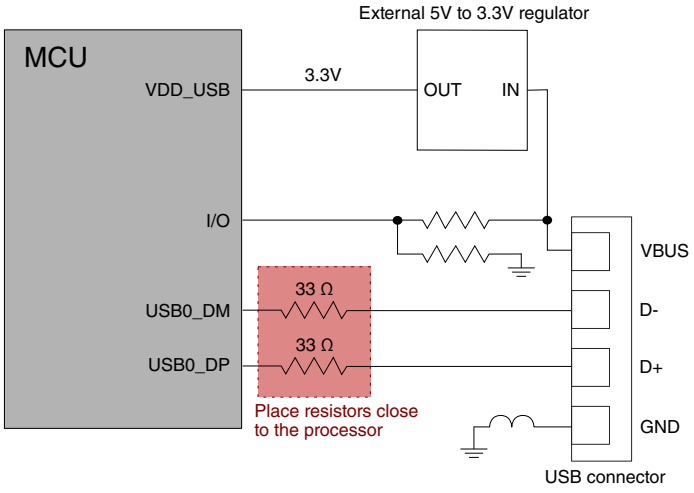


Figure 141. Typical Device-only block diagram (bus-powered with external regulator)

35.3.2 Programmer interface

USBFS communicates with the processor core through status registers, control registers, and data structures in memory. This section discusses the major components of the programming model for USBFS.

35.3.2.1 Buffer descriptor table (BDT)

The device operation's function is to transfer a request in the memory image to and from the USB. USBFS implements a BDT to manage USB endpoint communications in system memory efficiently.

- The BDT resides on a 512-byte boundary in system memory and is pointed to by the BDT page registers.
- Every endpoint direction requires two 8-byte buffer descriptor (BD) entries.

Therefore, a system with 16 fully bidirectional endpoints requires 512 bytes of system memory to implement the BDT. The two BD entries allow for an EVEN BD and ODD BD entry for each endpoint direction, which enables the microprocessor to process one BD while USBFS is processing the other BD. Double buffering BDs allow USBFS to transfer data easily at the maximum throughput provided by USB.

You must manage buffers for USBFS by updating the BDT when needed. This allows USBFS to efficiently manage data transmission and reception while the microprocessor performs communication overhead processing and other function-dependent applications.

Because the buffers are shared between the microprocessor and USBFS, a simple semaphore mechanism distinguishes who is allowed to update the BDT and buffers in system memory. A semaphore (the OWN field) becomes 0 when the microprocessor owns the BD entry.

- When OWN = 0, the microprocessor is allowed read and write access to the BD entry and the buffer in system memory.
- When OWN = 1, USBFS owns the BD entry and the buffer in system memory. USBFS now has full read and write access, and the microprocessor must not modify the BD or its corresponding data buffer.

The BD also contains indirect address pointers where the actual buffer resides in system memory. The following diagram shows this indirect address mechanism.

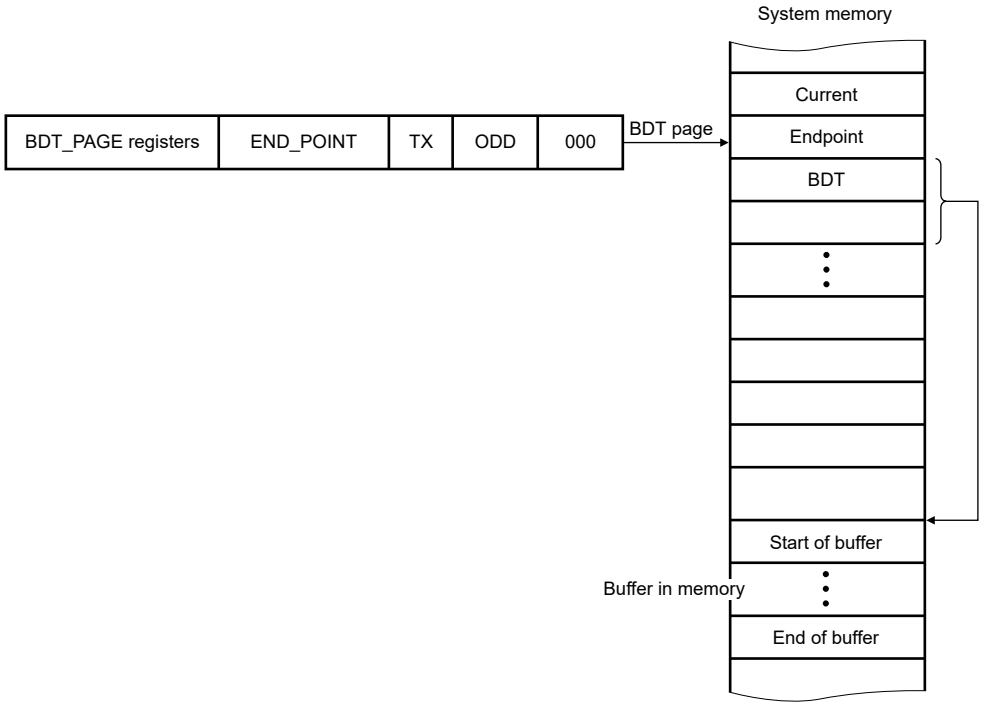


Figure 142. Buffer descriptor table

35.3.2.2 Addressing BDT entries

An understanding of the addressing mechanism of BDT is useful when accessing endpoint data via USBFS or microprocessor:

- The BDT occupies up to 512 bytes of system memory.
- You can support 16 bidirectional endpoints with a full BDT of 512 bytes.
- 16 bytes are needed for each USB endpoint direction.
- Applications with fewer than 16 endpoints require less RAM to implement the BDT.
- The BDT page registers (BDT_PAGE) point to the starting location of the BDT.
- You must locate the BDT on a 512-byte boundary in system memory.
- All enabled transit and receive endpoint BD entries are indexed into the BDT to allow easy access via USBFS or MCU core.

When you receive a USB token on an enabled endpoint, USBFS uses its integrated DMA controller to interrogate the BDT. USBFS reads the corresponding endpoint BD entry to determine whether it (USBFS) owns the BD and corresponding buffer in system memory.

To compute the entry point into the BDT, the BDT_PAGE registers are concatenated with the current endpoint and the TX and ODD fields to form a 32-bit address. This following tables show this address mechanism:

Table 188. BDT address calculation

31:24	23:16	15:9	8:5	4	3	2:0
BDT_PAGE_03	BDT_PAGE_02	BDT_PAGE_01[7:1]	Endpoint	TX	ODD	000

Table 189. BDT address calculation fields

Field	Description
BDT_PAGE	BDT_PAGE registers in the control register block
ENDPOINT	ENDPOINT field from the USB TOKEN
TX	1 for transmit transfers; 0 for receive transfers
ODD	It corresponds to the buffer currently in use; the buffers are used in a ping-pong fashion. Maintained within the USBFS SIE.

35.3.2.3 Buffer descriptors (BDs)

A buffer descriptor provides endpoint buffer control information for USBFS and the processor. The BDs have different meanings, depending on whether USBFS or the processor reads the BD in memory.

The USBFS controller uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- Whether to release ownership upon packet completion
- No address increment (FIFO mode)
- Whether data toggle synchronization is enabled
- How much data to transmit or receive
- Where the buffer resides in system memory

The processor uses the data stored in the BDs to determine:

- Who owns the buffer in system memory

- Data0 or Data1 PID
- The received TOKEN PID
- How much data was transmitted or received
- Where the buffer resides in system memory

The format for the BD is shown in the following table.

Table 190. Buffer descriptor format

31:26	25:16	15:8	7	6	5	4	3	2	1	0
RSVD	BC (10 bits)	RSVD	OWN	DATA0/1	KEEP/ TOK_PID[3]	NINC/ TOK_PID[2]	DTS/ TOK_PID[1]	BDT_STALL/ TOK_PID[0]	0	0
Buffer Address (32 bits)										

Table 191. Buffer descriptor fields

Field	Description
31–26 RSVD	Reserved
25–16 BC	Byte Count Represents the 10-bit byte count. USBFS SIE changes this field upon completing a receive transfer with the byte count of the data received.
15–8 RSVD	Reserved
7 OWN	Determines whether the processor or USBFS currently owns the buffer. Except when KEEP = 1, the SIE hands ownership back to the processor after completing the token by writing 0 to this field. This must always be the last byte of the BD that the processor updates when it initializes a BD. 0 The processor has access to the BD. USBFS ignores all other fields in the BD. 1 USBFS has access to the BD. When USBFS owns the BD, the processor must not modify any other fields in the BD.
6 DATA0/1	Defines whether you transmit or receive a DATA0 field (DATA0/1=0) or a DATA1 (DATA0/1=1) field. It is unchanged by USBFS.
5 KEEP/ TOK_PID[3]	Has two functions (KEEP or TOK_PID[3]): • KEEP—When written by the processor, it serves as the KEEP bit. Typically, this bit is 1 with ISO endpoints feeding a FIFO. The microprocessor is not informed that a token has been processed. Transfer the data to or from the FIFO. When KEEP becomes 1, the NINC field usually becomes 1 to prevent address increment. 0 Allows USBFS to release the BD when a token has been processed. 1 This field is unchanged by USBFS. Bit 3 of the current token PID is written back to the BD by USBFS.

Table continues on the next page...

Table 191. Buffer descriptor fields (continued)

Field	Description
	• TOK_PID[3]—If the OWN field also becomes 1, then the BD remains owned by USBFS indefinitely; when written by USB, it serves as TOK_PID[3]. Typically, this field is 1 with ISO endpoints feeding a FIFO. The microprocessor is not informed that a token has been processed. Transfer the data to or from the FIFO. When KEEP becomes 1, the NINC field usually becomes 1 to prevent address increment. 0 or 1 USBFS writes back bit 3 of the current token PID to the BD.
4 NINC/ TOK_PID[2]	No Increment (NINC) Disables the DMA engine address increment. This forces the DMA engine to read or write from the same address. This is useful for endpoints when data is *read from* or *written to* a single location (such as a FIFO). Typically this field becomes 1 with the KEEP field for ISO endpoints that are interfacing to a FIFO. 0 USBFS writes bit 2 of the current token PID to the BD. 1 This bit is unchanged by USBFS.
3 DTS/ TOK_PID[1]	Enables USBFS to perform data toggle synchronization when you write 1 to this field. • If KEEP = 0, then bit 1 of the current token PID is written back to the BD. • If KEEP = 1, then this field is unchanged by USBFS. 0 Data toggle synchronization is disabled. 1 Enables USBFS to perform data toggle synchronization.
2 BDT_STALL TOK_PID[0]	Causes USBFS to issue a STALL handshake if the SIE receives a token that uses the BDT in this location when you write 1 to this field. The SIE does not consume the BDT (the OWN field remains 1 and the rest of the BDT is unchanged) when a BDT_STALL field becomes 1. • If KEEP = 0, write back bit 0 of the current token PID to the BD. • If KEEP = 1, then this field is unchanged by USBFS. 0 No stall issued. 1 The SIE does not consume the BDT (the OWN field remains 1, and the rest of the BDT is unchanged). Writing 1 to BDT_STALL also causes the corresponding USB_ENDPTn[EPSTALL] to become 1. This causes USB OTG to issue a STALL handshake for both directions of the associated endpoint. To clear the stall condition: 1. Write 0 to the associated USB_ENDPTn[EPSTALL]. 2. Write the BDT to clear OWN and BDT_STALL.
TOK_PID[n]	Bits [5:2] can also represent the current token PID. When a transfer completes, USBFS writes back the current token PID into the BD. The values written back are the token PID values from the USB specification: • 1h for an OUT token. • 9h for an IN token. • Dh for a SETUP token.

Table continues on the next page...

Table 191. Buffer descriptor fields (continued)

Field	Description
1–0 Reserved	Reserved
ADDR[31:0]	Address Represents the 32-bit buffer address in system memory; this address must also be 32-bit aligned. USBFS does not change this field.

35.3.2.4 USB transaction

When USBFS transmits or receives data, it computes the BDT address using the address generation shown in Addressing BDT entries.

If OWN = 1, then the following process occurs:

1. USBFS reads the BDT.
2. The SIE transfers the data via the DMA to or from the buffer pointed to by Address (ADDR) of the BD.
3. When the TOKEN completes, USBFS updates the BDT, and if KEEP = 0, then USBFS changes the OWN field to 0.
4. Update the STAT register, and the TOK_DNE interrupt occurs.
5. When the processor processes the TOK_DNE interrupt, it reads from the status register all the information required to process the endpoint.
6. At this point, the processor allocates a new BD so that you can transmit or receive the additional USB data for that endpoint and then processes the last BD.

The following figure shows how a typical USB token is processed after the BDT is read and OWN = 1.

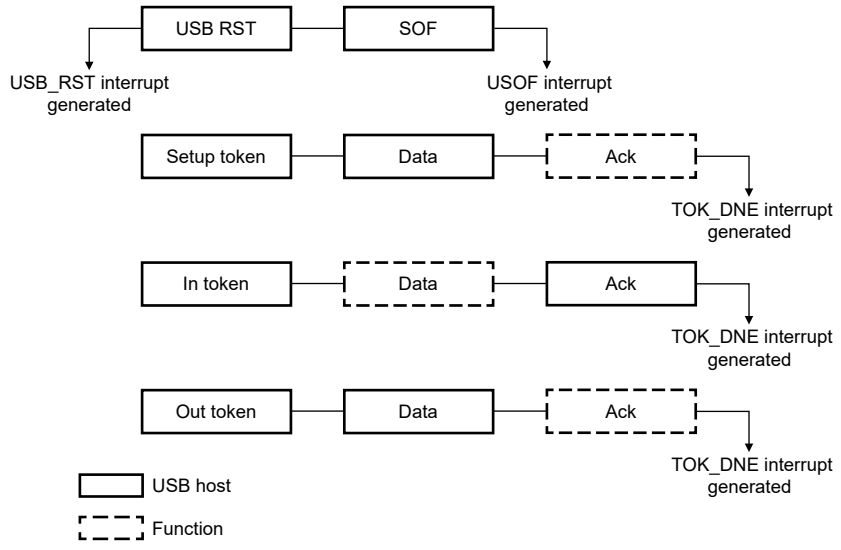


Figure 143. USB token transaction

USB has two sources for the DMA overrun error:

- **Memory latency:** The memory latency may be too high and cause the receive FIFO to overflow. This is predominantly a hardware performance issue, usually caused by transient memory access issues.

- **Oversized packets:** The packet received may be larger than the negotiated MaxPacket size. Typically, a software bug causes this. The USB specification is ambiguous for DMA overrun errors due to oversized data packets because it assumes correct software drivers on both sides. NAKing the packet can result in the retransmission of the already oversized packet data. Therefore, in response to oversized packets, the USB core continues ACKing the packet for non-isochronous transfers.

Table 192. USB responses to DMA overrun errors

Errors due to memory latency	Errors due to oversized packets
Non-acknowledgment (NAK) or bus timeout (BTO) — See bit 4 in Error Interrupt Status (ERRSTAT) as appropriate for the class of transaction.	Continues acknowledging (ACKing) the packet for non-isochronous transfers.
—	Clip the data written to memory to the MaxPacket size so as not to corrupt system memory.
ERRSTAT[DMAERR] becomes 1 for Device mode of operation. Depending on the INTENB and ERRENB register's values, the core may assert an interrupt to notify the processor of the DMA error.	Asserts ERRSTAT[DMAERR], which can trigger an interrupt and ISTAT[TOKDNE] fires. The TOK_PID field of the BDT is not 1111 because the DMAERR is not due to latency.
• In Device mode: The BDT is not written back nor ISTAT[TOKDNE] triggered because it is assumed that a second attempt is queued and will succeed in the future.	The packet length field written back to the BDT is the MaxPacket value representing the length of the clipped data written to memory.
From here, you can decide an appropriate course of action for future transactions, such as stalling the endpoint, canceling the transfer, disabling the endpoint, and others.	

35.3.3 Operations

35.3.3.1 Device mode FIRC operation

The following are the FIRC initialization code steps for USB crystal-less Device mode operation:

1. Enable the FIRC nominal 48 MHz clock by ensuring that the FIRCCSR[FIRCEN] and FIRCTCFG[TRIMSRC] fields in the SCG registers have the correct values for crystal-less FS USB Device mode operation.
2. Enable the USB clock recovery tuning by writing 1 to CLK_RECOVER_CTRL[CLOCK_RECOVER_EN].
3. Complete the configuration of the USB clock recovery logic by writing 1 to CLK_RECOVER_IRC_EN[IRC_EN].
4. This chip does not have alternate USB input clock sources from clock muxes, so no additional 48 MHz clock path configuration is required.

35.3.3.2 USB and VREGIN pin status detection

This device does not have a dedicated VBUS detect pin. Still, it does have the capability through CONTROL[SESS_VLD] to report the status of VBUS using one of two monitoring methods with different package pins. The choice between the two VBUS monitoring methods is made with CONTROL[VBUS_SOURCE_SEL]. For VBUS detection, use a specified GPIO pin for both bus-powered and self-powered USB cases. If the GPIO pins on this device do not directly support a 5 V input, use an external resistive voltage divider to keep the input voltage within the valid range if you use the GPIO pin method for VBUS detection. The voltage status of the VDD_LDO_USB pin is passed internally from the USB VREG to the USB FS subsystem, so no additional board connections are required if you choose the VBUS detection method.

See chip-specific information for the specified GPIO package pin for VBUS monitoring.

This device does not have a dedicated OTG ID detect pin. For OTG ID pin detection, if needed, use one of the GPIO pins connected to the WUU configured as an input pin with pullup enabled.

35.3.3.3 LPUART over USB capability

This device supports a connection whereby any LPUART may connect to the FS USB physical layer (and the USB controller disconnected) to allow simple debugging capabilities for applications that do not have direct physical access to other LPUART Tx/Rx pins.

When you write 1 to USBCTRL[UARTSEL], the selected LPUART connects to the FS USB physical layer. This configures the FS USB0_DP/DM signals to configure as normal LPUART signals, which do not operate in differential mode. When USBCTRL[UARTCHLS] becomes 0, FS USB0_DP/DM signal pins are configured for LPUART Tx/Rx functions respectively. When USBCTRL[UARTCHLS] becomes 1, FS USB0_DP/DM signal pins are configured for LPUART Rx/Tx functions.

35.3.3.4 Lane reversal feature

This device supports lane reversal for the USB0_DP and USB0_DM pins for flexibility in PC board design depending on the USB receptacle type and location choices.

USBCTRL[DPDM_LANE_REVERSE] controls the lane reversal. After USBCTRL[DPDM_LANE_REVERSE] becomes 0, the USB0_DP and USB0_DM external package pins have their normal roles. After USBCTRL[DPDM_LANE_REVERSE] becomes 1, the I/O pin roles are switched so that the USB0_DP external package pin operates as USB_DM and the USB0_DM external package pin operates as USB_DP.

Lane reversal impacts all USB standard data communication functions along with battery charging detection and signaling. Lane reversal does **not** impact LPUART over USB mode selections for USB0_DP/DM pin functions because there is already a pin configuration for that mode using USBCTRL[UARTCHLS].

35.3.4 Modes of operation

USBFS includes the following main modes of operation:

- Active mode
- Low-Power mode

35.3.4.1 Active mode

This is the basic mode of operation.

A USBFS core-centric nomenclature is used to describe the direction of the data transfer between USBFS core and USB. Throughout this chapter:

- Receive (or "Rx") describes transfers that move data from USB to memory.
- Transmit (or "Tx") describes transfers that move data from memory to USB.

The following table shows the data directions for USBFS.

Table 193. Data direction for USB host or USB target

Rx	Tx
OUT or SETUP	IN

35.3.4.2 Low-Power mode

The USB controller supports Sleep mode, Deep Sleep mode, and Power Down mode to save power consumption.

When the USB subsystem detects no activity on the USB bus for more than 3 ms, ISTAT[SLEEP] becomes 1. This field can cause an interrupt, and you decide on the appropriate action.

Waking from a Sleep Low-Power mode when the USB subsystem is powered occurs through an asynchronous interrupt triggered by activity on the USB bus. Writing 1 to USBTRC0[USBRESMEN] enables this function.

The following wakeup features are also supported:

- In Active and Sleep modes, the USB controller can generate an interrupt on VREGIN detection using MISCCTRL[VFEDG_EN], MISCCTRL[VREDG_EN], USBTRC0[VFEDG_DET], and USBTRC0[VREDG_DET].
- In Deep Sleep and Power Down modes, VDD_LDO_USB is an input pin to WUU, and a transition on it can generate a wakeup.
- In Deep Sleep and Power Down modes, the GPIO pins chosen to detect VBUS and OTG ID are also input to WUU, so transitions on them can generate a wakeup.

In Deep Power Down mode, the pads become isolated, and USB bus activity cannot wake the system.

35.3.5 Clocking

This section describes clocks and special clocking requirements of USBFS.

Table 194. Clocking

Clock name	Description
Bus clock	The bus clock is system-dependent. It must be greater than 24 MHz.
Function clock	USB 48 MHz functional clock

35.3.6 Reset

Chip reset

The logic and registers for USBFS reset to their default states on a chip reset.

Software reset

USBFS implements a software reset field in its control register. See USBTRC0[USBRESET] for more information.

35.3.7 Interrupts

The following table illustrates the status flags that can generate USBFS interrupts.

Table 195. Interrupts

Flag	Description	Low-power wakeup
STALL	Stall handshake flag	No
RESUME	Resume flag	No
SLEEP	Sleep flag	No
TOKDNE	Token processing done flag	No
SOFTOK	SOF token flag	No
ERROR	Error flag	No
USBRST	USB reset flag	No
BTSERR	Bit stuff error flag	No
OWNERR	BD unavailable error flag	No
DMAERR	DMA access error flag	No
BTOERR	Bus turnaround timeout error flag	No
DFN8	Data field bit width error flag	No

Table continues on the next page...

Table 195. Interrupts (continued)

Flag	Description	Low-power wakeup
CRC16	CRC16 error flag	No
CRC5EOF	EOF error flag	No
PIDERR	PID error flag	No
USB_RESUME_INT	Asynchronous resume wakeup flag	Yes
VFEDG_DET	VREGIN falling edge flag	Yes
VREDG_DET	VREGIN rising edge flag	Yes
OVF_ERROR	Frequency trim adjustment flag	No

35.3.8 DMA

USBFS provides an integrated DMA controller that transfers the packet data to and from memory. The DMA controller allows USB endpoint data to transfer very efficiently, limiting processor involvement.

35.4 External signals

This section describes clocks and special clocking requirements of USBFS.

Table 196. External signals

Signal	Description
USB_DP	USB D+ pin
USB_DM	USB D- pin
VDD_USB	3.3 V power supply for USB transceiver

35.5 Initialization

For Device mode initialization, see Device mode operation examples.

35.6 Application information

35.6.1 Device mode operation examples

The following sections list the steps required to perform USB device functions using the USBFS core.

To enable Device mode and discover a connected USB host:

1. Pullup DP (CONTROL[DPPULLUPNONOTG] = 1b).
2. Disable weak pulldown on DP and DM (USBCTRL[PDE] = 0b).
3. Disable suspend (USBCTRL[SUSP] = 1'b0).
4. Clear status flags.
5. Enable USBRST interrupt (INTEN[USBRSTEN] = 1b).
6. Enable USBFS (CTL[USBENSOFEN] = 1'b1).
7. Wait for reset request from USB host (ISTAT[USBRST] = 1b).
8. Detect USBRST interrupt, device connected to USB host.

To complete a packet transfer:

1. Complete all the steps to discover a connected USB host.
2. Set up the endpoint control register for packet transfer (ENDPT0[4:0] = 0dh).
3. Enable TOKDNE interrupt (INTEN[TOKDNEEN] = 1b).
4. Initialize the endpoint BDT in USB RAM including setting OWN = 1b.
5. Clear TXD_SUSPEND flag (write 1 to clear CTL[TXSUSPENDTOKENBUSY]).
6. Wait packet transfer to complete (ISTAT[TOKDNE] = 1b).
7. Detect TOKDNE interrupt, check BDT and status flags.

35.7 Memory map and register definitions

This section provides the memory map and detailed descriptions of all USBFS interface registers.

NOTE

Control (CTL), USB Control (USBCTRL), and USB OTG Control (CONTROL) have similar names but these are all separate registers.

35.7.1 USBFS register descriptions

35.7.1.1 USBFS memory map

USB0 base address: 400A_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Peripheral ID (PERID)	8	R	04h
4h	Peripheral ID Complement (IDCOMP)	8	R	FBh
8h	Peripheral Revision (REV)	8	R	33h
1Ch	OTG Control (OTGCTL)	8	RW	00h
80h	Interrupt Status (ISTAT)	8	RW	00h
84h	Interrupt Enable (INTEN)	8	RW	00h
88h	Error Interrupt Status (ERRSTAT)	8	RW	00h
8Ch	Error Interrupt Enable (ERREN)	8	RW	00h
90h	Status (STAT)	8	R	00h
94h	Control (CTL)	8	RW	00h
98h	Address (ADDR)	8	RW	00h
9Ch	BDT Page 1 (BDTPAGE1)	8	RW	00h
A0h	Frame Number Register Low (FRMNUML)	8	R	00h
A4h	Frame Number Register High (FRMNUMH)	8	R	00h
B0h	BDT Page 2 (BDTPAGE2)	8	RW	00h
B4h	BDT Page 3 (BDTPAGE3)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C0h	Endpoint Control (ENDPT0)	8	RW	00h
C4h	Endpoint Control (ENDPT1)	8	RW	00h
C8h	Endpoint Control (ENDPT2)	8	RW	00h
CCh	Endpoint Control (ENDPT3)	8	RW	00h
D0h	Endpoint Control (ENDPT4)	8	RW	00h
D4h	Endpoint Control (ENDPT5)	8	RW	00h
D8h	Endpoint Control (ENDPT6)	8	RW	00h
DCh	Endpoint Control (ENDPT7)	8	RW	00h
E0h	Endpoint Control (ENDPT8)	8	RW	00h
E4h	Endpoint Control (ENDPT9)	8	RW	00h
E8h	Endpoint Control (ENDPT10)	8	RW	00h
ECh	Endpoint Control (ENDPT11)	8	RW	00h
F0h	Endpoint Control (ENDPT12)	8	RW	00h
F4h	Endpoint Control (ENDPT13)	8	RW	00h
F8h	Endpoint Control (ENDPT14)	8	RW	00h
FCh	Endpoint Control (ENDPT15)	8	RW	00h
100h	USB Control (USBCTRL)	8	RW	C0h
104h	USB OTG Observe (OBSERVE)	8	R	50h
108h	USB OTG Control (CONTROL)	8	RW	00h
10Ch	USB Transceiver Control 0 (USBTRC0)	8	RW	00h
124h	Reserved (KEEP_ALIVE_CTRL_RSVD)	8	RW	08h
128h	Reserved (KEEP_ALIVE_WKCTRL_RSVD)	8	RW	01h
12Ch	Miscellaneous Control (MISCCTRL)	8	RW	00h
130h	Peripheral Mode Stall Disable for Endpoints 7 to 0 in IN Direction (STALL_IL_DIS)	8	RW	00h
134h	Peripheral Mode Stall Disable for Endpoints 15 to 8 in IN Direction (STALL_IH_DIS)	8	RW	00h
138h	Peripheral Mode Stall Disable for Endpoints 7 to 0 in OUT Direction (STALL_OL_DIS)	8	RW	00h
13Ch	Peripheral Mode Stall Disable for Endpoints 15 to 8 in OUT Direction (STALL_OH_DIS)	8	RW	00h
140h	USB Clock Recovery Control (CLK_RECOVER_CTRL)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
144h	FIRC Oscillator Enable (CLK_RECOVER_IRC_EN)	8	RW	01h
154h	Clock Recovery Combined Interrupt Enable (CLK_RECOVER_INT_EN)	8	RW	10h
15Ch	Clock Recovery Separated Interrupt Status (CLK_RECOVER_INT_STATUS)	8	RW	00h

35.7.1.2 Peripheral ID (PERID)

Offset

Register	Offset
PERID	0h

Function

Reads back the value of 04h. This value is defined for the USB peripheral.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0				ID			
W								
Reset	0	0	0	0	0	1	0	0

Fields

Field	Function
7-6 —	Reserved
5-0 ID	Peripheral Identification

35.7.1.3 Peripheral ID Complement (IDCOMP)

Offset

Register	Offset
IDCOMP	4h

Function

Indicates the one's complement of Peripheral ID (PERID). For the USB peripheral, the value is FBh.

Diagram

Bits	7	6	5	4	3	2	1	0
R	1				NID			
W								
Reset	1	1	1	1	1	0	1	1

Fields

Field	Function
7-6 —	Reserved
5-0 NID	Negative Peripheral ID Indicates the one's complement of PERID[ID].

35.7.1.4 Peripheral Revision (REV)

Offset

Register	Offset
REV	8h

Function

Indicates the revision number of USBFS.

Diagram

Bits	7	6	5	4	3	2	1	0
R	REV							
W								
Reset	0	0	1	1	0	0	1	1

Fields

Field	Function
7-0 REV	Revision Contains the USBFS revision number.

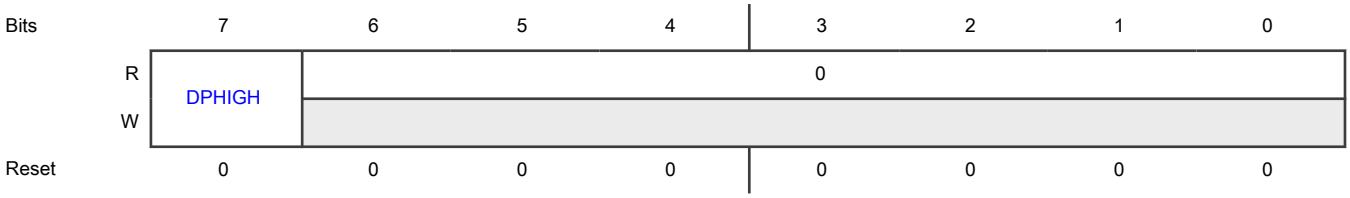
35.7.1.5 OTG Control (OTGCTL)

Offset

Register	Offset
OTGCTL	1Ch

Function
Controls the operation of data line termination resistors.

Diagram



Fields

Field	Function
7 DPHIGH	D+ Data Line Pullup Resistor Enable 0b - Disable 1b - Enable
6-0 —	Reserved

35.7.1.6 Interrupt Status (ISTAT)

Offset

Register	Offset
ISTAT	80h

Function

Contains flag fields for each of the interrupt sources within USBFS. Each of these fields is qualified with their respective interrupt enable fields.

Diagram

Bits	7	6	5	4	3	2	1	0
R	STALL	0	RESUME	SLEEP	TOKDNE	SOFTOK	ERROR	USBRST
W	W1C		W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STALL	Stall Interrupt Flag In Device mode, becomes 1 when SIE sent a STALL handshake. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt did not occur 1b - Interrupt occurred When writing 0b - No effect 1b - Clear the flag
6 —	Reserved
5 RESUME	Resume Flag Indicates when a K-state is observed on the DP or DM signals for 2.5 µs. You must disable this interrupt when not in Suspend mode. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt did not occur 1b - Interrupt occurred When writing 0b - No effect 1b - Clear the flag
4 SLEEP	Sleep Flag Indicates when USBFS detects a constant idle on the USB bus for 3 ms. Activity on the USB bus resets the sleep timer.

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 - Interrupt did not occur 1b - Interrupt occurred When writing 0b - No effect 1b - Clear the flag
3 TOKDNE	Current Token Processing Flag Indicates when the current token being processed has completed. The processor must immediately read Status (STAT) to determine the EndPoint and BD used for this token. Clearing this flag causes one of the following events: • STAT becomes 0. • USBFS loads the STAT holding register into Status (STAT). **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not processed 1b - Processed When writing 0b - No effect 1b - Clear the flag
2 SOFTOK	Start Of Frame (SOF) Token Flag Indicates when USBFS receives a SOF token. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not receive 1b - Received When writing 0b - No effect 1b - Clear the flag
1	Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERROR	Indicates when any of the error conditions within Error Interrupt Status (ERRSTAT) occur. The processor must then read Error Interrupt Status (ERRSTAT) to determine the source of the error. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Error did not occur 1b - Error occurred When writing 0b - No effect 1b - Clear the flag
0 USBRST	USB Reset Flag Indicates when USBFS detects a valid USB reset. This informs the processor that it must write 00h into the address register and enable endpoint 0. This field becomes 1 after a USB reset is detected for 2.5 µs. This field does not become 1 again until the USB reset condition is removed and the USB reset condition is reasserted. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag

35.7.1.7 Interrupt Enable (INTEN)

Offset

Register	Offset
INTEN	84h

Function

Enables interrupt sources within USBFS. Writing 1 to any of these fields enables the respective interrupt source in Interrupt Status (ISTAT).

Diagram

Bits	7	6	5	4	3	2	1	0
R	STALLEN	0	RESUMEEN	SLEEPEN	TOKDNEEN	SOFTOKEN	ERROREN	USBRSTEN
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STALLEN	STALL Interrupt Enable Enables the STALL interrupt. 0b - Disable 1b - Enable
6 —	Reserved
5 RESUMEEN	RESUME Interrupt Enable Enables the RESUME interrupt. 0b - Disable 1b - Enable
4 SLEEPEN	SLEEP Interrupt Enable Enables the SLEEP interrupt. 0b - Disable 1b - Enable
3 TOKDNEEN	TOKDNE Interrupt Enable Enables the TOKDNE interrupt. 0b - Disable 1b - Enable
2 SOFTOKEN	SOFTOK Interrupt Enable Enables the SOFTOK interrupt. 0b - Disable 1b - Enable
1 ERROREN	ERROR Interrupt Enable Enables the ERROR interrupt. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 USBRSTEN	USBRST Interrupt Enable Enables the USBRST interrupt. 0b - Disable 1b - Enable

35.7.1.8 Error Interrupt Status (ERRSTAT)

Offset

Register	Offset
ERRSTAT	88h

Function

Enables interrupt error sources within USBFS. Each of these fields are qualified with their respective error enable fields.

All fields of this register are logically OR'ed together and the result placed in ISTAT[ERROR].

Each field becomes 1 as soon as you detect the error condition; therefore, the interrupt does not typically correspond with the end of a token being processed.

Diagram

Bits		7	6	5	4	3	2	1	0
	R	BTSERR	OWNERR	DMAERR	BTOERR	DFN8	CRC16	CRC5EOF	PIDERR
	W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset		0	0	0	0	0	0	0	0

Fields

Field	Function
7 BTSERR	Bit Stuff Error Flag Indicates when a bit stuff error is detected. If this field is 1, then the corresponding packet is rejected due to the error. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Packet not rejected due to the error 1b - Packet rejected due to the error When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
6 OWNERR	BD Unavailable Error Flag Indicates when USBFS operates in Peripheral mode. This field becomes 1 if USBFS requires a new BD for SETUP, ISO IN, or ISO OUT transfer when a new BD is not available. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt did not occur 1b - Interrupt occurred When writing 0b - No effect 1b - Clear the flag
5 DMAERR	DMA Access Error Flag Indicates if USBFS has requested a DMA access to read a new BDT, but has not been given the bus before it needs to receive or transmit data. • If processing a transit transfer, this causes a transmit data underflow condition. • If processing a receive transfer, this causes a receive data overflow condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt did not occur 1b - Interrupt occurred When writing 0b - No effect 1b - Clear the flag
4 BTOERR	Bus Turnaround Timeout Error Flag Indicates when a bus turnaround timeout error occurs. USBFS contains a bus turnaround timer that keeps track of the amount of time elapsed between the token and data phases of a SETUP or OUT TOKEN or the data and handshake phases of a IN TOKEN. A bus turnaround timeout error occurs if more than 16 bit times are counted from the previous EOP before a transition from IDLE. **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
	0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag
3 DFN8	Data Field Not 8 Bits Flag Indicates if the data field received was not 8 bits in length. USB Specification 1.0 requires that data fields be an integer number of bytes. If the data field is not an integer number of bytes, then this field becomes 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Integer number of bytes 1b - Not an integer number of bytes When writing 0b - No effect 1b - Clear the flag
2 CRC16	CRC16 Error Flag Indicates when a data packet is rejected due to a CRC16 error. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not rejected 1b - Rejected When writing 0b - No effect 1b - Clear the flag
1 CRC5EOF	CRC5 Error or End of Frame Error Flag When USBFS operates in Peripheral mode, this interrupt detects CRC5 errors in the token packets generated by the host. If this field is 1, then the token packet is rejected because of a CRC5 error. **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
	0b - Interrupt did not occur 1b - Interrupt occurred When writing 0b - No effect 1b - Clear the flag
0 PIDERR	PID Error Flag Indicates when the PID check field fails. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not fail 1b - Failed When writing 0b - No effect 1b - Clear the flag

35.7.1.9 Error Interrupt Enable (ERREN)

Offset

Register	Offset
ERREN	8Ch

Function

Enables the error interrupt sources within USBFS.

Writing 1 to any of these fields enables the respective interrupt source in ERRSTAT.

Each field becomes 1 as soon as the error condition is detected; therefore, the interrupt does not typically correspond with the end of a token processed.

Diagram

Bits	7	6	5	4	3	2	1	0
R W	BTSERREN	OWNERREN	DMAERREN	BTOERREN	DFN8EN	CRC16EN	CRC5EOF...	PIDERREN
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 BTSERREN	BTSERR (Bit Stuff Error) Interrupt Enable 0b - Disable 1b - Enable
6 OWNERREN	OWNERR Interrupt Enable This field is valid when USBFS operates in Peripheral mode. 0b - Disable 1b - Enable
5 DMAERREN	DMAERR Interrupt Enable 0b - Disable 1b - Enable
4 BTOERREN	BTOERR (Bus Timeout Error) Interrupt Enable 0b - Disable 1b - Enable
3 DFN8EN	DFN8 (Data Field Not Integer Number of Bytes) Interrupt Enable 0b - Disable 1b - Enable
2 CRC16EN	CRC16 Interrupt Enable 0b - Disable 1b - Enable
1 CRC5EOFEN	CRC5/EOF Interrupt Enable 0b - Disable 1b - Enable
0 PIDERREN	PIDERR Interrupt Enable 0b - Disable 1b - Enable

35.7.1.10 Status (STAT)

Offset

Register	Offset
STAT	90h

Function

Reports the transaction status within USBFS.

When the processor's interrupt controller receives ISTAT[TOKDNE], you must read Interrupt Status (ISTAT) to determine the status of the previous endpoint communication. The data in this register is valid when the TOKDNE interrupt is asserted. This register is a read window into a group FIFO that USBFS maintains. After USBFS uses a BD, it updates this register.

If you perform another USB transaction before servicing the TOKDNE interrupt, then USBFS stores the status of the next transaction in the STAT FIFO. Thus, STAT is a 4-byte FIFO that allows the processor core to process one transaction when the SIE processes the next transaction. Writing 0 to ISTAT[TOKDNE] causes the SIE to update this register with the contents of the next STAT value. If the data in the STAT holding register is valid, then the SIE immediately reasserts to TOKDNE interrupt.

Diagram

Bits	7	6	5	4	3	2	1	0
R	ENDP				TX	ODD	0	
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-4 ENDP	Endpoint address Encodes the endpoint address that receives or transmits the previous token. This allows the processor core to determine the BDT entry that the last USB transaction update.
3 TX	Transmit Indicator Indicates whether the most recent transaction was a receive or transmit operation. 0b - Receive 1b - Transmit
2 ODD	Odd Bank Indicates whether the last BD that you updated is in the odd bank of the BDT. 0b - Not in the odd bank 1b - In the odd bank
1-0 —	Reserved

35.7.1.11 Control (CTL)

Offset

Register	Offset
CTL	94h

Function
Provides various control and configuration information for USBFS.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0	SE0	TXSUSPE...	0	0	RESUME	ODDRST	USBENSO...
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 —	Reserved
6 SE0	Live USB Single-Ended Zero signal
5 TXSUSPENDTOKENBUSY	TXD Suspend And Token Busy In Device mode, disables packet transmission and reception. SIE writes 1 to this field to disable this functionality. SIE does this when a SETUP token is received. This allows you to dequeue any pending packet transactions in the BDT before resuming token processing. Write 0 to this field to allow SIE to continue.
4 —	Reserved
3 —	Reserved
2 RESUME	Resume Enables USBFS to execute resume signaling when this field becomes 1. This allows USBFS to perform remote wake-up. You must write 1 to this field for the required time and then write 0 to this field.
1 ODDRST	Odd Reset Resets all the BDT ODD ping/pong fields to 0 when you write 1 to this field, which specifies the EVEN BDT bank.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 USBENSOFEN	USB Enable When Device mode is enabled, writing 1 to this field enables USBFS to operate. Writing 0 to this field disables USBFS. Writing 1 to this field causes the SIE to reset all of its STAT[ODD] to the BDTs and also resets most of the logic in the SIE. 0b - Disable 1b - Enable

35.7.1.12 Address (ADDR)

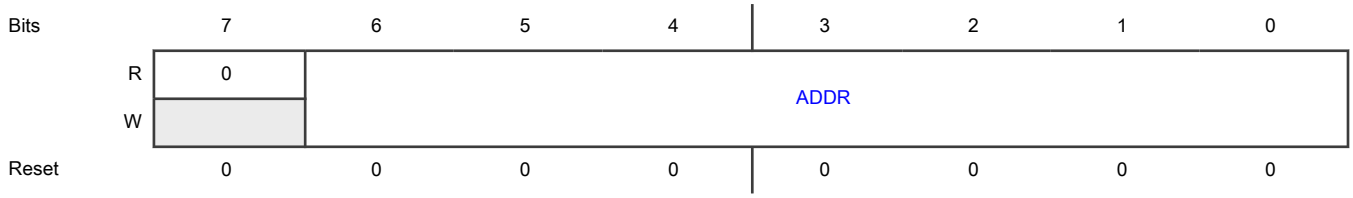
Offset

Register	Offset
ADDR	98h

Function

- In Peripheral mode, this register holds the unique USB address that USBFS decodes. CTL[USBEN] must be 1.
- After the reset input becomes active or USBFS decodes a USB reset signal, this register resets to 00h. This action initializes this register to decode address 00h, as required by the USB specification.

Diagram



Fields

Field	Function
7 —	Reserved
6-0 ADDR	USB Address Defines the USB address that USBFS decodes in Peripheral mode.

35.7.1.13 BDT Page 1 (BDTPAGE1)

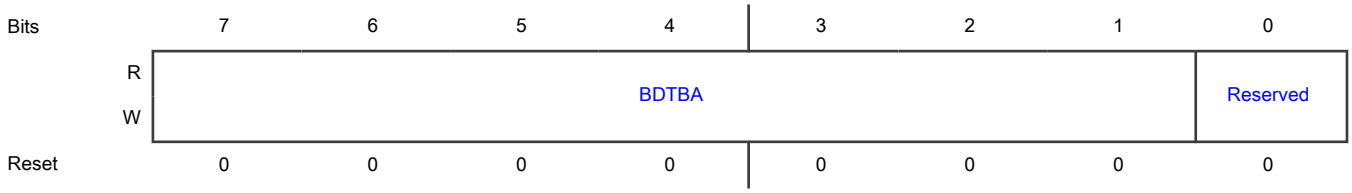
Offset

Register	Offset
BDTPAGE1	9Ch

Function

Provides address bits 15 through 9 of the base address where the current BDT resides in system memory. See Buffer descriptor table (BDT) for more information. The 32-bit BDT base address is always aligned on 512-byte boundaries, so bits 8 through 0 of the base address are always zero.

Diagram



Fields

Field	Function
7-1 BDTBA	BDT Base Address Provides address bits 15 through 9 of the BDT base address.
0 —	Reserved

35.7.1.14 Frame Number Register Low (FRMNUML)

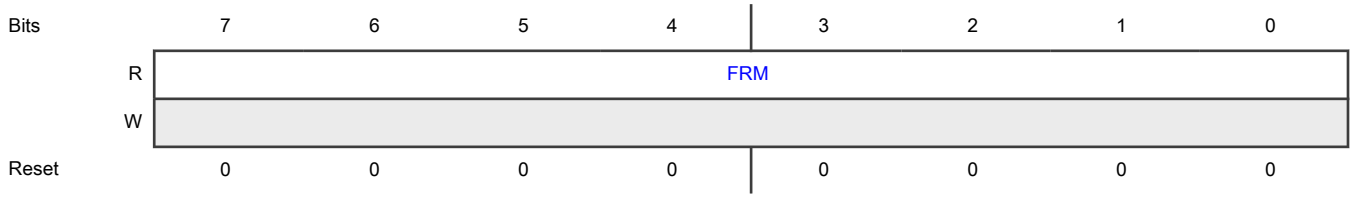
Offset

Register	Offset
FRMNUML	A0h

Function

Works with Frame Number Register High (FRMNUMH) to indicate the 11-bit frame number that determines the address of the BDT in system memory. When you receive ISTAT[SOFTOK], update these registers with the current frame number.

Diagram



Fields

Field	Function
7-0 FRM	Frame Number, Bits 0-7

35.7.1.15 Frame Number Register High (FRMNUMH)

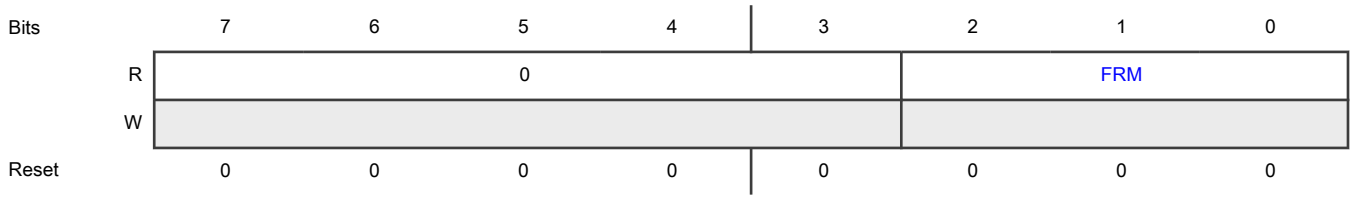
Offset

Register	Offset
FRMNUMH	A4h

Function

Works with Frame Number Register Low (FRMNUML) to indicate the 11-bit frame number that determines the address of the BDT in system memory. These registers are updated with the current frame number when you receive ISTAT[SOFTOK].

Diagram



Fields

Field	Function
7-3 —	Reserved
2-0 FRM	Frame Number, Bits 8-10

35.7.1.16 BDT Page 2 (BDTPAGE2)

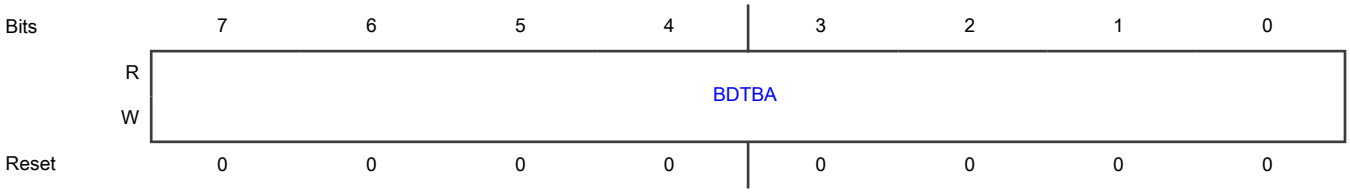
Offset

Register	Offset
BDTPAGE2	B0h

Function

Works with BDT Page 3 (BDTPAGE3) to specify address where the current BDT resides in system memory. See Buffer descriptor table (BDT) for more information.

Diagram



Fields

Field	Function
7-0 BDTBA	BDT Base Address Provides address bits 23 through 16 of the BDT base address in system memory.

35.7.1.17 BDT Page 3 (BDTPAGE3)

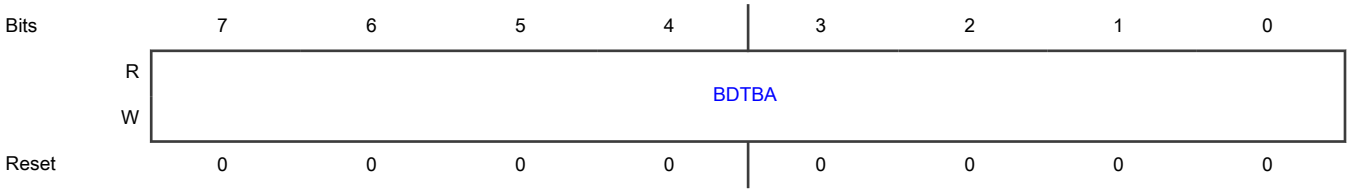
Offset

Register	Offset
BDTPAGE3	B4h

Function

Works with BDT Page 2 (BDTPAGE2) to specify the address where the current BDT resides in system memory. See Buffer descriptor table (BDT) for more information.

Diagram



Fields

Field	Function
7-0 BDTBA	BDT Base Address Provides address bits 31 through 24 of the BDT base address in system memory.

35.7.1.18 Endpoint Control (ENDPT0 - ENDPT15)

Offset

For a = 0 to 15:

Register	Offset
ENDPTa	C0h + (a × 4h)

Function

Contains the endpoint control fields for each of the 16 endpoints available within USBFS for a decoded address. The format for these registers is shown in the following figure.

- Endpoint 0 (ENDPT0) is associated with control pipe 0, which is required for all USB functions. Therefore, after a USBRST interrupt occurs, the processor core must write 1 to ENDPT0 to contain 0Dh.
- For control, bulk, and interrupt transfers, ENDPTn[EPHSHK] = 1. For isochronous transfers, ENDPTn[EPHSHK] = 0.
- ENDPTn[EPCTLDIS], ENDPTn[EPRXEN], and ENDPTn[EPTXEN] define if an endpoint is enabled and the endpoint's direction. The endpoint enables and direction control is defined in the following table.

Table 197. Endpoint enable and direction control

EPCTLDIS	EPRXEN	EPTXEN	Endpoint enable and direction control
X	0	0	Disable endpoint
X	0	1	Enable endpoint for transit transfers only
X	1	0	Enable endpoint for receive transfers only
1	1	1	Enable endpoint for receive and transit transfers
0	1	1	Enable endpoint for receive and transit as well as control (SETUP) transfers.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0			EPCTLDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-5 —	Reserved
4 EPCTLDIS	Control Transfer Disable Disables control (SETUP) transfers. When this field is 0, control transfers are enabled only if ENDPTn[EPRXEN] and ENDPTn[EPTXEN] are both 1. See Table 197. 0b - Enable 1b - Disable
3 EPRXEN	Endpoint for RX transfers enable Enables the endpoint for RX transfers. See Table 197.
2 EPTXEN	Endpoint for TX transfers enable Enables the endpoint for TX transfers. See Table 197.
1 EPSTALL	Endpoint Stalled Indicates that the endpoint is stalled. • This field has priority over all other control fields in this register, but this field is only valid if ENDPTn[EPTXEN] = 1 or ENDPTn[EPRXEN] = 1. • Any access to this endpoint causes USBFS to return a STALL handshake. • After an endpoint is stalled, the host controller must intervene.
0 EPHSHK	Endpoint Handshaking Enable Enables an endpoint to perform handshaking during a transaction to this endpoint when this field becomes 1. This field is generally 1 unless the endpoint is isochronous.

35.7.1.19 USB Control (USBCTRL)

Offset

Register	Offset
USBCTRL	100h

Function

Configures:

- The USB FS transceiver
- LPUART over USB pin mode
- DP and DM lane reversal

Diagram

Bits	7	6	5	4	3	2	1	0
R	SUSP	PDE	UARTCHLS	UARTSEL	0	DPDM_LA...	0	
W								
Reset	1	1	0	0	0	0	0	0

Fields

Field	Function
7 SUSP	Suspend Places the USB transceiver into Suspend state. 0b - Not in Suspend state 1b - In Suspend state
6 PDE	Pulldown Enable Enables the weak pulldowns on the USB transceiver. 0b - Disable on D+ and D– 1b - Enable on D+ and D–
5 UARTCHLS	UART Signal Channel Select Selects the UART signal channel. This field is valid only when you select to use USB signals as UART signals. 0b - USB DP and DM signals are used as UART TX/RX. 1b - USB DP and DM signals are used as UART RX/TX.
4 UARTSEL	UART Select Selects USB external package pins to be used as UART signals. You must not change this field when USB data communication is active. 0b - USB DP and DM external package pins are used for USB signaling. 1b - USB DP and DM external package pins are used for UART signaling.
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 DPDM_LANE_REVERSE	DP and DM Lane Reversal Control Specifies an external USB DP and DM package pin assignment configuration for your flexibility in board design. You must not change this field when USB data communication is active. 0b - Standard USB DP and DM package pin assignment 1b - Reverse roles of USB DP and DM package pins
1-0 —	Reserved

35.7.1.20 USB OTG Observe (OBSERVE)

Offset

Register	Offset
OBSERVE	104h

Function

Indicates the state of the pullups and pulldowns at the transceiver, which is useful when interfacing with an external OTG control module via a serial interface.

Diagram

Bits	7	6	5	4	3	2	1	0
R	DPPU	DPPD	0	DMPD	0			
W								
Reset	0	1	0	1	0	0	0	0

Fields

Field	Function
7 DPPU	D+ Pullup Indicates the state of the D+ pullup enable at the USB transceiver. 0b - Disabled 1b - Enabled
6 DPPD	D+ Pulldown Indicates the state of the D+ pulldown enable at the USB transceiver.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
5 —	Reserved
4 DMPD	D- Pulldown Indicates the state of the D- pulldown enable at the USB transceiver. 0b - Disabled 1b - Enabled
3-0 —	Reserved

35.7.1.21 USB OTG Control (CONTROL)

Offset

Register	Offset
CONTROL	108h

Function

Controls VBUS monitoring and the Device-mode DP pullup in the USBFS transceiver.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0			DPPULLU...	0		SESS_VLD	VBUS_SO...
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-5 —	Reserved
4	DP Pullup in Non-OTG Device Mode Provides control of the DP pullup in USB , if you configure USB in a non-OTG device mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
DPPULLUPNONOTG	0b - Disable 1b - Enabled
3-2 —	Reserved
1 SESS_VLD	VBUS Session Valid status Indicates whether the VBUS monitoring method reports that VBUS is above or below the session-valid threshold. CONTROL[VBUS_SOURCE_SEL] specifies which of the following options USBFS uses to monitor the VBUS voltage: • The input to the 3.3-5 V USB regulator • A resistive divider between VBUS and the ground connected to a GPIO pin 0b - Below 1b - Above
0 VBUS_SOURCE_SEL	VBUS Monitoring Source Select Specifies the source of the VBUS monitoring signal. This source determines the value of CONTROL[SESS_VLD]. 0b - Reserved 1b - Resistive divider attached to a GPIO pin

35.7.1.22 USB Transceiver Control 0 (USBTRC0)

Offset

Register	Offset
USBTRC0	10Ch

Function

Controls the following operational aspects that the USB Full-Speed Controller registers do not provide:

- Basic operation of the on-chip USB full-speed transceiver
- USB data connection

Diagram

Bits	7	6	5	4	3	2	1	0
R		VREGIN_...	USBRESM...	VFEDG_D...	VREDG_D...	USB_CLK...	SYNC_DET	USB_RES...
W	USBRESET							
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 USBRESET	USB Reset Generates a hard reset to USB . After this field becomes 1 and the reset occurs, this field becomes 0 automatically. **NOTE** **This field always reads 0. After writing 1 to this field, wait for 2 USB clock cycles before accessing other USB register fields.** 0b - Normal USBFS operation 1b - Returns USBFS to its reset state
6 VREGIN_STS	VREGIN Status Indicates the VREGIN status.
5 USBRESMEN	Asynchronous Resume Interrupt Enable Enables the USB asynchronous wake-up from Suspend mode. When this field becomes 1, after detection of resume signaling on the USB bus, it allows USBFS to send an asynchronous wake-up event to the MCU. The MCU then re-enables clocks to USBFS. This is used for low-power Suspend mode when USB module clocks are stopped or when the USB transceiver is in Suspend mode. Async wake-up only works in Device mode. • The asynchronous resume interrupt differs from the synchronous resume interrupt in that it asynchronously detects K-state using the unfiltered state of the D+ and D– pins. • You must only enable the asynchronous resume interrupt after you suspend the transceiver. 0b - Disable 1b - Enable
4 VFEDG_DET	VREGIN Falling Edge Interrupt Detect Detects the VREGIN falling edge interrupt. To enable this field, use USB *x* _MISCCTRL[VFEDG_EN]. 0b - Not detected 1b - Detected
3 VREDG_DET	VREGIN Rising Edge Interrupt Detect Detects the VREGIN rising edge interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	To enable this field, use USB *x* _MISCCTRL[VREDG_EN]. 0b - Not detected 1b - Detected
2 USB_CLK_RECOVERY_INT	Combined USB Clock Recovery interrupt status The read-only USB_CLK_RECOVERY_INT field will be set to 1'b1 when any of USB clock recovery interrupt conditions are detected and those interrupts are unmasked. • The only unmasked USB clock recovery interrupt condition results from an overflow of the frequency trim setting values, indicating that the frequency trim calculated is out of the adjustment range of the FIRC output clock. • To clear USB_CLK_RECOVERY_INT bit after it has been set, write 0xFF to register USB_CLK_RECOVER_INT_STATUS.
1 SYNC_DET	Synchronous USB Interrupt Detect Detects the synchronous USB interrupt. 0b - Not detected 1b - Detected
0 USB_RESUME_INT	USB Asynchronous Interrupt Indicates whether the interrupt is generated. 0b - Not generated 1b - Generated because of the USB asynchronous interrupt

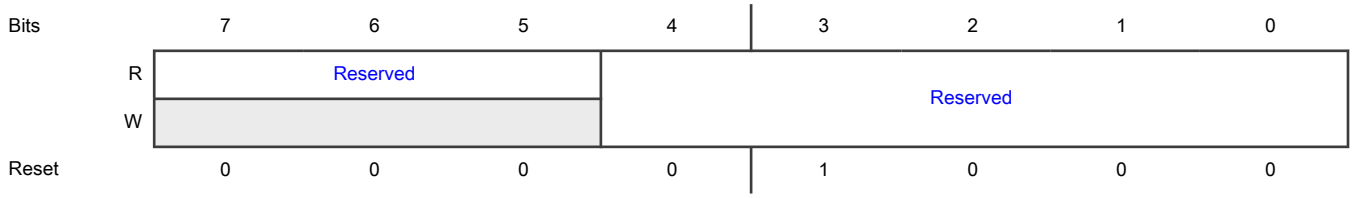
35.7.1.23 Reserved (KEEP_ALIVE_CTRL_RSVD)

Offset

Register	Offset
KEEP_ALIVE_CTRL_RSVD	124h

Function
This register is reserved for this SOC.

Diagram



Fields

Field	Function
7-5 —	Reserved
4-0 —	Reserved

35.7.1.24 Reserved (KEEP_ALIVE_WKCTRL_RSVD)

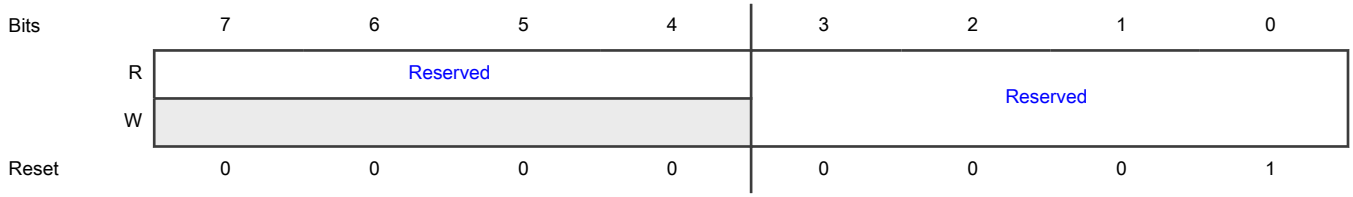
Offset

Register	Offset
KEEP_ALIVE_WKCTRL_RSVD	128h

Function

This register is reserved for this SOC.

Diagram



Fields

Field	Function
7-4 —	Reserved
3-0 —	Reserved

35.7.1.25 Miscellaneous Control (MISCCTRL)

Offset

Register	Offset
MISCCTRL	12Ch

Function

Provides additional controls for the USBFS subsystem.

Diagram

Bits	7	6	5	4	3	2	1	0
R	STL_ADJ...	Reserved		VFEDG_EN	VREDG_EN	OWNERRI...	0	0
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STL_ADJ_EN	USB Peripheral Mode Stall Adjust Enable Specifies that this field is valid only in Peripheral mode. By default (STL_ADJ_EN = 0), when you stall an endpoint (ENDPTn[END_STALL] = 1), both IN and OUT directions of the endpoint are stalled. If this field is 1, then when an endpoint is stalled (ENDPTn[END_STALL] = 1), then the USBx_STALL_xx_DIS registers can control which endpoint direction(s) are affected. 0b - If ENDPTn[END_STALL] = 1, both IN and OUT directions for the associated endpoint stalls. 1b - If ENDPTn[END_STALL] = 1, the STALL_xx_DIS registers control which directions for the associated endpoint stalls.
6-5 —	Reserved
4 VFEDG_EN	VREGIN Falling Edge Interrupt Enable Enables VREGIN falling edge interrupt. 0b - Disable 1b - Enable
3 VREDG_EN	VREGIN Rising Edge Interrupt Enable Enables VREGIN rising edge interrupt. 0b - Disable 1b - Enable
2	OWN Error Detect for ISO IN and ISO OUT Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
OWNERRISODIS	Disables OWN error detect for ISO IN, and ISO out. This field is only valid for Peripheral mode. 0b - Enable 1b - Disable
1 —	Reserved
0 —	Reserved

35.7.1.26 Peripheral Mode Stall Disable for Endpoints 7 to 0 in IN Direction (STALL_IL_DIS)

Offset

Register	Offset
STALL_IL_DIS	130h

Function

Meaningful only in Peripheral mode and when MISCCTRL[STL_ADJ_EN] = 1.

When you stall an endpoint (ENDPTn[END_STALL] = 1), the fields in this register enable or disable stalling of the IN direction of the endpoints 7 to 0.

Diagram

Bits	7	6	5	4	3	2	1	0
R W	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STALL_I_DIS7	Disable Endpoint 7 IN Direction Disables USB Device mode stall for endpoint 7 IN direction. 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 STALL_I_DIS6	Disable Endpoint 6 IN Direction Disables USB Device mode stall for endpoint 6 IN direction. 0b - Enable 1b - Disable
5 STALL_I_DIS5	Disable Endpoint 5 IN Direction Disables USB Device mode stall for endpoint 5 IN direction. 0b - Enable 1b - Disable
4 STALL_I_DIS4	Disable Endpoint 4 IN Direction Disables USB Device mode stall for endpoint 4 IN direction. 0b - Enable 1b - Disable
3 STALL_I_DIS3	Disable Endpoint 3 IN Direction Disables USB Device mode stall for endpoint 3 IN direction. 0b - Enable 1b - Disable
2 STALL_I_DIS2	Disable Endpoint 2 IN Direction Disables USB Device mode stall for endpoint 2 IN direction. 0b - Enable 1b - Disable
1 STALL_I_DIS1	Disable Endpoint 1 IN Direction Disables USB Device mode stall for endpoint 1 IN direction. 0b - Enable 1b - Disable
0 STALL_I_DIS0	Disable Endpoint 0 IN Direction Disables USB Device mode stall for endpoint 0 IN direction. 0b - Enable 1b - Disable

35.7.1.27 Peripheral Mode Stall Disable for Endpoints 15 to 8 in IN Direction (STALL_IH_DIS)

Offset

Register	Offset
STALL_IH_DIS	134h

Function

Meaningful only in Peripheral mode and when MISCCTRL[STL_ADJ_EN] = 1.

When you stall an endpoint (ENDPTn[END_STALL] = 1), the fields in this register enable or disable stalling of the IN direction of the endpoints 15 to 8.

Diagram

Bits	7	6	5	4	3	2	1	0
R W	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...	STALL_I...
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STALL_I_DIS15	Disable Endpoint 15 IN Direction Disables USB Device mode stall for endpoint 15 IN direction. 0b - Enable 1b - Disable
6 STALL_I_DIS14	Disable Endpoint 14 IN Direction Disables USB Device mode stall for endpoint 14 IN direction. 0b - Enable 1b - Disable
5 STALL_I_DIS13	Disable Endpoint 13 IN Direction Disables USB Device mode stall for endpoint 13 IN direction. 0b - Enable 1b - Disable
4 STALL_I_DIS12	Disable Endpoint 12 IN Direction Disables USB Device mode stall for endpoint 12 IN direction. 0b - Enable 1b - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 STALL_I_DIS11	Disable Endpoint 11 IN Direction Disables USB Device mode stall for endpoint 11 IN direction. 0b - Enable 1b - Disable
2 STALL_I_DIS10	Disable Endpoint 10 IN Direction Disables USB Device mode stall for endpoint 10 IN direction. 0b - Enable 1b - Disable
1 STALL_I_DIS9	Disable Endpoint 9 IN Direction Disables USB Device mode stall for endpoint 9 IN direction. 0b - Enable 1b - Disable
0 STALL_I_DIS8	Disable Endpoint 8 IN Direction Disables USB Device mode stall for endpoint 8 IN direction. 0b - Enable 1b - Disable

35.7.1.28 Peripheral Mode Stall Disable for Endpoints 7 to 0 in OUT Direction (STALL_OL_DIS)

Offset

Register	Offset
STALL_OL_DIS	138h

Function

Meaningful only in Peripheral mode and when MISCCTRL[STL_ADJ_EN] = 1.

When you stall an endpoint (ENDPTn[END_STALL] = 1), the fields in this register enable or disable stalling of the OUT direction for the endpoints 7 to 0.

Diagram

Bits	7	6	5	4	3	2	1	0
R W	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STALL_O_DIS7	Disable Endpoint 7 OUT Direction Disables USB Device mode stall for endpoint 7 OUT direction. 0b - Enable 1b - Disable
6 STALL_O_DIS6	Disable Endpoint 6 OUT Direction Disables USB Device mode stall for endpoint 6 OUT direction. 0b - Enable 1b - Disable
5 STALL_O_DIS5	Disable Endpoint 5 OUT Direction Disables USB Device mode stall for endpoint 5 OUT direction. 0b - Enable 1b - Disable
4 STALL_O_DIS4	Disable Endpoint 4 OUT Direction Disables USB Device mode stall for endpoint 4 OUT direction. 0b - Enable 1b - Disable
3 STALL_O_DIS3	Disable Endpoint 3 OUT Direction Disables USB Device mode stall for endpoint 3 OUT direction. 0b - Enable 1b - Disable
2 STALL_O_DIS2	Disable Endpoint 2 OUT Direction Disables USB Device mode stall for endpoint 2 OUT direction. 0b - Enable 1b - Disable
1 STALL_O_DIS1	Disable Endpoint 1 OUT Direction Disables USB Device mode stall for endpoint 1 OUT direction.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
0 STALL_O_DIS0	Disable Endpoint 0 OUT Direction Disables USB Device mode stall for endpoint 0 OUT direction. 0b - Enable 1b - Disable

35.7.1.29 Peripheral Mode Stall Disable for Endpoints 15 to 8 in OUT Direction (STALL_OH_DIS)

Offset

Register	Offset
STALL_OH_DIS	13Ch

Function

Meaningful only in Peripheral mode and when MISCCTRL[STL_ADJ_EN] = 1.

When you stall an endpoint (ENDPTn[END_STALL] = 1), the fields in this register enable or disable stalling of the OUT direction for the endpoints 15 to 8.

Diagram

Bits	7	6	5	4	3	2	1	0
R W	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...	STALL_O...
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 STALL_O_DIS15	Disable Endpoint 15 OUT Direction Disables USB Device mode stall for endpoint 15 OUT direction. 0b - Enable 1b - Disable
6 STALL_O_DIS14	Disable Endpoint 14 OUT Direction Disables USB Device mode stall for endpoint 14 OUT direction.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable 1b - Disable
5 STALL_O_DIS13	Disable Endpoint 13 OUT Direction Disables USB Device mode stall for endpoint 13 OUT direction. 0b - Enable 1b - Disable
4 STALL_O_DIS12	Disable endpoint 12 OUT direction Disables USB Device mode stall for endpoint 12 OUT direction. 0b - Enable 1b - Disable
3 STALL_O_DIS11	Disable Endpoint 11 OUT Direction Disables USB Device mode stall for endpoint 11 OUT direction. 0b - Enable 1b - Disable
2 STALL_O_DIS10	Disable Endpoint 10 OUT Direction Disables USB Device mode stall for endpoint 10 OUT direction. 0b - Enable 1b - Disable
1 STALL_O_DIS9	Disable Endpoint 9 OUT Direction Disables USB Device mode stall for endpoint 9 OUT direction. 0b - Enable 1b - Disable
0 STALL_O_DIS8	Disable Endpoint 8 OUT Direction Disables USB Device mode stall for endpoint 0 OUT direction. 0b - Enable 1b - Disable

35.7.1.30 USB Clock Recovery Control (CLK_RECOVER_CTRL)

Offset

Register	Offset
CLK_RECOVER_CTRL	140h

Function

Controls the crystal-less USB clock mode in which the internal FIRC oscillator is tuned to match the clock extracted from the incoming USB data stream.

You must configure the FIRC internal oscillator module in the SCG module and enable SOF frequency tuning in this register and FIRC Oscillator Enable (CLK_RECOVER_IRC_EN) for this mode.

Diagram

Bits	7	6	5	4	3	2	1	0
R	CLOCK_R...	RESET_R...	RESTART...	0	TRIM_IN...	Reserved	Reserved	Reserved
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 CLOCK_RECOVER_EN	Crystal-Less USB Enable Enables the frequency trimming logic for the FIRC module to generate clocks for crystal-less FS USB Device mode operation. You must initialize the FIRCCSR[FIRCEN] and FIRCTCFG[TRIMSRC] to enable the FIRC module itself. Then this field and CLK_RECOVER_IRC_EN[IRC_EN] must become 1 before using the crystal-less USB clock configuration. 0b - Disable 1b - Enable
6 RESET_RESUME_ROUGH_EN	Reset or Resume to Rough Phase Enable Indicates that the clock recovery block tracks the FIRC to get an accurate 48 MHz clock. It has 2 phases after you enable CLK_RECOVER_CTRL[CLOCK_RECOVER_EN], rough phase and tracking phase. The step to fine-tune the FIRC by adjusting the trim fine value differs during these 2 phases. The rough step-in phase is larger than that in the tracking phase. Switch back to the rough stage whenever a USB bus reset or bus resume occurs. 0b - Always works in tracking phase after the first time rough phase, to track transition. 1b - Go back to rough stage whenever a bus reset or bus resume occurs.
5 RESTART_IFR TRIM_EN	Restart from IFR Trim Value FIRC has a default trim fine value whose default value is factory trimmed (the IFR trim value). The clock recover block tracks the clock's accuracy at 48 MHz and keeps updating the trim fine value accordingly. 0b - Trim fine adjustment always works based on the previous updated trim fine value. 1b - Trim fine restarts from the IFR trim value whenever you detect bus_reset or bus_resume or deassert module enable.
4 —	Reserved
3	Selects the source for the initial FIRC trim fine value used after a reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
TRIM_INIT_VAL_SEL	0b - Mid-scale 1b - IFR
2 —	Reserved
1 —	Reserved
0 —	Reserved

35.7.1.31 FIRC Oscillator Enable (CLK_RECOVER_IRC_EN)

Offset

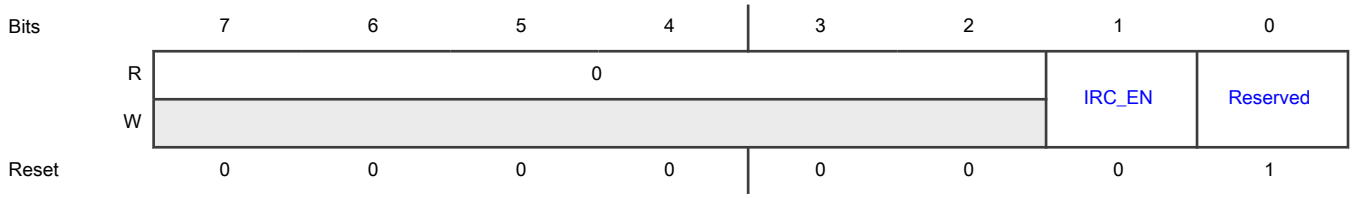
Register	Offset
CLK_RECOVER_IRC_EN	144h

Function

Controls the USB-specific frequency trimming behavior for the on-chip FIRC module used to produce nominal 48 MHz clocks for USB crystal-less operation and other functions.

See the FIRC operation in the "Clock Distribution" section in the SCG chapter for more information.

Diagram



Fields

Field	Function
7-2 —	Reserved
1	Fast IRC enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
IRC_EN	Configures the frequency trimming logic for the FIRC module to generate clocks for crystal-less FS USB Device mode operation. You must initialize FIRCCSR[FIRCEN] and FIRCTCFG[TRIMSRC] for the SCG module to enable the FIRC module itself. Then this field and CLK_RECOVER_CTRL[CLOCK_RECOVER_EN] must become 1 before using the crystal-less USB clock configuration. 0b - Disable 1b - Enable
0 —	Reserved

35.7.1.32 Clock Recovery Combined Interrupt Enable (CLK_RECOVER_INT_EN)

Offset

Register	Offset
CLK_RECOVER_INT_EN	154h

Function

Enables or masks the individual interrupt flags, which are logically OR'ed together, to produce the combined interrupt indication at USBTRC0[USB_CLK_RECOVERY_INT] if you detect the indicated conditions in the USB clock recovery algorithm operation.

Diagram

Bits	7	6	5	4	3	2	1	0
R	Reserved			OVF_ERR...	Reserved			
W								
Reset	0	0	0	1	0	0	0	0

Fields

Field	Function
7-5 —	Reserved
4 OVF_ERROR_EN	Overflow error interrupt enable Determines whether the OVF_ERROR condition signal is used in generation of USB_CLK_RECOVERY_INT.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The interrupt is masked 1b - The interrupt is enabled
3-0 —	Reserved

35.7.1.33 Clock Recovery Separated Interrupt Status (CLK_RECOVER_INT_STATUS)

Offset

Register	Offset
CLK_RECOVER_INT_STATUS	15Ch

Function

Contains flags for clock-recovery interrupts.

Diagram

Bits	7	6	5	4	3	2	1	0
R	Reserved			OVF_ERR...	Reserved			
W	W1C			W1C	W1C			
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-5 —	Reserved
4 OVF_ERROR	Overflow Error Interrupt Status Flag Indicates that the USB clock recovery algorithm detects that the frequency trim adjustment needed for the FIRC output clock is outside the available TRIM_FINE adjustment range for FIRC. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Interrupt did not occur

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Unmasked interrupt occurred When writing 0b - No effect 1b - Clear the flag
3-0 —	Reserved

Chapter 36
Flexible I/O (FLEXIO)

36.1 Chip-specific FlexIO information

Table 198. Reference links to related information

Topic	Related module	Reference
Full description	FlexIO	FlexIO
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

36.2 Overview

FLEXIO is a highly configurable module that provides:

- 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 that allow 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 the CPU.

36.2.1 Block diagram

The following diagram provides a high-level overview of the FLEXIO timer and shifter configuration.

FLEXIO uses shifters, timers, and external triggers to shift data into or out of FLEXIO. As shown in the block diagram, timers control the timing of this data shift. You can configure the timers to use generic timer functions, external triggers, or various other conditions to determine the control.

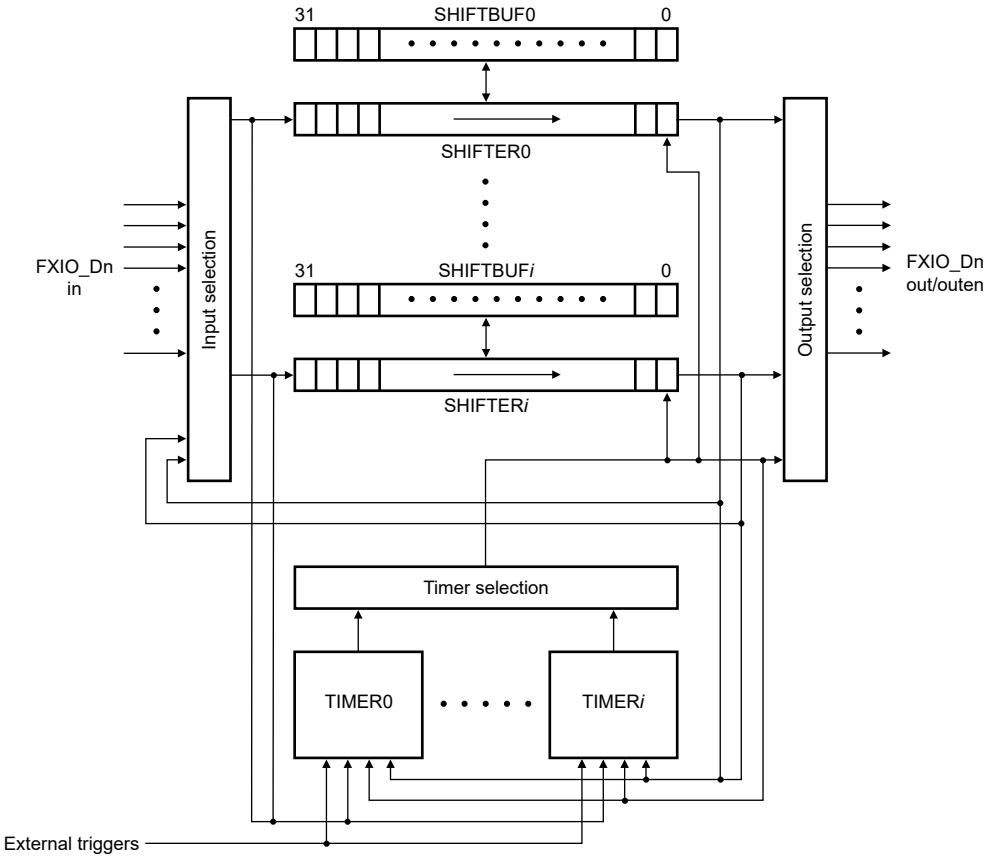


Figure 144. Block diagram

36.2.2 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
- Highly flexible 16-bit timers with support for various internal or external triggers, reset, enable, and disable conditions:
  - Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during Stop mode
  - 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 I/O registers and pin rising or falling edge interrupts to simplify software support
- Support for 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

36.3 Functional description

36.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of FLEXIO. The timer assigned to the shifter controls the timing of shift, load, and store events via SHIFTCTL*n*[TIMSEL]. Shifters are designed to support either DMA, interrupt, or polled operations. The following figure provides a detailed view of the shifter microarchitecture.

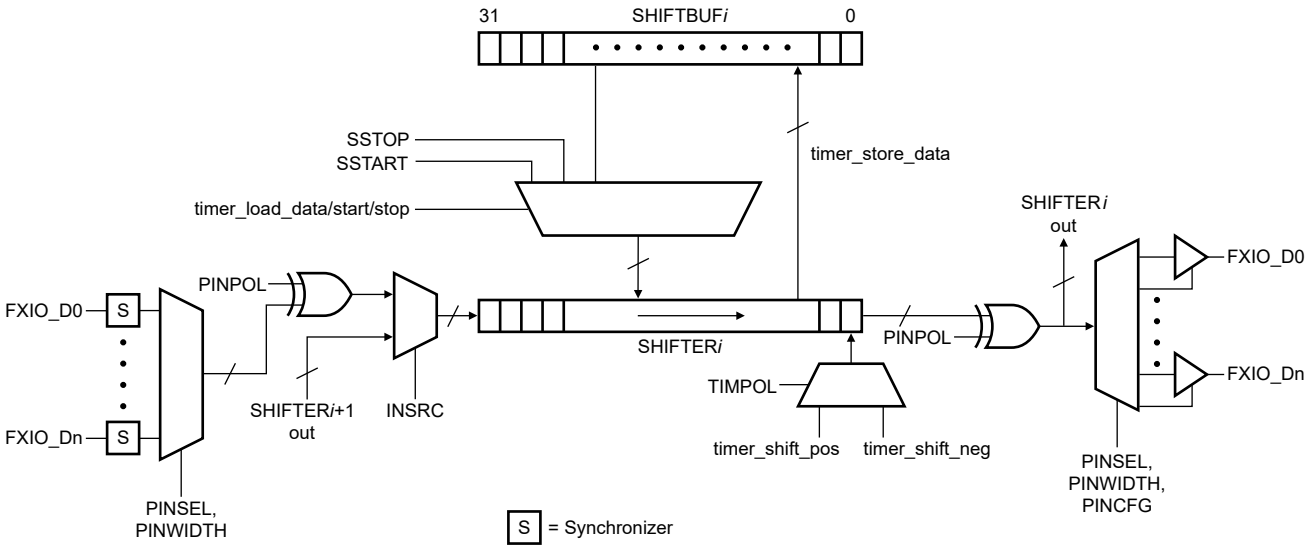


Figure 145. Shifter microarchitecture

36.3.1.1 Transmit mode

In Transmit mode (SHIFTCTL*n*[SMOD] = 010b), the shifter loads data from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) and shifts data out when the assigned timer signals a load event. An optional start and stop bit can be automatically loaded before or after SHIFTBUF register data by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] in the shifter and timer.

NOTE

If a stop bit is enabled, the shifter immediately loads a stop bit when it is initially configured for Transmit mode.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has either been loaded from the SHIFTBUF register into the shifter or when the shifter is initially configured for Transmit mode. To clear the flag, write 1 or write new data to SHIFTBUF. In Transmit mode, write any value to the SHIFTBUF register to clear the corresponding shifter status flag, which is cleared regardless of what is writing to the register (DMA or interrupt), or the state of the DMA or interrupt enables. See the functional description of SHIFTSTAT[SSF] for information on how the flag is set and cleared for each mode.

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). Clear the flag by writing 1.

36.3.1.2 Receive mode

When the assigned timer signals a store event in Receive mode (SHIFTCTL*n*[SMOD] = 001b), the shifter shifts and stores data in Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). You can check for a start and stop bit before or after the shifter data is sampled by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] in the shifter and timer.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data is stored in the SHIFTBUF register from the shifter. To clear the flag, write 1 to or read the data from SHIFTBUF. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is in Receive mode. The flag is cleared regardless of what is reading the register (DMA or interrupt) or the state of the DMA or interrupt enables. See the functional description of SHIFTSTAT[SSF] for information on how the flag is set or cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set either when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start or stop bit check. Write 1 to clear the flag.

36.3.1.3 Match Store mode

In Match Store mode (SHIFTCTL*n*[SMOD] = 100b), the shifter shifts data in, checks for a match result, and stores matched data in Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) when the assigned timer signals a store event. By configuring either SHIFTCFG[SSTART], TIMCFG[TSTART], and SHIFTCFG[SSTOP], or TIMCFG[TSTOP] in the shifter and timer, you can check for a start and stop bit before or after the shifter data is sampled. You can compare up to 16 bits of data 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 the matched data is stored in the SHIFTBUF register from the shifter. To clear the flag, read the matched data from the SHIFTBUF register or write 1 to the flag. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is configured in Match Store mode. The flag is cleared regardless of what is reading the register (DMA or interrupt) or the state of the DMA or interrupt enables. See the functional description for SHIFTSTAT[SSF] to know how the flag is set or cleared for each mode.

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 or stop bit check. Write 1 to clear the flag.

36.3.1.4 Match Continuous mode

In Match Continuous mode (SHIFTCTL*n*[SMOD] = 101b), the shifter shifts data in and continuously checks for a match result whenever a shift event is signaled by the assigned timer. You can compare up to 16 bits of data 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 no match exists between the shifter data and Shifter Buffer (SHIFTBUF0 - SHIFTBUF3).

You cannot clear the flag by reading the SHIFTBUF register.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when a match occurs. To clear the flag, write 1 or perform a read from the SHIFTBUF register.

36.3.1.5 State mode

State mode enables you to implement any state machine with up to eight states, eight outputs, and three selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU, which can remain in a low-power mode.

In State mode (SHIFTCTL*n*[SMOD] = 110b), when the shifter is selected by the current state pointer (SHIFTSTATE[STATE]), use the SHIFTBUF register to drive the output and compute the next state values. The following figure provides a detailed view of the shifter microarchitecture when configured for State mode.

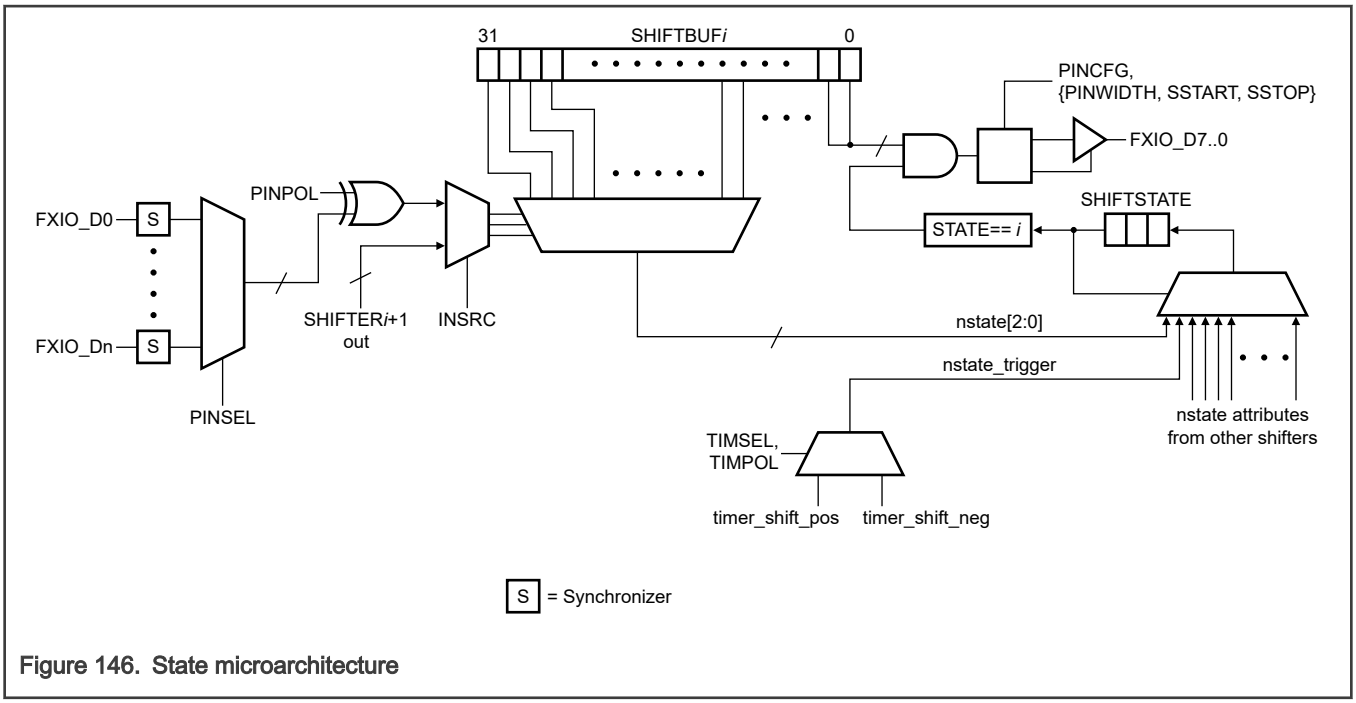


Figure 146. State microarchitecture

When the current state pointer selects a specific shifter (shifter *n*), output pins FXIO_D[7:0] are driven by SHIFTBUF*n*[31:24]; the configuration is defined by SHIFTCTL*n*[PINCFG]. Write 1 to Shifter Configuration (SHIFTCFG0 - SHIFTCFG3) {PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} to disable the output drive on pins FXIO_D[7:0] for state machine applications that require less than eight output pins.

Use the three input pins selected by SHIFTCTL*n*[PINSEL] and SHIFTBUF*n*[23:0] to compute the next state value.

NOTE

Each state can use a different set of three input pins.

The following table shows how the next state value is computed when the current state pointer is pointing to shifter *n*.

Table 199. Next state computation for SHIFTSTATE[STATE] = *n*

FXIO_D[PINSEL + 2]	FXIO_D[PINSEL + 1]	FXIO_D[PINSEL]	Next state value
0	0	0	SHIFTBUF*n*[2:0]
0	0	1	SHIFTBUF*n*[5:3]
0	1	0	SHIFTBUF*n*[8:6]
0	1	1	SHIFTBUF*n*[11:9]
...	...	...	...
1	1	1	SHIFTBUF*n*[23:21]

NOTE

You can configure other shifters and timers to drive the input pins of a given state, allowing you to create complex combinations of shifters and timers as needed. For example, the output of a shifter configured for Logic mode can be used to drive a state machine input.

The next state transition is triggered using the timer output selected by SHIFTCTL*n*[TIMSEL], with polarity controlled by SHIFTCTL*n*[TIMPOL].

NOTE

Each state can use a different timer to trigger each next state transition, allowing various internal or external trigger sources and clocking configurations to be used. See Timer section for more information.

The current state pointer defaults to shifter 0 at reset; however, you can write to select a different shifter for the initial state. If the current state pointer selects a shifter that is not configured for State mode, then outputs are not driven and the next state transition is never triggered.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when the shifter is selected by the current state pointer. The flag is cleared when the current state pointer is updated to a different shifter.

36.3.1.6 Logic mode

Logic mode enables you to implement a small amount of programmable digital logic within a FLEXIO shifter.

In Logic mode (SHIFTCTL*n*[SMOD] = 111b), use Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) to implement a 5-input, 32-bit programmable logic lookup table. The following figure provides a detailed view of shifter microarchitecture when configured for Logic mode.

Use the SHIFTBUF register to configure the lookup table for the four pin inputs. You can also use SHIFTER*n* to configure a feedback or delayed pin source as the fifth input to the lookup table.

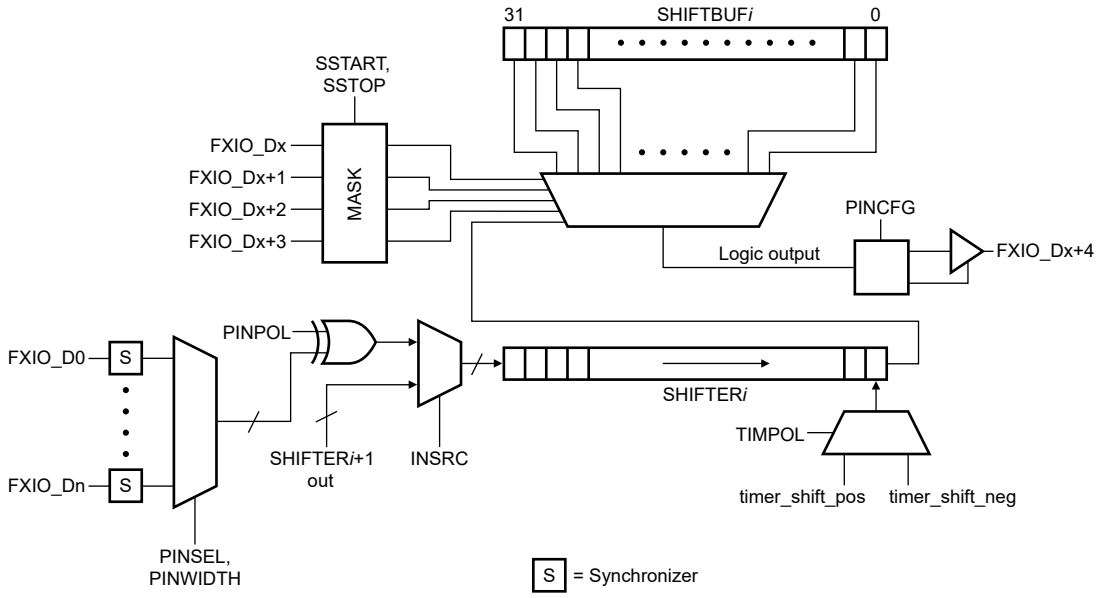


Figure 147. Logic microarchitecture

The lookup table is driven using four pin inputs (maskable using SHIFTCFG[SSTOP] and SHIFTCFG[SSTART]), plus one input from the internal shifter. It can be configured to drive an output pin using SHIFTCTL[PINCFG]. Pin inputs and outputs are fixed for each logic lookup table and are not selectable. The following table lists the logic output value selected by the lookup table for shifter *n*.

Table 200. Logic lookup table for shifter *n*

SHIFTER*n*[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	SHIFTBUF*n*[0]
0	0	0	0	1	SHIFTBUF*n*[1]

Table continues on the next page...

Table 200. Logic lookup table for shifter *n* (continued)

SHIFTER*n*[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	1	0	SHIFTBUF*n*[2]
0	0	0	1	1	SHIFTBUF*n*[3]
...	...	...	...	...	...
1	1	1	1	1	SHIFTBUF*n*[31]

1. for shifters *n* = 0...3, *x* = *n*

To minimize output glitches, use SHIFTCFG*n*[SSTOP] and SHIFTCFG*n*[SSTART] to mask unused input pins. When these fields are 1, {SSTOP[1:0] and SSTART[1:0]} mask FXIO_D[*x* + 3]...FXIO_D[*x*] inputs respectively so that any transitions on these pins do not cause the logic output to glitch.

NOTE

You can configure other shifters and timers to drive the input pins of a given lookup table, allowing you to concatenate lookup tables or create complex combinations of shifters and timers as needed.

SHIFTCFG[PWIDTH] controls the number of delay stages introduced by the internal shifter input (SHIFTER*n*[0]). For example, when configured for a 1-bit shift (SHIFTCFG[PWIDTH] = 0), the internal shifter introduces a 32-shift clock delay before passing its input (selected by SHIFTCTL[PINSEL]) to the lookup table. When configured for a 32-bit shift (SHIFTCFG[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 lookup table has a value of 1 after being synchronized with the FLEXIO clock. The flag clears when the output pin has a value of 0. This also allows SHIFTSTAT[SSF] to be used as a trigger to a timer if needed.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when the output pin allocated to the logic lookup table is asserted. Clear the flag by writing 1 to it.

The Logic mode input pins, including pins driven by other shifters and timers, are synchronized with the FLEXIO functional clock before they are input to the programmable logic lookup table.

36.3.2 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, select output, or a PWM waveform. You can configure these timers to perform any of the following functions:

- 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.
- 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, you can configure a timer to enable or disable at the same time as the previous timer (for example, timer 1 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 as a timer output, shifter status flag, pin input, or an external trigger input. The trigger configuration is separate from pin configuration; you can perform it to configure input, output data, or output enable. See the chip-specific FLEXIO information for information on external trigger connections.

You must configure Timer Configuration (TIMCFG0 - TIMCFG3) before writing 1 to TIMCTL*n*[TIMOD].

36.3.2.1 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 become zero and decrement.

NOTE

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, which toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8 bits equal zero and decrement. The timer status flag is set on a timer compare event.

36.3.2.2 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 become 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 become 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 become zero and decrement. The timer status flag is set on a timer compare event.

36.3.2.3 Timer 16-bit Counter mode

In 16-bit Counter mode, you can use the 16-bit counter to configure either the baud rate of the shift clock (for example, TIMDEC[1:0] ≠ 10 or 11) or the number of shift clock edges in the transfer (for example, 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.

36.3.2.4 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 (for example, TIMDEC[1:0] ≠ 10 or 11) or the number of shift clock edges in the transfer (for example, 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.

36.3.2.5 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 become zero and decrement.

A timer compare event occurs when the lower 8 bits become zero and decrement. The timer status flag is set when the upper 8 bits become zero and decrement.

36.3.2.6 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 become 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 become 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 become zero and decrement. The timer status flag is set on a timer compare event.

36.3.2.7 Timer enable and start functions

The following events occur when you configure TIMCTL*n*[TIMOD] for the desired mode and the condition configured by the timer enable (TIMCFG*n*[TIMENA]) is detected. When TIMCTL*n*[ONETIM] is 1, the timer status flag must be clear to generate a timer enable event; otherwise, the timer enable event is blocked. You can use this to enforce software intervention after each timer iteration:

- The timer counter loads the current value of the compare register and starts decrementing, as configured by TIMCFG*n*[TIMDEC].
- The timer output may update to its initial state depending on the configuration of TIMCFG*n*[TIMOUT]. 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 SHIFTCFG*n*[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 TIMCFG*n*[TIMOUT] = 1), a shifter that is configured to shift on the falling edge and load on the first shift does not load correctly.

36.3.2.8 Timer decrement and reset functions

The timer generates the timer output and timer shift clock depending on the fields, TIMCTL*n*[TIMOD] and TIMCFG*n*[TIMDEC]. The shifter clock is either equal to the timer output (when TIMCFG*n*[TIMDEC] ≠ 10 or 11) or equal to the decrement clock (when TIMCFG*n*[TIMDEC] = 10 or 11). If you configure TIMCFG*n*[TIMDEC] 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 TIMCFG*n*[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 not to use the prescaler (TIMCFG*n*[TIMDEC] ≠ 100 or 101).

If you configure the timer to reset as determined by TIMCFG*n*[TIMRST], then the timer counter loads the current value of the compare register again. You can configure the timer output and timer shift clock to update on timer reset, as configured by TIMCFG*n*[TIMOUT]. If the time output toggles as a result of the timer reset, this can result in a timer shift clock edge. In 8-bit Baud Counter mode, this also decrements 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.
- Any configured receive shift registers to store.

Depending on the timer mode, the timer status flag may also be set.

36.3.2.9 Timer disable and stop functions

When the 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 SHIFTCFG*n*[SSTOP]).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SHIFTCFG*n*[SSTOP].

- The timer counter reloads the current value of the compare register on the first rising edge of the shifter clock after the compare.

If you configure the timer to insert a stop bit on each compare, you must configure the transmit shifters to load on the first shift.

When the condition configured by timer disable (TIMCFG*n*[TIMDIS]) is detected, the following events occur:

- Timer counter reloads the current value of the compare register and starts decrementing as configured by TIMCFG*n*[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 otherwise generates one.
- Transmit shifters controlled by this timer output their stop bit value (if configured by SHIFTCFG*n*[SSTOP]).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SHIFTCFG*n*[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 decrementing. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit. The timer output does not generate shift events during the stop bit.

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 TIMCTL*n*[ONETIM] is 1, the timer status flag must be clear before the next timer enable condition is detected. When the timer is in the stop state condition, 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. 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.

36.3.3 Pin operation

The pin configuration for each timer and shifter can be set to use any FLEXIO pin with either polarity. You can configure each timer and shifter 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 because the output enable assertion causes logic zero to be output on the pin) or to control the enable on the bidirectional output. You can configure any timer or shifter to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

When more than one shifter or timer is configured to use the same pin for output data, then the output data to the pin is combined using an OR function. This also applies if more than one timer or shifter is configured to use the same pin for output enable.

36.3.3.1 Parallel interface

You can configure shifters to use multiple FLEXIO pins simultaneously by using SHIFTCFG*n*[PWIDTH], which 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—that is, SHIFTER0.)
3. Number of pins sampled by the shifter per shift clock (only on shifters supporting parallel receive—that is, SHIFTER3.)

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*n*[INSRC] = 1), the least significant 4, 8, 16, or 32 bits from the adjacent shifter are sampled on each shift clock.

For shifters supporting parallel receive (SHIFTER3), you can configure the shifter to sample multiple pins (INSRC = 0), with PWIDTH and PINSEL selecting the pins as FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL].

NOTE

If PWIDTH is less than the number of bits being shifted on each shift clock, then the most significant bits are masked with 0. For example, 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), you can configure the shifter to drive multiple pins using SHIFTCTL*n*[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL].

NOTE

If PWIDTH is less than the number of bits being shifted on each shift clock, then the most significant pins are not driven. For example, if PINSEL = 7 and PWIDTH = 6, then SHIFTER[5:0] drives only FXIO_D[12:7] on each shift clock.

36.3.3.2 Pin synchronization

When you configure a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized with the FLEXIO clock before a timer or shifter could use the signal. This introduces a small latency of 0.5–1.5 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

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 timers to use this pin as an input. This allows a shifter output to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized with the FLEXIO clock and therefore incurs a one-cycle latency.

When using a pin input as a timer trigger, timer clock, or shifter data input, the following synchronization delays occur:

- 0.5–1.5 FLEXIO clock cycles for an external pin
- One FLEXIO clock cycle for an internally driven pin

See Application information for timing considerations such as output valid time and input setup time for specific applications (SPI controller, SPI target, I2C controller, I2S controller, and I2S target).

36.3.3.3 Pin override

You can change the state of any FLEXIO pin at any time. Pin Output Enable (PINOUTE) configures any pin as an output and drives that pin with the value in Pin Output Data (PINOUTD).

Alias registers for PINOUTE and data registers also exist. Writing a logic 1 to an alias register updates the corresponding register fields in both PINOUTE and PINOUTD as follows:

- Pin Output Disable (PINOUTDIS) clears Pin Output Enable (PINOUTE) and Pin Output Data (PINOUTD).
- Pin Output Clear (PINOUTCLR) sets Pin Output Enable (PINOUTE) and clears Pin Output Data (PINOUTD).
- Pin Output Set (PINOUTSET) sets Pin Output Enable (PINOUTE) and Pin Output Data (PINOUTD).
- Pin Output Toggle (PINOUTTOG) sets Pin Output Enable (PINOUTE) and toggles Pin Output Data (PINOUTD).

36.3.3.4 Pin interrupt

You can read the state of any FLEXIO pin at any time and also configure any pin to set a status flag when either a rising or falling edge is detected on that pin. Additionally, you can configure the pin status flag to generate an interrupt.

36.3.4 Low-power modes

FLEXIO remains functional during low-power modes, if CTRL[DOZEN] is 0 and the FLEXIO functional clock remains enabled.

The exception to this is in Power Down mode. In this case, CTRL[DOZEN] is ignored and FLEXIO waits for all timers to complete any pending operation before acknowledging Power Down mode entry.

36.3.5 Debug mode

FLEXIO remains functional in Debug mode, provided the value of CTRL[DBGE] is 1.

36.3.6 Clocking

Table 201. FLEXIO clocks

Clock	Description
Functional clock	Is asynchronous to the bus clock and can remain enabled in low-power modes. You must enable the FLEXIO functional clock before accessing any of the FLEXIO registers. Provided the FLEXIO functional clock is at least equal to the bus clock, you can configure CTRL[FASTACC] to support fast register accesses.
Bus clock	Is used only for bus accesses to the control and configuration registers.

36.3.7 Reset

Table 202. FLEXIO reset types

Reset	Description
Chip reset	Resets the FLEXIO logic and registers to their default states on chip reset.
Software reset	Resets, using CTRL[SWRST], all logic and registers to their default states, except for the Control register.

36.3.8 Interrupts and DMA requests

The following table shows the status flags that generate the FLEXIO interrupt and DMA requests.

Table 203. FLEXIO interrupts and DMA requests

Flag	Description	Interrupt	DMA request	Low-power wake-up
SHIFTSTAT[SSF]	Shifter status flag	Y	Y	Y
SHIFTERR[SEF]	Shifter error flag	Y	N	Y
TIMSTAT[TSF]	Timer status flag	Y	Y	Y
PINSTAT[PSF]	Pin status flag	Y	N	Y
TRGSTAT[ETSF]	External trigger status flag	Y	N	Y

36.3.9 Peripheral triggers

The connection between FLEXIO peripheral triggers and other peripherals is device-specific.

36.3.9.1 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.

36.3.9.2 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.

36.4 External signals

Table 204. External signals

Signal	Description	Direction
FXIO_D*n* (*n* = 0...31)	Bidirectional FLEXIO shifter and timer pin	Input or output

36.5 Initialization

Perform the following procedure to initialize FLEXIO registers:

1. Enable FLEXIO by writing 1 to CTRL[FLEXEN].
2. Configure shift registers for the given application. It is recommended to write to Shifter Configuration (SHIFTCFG0 - SHIFTCFG3) before writing to the corresponding register, Shifter Control (SHIFTCTL0 - SHIFTCTL3).
3. Configure timer registers for the given application. It is recommended to write to Timer Compare (TIMCMP0 - TIMCMP3) and Timer Configuration (TIMCFG0 - TIMCFG3) before writing to the corresponding register, Timer Control (TIMCTL0 - TIMCTL3).
4. Enable interrupts and/or DMA requests, as appropriate, for the given application.
5. Write transmit data to initiate a transfer (depending on the given application).

36.6 Application information

This section provides examples for a variety of FLEXIO module applications. See FLEXIO register descriptions for more information.

36.6.1 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 the 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, you must modify SHIFTCFG*n*[SSTART] and SHIFTCFG*n*[SSTOP] 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 Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) (or Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) 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 205. UART transmit configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0. You can configure the PINPOL field to invert output data, or support open-drain by writing 1h to the PINPOL and PINCFG fields.

Table continues on the next page...

Table 205. UART transmit configuration (continued)

Register	Value	Configuration
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0000_2222h	Configure start bit, stop bit, enable on trigger asserted and disable on compare. You can support CTS by configuring the TIMENA field as 3h.
TIMCTL*n*	01C0_0001h	Configure the dual 8-bit counter using the shifter 0 status flag as an inverted internal trigger source. To support CTS, configure the PINSEL (for pin 1) and PINPOL fields as 1h.
SHIFTBUF*n*	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 an interrupt or a DMA request. Write to SHIFTBUFBBS[7:0] instead to support MSB first transfer.

The following table shows an alternative configuration that supports slower baud rates. This configuration requires two timers.

Table 206. UART transmit configuration for slow baud rate

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0. Invert output data by writing 1 to the PINPOL field. Support open-drain by configuring the PINPOL and PINCFG fields as 1h.
TIMCMP*n*	0000_000Fh	Configure for 8-bit transfer, and configure TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG*n*	0030_2622h	Configure start bit, stop bit, enable on trigger rising edge, decrement on trigger and disable on compare.
TIMCTL*n*	0740_0003h	Configure the 16-bit counter using the timer 1 output as an internal trigger source.
TIMCMP(*n* + 1)	0000_0001h	Configure baud rate of divide by 4 of the FLEXIO clock, and configure

Table continues on the next page...

Table 206. UART transmit configuration for slow baud rate (continued)

Register	Value	Configuration
		TIMCMP[15:0] as (baud rate divider ÷ 2) - 1.
TIMCFG(*n* + 1)	0000_1200h	Configure enable on trigger asserted and disable on timer 0 disable. You can configure the TIMEN field as 3h to support CTS.
TIMCTL(*n* + 1)	01C0_0003h	Configure the 16-bit counter using the shifter 0 status flag as an inverted internal trigger source. You can support CTS by configuring the PINSEL (for pin 1) and PINPOL fields as 1h.
SHIFTBUF*n*	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 an interrupt or a DMA request. Write to SHIFTBUFBBS[7:0] instead to support MSB first transfer.

36.6.2 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.

FLEXIO does not support triple voting of the received data, which is sampled only once in the middle of each bit. You can use a timer to implement a glitch filter on the incoming data and a different timer 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 207. UART receiver configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using timer 0 on the negative edge of clock with input data on pin 0. You can invert input data by writing 1 to the PINPOL field.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0204_2422h	Configure start bit, stop bit, enable on pin positive edge and disable on

Table continues on the next page...

Table 207. UART receiver configuration (continued)

Register	Value	Configuration
		compare. Enable resynchronization to received data with TIMOUT = 2h and TIMRST = 4h.
TIMCTL*n*	0000_0081h	Configure the dual 8-bit counter using the inverted pin 0 input.
SHIFTBUF*n*	Data to receive	You can read received data from SHIFTBUFBYS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from SHIFTBUFBIS[7:0] instead to support MSB first transfer.

The UART receiver with RTS configuration uses a second timer to generate the RTS output. 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 when the RTS is asserted, the received data is ignored.

Table 208. UART receiver with RTS configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using timer 0 on the negative edge of clock with input data on pin 0. Invert input data by writing 1 to the PINPOL field.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0204_2522h	Configure start bit, stop bit, enable on pin positive edge with trigger asserted and disable on compare. Enable resynchronization to received data with TIMOUT = 2h and TIMRST = 4h.
TIMCTL*n*	02C0_0081h	Configure dual 8-bit counter using the inverted pin 0 input. Trigger is internal using the inverted pin 1 input.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0030_6100h	Enable on timer *n* enable and disable on the 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 the shifter 0 flag.

Table continues on the next page...

Table 208. UART receiver with RTS configuration (continued)

Register	Value	Configuration
SHIFTBUF*n*	Data to receive	You can read received data using SHIFTBUFBYS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from SHIFTBUFBIS[7:0] instead to support MSB first transfer.

36.6.3 SPI controller

SPI Controller mode can be supported using two timers, two shifters, and four pins. Using the DMA controller, either CPHA = 0 or CPHA = 1 and transfers can be supported. For CPHA = 1, the chip 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 target chip select negating and before the next transfer. To initiate each transfer, either the core or DMA writes to the transmit buffer.

NOTE

Because of 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 209. SPI controller (CPHA = 0) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using timer 0 on the negative edge 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 the positive edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare; initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using the pin 2 output (shift clock), with shifter 0 flag as the inverted trigger. Write 1 to the PINPOL field to invert the output shift clock.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.

Table continues on the next page...

Table 209. SPI controller (CPHA = 0) configuration (continued)

Register	Value	Configuration
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 the inverted pin 3 output as target select.
SHIFTBUF*n*	Data to transmit	You can write transmit data to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

Table 210. SPI controller (CPHA = 1) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0021h	Start bit loads data on first shift.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive 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 the negative edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits x 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare; initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using pin 2 output (shift clock), with the shifter 0 flag as the inverted trigger. Write 1 to the

Table continues on the next page...

Table 210. SPI controller (CPHA = 1) configuration (continued)

Register	Value	Configuration
		PINPOL field to invert the output shift clock, and set the TIMDIS field as 3 to keep target 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 target select).
SHIFTBUF*n*	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. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

36.6.4 SPI target

SPI Target 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.

You must write the transmit data to the transmit buffer register before the external target select asserts; otherwise, the shifter error flag is set.

NOTE

Because of 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 211. SPI target (CPHA = 0) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using timer 0 on the falling edge of shift clock with output data on pin 0.

Table continues on the next page...

Table 211. SPI target (CPHA = 0) configuration (continued)

Register	Value	Configuration
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0000_0101h	Configure receive using timer 0 on the rising edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG*n*	0120_0600h	Configure enable on trigger rising edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using pin 2 input (shift clock), with pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

Table 212. SPI target (CPHA = 1) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0001h	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTL*n*	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 212. SPI target (CPHA = 1) configuration (continued)

Register	Value	Configuration
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] as (number of bits × 2) - 1).
TIMCFG*n*	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.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using pin 2 input (shift clock), with pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

36.6.5 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 another one is used to control the shifters. The two shifters that 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 condition), so you must program the compare register 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 one 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 two 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 the SCL pin. When generating a stop condition or a repeated start condition, this word must be 00h and FFh, respectively. During the last word of a controller-receiver transfer, you must set the transmit SHIFTCFG*n*[SSTOP] field 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 controller-transmit, the interrupt routine must immediately write 00h (when generating a stop condition) or FFh (when generating a repeated start condition) to the transmit shifter register. You must wait for

the next rising edge on SCL before disabling both timers. The transmit shifter must be disabled after the setup delay for a repeated start or stop condition.

NOTE

Because of 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 by 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 an additional cycle is involved to generate the output.

Table 213. I2C controller configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTL*n*	0101_0082h	Configure transmit using timer 1 on the 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 the falling edge of clock with input data on pin 0.
TIMCMP*n*	0000_2501h	Configure 2 word transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of words × 18) + 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	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.
TIMCTL*n*	01C1_0101h	Configure dual 8-bit counter using pin 1 output enable (SCL open-drain), with the shifter 0 flag as the inverted trigger.
TIMCMP(*n* + 1)	0000_000Fh	Configure 8-bit transfer. Set TIMCMP[15:0] as (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 the end of each word and decrement on pin input.
TIMCTL(*n* + 1)	01C0_0183h	Configure 16-bit counter using inverted pin 1 input (SCL).
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0]. Use the shifter

Table continues on the next page...

Table 213. I2C controller configuration (continued)

Register	Value	Configuration
		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.

36.6.6 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 another 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

Because of synchronization delays, the setup time for the receiver input is 1.5 FLEXIO clock cycles. This means that the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 214. I2S controller configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0001h	Load transmit data on first shift and stop bit disabled.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the 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 the falling edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0000_0202h	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTL*n*	01C3_0281h	Configure dual 8-bit counter using inverted pin 2 output (bit clock), with shifter 0 flag as the inverted trigger.

Table continues on the next page...

Table 214. I2S controller configuration (continued)

Register	Value	Configuration
		Write 0 to the PINPOL field 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] as (number of bits × 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). Write 0 to the PINPOL field to invert the polarity of the output frame sync.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) instead to support LSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) instead to support LSB first transfer.

36.6.7 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

Because of 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 target is maximum 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization, plus one cycle to output the data.

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until the rising edge of bit clock (when the frame sync is normally sampled). Timer 0 detects the rising edge of bit clock with timer 2 output asserted and asserts output for length of frame. Timer 1 detects the falling edge of bit clock with timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 215. I2S target configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0103_0002h	Configure transmit using timer 1 on the 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 the falling edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_007Fh	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] as (number of bits × 4) - 1.
TIMCFG*n*	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 decrements on pin input (bit clock).
TIMCTL*n*	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] as (number of bits × 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.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be written

Table continues on the next page...

Table 215. I2S target configuration (continued)

Register	Value	Configuration
		using interrupt or DMA request. Write to the SHIFTBUF register instead to support LSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from the SHIFTBUF register instead to support LSB first transfer.

36.6.8 SENT receiver

The 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 the 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 the previous falling edge and current falling edge. You can configure the CPU interrupt or DMA trigger at each capture of the counter. The CPU software performs the entire SENT frame decoding with the latest tick width adjustment.

Table 216. SENT receiver configuration

Register	Value	Configuration
TIMCMP*n*	—	Stores counter value at the falling edge of the pin.
TIMCFG*n*	0000_6000h	Timer is always enabled. It is disabled on trigger falling edge, decrement counter on FLEXIO clock.
TIMCTL*n*	0040_0007h	Single 16-bit input capture mode. The PINSEL field selects the timer pin input and output. Timer pin output disabled, internal trigger selected, select pin 0 as trigger.

36.6.9 Camera interface

Camera interface can be supported using one timer, one or more shifters, and multiple pins. Multiple transfers can be supported using the DMA controller.

The example configuration shown in the following table describes the FLEXIO configuration for interfacing with 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 an 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 217. Camera interface configuration for 8-bit CMOS sensor

Register	Value	Configuration
SHIFTCFG*n*...*n*+2[1]	0007_0100h	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFG*n*+3	0007_0000h	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTL*n*...*n*+3	0080_0001h	Configure receive using timer 0 on the negative edge of clock.
TIMCMP*n*	0000_001Fh	Configure 16 pixel (8 bits/pixel × 16 pixels = 128 bits) transfer. Set TIMCMP[15:0] as (number of pixels × 2) - 1.
TIMCFG*n*	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.
TIMCTL*n*	12C0_0803h	Configure 16-bit counter using FXIO_D[8] input (PCLK), with FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUF*n*...*n*+3	Data to receive	Received data can be read from SHIFTBUF*n*...*n*+3. Use the shifter status flag to indicate when data can be read using interrupt or DMA request.

1. *n* = 0 or 4

36.6.10 Motorola 68K and Intel 8080 bus interface

Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by smart and 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 following table 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 218. Motorola 68K and Intel 8080 write configuration

Register	Value	Configuration
SHIFTCFG0...3	000F_0100h	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with data output to FXIO_D[15:0].
SHIFTCTL1...3	0000_0002h	Configure transmit using timer 0 on the positive edge of clock.
TIMCMP0	0000_0101h (1 beat)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FLEXIO

Table continues on the next page...

Table 218. Motorola 68K and Intel 8080 write configuration (continued)

Register	Value	Configuration
	0000_1F01h (16 beats)	clock. Set TIMCMP[15:8] as (number of beats × 2) - 1, and set TIMCMP[7:0] as (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 3 (16 beats) flag as the inverted trigger.
SHIFTBUF0...3	Data to transmit	Transmit data can be written to SHIFTBUF0 (1 beat) or SHIFTBUF0...3(16-beats) to initiate a transfer; use the shifter status flag to indicate when data can be written using interrupt or DMA request.

Table 219. Motorola 68K and Intel 8080 read configuration

Register	Value	Configuration
SHIFTCFG0...2	000F_0100h	Configure 16-bit parallel shift in from the adjacent shifter.
SHIFTCFG3	000F_0000h	Configure 16-bit parallel shift in from pin.
SHIFTCTL0...3	0080_0001h	Configure receive using timer 0 on the negative edge 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 × 2) - 1, and 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 3 flag (1-beat) or shifter 0 flag (16 beats) as the inverted trigger.
SHIFTBUF0...3	Data received	Received data can be read from SHIFTBUF0 (1 beat) or SHIFTBUF0...3

Table continues on the next page...

Table 219. Motorola 68K and Intel 8080 read configuration (continued)

Register	Value	Configuration
		(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 target begins with a register write cycle followed by one or more data read or write cycles. To accomplish this, perform the following procedure:

1. Configure FLEXIO with 1-beat write configuration.
2. Configure GPIO to assert the nCS and RS pins (and deassert the RDWR pin for 68K).
3. Write register index data to SHIFTBUF0[15:0].
4. Configure GPIO to deassert the RS pin (and assert the RDWR pin for 68K data read).
5. Configure FLEXIO with the 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 Shifter Buffer (SHIFTBUF0 - SHIFTBUF3).
7. Configure GPIO to deassert the nCS pin.

36.6.11 Low-power state machine

Table 220 shows an example of a hypothetical state machine 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 is 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 writing 0 to CTRL[DOZEN] and ensuring that FLEXIO_CLK is enabled. Figure 148 shows the states and transitions implemented by this example.

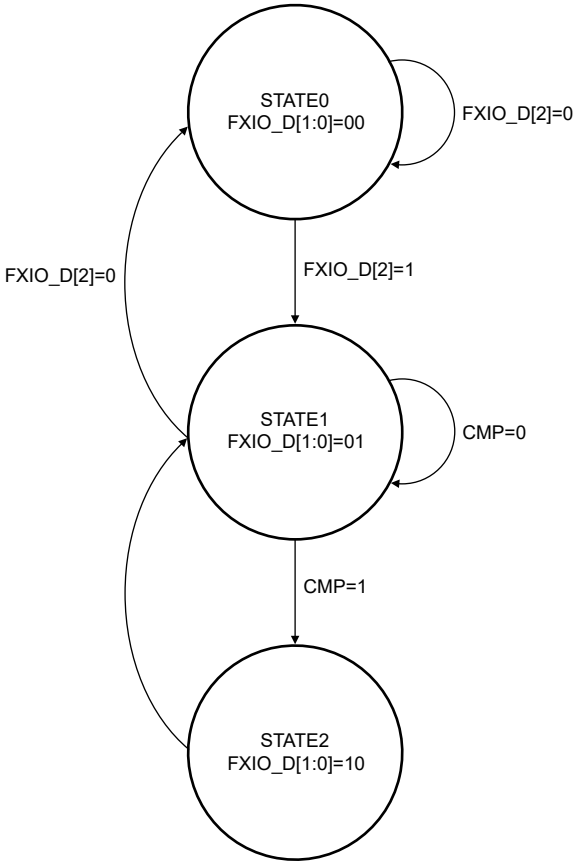


Figure 148. State diagram

Table 220. State machine configuration

Register	Value	Configuration
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 timer 0 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 timer 0 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 220. State machine configuration (continued)

Register	Value	Configuration
SHIFTCTL2	0080_0206h	Configure for State mode using FXIO_D[4:2] as inputs to select next state and timer 0 output low to trigger next state.
SHIFTBUF2	0224_9249h	State2: Drive FXIO_D[1:0] = 10, transition to state1 with timer 0 output low.
TIMCMP0	0000_FFFFh	Configure baud rate of divide by 131072 of the FLEXIO clock. Set TIMCMP[7:0] as (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.

36.6.12 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 so that you can 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 a single keypress. To support debouncing, you can decide how many consecutive scans must be considered as a single keypress.

NOTE

Because the pins used in Logic mode are fixed per shifter and the shifters that support parallel shifts are limited, this configuration is restricted to 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 221. Keypad interface configuration

Register	Value	Configuration
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 the 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	Configure prescanning glitch filter to 256 FLEXIO clock cycles. For different filter cycles, configure TIMCMP[15:0] as (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 the shifter 1 flag (logic state) as trigger.
TIMCMP1	0000_0F3Fh	Configure eight shifts (twice for each column) at column scan rate of divide by 128. For a different scan frequency, define TIMCMP[7:0] as (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 four shifts at half the frequency of the timer 1 trigger.

Table continues on the next page...

Table 221. Keypad interface configuration (continued)

Register	Value	Configuration
TIMCFG2	0110_2100h	Enable on timer 1 enable, disable on compare, decrement on trigger input 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.

36.7 Memory map and registers

36.7.1 FLEXIO register descriptions

NOTE

Invalid register accesses, which include reading a write-only register, writing to a read-only register, or accessing an invalid address, result in a bus error.

36.7.1.1 FLEXIO memory map

FLEXIO0 base address: 4009_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0201_0003h
4h	Parameter (PARAM)	32	R	0420_0404h
8h	FLEXIO Control (CTRL)	32	RW	0000_0000h
Ch	Pin State (PIN)	32	R	0000_0000h
10h	Shifter Status (SHIFTSTAT)	32	RW	0000_0000h
14h	Shifter Error (SHIFTERR)	32	RW	0000_0000h
18h	Timer Status Flag (TIMSTAT)	32	RW	0000_0000h
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable (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

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	Shifter State (SHIFTSTATE)	32	RW	0000_0000h
48h	Trigger Status (TRGSTAT)	32	RW	0000_0000h
4Ch	External Trigger Interrupt Enable (TRIGIEN)	32	RW	0000_0000h
50h	Pin Status (PINSTAT)	32	RW	0000_0000h
54h	Pin Interrupt Enable (PINIEN)	32	RW	0000_0000h
58h	Pin Rising Edge Enable (PINREN)	32	RW	0000_0000h
5Ch	Pin Falling Edge Enable (PINFEN)	32	RW	0000_0000h
60h	Pin Output Data (PINOUTD)	32	RW	0000_0000h
64h	Pin Output Enable (PINOUTE)	32	RW	0000_0000h
68h	Pin Output Disable (PINOUTDIS)	32	RW	0000_0000h
6Ch	Pin Output Clear (PINOUTCLR)	32	RW	0000_0000h
70h	Pin Output Set (PINOUTSET)	32	RW	0000_0000h
74h	Pin Output Toggle (PINOUTTOG)	32	RW	0000_0000h
80h - 8Ch	Shifter Control (SHIFTCTL0 - SHIFTCTL3)	32	RW	0000_0000h
100h - 10Ch	Shifter Configuration (SHIFTCFG0 - SHIFTCFG3)	32	RW	0000_0000h
200h - 20Ch	Shifter Buffer (SHIFTBUF0 - SHIFTBUF3)	32	RW	0000_0000h
280h - 28Ch	Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3)	32	RW	0000_0000h
300h - 30Ch	Shifter Buffer Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS3)	32	RW	0000_0000h
380h - 38Ch	Shifter Buffer Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS3)	32	RW	0000_0000h
400h - 40Ch	Timer Control (TIMCTL0 - TIMCTL3)	32	RW	0000_0000h
480h - 48Ch	Timer Configuration (TIMCFG0 - TIMCFG3)	32	RW	0000_0000h
500h - 50Ch	Timer Compare (TIMCMP0 - TIMCMP3)	32	RW	0000_0000h
680h - 68Ch	Shifter Buffer Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS3)	32	RW	0000_0000h
700h - 70Ch	Shifter Buffer Halfword Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS3)	32	RW	0000_0000h
780h - 78Ch	Shifter Buffer Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS3)	32	RW	0000_0000h
800h - 80Ch	Shifter Buffer Odd Even Swapped (SHIFTBUFOES0 - SHIFTBUFOES3)	32	RW	0000_0000h
880h - 88Ch	Shifter Buffer Even Odd Swapped (SHIFTBUFEOS0 - SHIFTBUFEOS3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
900h - 90Ch	Shifter Buffer Halfword Byte Swapped (SHIFTBUFHBS0 - SHIFTBUFHBS3)	32	RW	0000_0000h

36.7.1.2 Version ID (VERID)

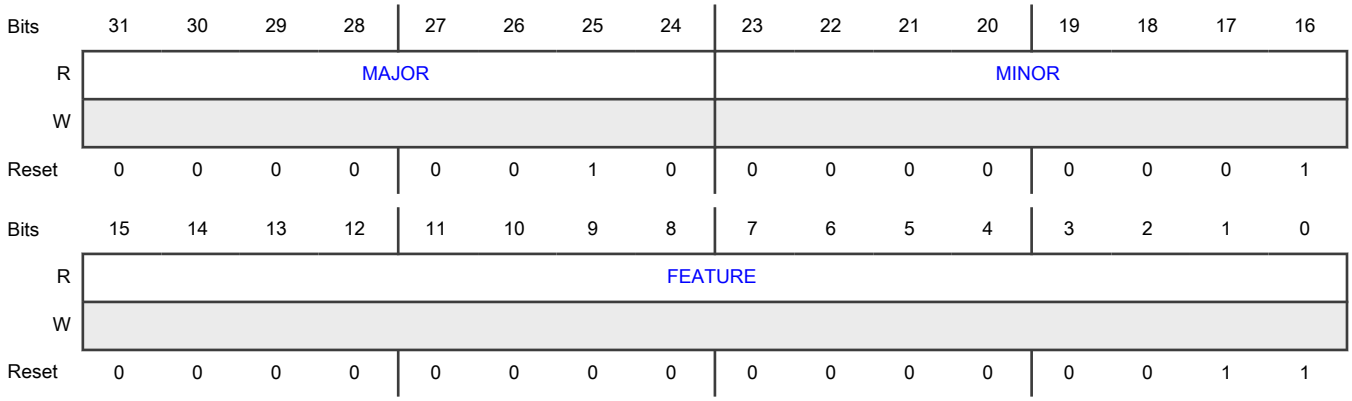
Offset

Register	Offset
VERID	0h

Function

Indicates the version of FLEXIO.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number of the module specification.
15-0 FEATURE	Feature Specification Number Indicates the feature set number. 0000_0000_0000_0000b - Standard features implemented

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000_0000_0000_0001b - State, logic, and parallel modes supported 0000_0000_0000_0010b - Pin control registers supported 0000_0000_0000_0011b - State, logic, and parallel modes, plus pin control registers supported

36.7.1.3 Parameter (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	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 TRIGGER	Trigger Number Indicates the number of external triggers implemented.
23-16 PIN	Pin Number Indicates the number of pins implemented.
15-8 TIMER	Timer Number Indicates the number of timers implemented.
7-0 SHIFTER	Shifter Number Indicates the number of shifters implemented.

36.7.1.4 FLEXIO Control (CTRL)

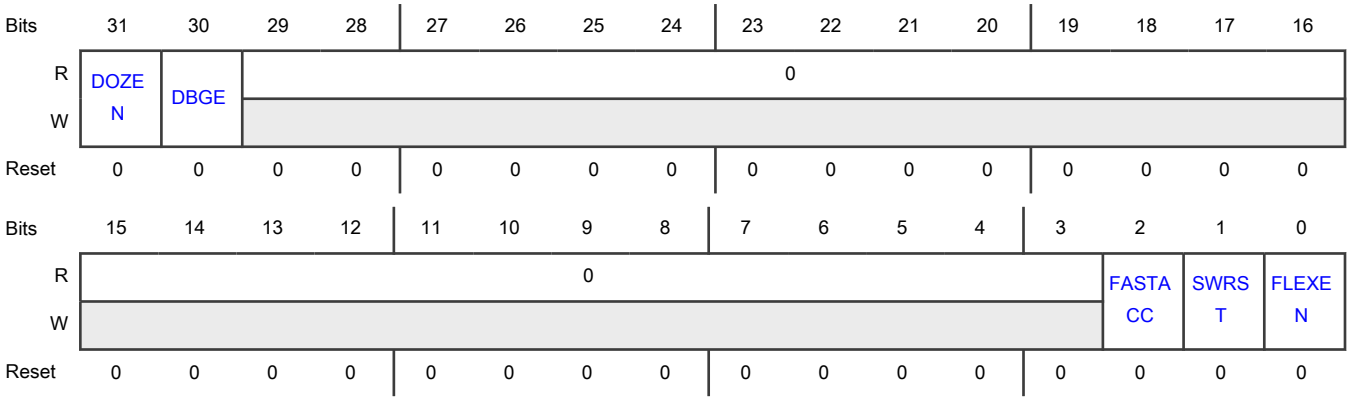
Offset

Register	Offset
CTRL	8h

Function

Controls various aspects of the FLEXIO operation.

Diagram



Fields

Field	Function
31 DOZEN	Doze Enable Disables FLEXIO operation in Doze modes. 0b - Enable 1b - Disable
30 DBGE	Debug Enable Enables the FLEXIO operation in Debug mode. 0b - Disable 1b - Enable
29-3 —	Reserved
2 FASTACC	Fast Access Configures fast or normal register accesses to FLEXIO registers, but requires the FLEXIO functional clock to be at least equal to the frequency of the bus clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Normal 1b - Fast
1 SWRST	Software Reset Specifies whether software reset is enabled. The software reset does not affect this register but it affects all other logic in FLEXIO. All other register accesses are ignored until this field is cleared. The field remains 1 until software clears it and the reset has cleared in the FLEXIO clock domain. If you write 1 to this field, all FLEXIO registers except the Control register are reset. 0b - Disabled 1b - Enabled
0 FLEXEN	FLEXIO Enable Enables FLEXIO. 0b - Disable 1b - Enable

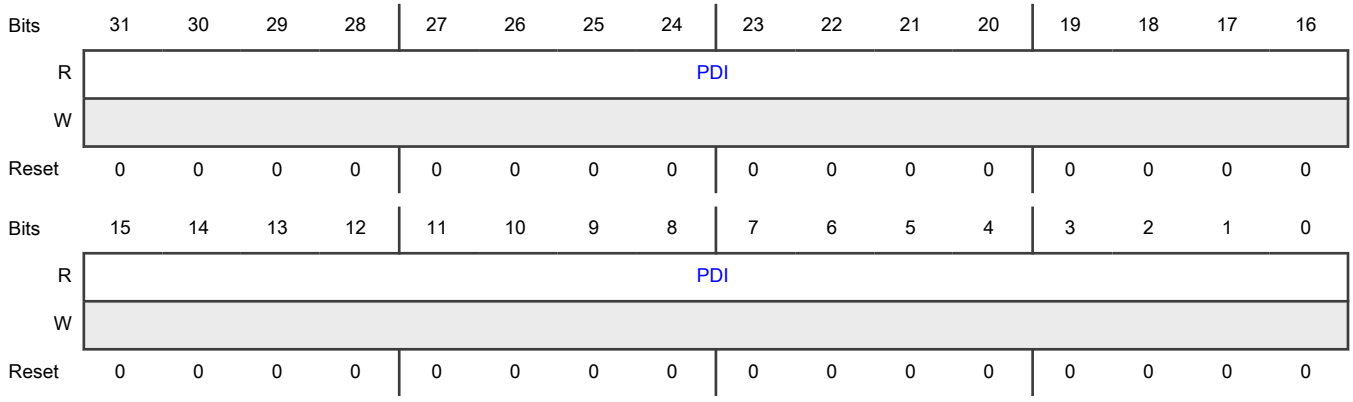
36.7.1.5 Pin State (PIN)

Offset

Register	Offset
PIN	Ch

Function
Indicates the status of the pin data input.

Diagram



Fields

Field	Function
31-0 PDI	Pin Data Input Indicates the input data on each of the FLEXIO pins.

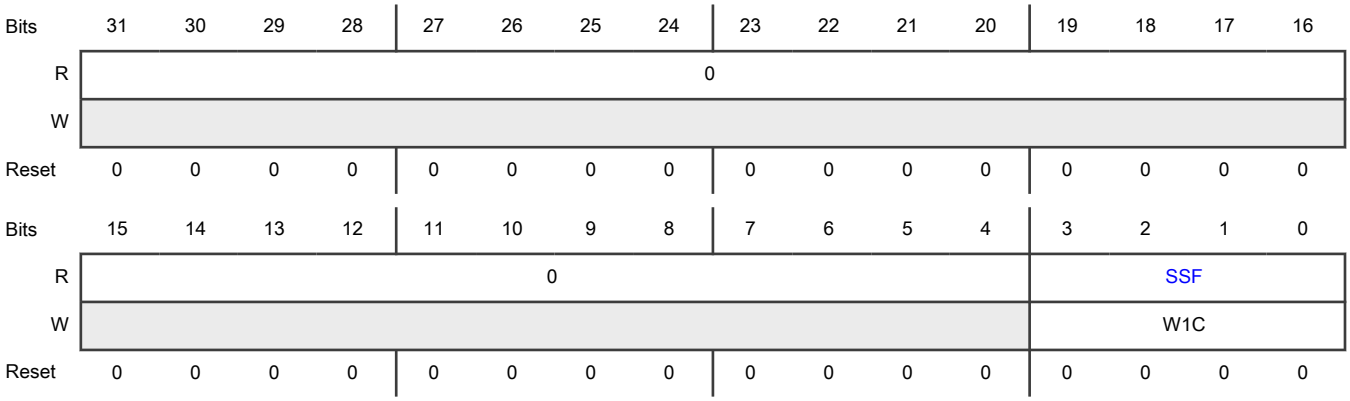
36.7.1.6 Shifter Status (SHIFTSTAT)

Offset

Register	Offset
SHIFTSTAT	10h

Function
Contains shifter status flags.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 SSF	Shifter Status Flag Indicates the shifter status. This flag is updated in one of the following cases: • If SHIFTCTL*n*[SMOD] = 001b (Receive mode), the status flag is set when SHIFTBUF is loaded with data from the shifter (SHIFTBUF is full). The status flag is cleared when you read Shifter Buffer (SHIFTBUF0 - SHIFTBUF3).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If SHIFTCTL*n*[SMOD] = 010b (Transmit mode), the status flag is set when SHIFTBUF data is transferred to the shifter (SHIFTBUF is empty) or when SHIFTCTL*n*[SMOD] is initially configured as 010b (Transmit mode). The status flag is cleared when you write to the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 100b (Match Store mode), the status flag is set when a match occurs between SHIFTBUF and the shifter. The status flag is cleared when you read the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the status flag returns the current match result between SHIFTBUF and the shifter. You cannot clear the status flag by reading the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 110b (State mode), the status flag for a shifter sets when it is selected by the current state pointer. • If SHIFTCTL*n*[SMOD] = 111b (Logic mode), the status flag returns the current value of the programmable logic block output. You can clear this status flag by writing a logic one to the flag for all modes except Match Continuous mode, State mode, and Logic mode. **NOTE** This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

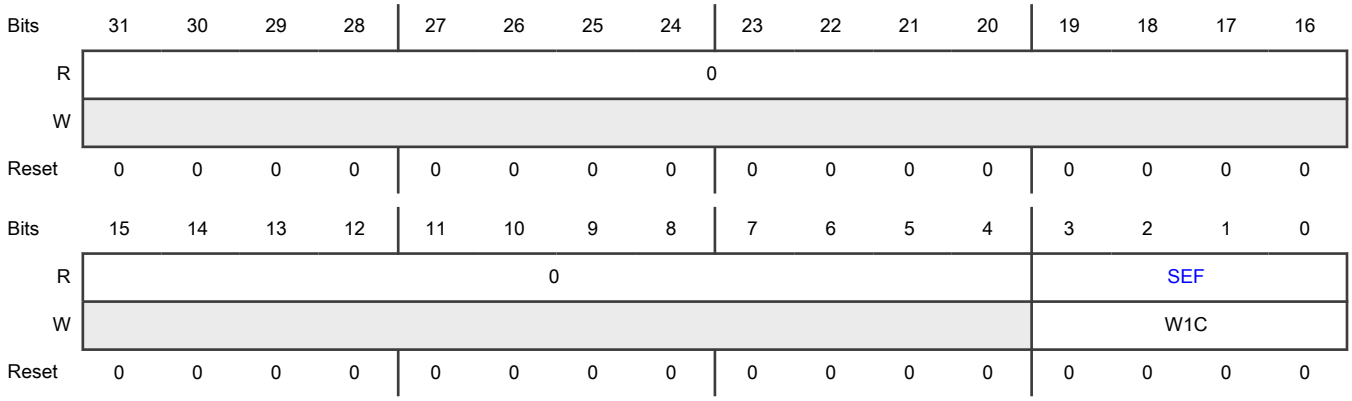
36.7.1.7 Shifter Error (SHIFTERR)

Offset

Register	Offset
SHIFTERR	14h

Function
Reports shifter errors.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 SEF	Shifter Error Flag Indicates shifter error flag status. This flag is set when one of the following events occurs: • If SHIFTCTL*n*[SMOD] = 001b (Receive mode), it indicates that either the shifter is ready to store new data into SHIFTBUF before the previous data is read from SHIFTBUF (SHIFTBUF overrun), or the received start or stop bit does not match the expected value. • If SHIFTCTL*n*[SMOD] = 010b (Transmit mode), it indicates that the shifter is ready to load new data from SHIFTBUF before new data is written into SHIFTBUF (SHIFTBUF underrun). • If SHIFTCTL*n*[SMOD] = 100b (Match Store mode), it indicates the occurrence of a match event before the previous match data is read from SHIFTBUF (SHIFTBUF overrun). • If SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the error flag is set when a match occurs between SHIFTBUF and the shifter. • If SHIFTCTL*n*[SMOD] = 111b (Logic mode), the error flag is set when the output of the programmable logic block is asserted. For SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the flag can also be cleared when you read Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). **NOTE** This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

36.7.1.8 Timer Status Flag (TIMSTAT)

Offset

Register	Offset
TIMSTAT	18h

Function
Reports timer 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												TSF			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 TSF	Timer Status Flag Indicates timer status. This flag is set depending on Timer mode: • In 8-bit baud counter mode, this flag is set when the upper 8-bit counter equals zero and decrements. • In 8-bit high PWM mode, this flag is set when the upper 8-bit counter equals zero and decrements. • In 16-bit counter mode, this flag is set when the 16-bit counter equals zero and decrements. • In 16-bit counter disable mode, TSF is set when a timer disable event is detected. • In 8-bit word counter mode, TSF is set when the upper 8-bit counter equals zero and decrements. • In 8-bit low PWM mode, TSF is set when the upper 8-bit counter equals zero and decrements. • In 16-bit input capture mode, TSF is set when a timer disable event is detected and the flag is clear. In this mode, you must read Timer Control (TIMCTL0 - TIMCTL3) only when TSF is set. **NOTE** This field behaves differently for register reads and writes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

36.7.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

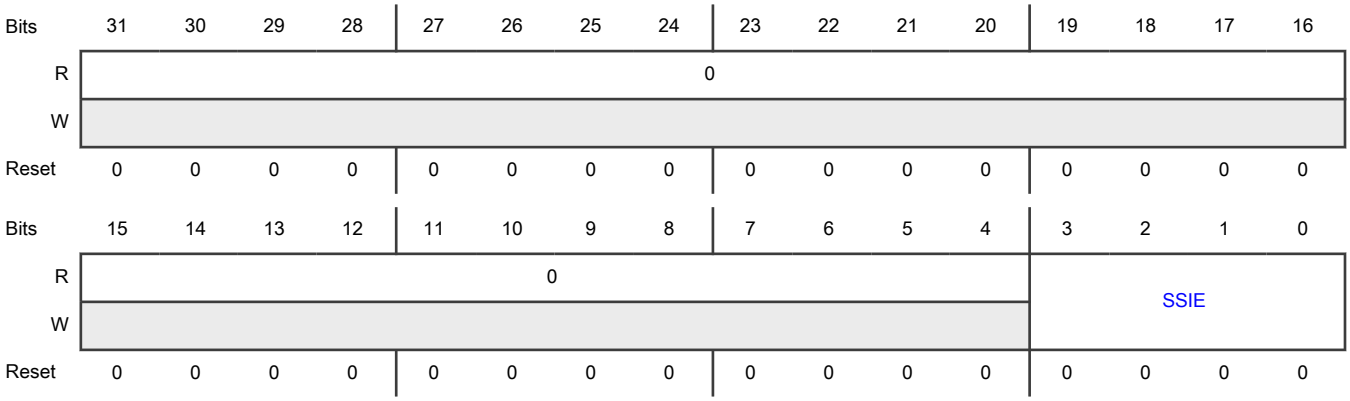
Offset

Register	Offset
SHIFTSIEN	20h

Function

Enables shifter status interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when the corresponding SHIFTSTAT[SSF] flag is set. If you write 0 to this field, SHIFTSTAT[SSF] is disabled; and if you write 1 to this field, SHIFTSTAT[SSF] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

36.7.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

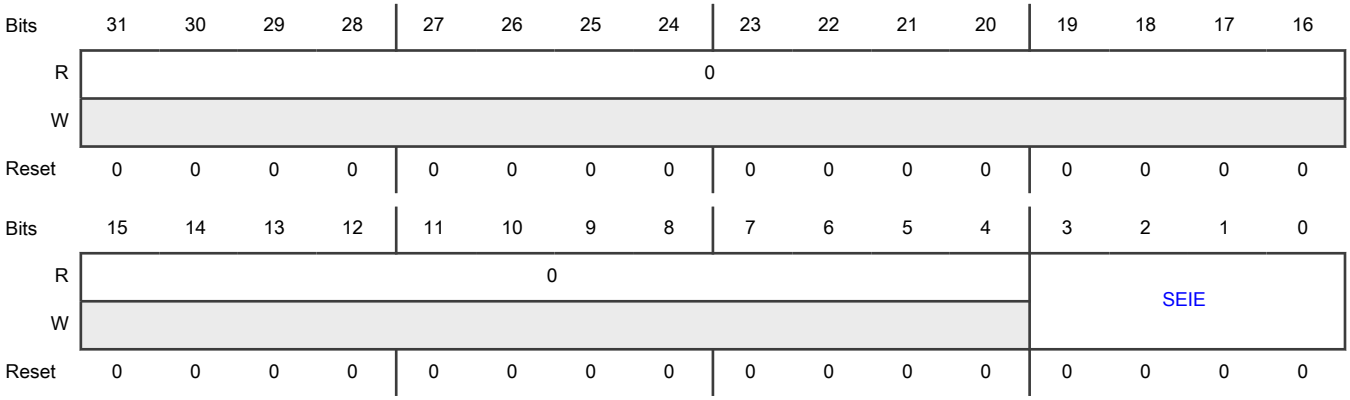
Offset

Register	Offset
SHIFTEIEN	24h

Function

Enables shifter error interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when the corresponding SHIFTERR[SEF] flag is set. If you write 0 to this field, SHIFTERR[SEF] is disabled; and if you write 1 to this field, SHIFTERR[SEF] is enabled. 0b - Disable 1b - Enable

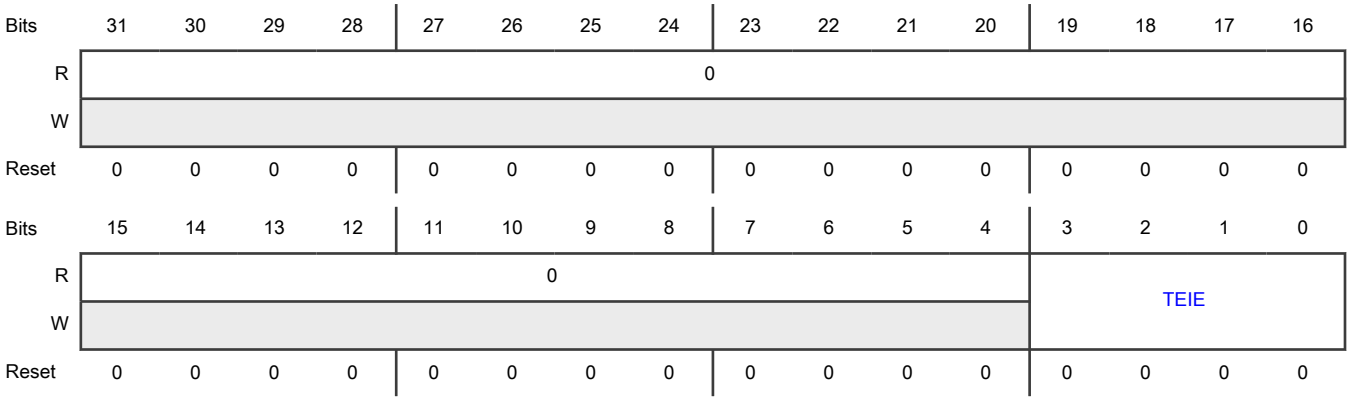
36.7.1.11 Timer Interrupt Enable (TIMIEN)

Offset

Register	Offset
TIMIEN	28h

Function
Enables timer status interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when the corresponding TIMSTAT[TSF] flag is set. If you write 0 to this field, TIMSTAT[TSF] is disabled; and if you write 1 to this field, TIMSTAT[TSF] is enabled. 0b - Disable 1b - Enable

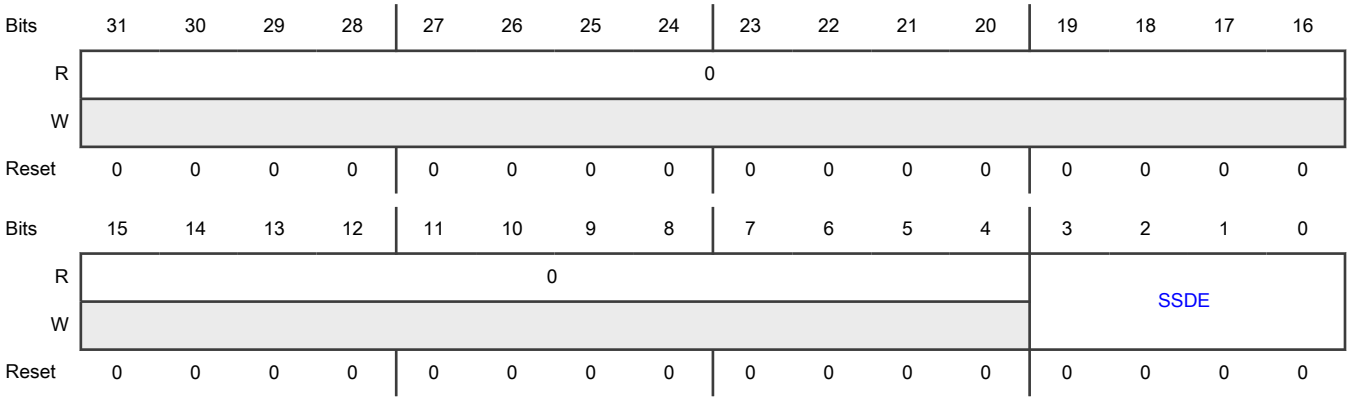
36.7.1.12 Shifter Status DMA Enable (SHIFTSDEN)

Offset

Register	Offset
SHIFTSDEN	30h

Function
Enables shifter DMA requests.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when the corresponding SHIFTSTAT[SSF] flag is set. If you write 0 to this field, SHIFTSTAT[SSF] is disabled; and if you write 1 to this field, SHIFTSTAT[SSF] is enabled. 0b - Disable 1b - Enable

36.7.1.13 Timer Status DMA Enable (TIMERSDEN)

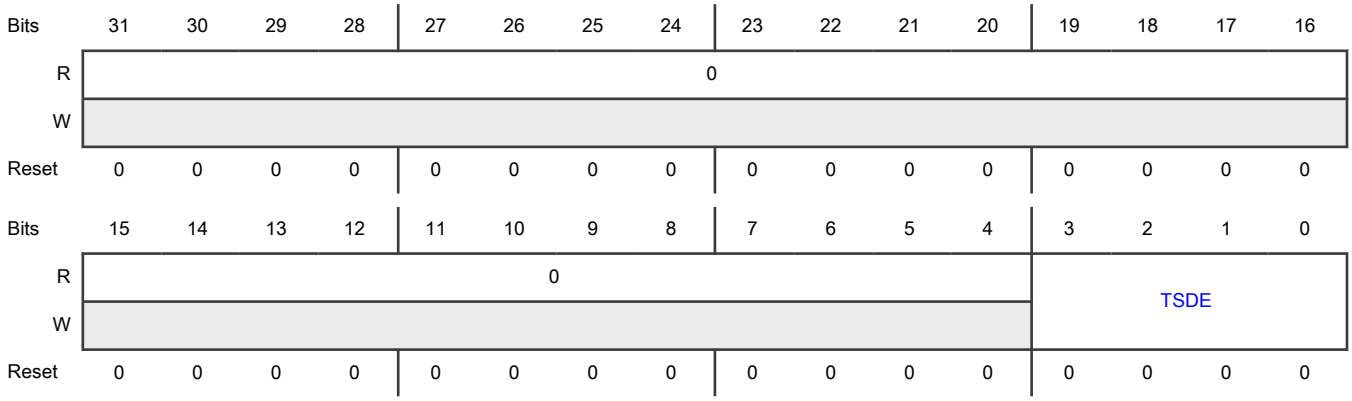
Offset

Register	Offset
TIMERSDEN	38h

Function

Enables DMA requests when the timer status flag is set.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TSDE	Timer Status DMA Enable Enables DMA request generation when the corresponding TIMSTAT[TSF] flag is set. When the timer status DMA request is enabled, reading or writing to a timer compare register clears the corresponding timer status register. The DMA must therefore read or write to the timer compare register as part of the DMA transfer; otherwise, the DMA request remains asserted. 0b - Disable 1b - Enable

36.7.1.14 Shifter State (SHIFTSTATE)

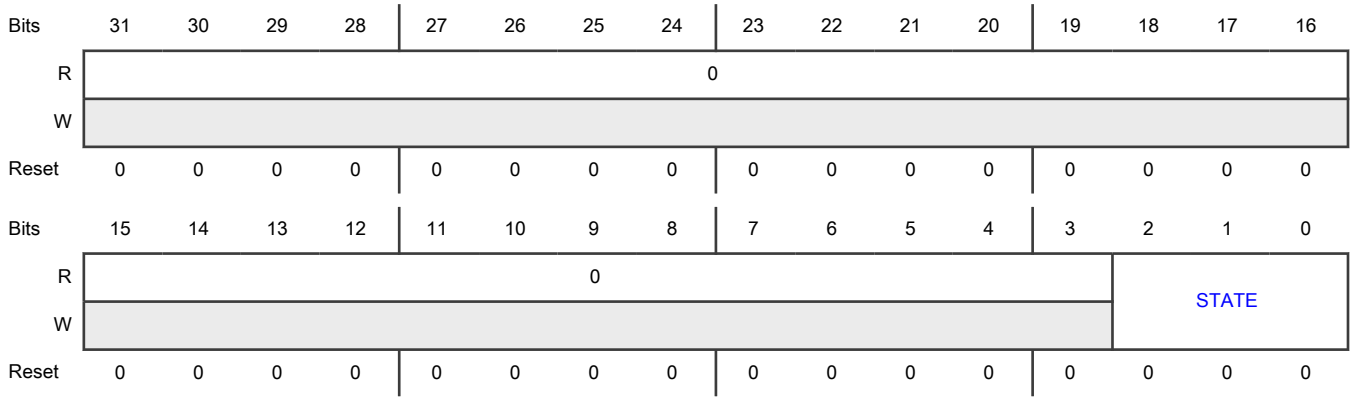
Offset

Register	Offset
SHIFTSTATE	40h

Function

Contains a pointer to track the current shifter.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 STATE	Current State Pointer Maintains a pointer to track 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 return of an incorrect state. The value that you write to this field overrides the current state.

36.7.1.15 Trigger Status (TRGSTAT)

Offset

Register	Offset
TRGSTAT	48h

Function

Contains external trigger 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												ETSF			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 ETSF	External Trigger Status Flag Specifies whether the external trigger status flag is set when a rising edge is detected on the corresponding external trigger input. **NOTE** This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

36.7.1.16 External Trigger Interrupt Enable (TRIGIEN)

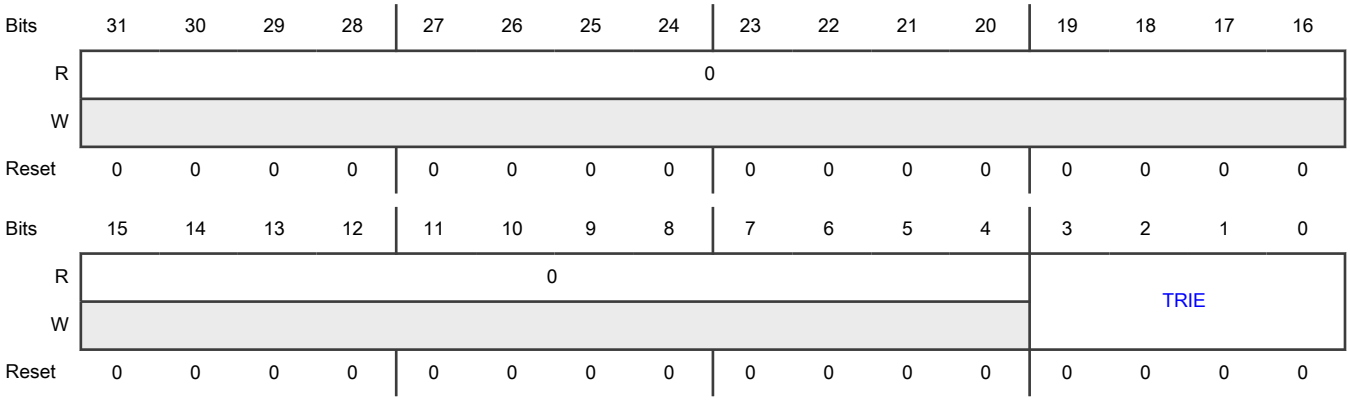
Offset

Register	Offset
TRIGIEN	4Ch

Function

Enables external trigger interrupts.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 TRIE	External Trigger Interrupt Enable Enables interrupt generation when the corresponding TRGSTAT[ETSF] flag is set. If you write 0 to this field, TRGSTAT[ETSF] is disabled, and if you write 1 to this field, TRGSTAT[ETSF] is enabled. 0b - Disable 1b - Enable

36.7.1.17 Pin Status (PINSTAT)

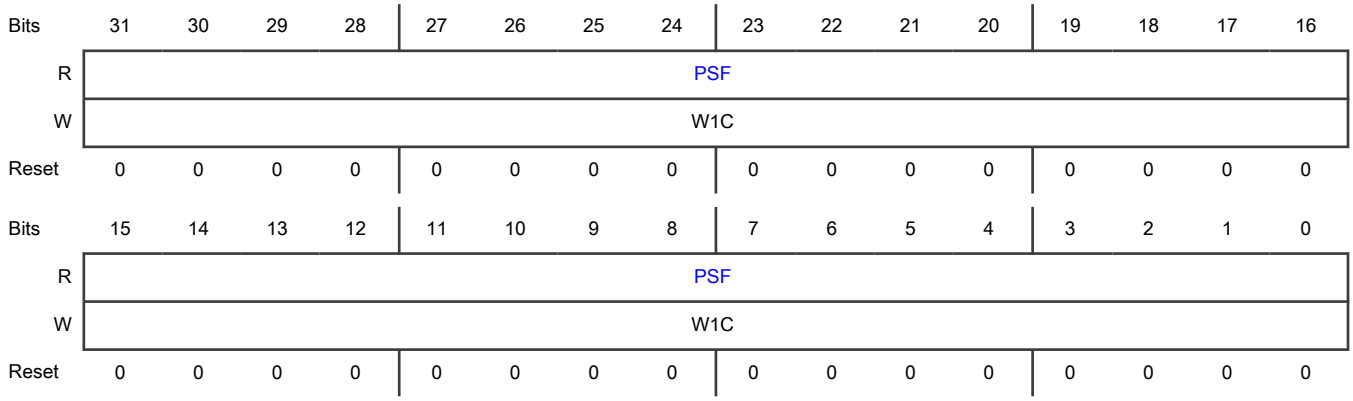
Offset

Register	Offset
PINSTAT	50h

Function

Contains pin status flags.

Diagram



Fields

Field	Function
31-0 PSF	Pin Status Flag Indicates whether the pin status flag is set when a rising edge or falling edge (if configured) is detected on the corresponding pin, as configured by the pin. **NOTE** This field behaves differently for register reads and writes. When reading 0000_0000_0000_0000_0000_0000_0000_0000b - Clear 0000_0000_0000_0000_0000_0000_0000_0001b - Set When writing 0000_0000_0000_0000_0000_0000_0000_0000b - No effect 0000_0000_0000_0000_0000_0000_0000_0001b - Clear the flag

36.7.1.18 Pin Interrupt Enable (PINIEN)

Offset

Register	Offset
PINIEN	54h

Function

Enables 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 the corresponding PINSTAT[PSF] flag is set. If you write 0 to this field, PINSTAT[PSF] is disabled, and if you write 1 to this field, PINSTAT[PSF] is enabled. 0b - Disable 1b - Enable

36.7.1.19 Pin Rising Edge Enable (PINREN)

Offset

Register	Offset
PINREN	58h

Function
Enables the pin status flag on a rising edge.

Diagram

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

Fields

Field	Function
31-0 PRE	Pin Rising Edge Specifies whether the pin status flag is set whenever a rising edge is detected on the pin. 0b - Not set 1b - Set

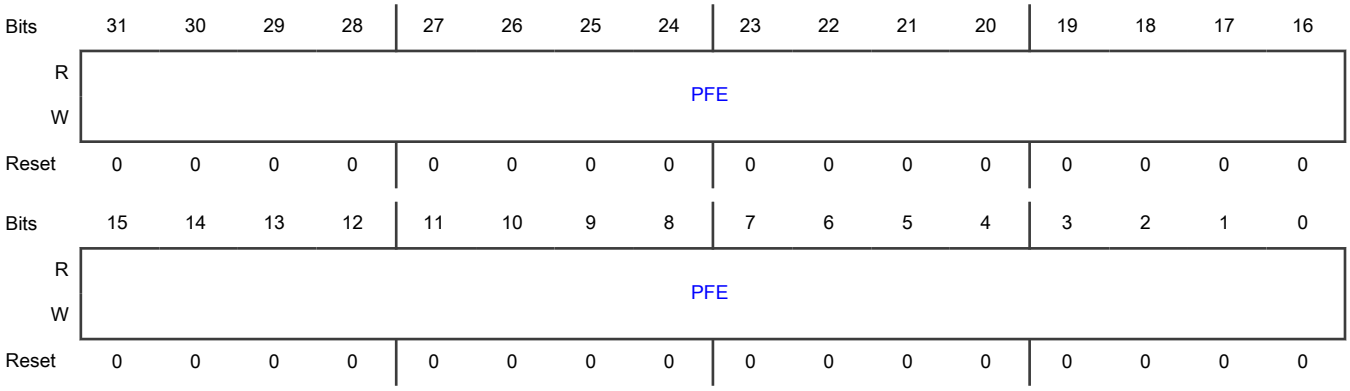
36.7.1.20 Pin Falling Edge Enable (PINFEN)

Offset

Register	Offset
PINFEN	5Ch

Function
Enables the pin status flag on a falling edge.

Diagram



Fields

Field	Function
31-0 PFE	Pin Falling Edge Specifies whether the pin status flag is set whenever a falling edge is detected on the pin. 0b - Not set 1b - Set

36.7.1.21 Pin Output Data (PINOUTD)

Offset

Register	Offset
PINOUTD	60h

Function
Contains data output when direct pin output is 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 1b - Logic one

36.7.1.22 Pin Output Enable (PINOUTE)

Offset

Register	Offset
PINOUTE	64h

Function
Enables pin output.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OUTE															
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	OUTE															
W																
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. If this field is 0, the pin is controlled by timer/shifter configuration, and if this field is 1, pin is an output and driven with the value of Pin Output Data (PINOUTD). 0b - Controlled by timer/shifter configuration 1b - Output; driven with value of PINOUTD

36.7.1.23 Pin Output Disable (PINOUTDIS)

Offset

Register	Offset
PINOUTDIS	68h

Function
Disables pin output.

Diagram

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

Fields

Field	Function
31-0 OUTDIS	Output Disable Configures the corresponding pins to disable direct output. If this field is 1, the corresponding fields in Pin Output Data (PINOUTD) and Pin Output Enable (PINOUTE) become 0. 0b - No effect 1b - Corresponding fields become 0

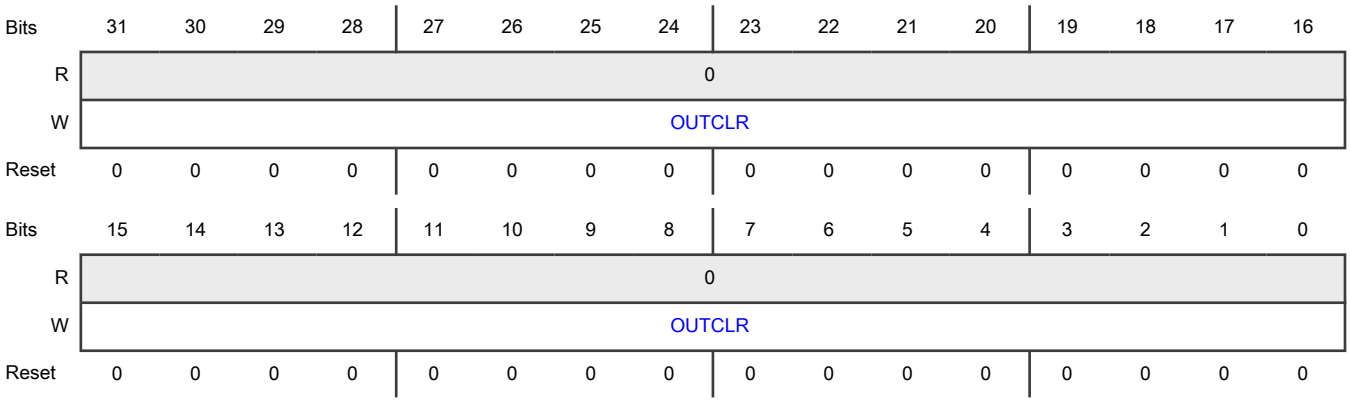
36.7.1.24 Pin Output Clear (PINOUTCLR)

Offset

Register	Offset
PINOUTCLR	6Ch

Function
Clears pin output.

Diagram



Fields

Field	Function
31-0 OUTCLR	Output Clear Configures the corresponding pins to output zero. If this field is 1, the corresponding field in Pin Output Data (PINOUTD) becomes 0 and the one in Pin Output Enable (PINOUTE) becomes 1. 0b - No effect 1b - Corresponding field in PINOUTD becomes 0; corresponding field in PINOUTE becomes 1

36.7.1.25 Pin Output Set (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 the corresponding pins to output logic one. If this field is 1, the corresponding fields in Pin Output Data (PINOUTD) and Pin Output Enable (PINOUTE) become 1. 0b - No effect 1b - Corresponding fields become 1

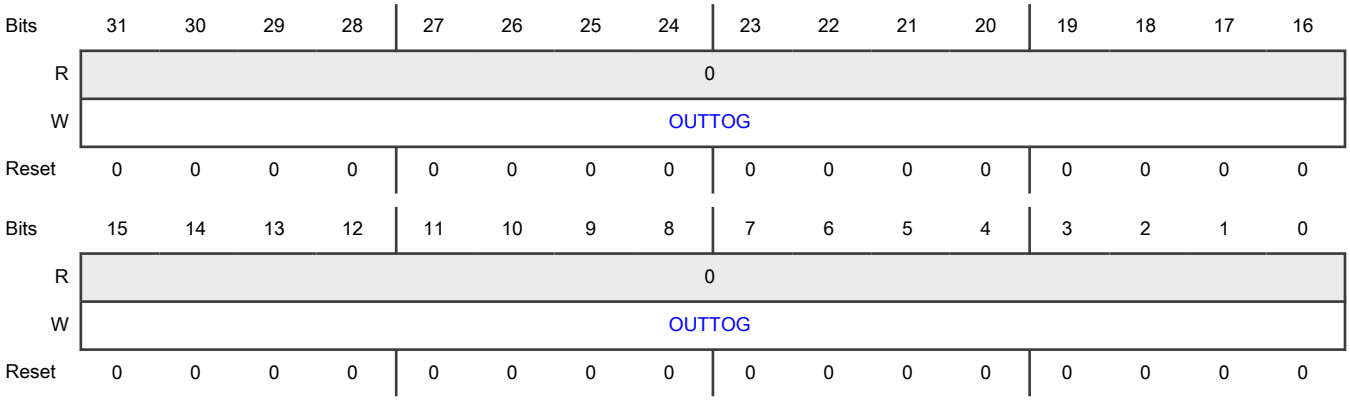
36.7.1.26 Pin Output Toggle (PINOUTTOG)

Offset

Register	Offset
PINOUTTOG	74h

Function
Toggles pin output.

Diagram



Fields

Field	Function
31-0 OUTTOG	Output Toggle Configures the corresponding pins to toggle. If this field is 1, the corresponding field in Pin Output Data (PINOUTD) is inverted and the one in Pin Output Enable (PINOUTE) becomes 1. 0b - No effect 1b - Corresponding field in PINOUTD is inverted; corresponding field in PINOUTE becomes 1

36.7.1.27 Shifter Control (SHIFTCTL0 - SHIFTCTL3)

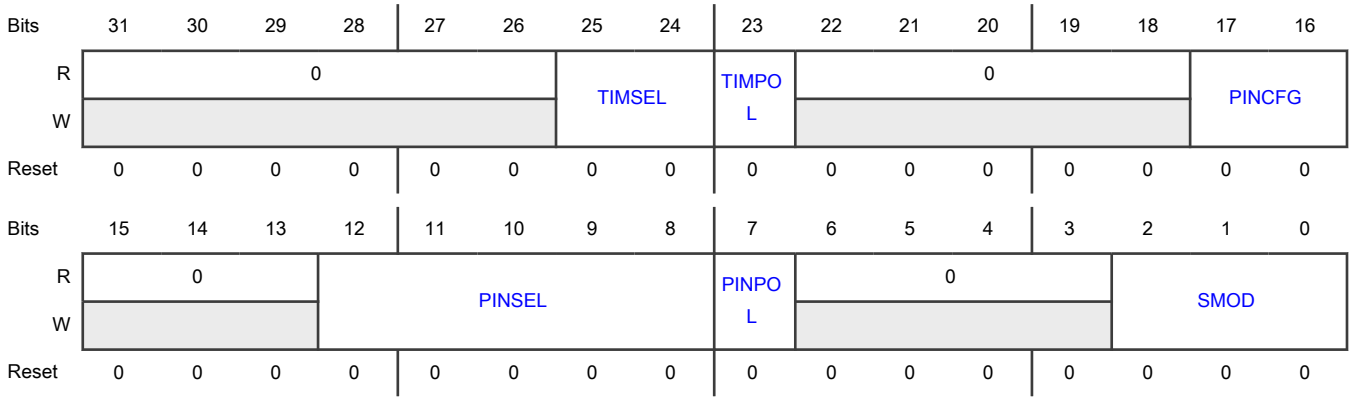
Offset

Register	Offset
SHIFTCTL0	80h
SHIFTCTL1	84h
SHIFTCTL2	88h
SHIFTCTL3	8Ch

Function

Provides shifter controls.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 TIMSEL	Timer Select Selects which timer is used for controlling the logic or shift register and generating the shift clock. TIMSEL = i selects TIMERi.
23 TIMPOL	Timer Polarity Determines whether the shift occurs on the positive edge or negative edge of the shift clock. 0b - Positive edge 1b - Negative edge
22-18 —	Reserved
17-16 PINCFG	Shifter Pin Configuration Specifies shifter pin configuration. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. **NOTE** When initially configuring PINCFG as 11b, FLEXIO may briefly drive the pin low. To avoid this, you can configure PINCFG as 10b along with the rest of the Control register and then perform a subsequent write to set PINCFG as 11b. Likewise, when changing the value of PINCFG from 11b to 00b, you must perform an initial write to set PINCFG as 10b and then perform a subsequent write to update the rest of the Control register with the value of PINCFG as 00b. 00b - Shifter pin output disabled 01b - Shifter pin open-drain or bidirectional output enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	Reserved
12-8 PINSEL	Shifter Pin Select Selects the pin that is used by the shifter input or output. PINSEL = i selects the FXIO_Di pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register.
7 PINPOL	Shifter Pin Polarity Specifies the shifter pin polarity. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to this register. 0b - Active high 1b - Active low
6-3 —	Reserved
2-0 SMOD	Shifter Mode Configures the mode of the shifter. 000b - Disable 001b - Receive mode; capture the current shifter content into SHIFTBUF on expiration of the timer 010b - Transmit mode; load SHIFTBUF contents into the shifter on expiration of the timer 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 store programmable state attributes 111b - Logic mode; SHIFTBUF contents implement programmable logic lookup table

36.7.1.28 Shifter Configuration (SHIFTCFG0 - SHIFTCFG3)

Offset

Register	Offset
SHIFTCFG0	100h
SHIFTCFG1	104h

Table continues on the next page...

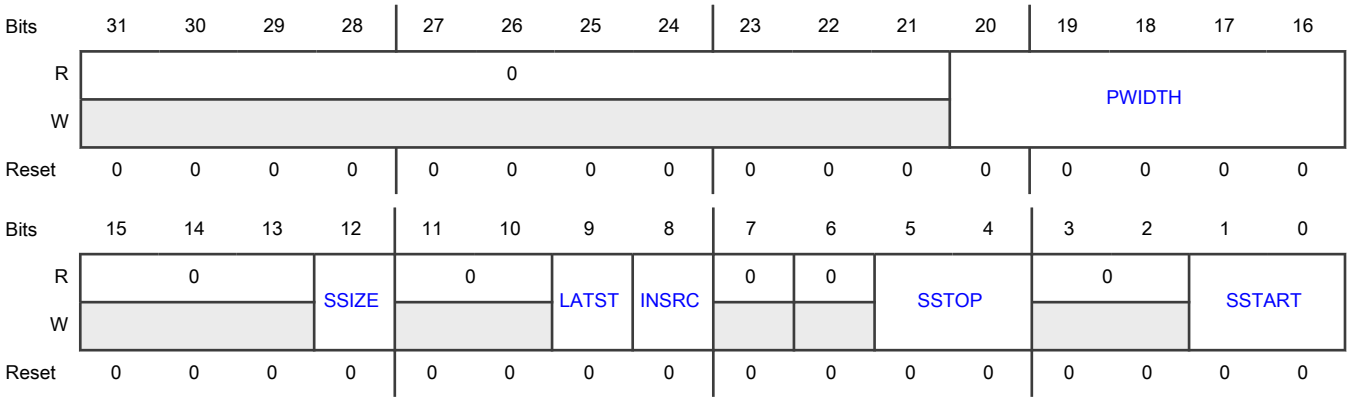
Table continued from the previous page...

Register	Offset
SHIFTCFG2	108h
SHIFTCFG3	10Ch

Function

Provides fields for shifter configuration.

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 PWIDTH	Parallel Width Configures the number of bits to be shifted on each shift clock for all shifters: • 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 that support parallel transmit (SHIFTER0, SHIFTER4, ...) or parallel receive (SHIFTER3, SHIFTER7, ...), this field, together with PINSEL, also selects the pins to be driven or sampled on each shift clock: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Shifters that do not support parallel transmit or parallel receive only support parallel shift when SHIFTCFG*n*[INSRC] = 1. If SHIFTCTL*n*[SMOD] = 110b (State mode), use this field to disable state outputs (see State mode).

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 —	Reserved
12 SSIZE	Shifter Size Configures the size of the Shift registers. A 24-bit Shift register shifts data only into bits [23:0] and does not update bits [31:24] during shift operations. When the Shift register is configured for a 24-bit shift, configuring PWIDTH as 8..15 performs a 12-bit shift and PWIDTH as 16..31 performs a 24-bit shift. 0b - 32-bit 1b - 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 - Store the pre-shift register state 1b - Store the post-shift register state
8 INSRC	Input Source Selects the input source for the shifter. Configuring this field as 1 is not supported for the last shifter. 0b - Pin 1b - Shifter n+1 output
7 —	Reserved
6 —	Reserved
5-4 SSTOP	Shifter Stop Allows automatic stop bit insertion, if the selected timer has also enabled a stop bit, when SHIFTCTL*n*[SMOD] is 10b (Transmit mode). If SHIFTCTL*n*[SMOD] is 1b or 100b (Receive mode or Match Store mode), this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. If SHIFTCTL*n*[SMOD] is 110b (State mode), this field disables state outputs (see State mode). If SHIFTCTL*n*[SMOD] is 111b (Logic mode), this field masks logic pin inputs (see Logic mode). 00b - Stop bit disabled for Transmitter, Receiver, and Match Store modes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Stop bit disabled for Transmitter, Receiver, and Match Store modes; when timer is in stop condition, Receiver and Match Store modes store receive data on the configured shift edge 10b - Transmitter mode outputs stop bit value 0 in Match Store mode; if stop bit is not 0, Receiver and Match Store modes set error flag (when timer is in stop condition, these modes also store receive data on the configured shift edge) 11b - Transmitter mode outputs stop bit value 1 in Match Store mode; if stop bit is not 1, Receiver and Match Store modes set error flag (when timer is in stop condition, these modes also store receive data on the configured shift edge)
3-2 —	Reserved
1-0 SSTART	Shifter Start Allows automatic start bit insertion, if the selected timer has also enabled a start bit, when SHIFTCTL*n*[SMOD] is 10b (Transmit mode). If SHIFTCTL*n*[SMOD] = 1b (Receive mode) or 100b (Match Store mode), this field allows automatic start bit checking if the selected timer has also enabled a start bit. If SHIFTCTL*n*[SMOD] is 110b (State mode), this field disables state outputs (see State mode). If SHIFTCTL*n*[SMOD] = 111b (Logic mode), this field masks logic pin inputs (see Logic mode). 00b - Start bit disabled for Transmitter, Receiver, and Match Store modes; Transmitter mode loads data on enable 01b - Start bit disabled for Transmitter, Receiver, and Match Store modes; Transmitter mode loads data on first shift 10b - Transmitter mode outputs start bit value 0 before loading data on first shift; if start bit is not 0, Receiver and Match Store modes set error flag 11b - Transmitter mode outputs start bit value 1 before loading data on first shift; if start bit is not 1, Receiver and Match Store modes set error flag

36.7.1.29 Shifter Buffer (SHIFTBUF0 - SHIFTBUF3)

Offset

Register	Offset
SHIFTBUF0	200h
SHIFTBUF1	204h
SHIFTBUF2	208h
SHIFTBUF3	20Ch

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 Contains the data to be matched with the shifter contents and is used for various other functions, depending on the setting of SHIFTCTL*n*[SMOD]: • If SHIFTCTL0[SMOD] is 1b (Receive mode), shifter data is transferred into SHIFTBUF at the expiration of the timer. You must read this register only when the corresponding SHIFTSTAT[SSF] flag is set, indicating that new shifter data is available. • If SHIFTCTL0[SMOD] is 10b (Transmit mode), SHIFTBUF data is transferred into the shifter before the timer begins. • If SHIFTCTL0[SMOD] is 100b (Match Store mode), 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. Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. You must read this register only when the corresponding shifter status flag is set, indicating that new shifter data is available. • If SHIFTCTL0[SMOD] is 101b (Match Continuous mode), 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). • If SHIFTCTL0[SMOD] is 111b (Logic mode), SHIFTBUF[31:0] implements a 5-input, 32-bit programmable logic lookup table (see Logic mode). • If SHIFTCTL0[SMOD] is 110b (State mode), use SHIFTBUF[31:24] to drive the output value when this shifter is selected by the current state pointer and use SHIFTBUF[23:0] to configure the value of the next state transition (see State mode).

36.7.1.30 Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3)

Offset

Register	Offset
SHIFTBUFBIS0	280h

Table continues on the next page...

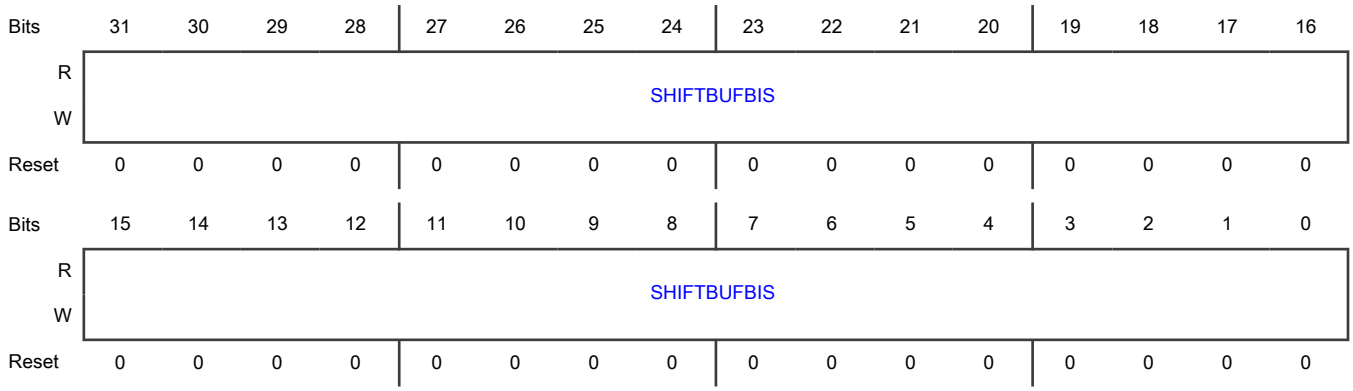
Table continued from the previous page...

Register	Offset
SHIFTBUFBIS1	284h
SHIFTBUFBIS2	288h
SHIFTBUFBIS3	28Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is bit-swapped.

Diagram



Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register are bit-swapped. Reads return SHIFTBUF[0:31].

36.7.1.31 Shifter Buffer Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS3)

Offset

Register	Offset
SHIFTBUFBYS0	300h
SHIFTBUFBYS1	304h
SHIFTBUFBYS2	308h
SHIFTBUFBYS3	30Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is 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 Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register are byte-swapped. Reads return {SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24]}.

36.7.1.32 Shifter Buffer Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS3)

Offset

Register	Offset
SHIFTBUFBBS0	380h
SHIFTBUFBBS1	384h
SHIFTBUFBBS2	388h
SHIFTBUFBBS3	38Ch

Function

Contains the register data for Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but it 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 Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), except that reads or writes to this register are bit-swapped within each byte. Reads return {SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7]}.

36.7.1.33 Timer Control (TIMCTL0 - TIMCTL3)

Offset

Register	Offset
TIMCTL0	400h
TIMCTL1	404h
TIMCTL2	408h
TIMCTL3	40Ch

Function
Controls various settings for timer *n*.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		TRGSEL						TRGPOL	TRGSRC	0		PINCFG			
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		PINSEL						PINPOL	PININS	ONETIM	0	TIMOD			
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 TRGSEL	Trigger Select Selects the trigger.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The valid values for TRGSEL depend on the configuration of Parameter (PARAM): • If TRGSRC = 1, the valid values for *n* depend on the settings of PARAM[PIN], PARAM[TIMER], and PARAM[SHIFTER]. • If TRGSRC = 0, the valid values for *n* depend on PARAM[TRIGGER]. See the chip-specific FLEXIO information for external trigger selection. **NOTE** For a pin, *n* = 0 to 31, for a shifter, *n* = 0 to 3, and for a timer, *n* = 0 to 3. If TRGSRC = 0, configure the trigger selection as *n* = external trigger *n* input. If TRGSRC = 1, you can configure the internal trigger to select an input pin as 2×*n* = pin *n* input. If TRGSRC = 1, you can configure the internal trigger to select a shifter or timer signal as: • 4×*n* + 1 = shifter *n* status flag • 4×*n* + 3 = timer *n* trigger output Following are the values for expanded internal trigger selection (TRGSRC = 1): • 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 pin 31, shifter 3, and timer 3.
23 TRGPOL	Trigger Polarity Specifies whether the trigger is active high or active low. 0b - Active high 1b - Active low
22 TRGSRC	Trigger Source Specifies whether the selected trigger source is external or internal. 0b - External 1b - Internal
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 = 11b), this field takes effect when you write to the register. **NOTE** When you initially configure PINCFG as 11b, FLEXIO may briefly drive the pin low. To avoid this, configure PINCFG as 10b along with the rest of the Control register and then perform a subsequent write to set the value of PINCFG as 11b. Likewise, when changing the value of PINCFG from 11b to 00b, you must perform an initial write to set PINCFG as 10b, and then perform a subsequent write to update the rest of the Control register with PINCFG as 00b. 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 the pin that is used by the timer input or output. PINSEL = i selects the FXIO_Di pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register.
7 PINPOL	Timer Pin Polarity Specifies the timer pin polarity. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. 0b - Active high 1b - Active low
6 PININS	Timer Pin Input Select Specifies what selects the timer pin input. If this field is 1, the timer input pin is different from the timer output pin. PINSEL must select an even-numbered pin when this field is 1, which means that the output pin is even-numbered and input pin is odd-numbered. 0b - PINSEL selects timer pin input and output 1b - PINSEL + 1 selects the timer pin input; timer pin output remains selected by PINSEL
5 ONETIM	Timer One Time Operation Configures the timer to perform a single enable or disable iteration. Clear the timer status flag for the timer to be enabled again. 0b - Generate the timer enable event as normal 1b - Block the timer enable event unless the timer status flag is clear

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-3 —	Reserved
2-0 TIMOD	Timer Mode Specifies the timer mode: • 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. 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. 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. 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. The upper 8 bits are used to configure the high period of the timer shift clock. Use another timer or external signal to configure the shifter bit count. • In 16-bit input capture mode, the inverted value of the 16-bit counter is latched into the compare register when a timer counter disable condition is detected (as configured by TIMCFG*n*[TIMDIS]). This also sets the timer status flag. The timer counter is immediately restarted from FFFFh. 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

36.7.1.34 Timer Configuration (TIMCFG0 - TIMCFG3)

Offset

Register	Offset
TIMCFG0	480h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
TIMCFG1	484h
TIMCFG2	488h
TIMCFG3	48Ch

Function

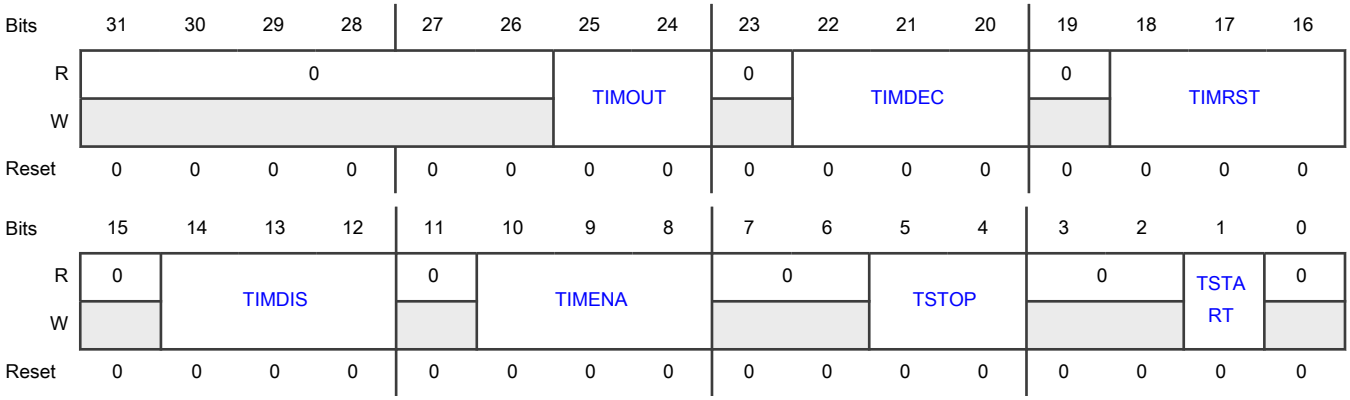
Controls various aspects of timer configuration.

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 (timer 0, for example).

NOTE

The pin and trigger level and edges specified in this register refer to the signal state after being modified by the settings of TIMCTL*n*[PINPOL] and TIMCTL*n*[TRGPOL]. For example, "trigger low" means that a trigger is actually at logic level 1 if TIMCTL*n*[TRGPOL] is 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 - Logic one when enabled; not affected by timer reset 01b - Logic zero when enabled; not affected by timer reset 10b - Logic one when enabled and on timer reset 11b - Logic zero when enabled and on timer reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 —	Reserved
22-20 TIMDEC	Timer Decrement Configures the source of the timer decrement and that 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 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 full 16 bits of the counter. 000b - Never reset timer 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. 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 Specifies whether the stop bit is enabled. 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 - Disabled 01b - Enabled on timer compare 10b - Enabled on timer disable 11b - Enabled on timer compare and timer disable
3-2 —	Reserved
1	Timer Start

Table continues on the next page...

Table continued from the previous page...

Field	Function
TSTART	Specifies whether the start bit is enabled. If it is enabled, configured shifters output the contents of the start bit when the timer is enabled. The timer counter reloads from the compare register on the first rising edge of the shift clock. 0b - Disabled 1b - Enabled
0 —	Reserved

36.7.1.35 Timer Compare (TIMCMP0 - TIMCMP3)

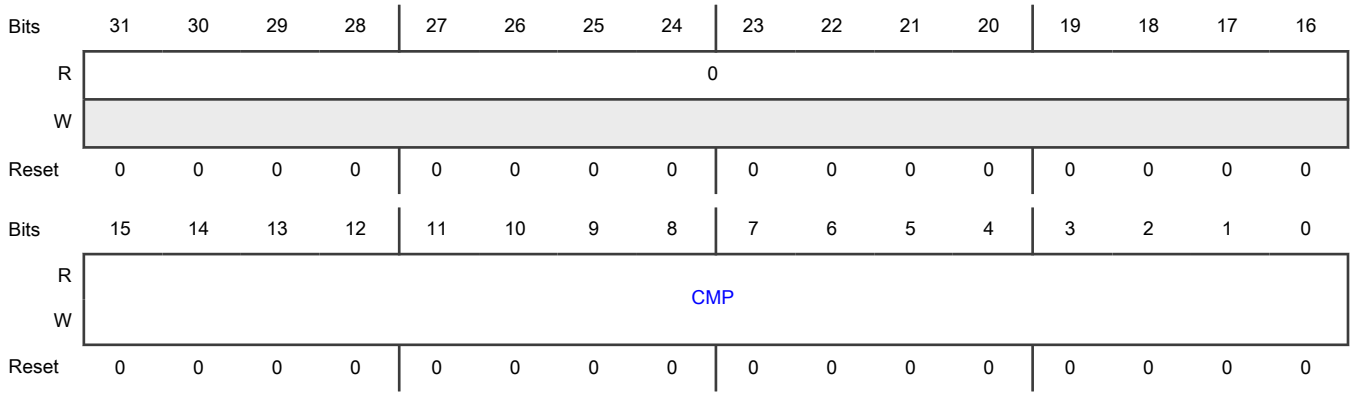
Offset

Register	Offset
TIMCMP0	500h
TIMCMP1	504h
TIMCMP2	508h
TIMCMP3	50Ch

Function

Contains the timer compare value.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15-0 CMP	Timer Compare Value Loads 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 as (CMP[7:0] + 1) × 2. The upper 8 bits configure the number of bits in each word as (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) as (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 as (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) as (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 as (CMP[15:0] + 1) ÷ 2. In 8-bit word counter mode, the lower 8 bits configure the number of bits in each word as (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). 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. You must read this register only when the timer status flag is set.

36.7.1.36 Shifter Buffer Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS3)

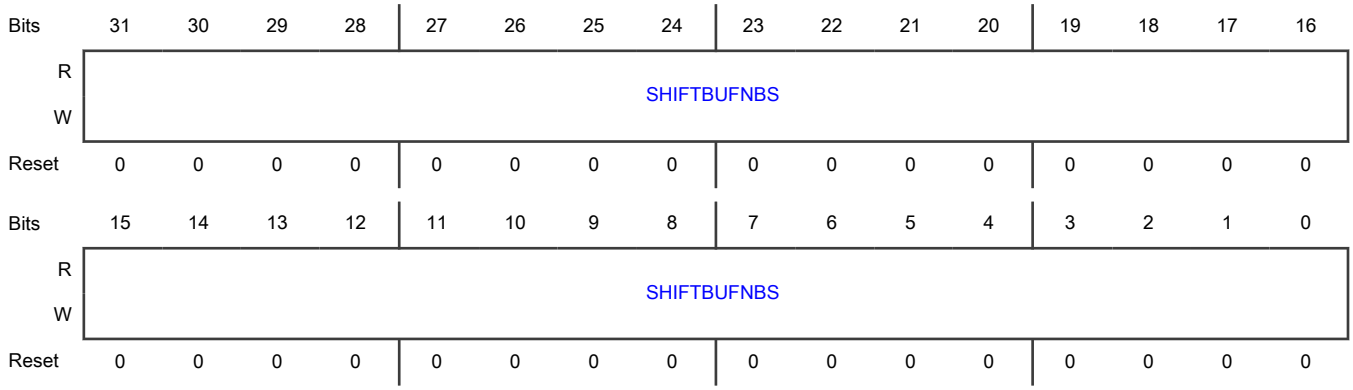
Offset

Register	Offset
SHIFTBUFNBS0	680h
SHIFTBUFNBS1	684h
SHIFTBUFNBS2	688h
SHIFTBUFNBS3	68Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is nibble-swapped within each byte.

Diagram



Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or 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]}.

36.7.1.37 Shifter Buffer Halfword Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS3)

Offset

Register	Offset
SHIFTBUFHWS0	700h
SHIFTBUFHWS1	704h
SHIFTBUFHWS2	708h
SHIFTBUFHWS3	70Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is halfword-swapped.

Diagram

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

Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register are halfword-swapped. Reads return {SHIFTBUF[15:0], SHIFTBUF[31:16]}.

36.7.1.38 Shifter Buffer Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS3)

Offset

Register	Offset
SHIFTBUFNIS0	780h
SHIFTBUFNIS1	784h
SHIFTBUFNIS2	788h
SHIFTBUFNIS3	78Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is nibble-swapped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SHIFTBUFNIS															
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	SHIFTBUFNIS															
W																
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 Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or 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]}.

36.7.1.39 Shifter Buffer Odd Even Swapped (SHIFTBUFOES0 - SHIFTBUFOES3)

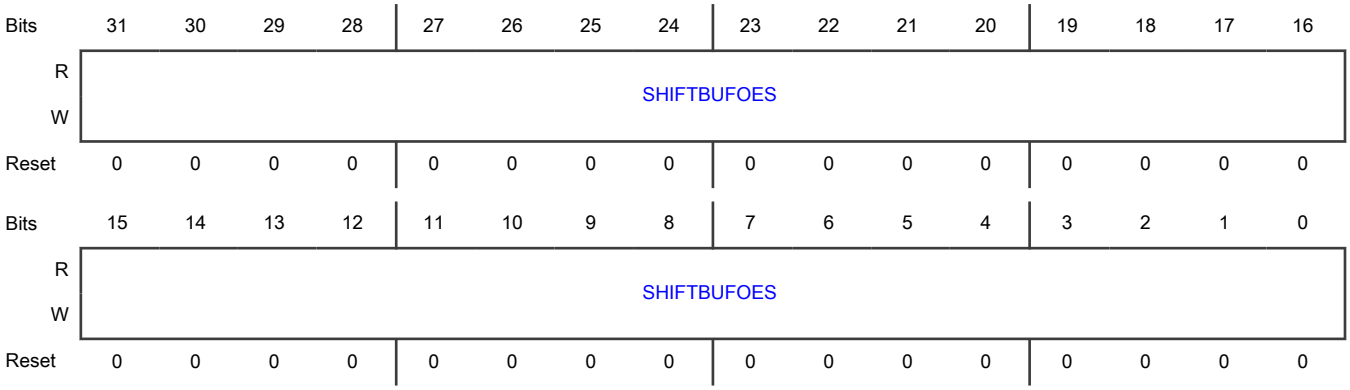
Offset

Register	Offset
SHIFTBUFOES0	800h
SHIFTBUFOES1	804h
SHIFTBUFOES2	808h
SHIFTBUFOES3	80Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it has odd and even bits partitioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFOES	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register have the odd and even bits partitioned separately. Only 32-bit accesses are supported for 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],

Table continues on the next page...

Field	Function
	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]}.

36.7.1.40 Shifter Buffer Even Odd Swapped (SHIFTBUFEOS0 - SHIFTBUFEOS3)

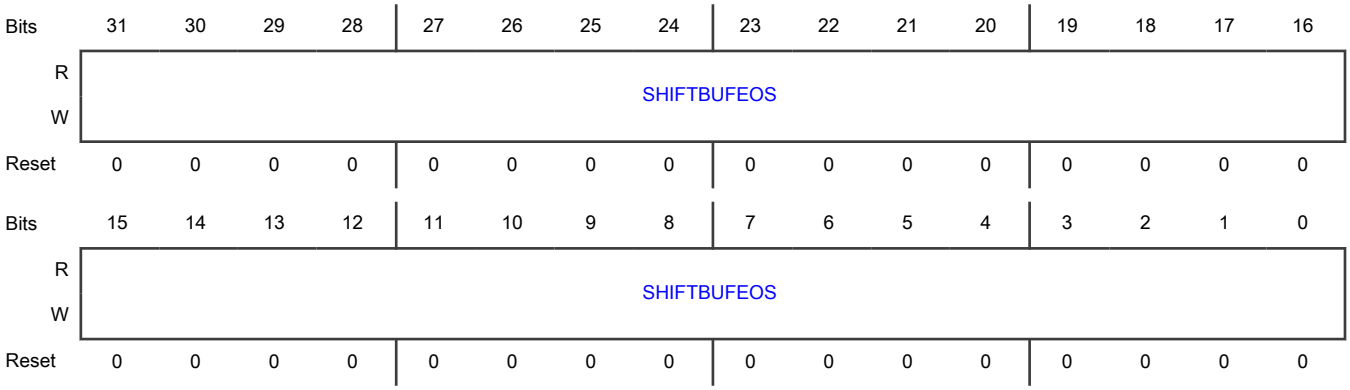
Offset

Register	Offset
SHIFTBUFEOS0	880h
SHIFTBUFEOS1	884h
SHIFTBUFEOS2	888h
SHIFTBUFEOS3	88Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, with even and odd bits partitioned separately.

Diagram



Fields

Field	Function
31-0 SHIFTBUFEOS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register have the even and odd bits partitioned separately. Only 32-bit accesses are supported for this register. Reads return {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], 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]}.

36.7.1.41 Shifter Buffer Halfword Byte Swapped (SHIFTBUFHBS0 - SHIFTBUFHBS3)

Offset

Register	Offset
SHIFTBUFHBS0	900h
SHIFTBUFHBS1	904h
SHIFTBUFHBS2	908h
SHIFTBUFHBS3	90Ch

Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is halfword byte-swapped.

Diagram

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

Fields

Field	Function
31-0 SHIFTBUFHBS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register are halfword byte-swapped. Reads return {SHIFTBUF[23:16], SHIFTBUF[31:24], SHIFTBUF[7:0], SHIFTBUF[15:8]}.

Chapter 37
Flexible Data Rate CAN (FlexCAN)

37.1 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.

37.1.1 Block diagram

Figure 149 shows the main submodules implemented in FlexCAN.

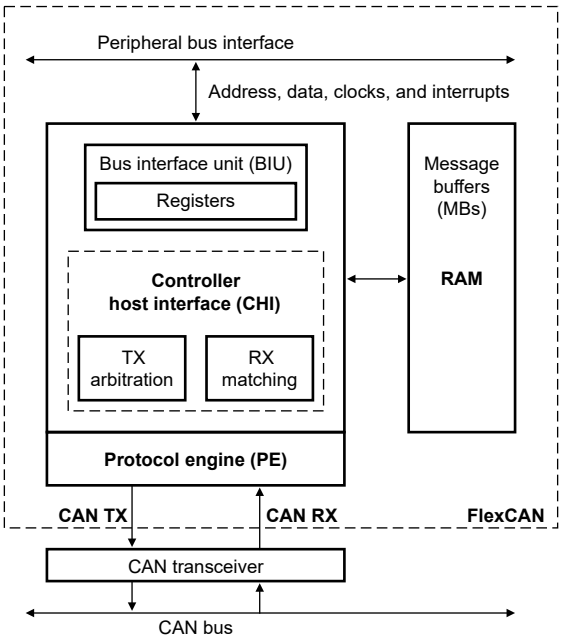


Figure 149. Block diagram

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

37.1.2 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

37.2 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 256). A transmit message buffer with a 1000b or 1001b code is also inactive (see Table 257).

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.

37.2.1 Modes of operation

FlexCAN has these functional modes:

Table 222. 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 Debug mode 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.

Table continues on the next page...

Table 222. Functional modes (continued)

Mode	Description
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 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 223. 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 224. 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.

Table continues on the next page...

Table 224. Low-power modes (continued)

Mode	Description
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 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).

37.2.1.1 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.

37.2.1.1.1 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.

37.2.1.1.2 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.

37.2.1.1.3 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 225. 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

37.2.1.1.4 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 226 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 226. 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.

37.2.1.1.5 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 223.

37.2.2 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 237.[1]
    d. Configure Data Length Code in bytes via MB_CS[DLC]. See Table 259 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.

[1] 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 256 and Table 257 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).

37.2.3 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.

37.2.3.1 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.

37.2.3.2 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].

37.2.3.2.1 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 227. 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)

37.2.3.2.2 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 228. 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.

37.2.3.3 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.

37.2.4 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 256 and Table 257 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 256. 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).

37.2.5 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 229.

Table 229. 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.

37.2.5.1 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

37.2.5.2 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.

37.2.6 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 230. 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.

37.2.6.1 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.

37.2.6.2 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.

37.2.6.3 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.

37.2.7 Move process

There are two types of move process: move-in and move-out.

37.2.7.1 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.

37.2.7.2 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.

37.2.8 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.

37.2.8.1 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.

37.2.8.2 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).

37.2.8.3 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.

37.2.9 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 150 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.

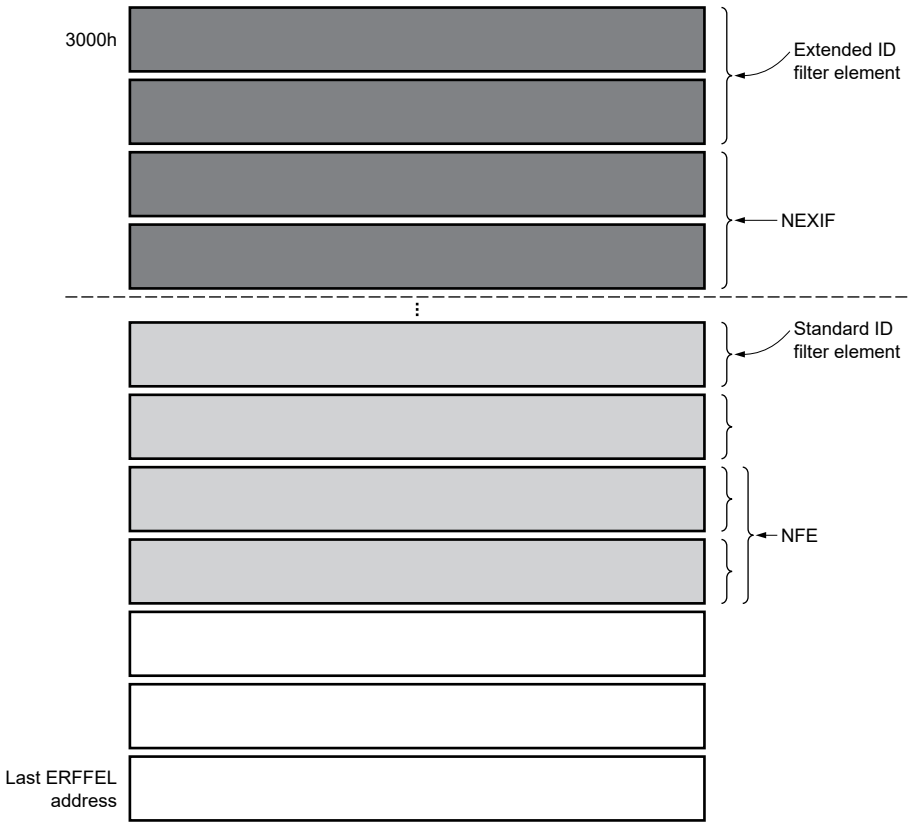


Figure 150. Number of enhanced RX FIFO filter elements

37.2.9.1 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 231.

Table 231. 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 232.

Table 232. 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 233.

Table 233. 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 234.

Table 234. 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 235.

Table 235. 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 236.

Table 236. 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	

37.2.9.2 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 259.

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 269 and Table 270. 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

37.2.9.3 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.

37.2.10 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 223 for information about the difference between FD and non-FD regarding this feature.

37.2.10.1 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 223 for information about the difference between FD and non-FD regarding this feature.

37.2.10.2 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.

37.2.11 ACK Suppression

FlexCAN does not send acknowledgment when connected through CAN HUB and ACK suppression is enabled by CTRL2[ASD]. When any of the on-chip CAN controllers (connected through CAN HUB) are transmitting, all the on-chip receivers do not send the acknowledgment of received frames. All the on-chip receivers accept the frames only when there is acknowledgment by external receivers. If there is no acknowledgment by external receivers, the frame is dropped and the BIT error is notified.

37.2.12 CAN protocol-related features

37.2.12.1 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].

37.2.12.2 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.

37.2.12.2.1 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 151 illustrates the message formats for Classical and FD frames with either standard or extended ID.

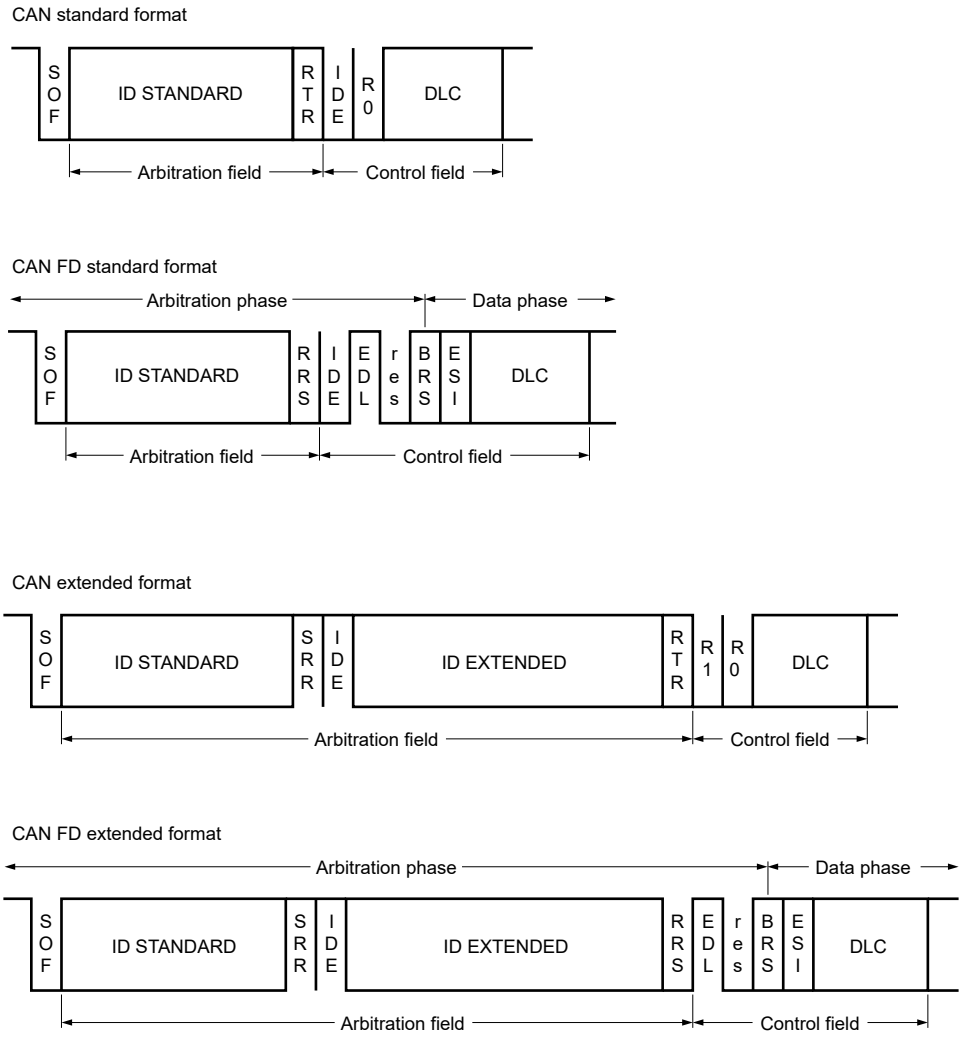


Figure 151. CAN message formats

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.

37.2.12.2.2 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 152 describes the mechanism for entering and leaving the data phase when the BRS bit is recessive.

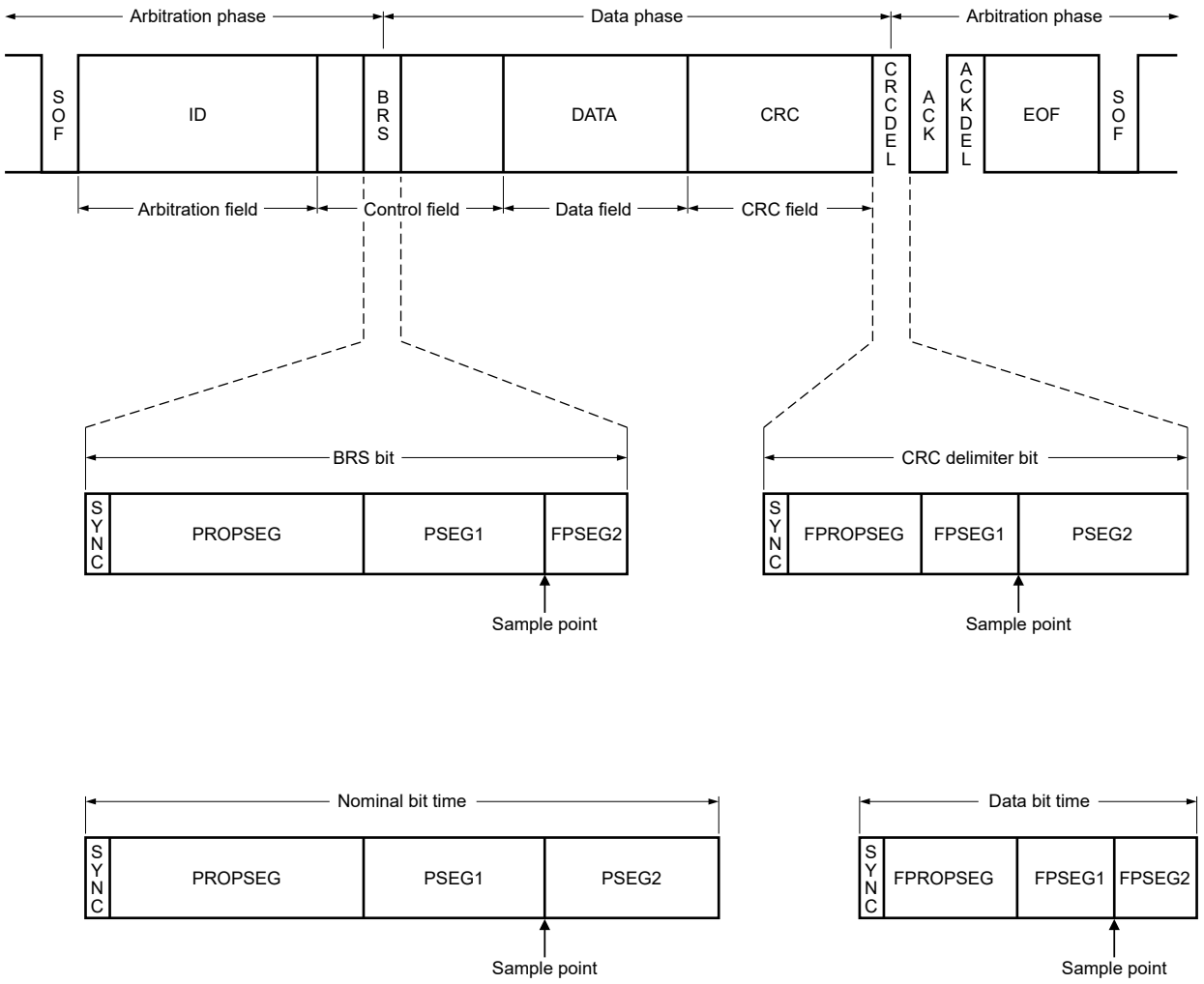


Figure 152. Bit rate switching mechanism for CAN FD messages

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.

37.2.12.2.3 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 237.

Table 237. 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)

37.2.12.2.4 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.

37.2.12.2.5 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.

37.2.12.2.6 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.

37.2.12.3 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 153). 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.

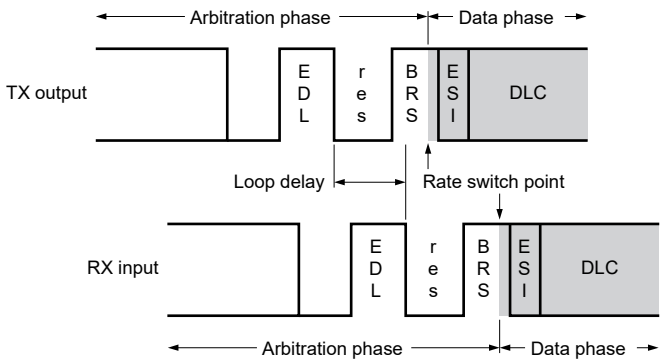


Figure 153. Transceiver loop delay measurement

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:

$$Offset = (FPSEG1 + FPROPSEG + 2) \times (FPRESDIV + 1)$$
$$or$$
$$Offset = (DTSEG1 + 2) \times (EDPRESDIV + 1), \; if \; ETDCEN$$

Equation 2. TDC offset calculation

Figure 154 shows the SSP position when these settings are used.

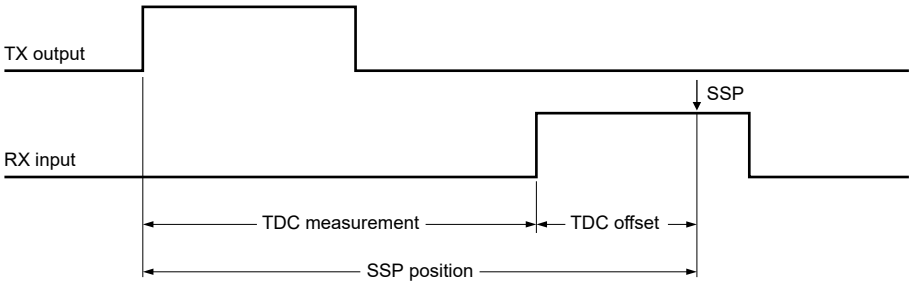


Figure 154. SSP position with optimal values

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 155 shows the secondary sample point position when the transceiver delay measurement is disabled.

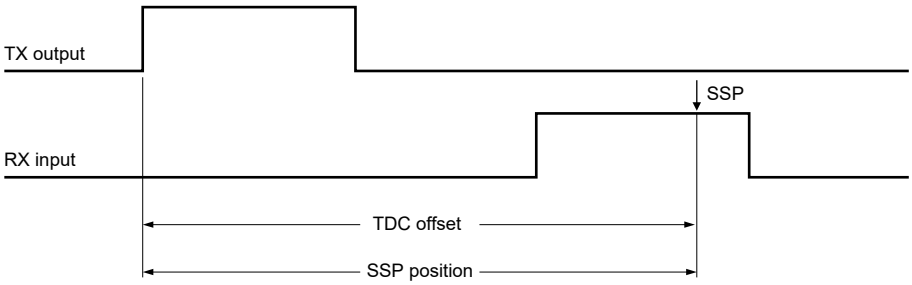


Figure 155. SSP position with TDC measurement disabled by ETDCEN and TDMDIS

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.

37.2.12.4 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.

37.2.12.5 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.

37.2.12.6 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].

37.2.12.7 Protocol timing

Figure 156 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).

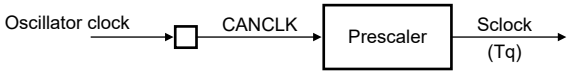


Figure 156. CAN engine clocking scheme

37.2.12.7.1 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 3.) 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 3. 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 4. CAN bit time and baud rate

37.2.12.7.2 Bit time segments

A bit time is subdivided into three segments as shown in Figure 157. See also Figure 158 , Figure 159, and Table 238.

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.

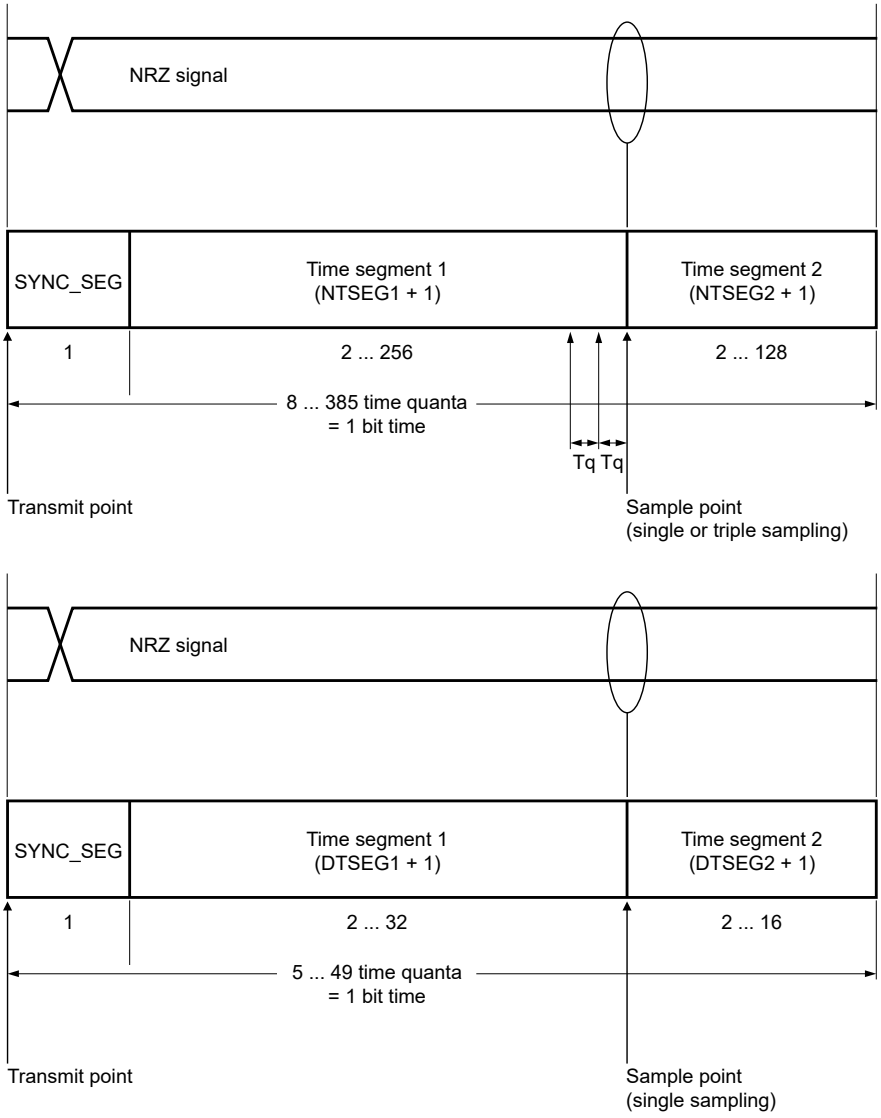


Figure 157. Segments within the bit timing (example using ENCBT and EDCBT bit timing variables)

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.

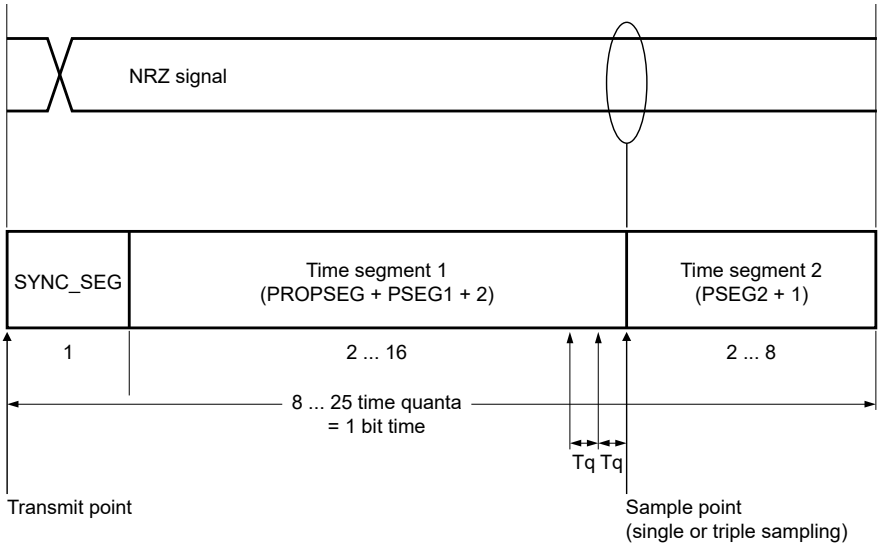


Figure 158. Segments within the bit time (example using CTRL1 bit timing variables for Classical CAN format)

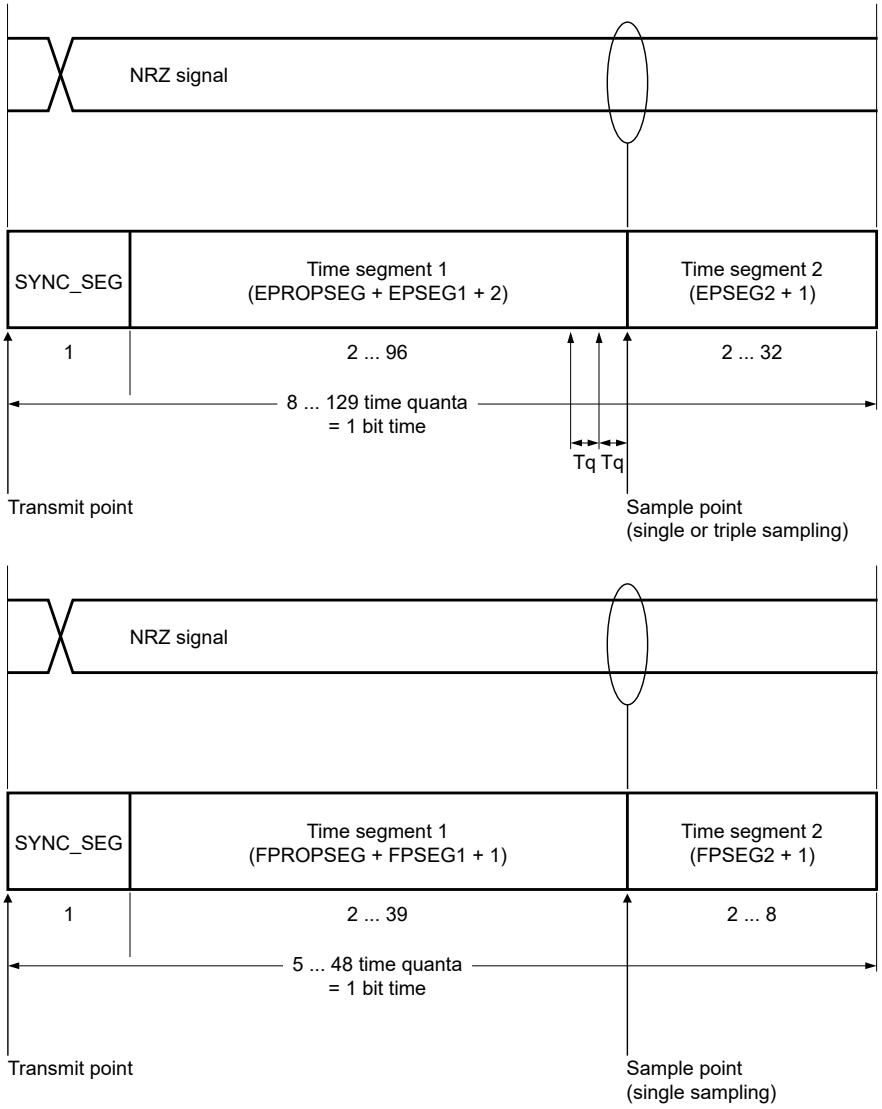


Figure 159. Segments within the bit time (example using CBT and FDCBT bit timing variables for CAN FD format)

Table 238. 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 239 gives some examples of the CAN-compliant segment settings for Classical CAN format (Bosch CAN 2.0B) (non-FD) messages.

Table 239. 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).

37.2.12.7.3 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 5. 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 6. 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 156), 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.

37.2.12.8 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.

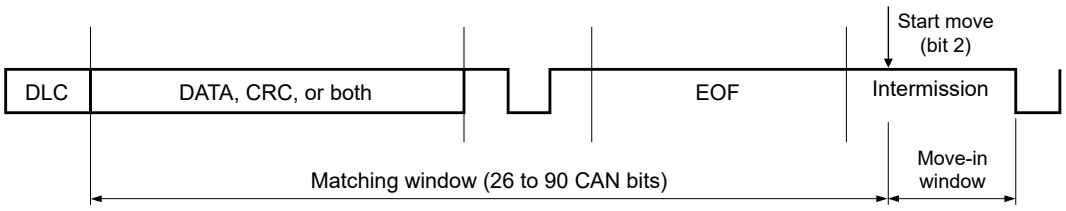


Figure 160. Matching and move-in time windows

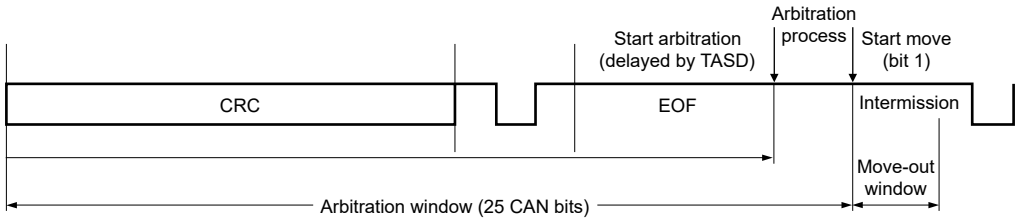


Figure 161. Arbitration and move-out time windows

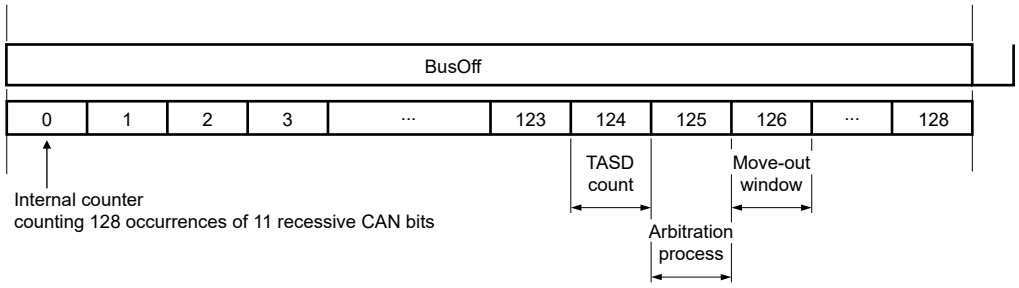


Figure 162. Arbitration at the end of bus off and move-out time windows

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.

37.2.12.9 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 163, 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

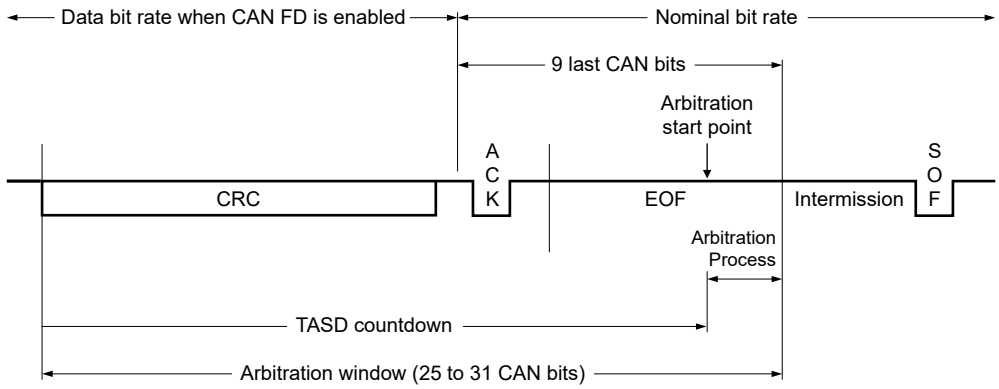


Figure 163. Optimal TX arbitration start point

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 7. 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 8. 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 8.)
- $BITRATE_N$ 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 9. Nominal baud rate when CTRL2[BTE] = 1

$$BITRATE_F = \frac{f_{CANCLK}}{[1 + (DTSEG1 + 1) + (DTSEG2 + 1)] \times (EDPRESDIV + 1)}$$

Equation 10. Fast baud rate when CTRL2[BTE] = 1

See also Protocol timing for more details.

37.2.12.9.1 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 240. 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 241. 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 242. 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

37.2.13 Clocks

The following table describes the clock sources for FlexCAN. See the chip clocking chapter for clock setting, configuration, and gating information.

Table 243. 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

37.2.13.1 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 244.

Table 244. 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 244. 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 244 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.

37.2.13.1.1 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 11. 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 245. 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 245. 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 245. 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 12. 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 13. 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 14. 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 15. 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 16. Calculation example for number of peripheral clocks per nominal CAN bit

2. The number of fast CAN bits (NumOfFastBits) is determined in Table 245. 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 17. 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 18. 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 19. 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 20. 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 246 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 246. 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

37.2.14 Reset

You can reset FlexCAN in the following ways:

- Chip-level hard reset, which resets all memory-mapped registers asynchronously.
- Soft reset:
  - MCR[SOFTRST], which resets some of the memory-mapped registers synchronously.
  - The chip level soft reset input is not supported, use the MCR[SOFTRST] to trigger a soft reset.

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.

37.2.15 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 7 of Interrupt Flags 1 (IFLAG1) becomes the Legacy FIFO Overflow flag
- Bit 6 becomes the Legacy FIFO Warning flag
- Bit 5 becomes the Frames Available in Legacy FIFO flag
- Bits 4–0 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 5 of IFLAG1 still indicates Frames Available in Legacy FIFO and generates a DMA request. Bits 7, 6, and 4–0 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).

37.2.16 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 247. 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

Table continues on the next page...

Table 247. Access permissions (continued)

Modes of operation	Normal	Freeze	Low-power
RXMGMASK[1]	Bus error for write operation	Read and write	Bus error for write operation
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]

Table continues on the next page...

Table 247. Access permissions (continued)

Modes of operation	Normal	Freeze	Low-power
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]
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.

37.3 External signal descriptions

FlexCAN has two I/O signals connected to the external chip pins. These signals are summarized in Table 248 and described in the following subsections.

Table 248. FlexCAN signal descriptions

Signal	Description	I/O
CAN RX	CAN receive pin	Input
CAN TX	CAN transmit pin	Output

37.3.1 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.

37.3.2 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.

37.4 Initialization and application information

37.4.1 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.

37.5 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.

37.5.1 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 249. Register reset information Statement

Register	Affected by hard reset	Affected by soft reset
Module Configuration (MCR)	Yes	Yes
Control 1 (CTRL1)	Yes	No
Free-Running Timer (TIMER)	Yes	Yes
RX Message buffers Global Mask (RXMGMASK)	No	No
RX Buffer 14 Mask (RX14MASK)	No	No

Table continues on the next page...

Table 249. Register reset information Statement (continued)

Register	Affected by hard reset	Affected by soft reset
RX Buffer 15 Mask (RX15MASK)	No	No
Error Counter (ECR)	Yes	Yes
Error and Status 1 (ESR1)	Yes	Yes
Interrupt Masks 1 (IMASK1)	Yes	Yes
Interrupt Flags 1 (IFLAG1)	Yes	Yes
Control 2 (CTRL2)	Yes	No
Error and Status 2 (ESR2)	Yes	Yes
Cyclic Redundancy Check (CRCR)	Yes	Yes
RX FIFO Global Mask (RXFGMASK)	No	No
RX FIFO Information (RXFIR)	No	No
CAN Bit Timing (CBT)	Yes	No
Message buffers	No	No
RX Individual Masks	No	No
Pretended Networking Control 1 (CTRL1_PN)	Yes	Yes
Pretended Networking Control 2 (CTRL2_PN)	Yes	Yes
Pretended Networking Wake-Up Match (WU_MTC)	Yes	Yes
Pretended Networking ID Filter 1 (FLT_ID1)	Yes	Yes
Pretended Networking DLC Filter (FLT_DLC)	Yes	Yes
Pretended Networking Payload Low Filter 1 (PL1_LO)	Yes	Yes
Pretended Networking Payload High Filter 1 (PL1_HI)	Yes	Yes
Pretended Networking ID Filter 2 or ID Mask (FLT_ID2_IDMASK)	Yes	Yes
Pretended Networking Payload Low Filter 2 or Payload Low Mask (PL2_PLMASK_LO)	Yes	Yes
Pretended Networking Payload High Filter 2 or Payload High Mask (PL2_PLMASK_HI)	Yes	Yes
Pretended Networking Wake Up Message Buffer 0 (WMB0)	Yes	No
Pretended Networking Wake Up Message Buffer 1 (WMB1)	Yes	No

Table continues on the next page...

Table 249. Register reset information Statement (continued)

Register	Affected by hard reset	Affected by soft reset
Pretended Networking Wake-Up Message Buffer 2 (WMB2)	Yes	No
Pretended Networking Wake-Up Message Buffer 3 (WMB3)	Yes	No
Enhanced CAN Bit Timing Prescalers (EPRS)	Yes	No
Enhanced Nominal CAN Bit Timing (ENCBT)	Yes	No
Enhanced Data Phase CAN bit Timing (EDCBT)	Yes	No
Enhanced Transceiver Delay Compensation (ETDC)	Yes	No
CAN FD Control (FDCTRL)	Yes	No
CAN FD Bit Timing (FDCBT)	Yes	No
CAN FD CRC (FDCRC)	Yes	Yes
Enhanced RX FIFO Control (ERFCR)	Yes	Yes
Enhanced RX FIFO Interrupt Enable (ERFIER)	Yes	Yes
Enhanced RX FIFO Status (ERFSR)	Yes	Yes
Enhanced RX FIFO	No	No
Enhanced RX FIFO Filter Element (ERFFEL)	No	No

FlexCAN can store CAN messages for transmission and reception using message buffers and RX FIFO structures.

37.5.2 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.

37.5.2.1 CAN memory map

CAN0 base address: 400C_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration (MCR)	32	RW	D890_000Fh
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_0000h
38h	Error and Status 2 (ESR2)	32	R	0000_0000h
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
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
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

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
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

37.5.2.2 Module Configuration (MCR)

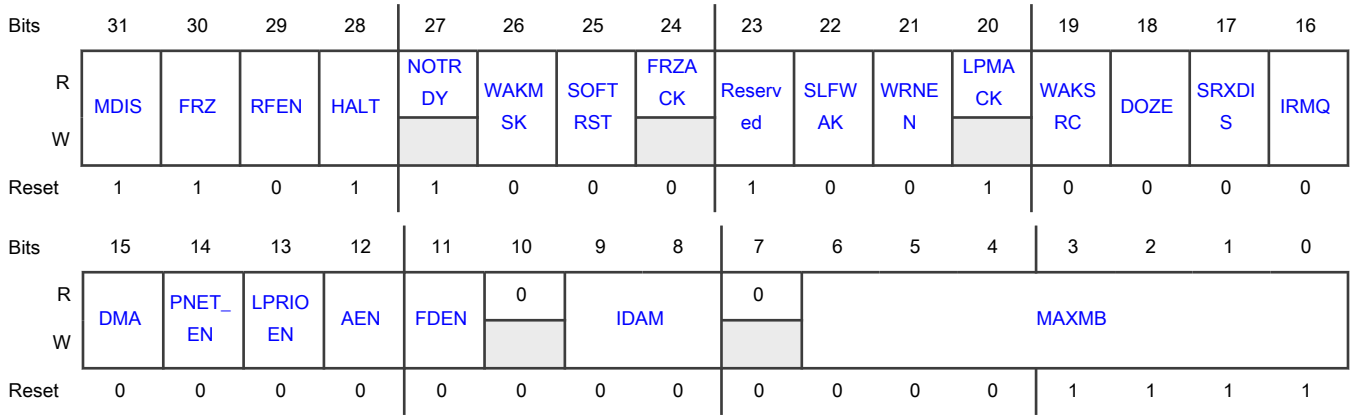
Offset

Register	Offset
MCR	0h

Function

Defines global system configurations, including the module operation modes and the maximum message buffer configuration.

Diagram



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. 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 244. 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. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
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 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.

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 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 —	Reserved
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 —	Reserved
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. 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 244.

37.5.2.3 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	BOFF MSK	ERRM SK	Reserv ed	LPB	TWRN MSK	RWRN MSK	0	0	SMP	BOFF REC	TSYN	LBUF	LOM	PROPSEG		
W																
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. PhaseBuffer 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 LPB	Loopback Mode 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 —	Reserved
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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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
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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.4 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.

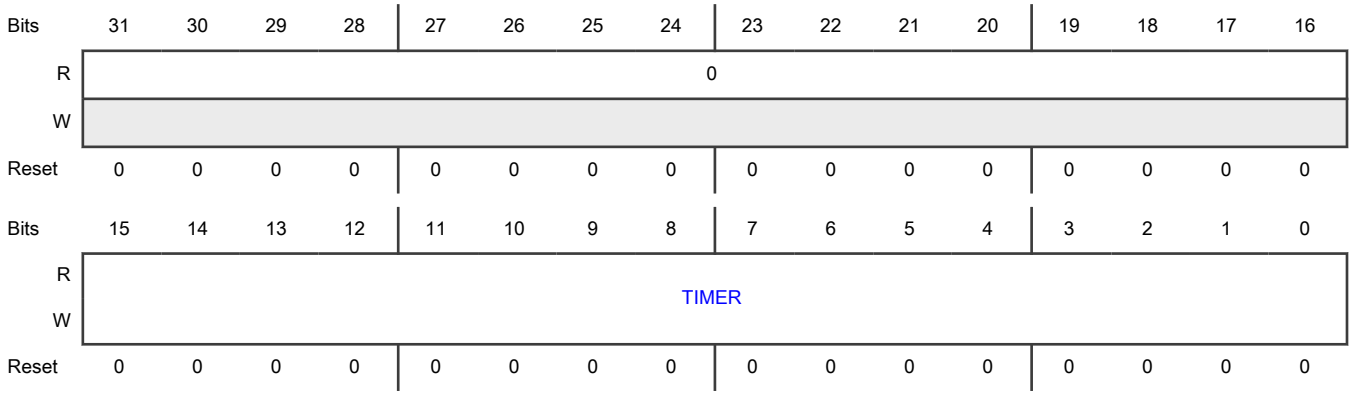
The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the timestamp entry in a message buffer after a successful reception or transmission of a message.

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



Fields

Field	Function
31-16 —	Reserved
15-0 TIMER	Timer Value Contains the free-running counter value.

37.5.2.5 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.

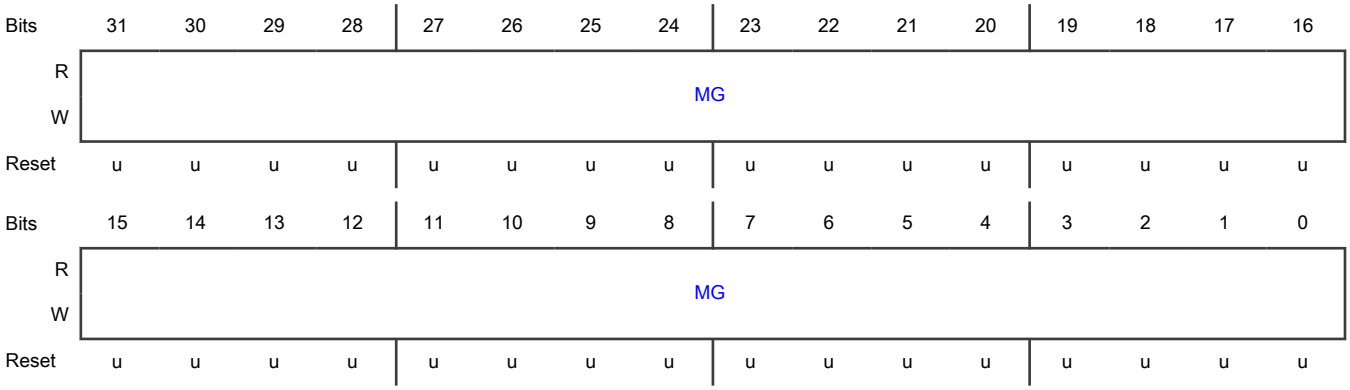
This register is located in RAM.

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.

37.5.2.6 Receive 14 Mask (RX14MASK)

Offset

Register	Offset
RX14MASK	14h

Function

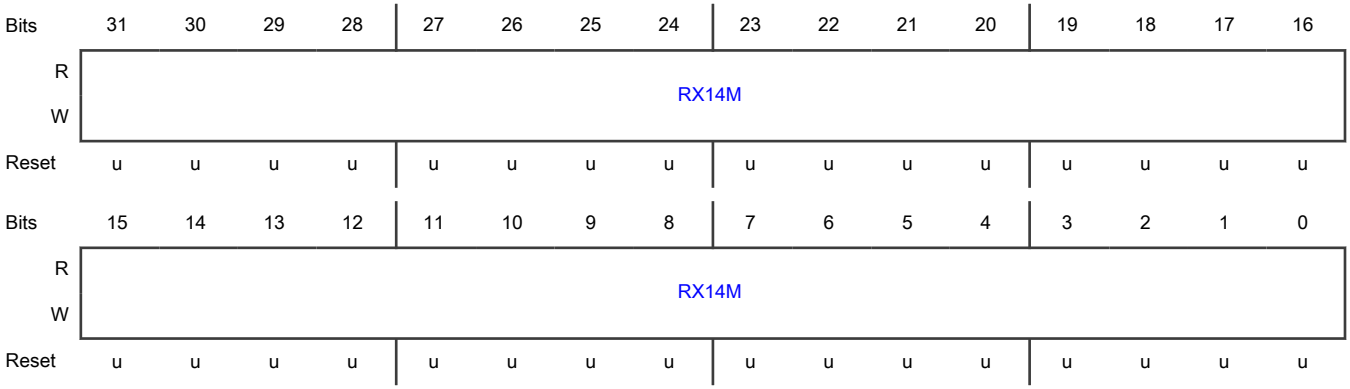
Masks the filter fields of MB14.

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.

37.5.2.7 Receive 15 Mask (RX15MASK)

Offset

Register	Offset
RX15MASK	18h

Function

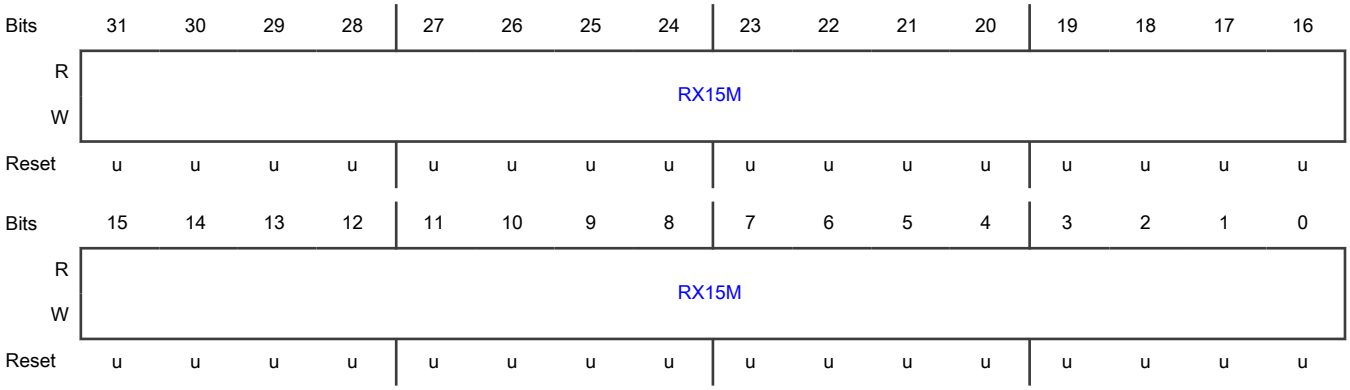
Masks the filter fields of MB15.

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.

37.5.2.8 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

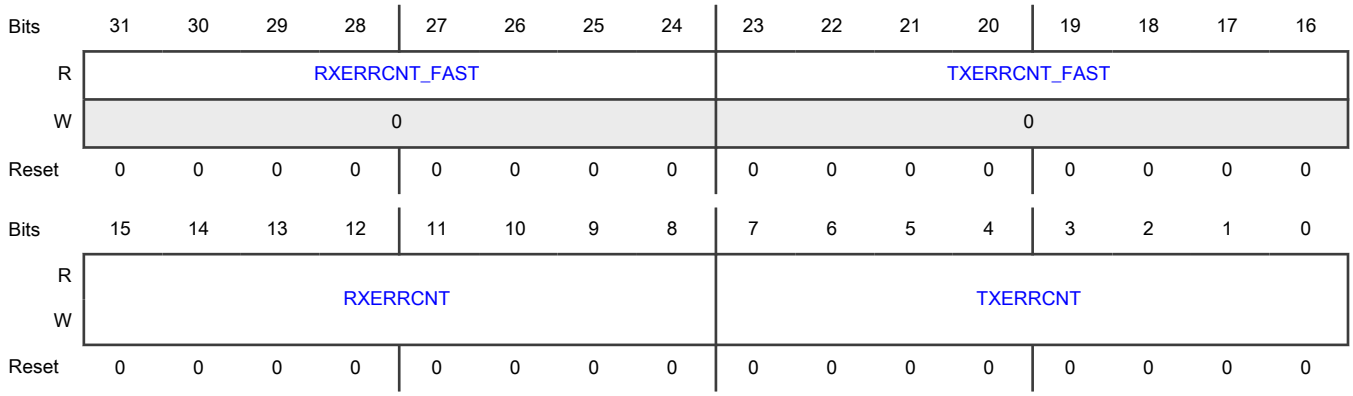
counts 11 such bits and then wraps around when incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, ESR1[FLTCONF] is updated to Error Active, and both error counters are reset to zero. Upon any instance of a dominant bit following a stream of less than 11 consecutive recessive bits, the 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.

37.5.2.9 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 250. 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		0	0	ERROVR	ERRINT_...	BOFFDON...	SYNCH	TWRNINT	RWRNINT
W											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 —	Reserved
22 —	Reserved
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.
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	CAN Synchronization Status Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNCH	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. 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
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. 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.

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.
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 250. 0b - Not IDLE 1b - IDLE
6 TX	FlexCAN In Transmission Indicates whether FlexCAN is transmitting a message. See Table 250. 0b - Not transmitting 1b - Transmitting
5-4 FLTCONF	Fault Confinement State Indicates the confinement state of FlexCAN. 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
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.

37.5.2.10 Interrupt Masks 1 (IMASK1)

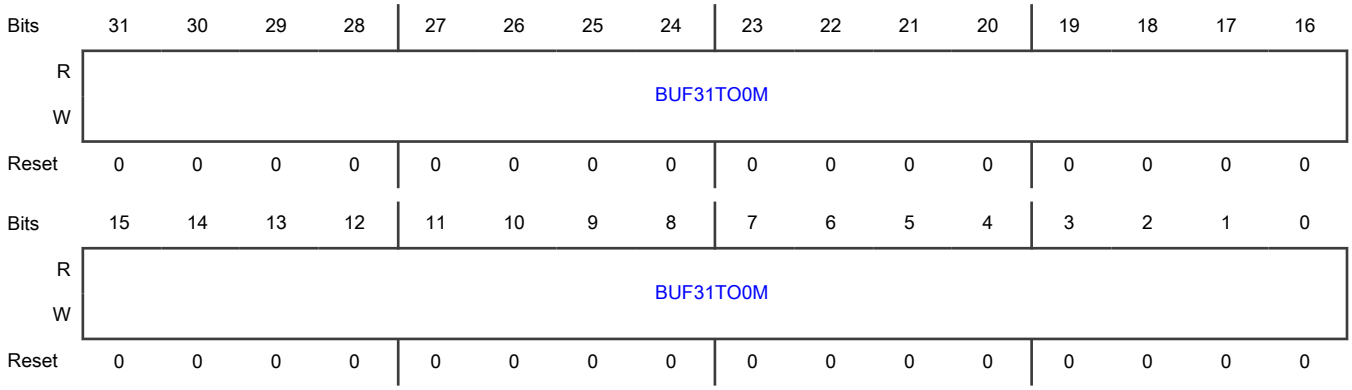
Offset

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.

37.5.2.11 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. 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
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). **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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.12 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.

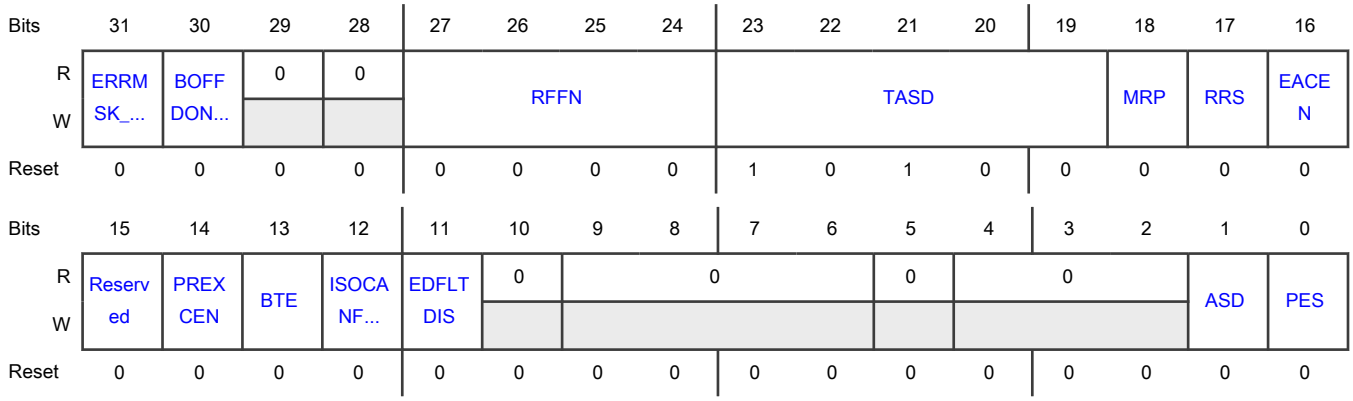
Soft reset does not affect the contents of this register.

Table 251 shows how the value of CTRL2[RFFN] determines the Legacy RX FIFO filter structure.

Table 251. 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
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 BOFFDONEMSK	Bus Off Done Interrupt Mask Enables the Bus Off Done interrupt, ESR1[BOFFDONEINT]. 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 251. 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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
10 —	Reserved
9-6 —	Reserved
5 —	Reserved
4-2 —	Reserved
1 ASD	ACK Suppression Disable Disables the ACK suppression logic. You can only write to this field in Freeze mode. 0b - Enabled 1b - Disabled
0 PES	Payload Byte and Bit Order Selection Selects the byte order for the payload of transmit and receive frames.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You can only write to this field in Freeze mode. 0b - Big-endian 1b - Little-endian

37.5.2.13 Error and Status 2 (ESR2)

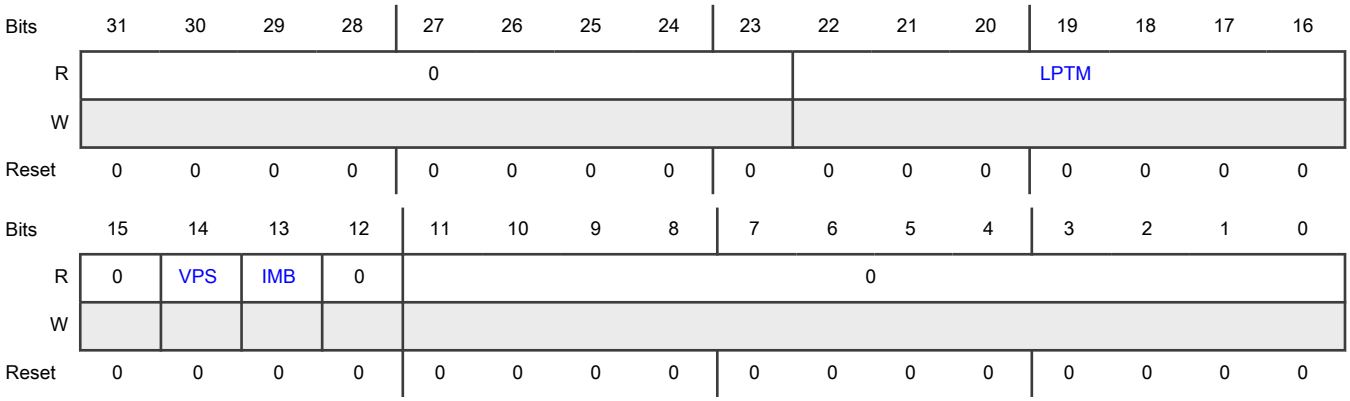
Offset

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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
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 —	Reserved
11-0 —	Reserved

37.5.2.14 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

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

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).

37.5.2.15 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.

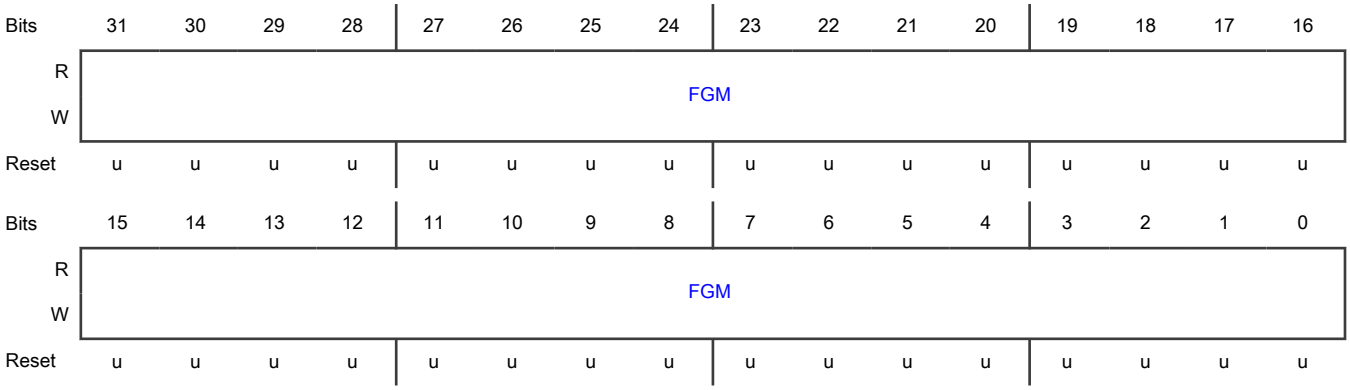
The following table shows how the FGM bits correspond to each IDAF field.

Table 252. 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.

37.5.2.16 Legacy RX FIFO Information (RXFIR)

Offset

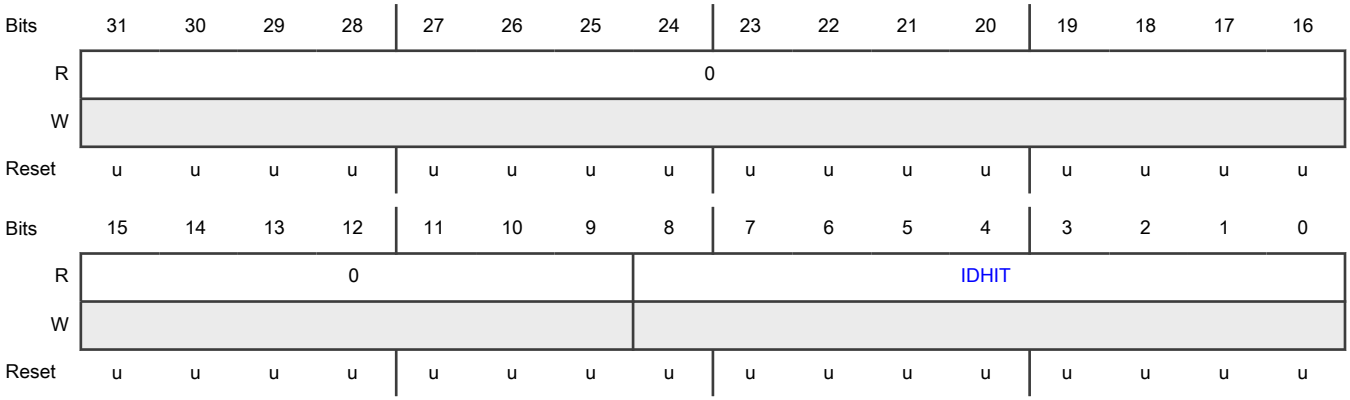
Register	Offset
RXFIR	4Ch

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.17 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.

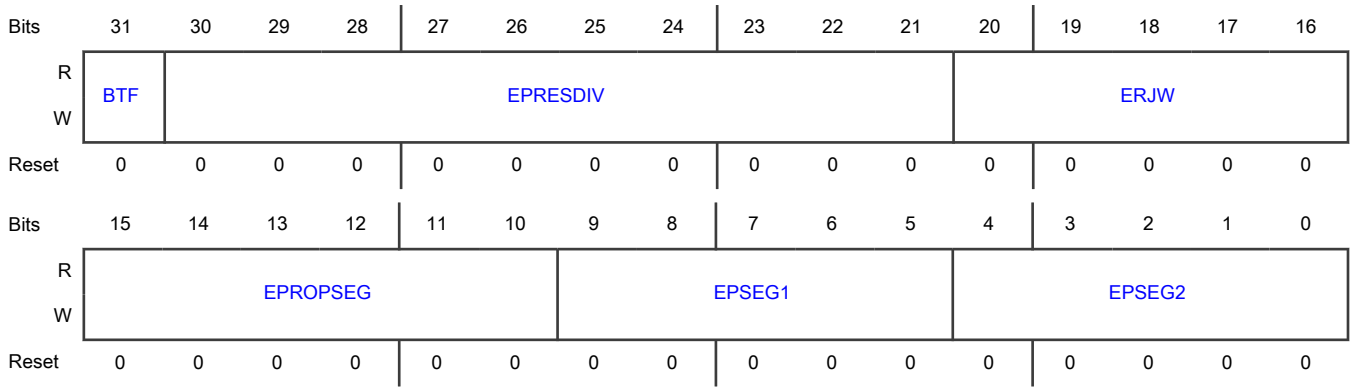
NOTE

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. 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	One Time Quantum = one Sclock period.

37.5.2.18 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.

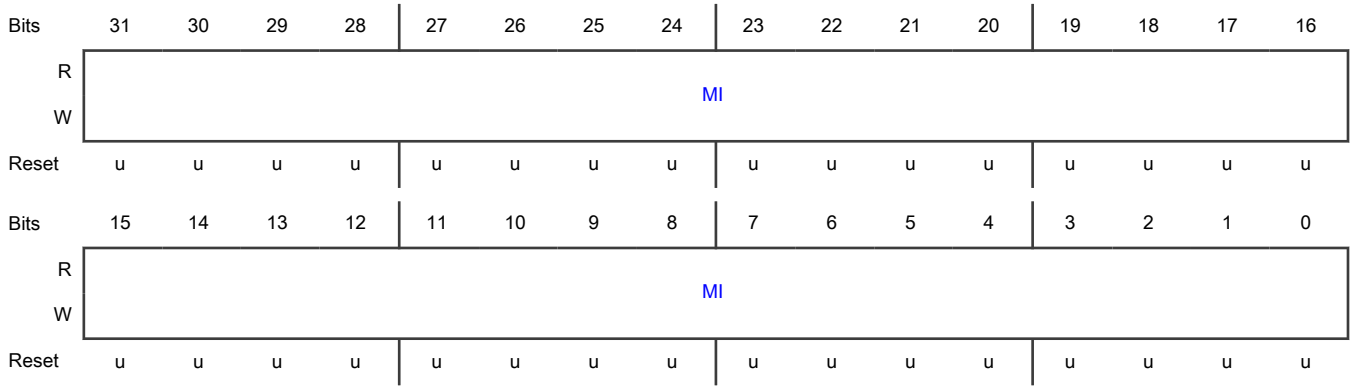
When the Legacy RX FIFO is disabled (MCR[RFEN] = 0), an individual mask is provided for each available RX message buffer on a one-to-one correspondence. When the Legacy RX FIFO is enabled (MCR[RFEN] = 1), an individual mask is provided for each Legacy RX FIFO ID filter table element on a one-to-one correspondence. 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	Individual Mask Bits

Table continues on the next page...

Field	Function
MI	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.

37.5.2.19 Pretended Networking Control 1 (CTRL1_PN)

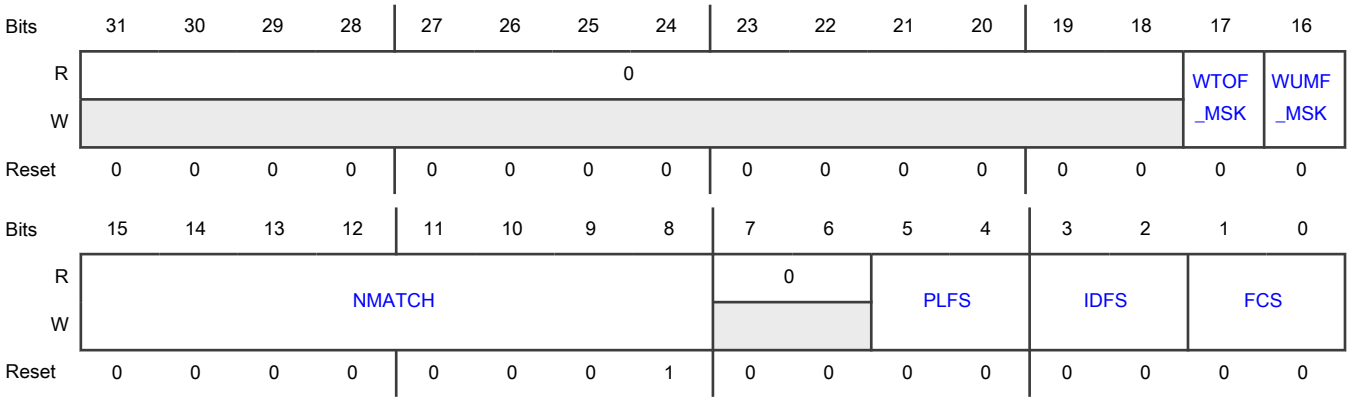
Offset

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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). 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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

37.5.2.20 Pretended Networking Control 2 (CTRL2_PN)

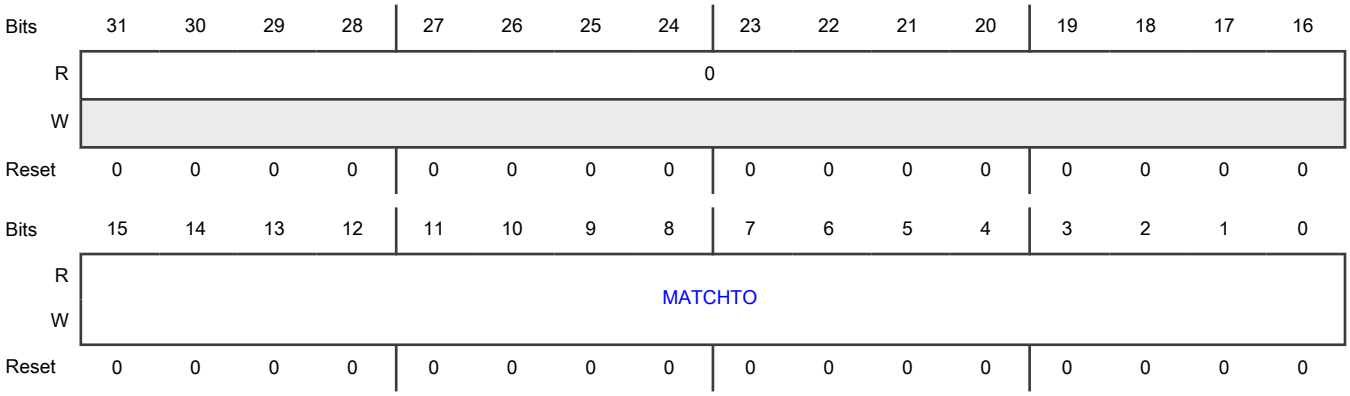
Offset

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.

37.5.2.21 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

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														WTOF	WUMF
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	MCOUNTER								0							
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 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	Number of Matches in Pretended Networking

Table continues on the next page...

Table continued from the previous page...

Field	Function
MCOUNTER	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

37.5.2.22 Pretended Networking ID Filter 1 (FLT_ID1)

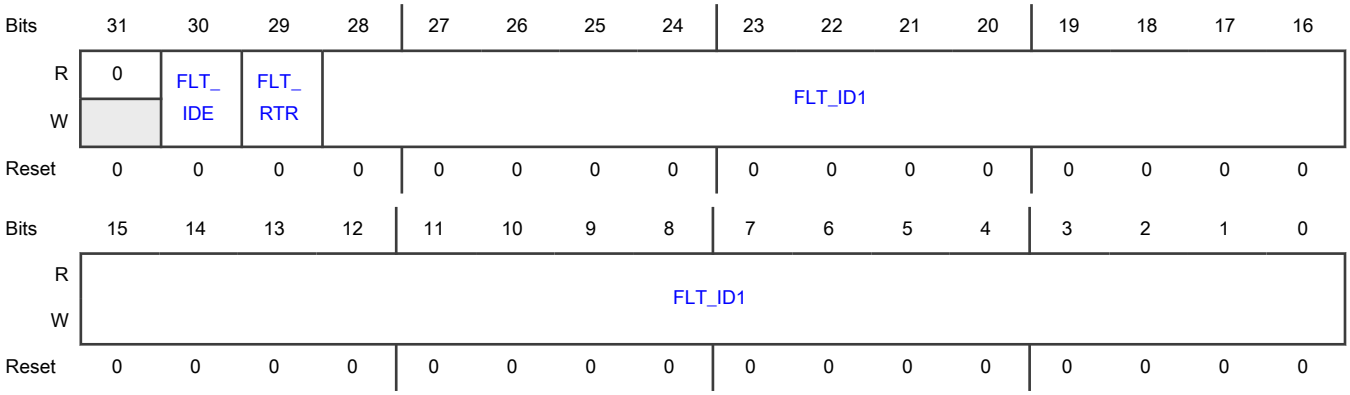
Offset

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



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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.23 Pretended Networking Data Length Code (DLC) Filter (FLT_DLC)

Offset

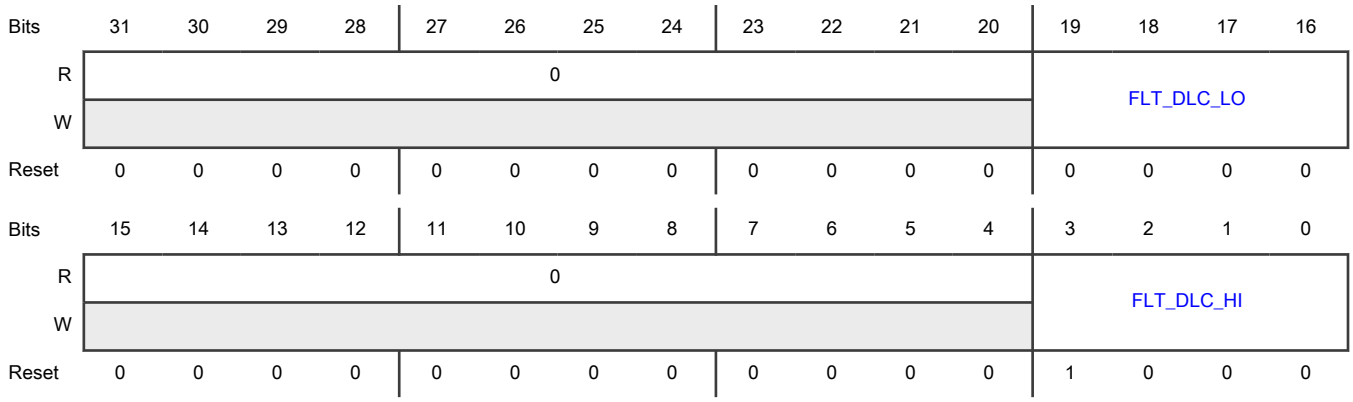
Register	Offset
FLT_DLC	B10h

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



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.

37.5.2.24 Pretended Networking Payload Low Filter 1 (PL1_LO)

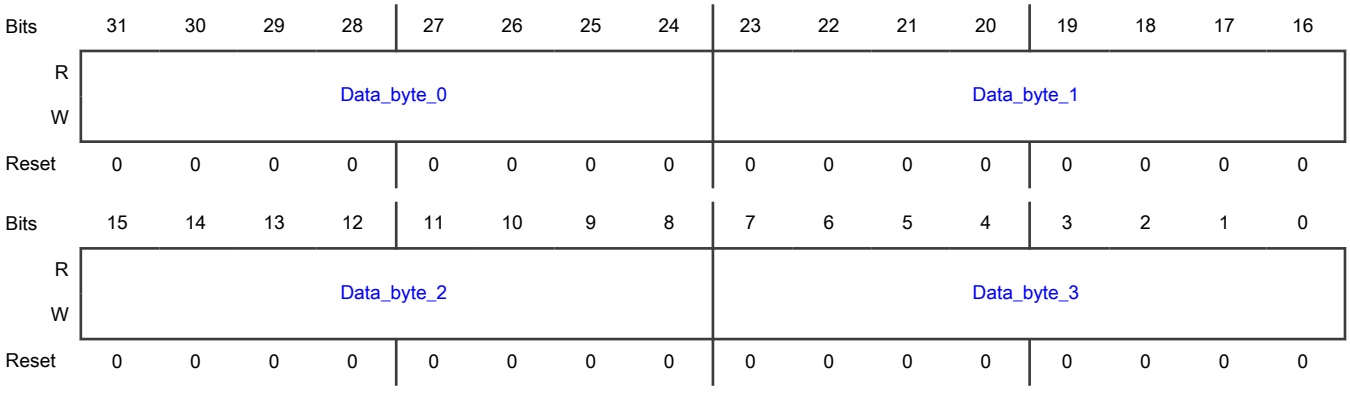
Offset

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



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.

37.5.2.25 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
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.

37.5.2.26 Pretended Networking ID Filter 2 or ID Mask (FLT_ID2_IDMASK)

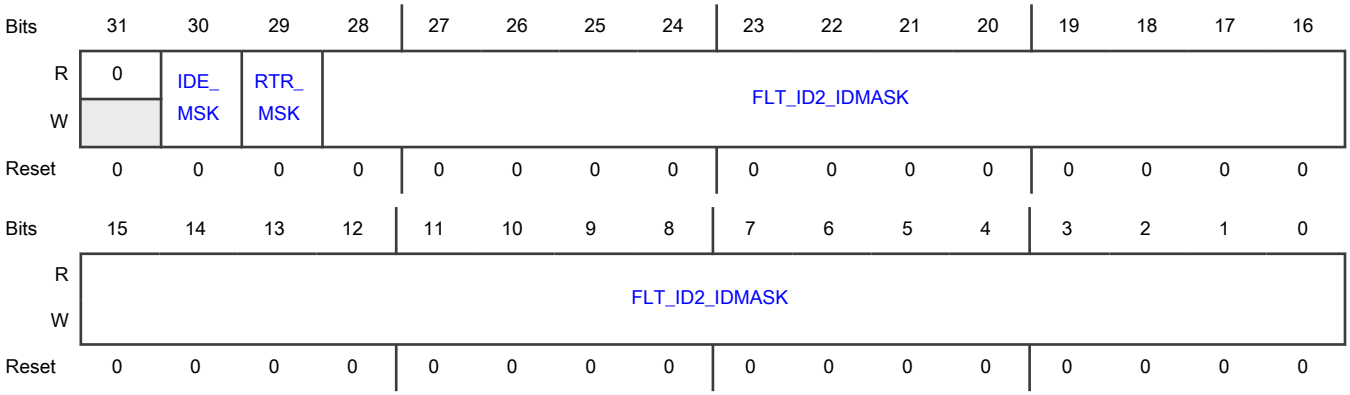
Offset

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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.

37.5.2.27 Pretended Networking Payload Low Filter 2 and Payload Low Mask (PL2_PLMASK_LO)

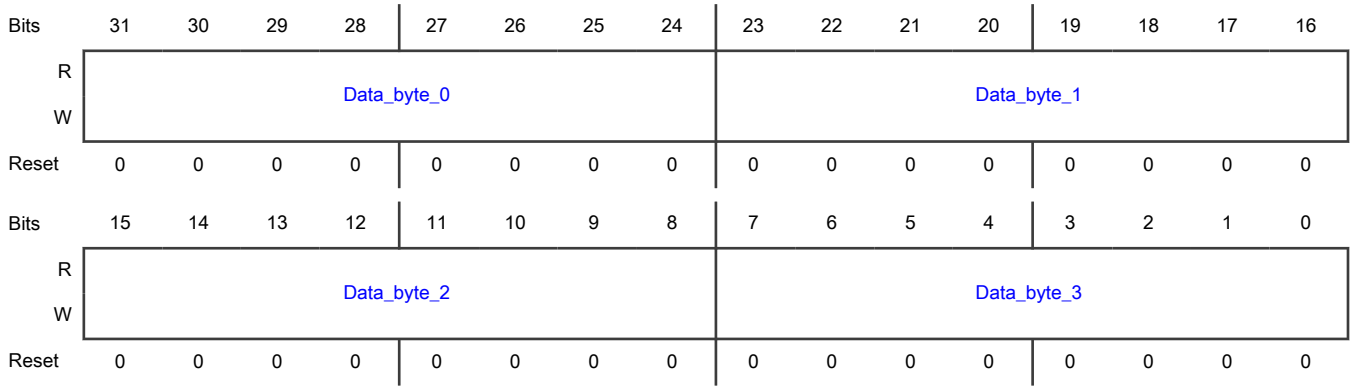
Offset

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

37.5.2.28 Pretended Networking Payload High Filter 2 and Payload High Mask (PL2_PLMASK_HI)

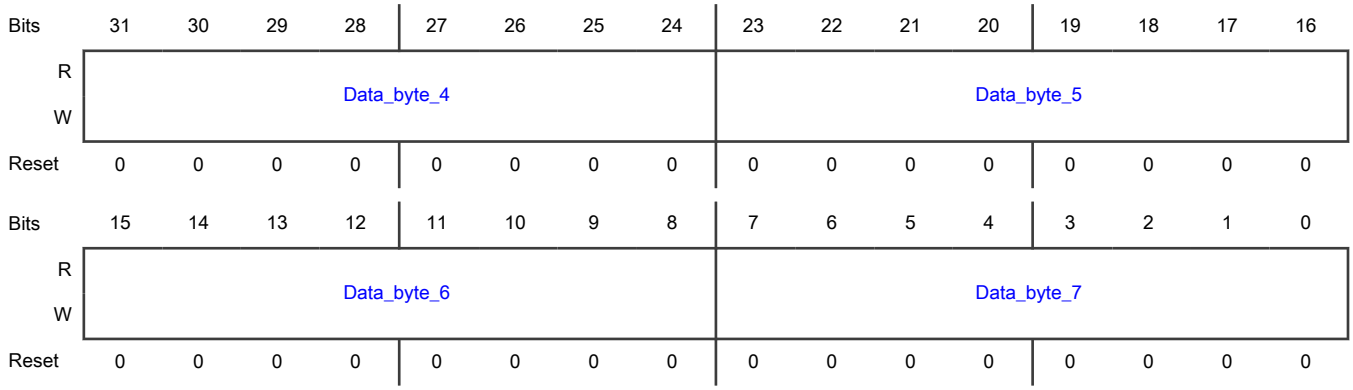
Offset

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



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.

37.5.2.29 Wake-Up Message Buffer (WMB0_CS - WMB3_CS)

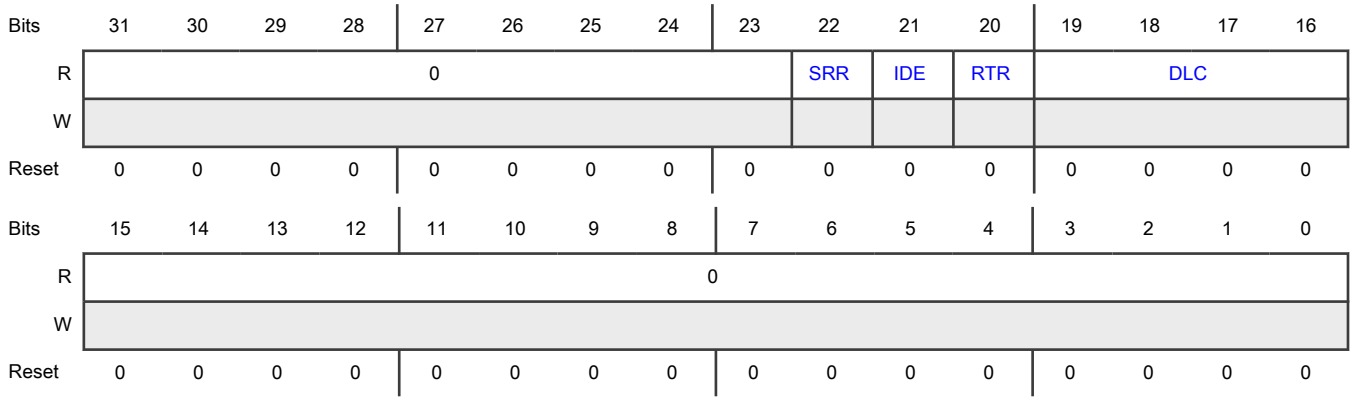
Offset

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. 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

37.5.2.30 Wake-Up Message Buffer for ID (WMB0_ID - WMB3_ID)

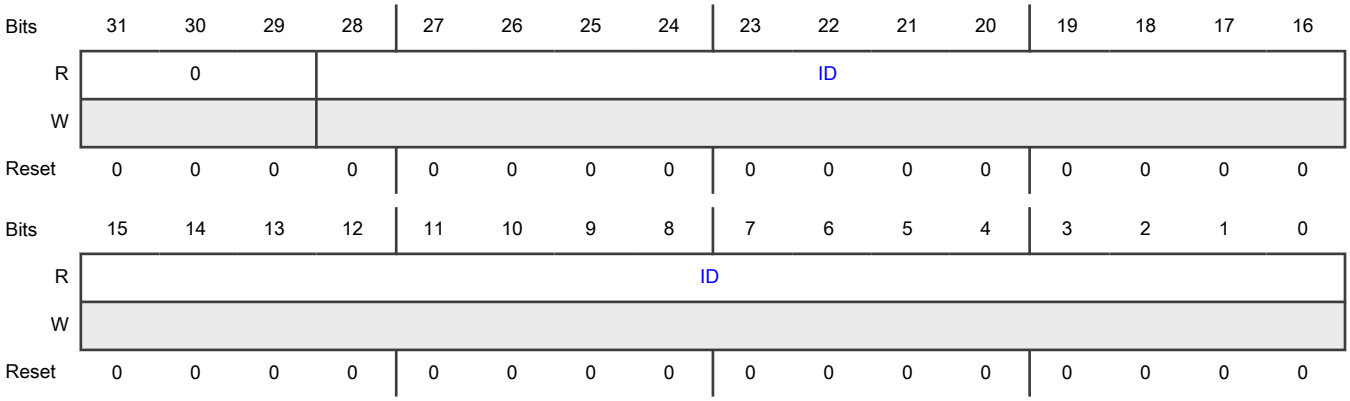
Offset

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 ID[28:0]) • The 11 bits of the standard frame format (considering ID[28:18] only; the remaining bits in the ID[17:0] range are meaningless).

37.5.2.31 Wake-Up Message Buffer for Data 0–3 (WMB0_D03 - WMB3_D03)

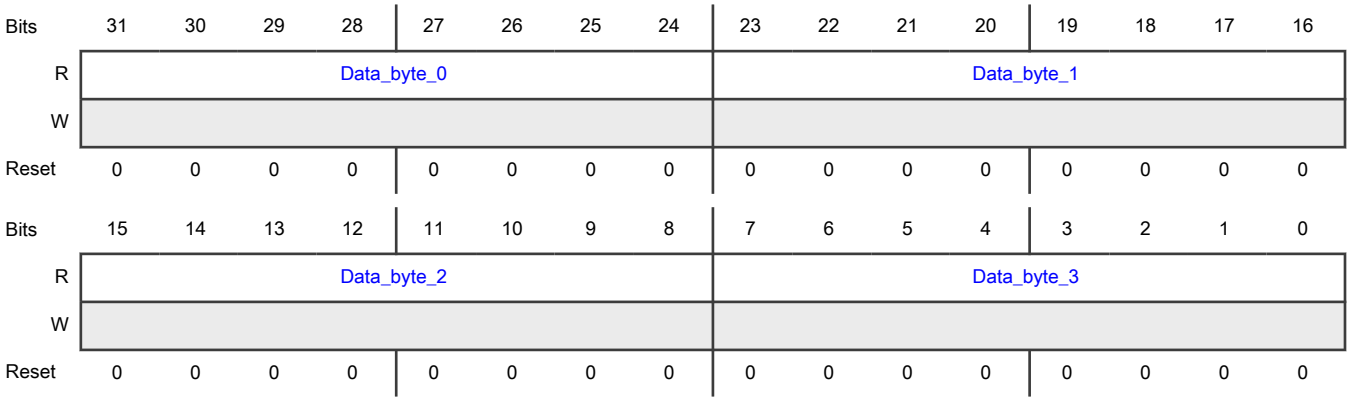
Offset

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



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

37.5.2.32 Wake-Up Message Buffer Register Data 4–7 (WMB0_D47 - WMB3_D47)

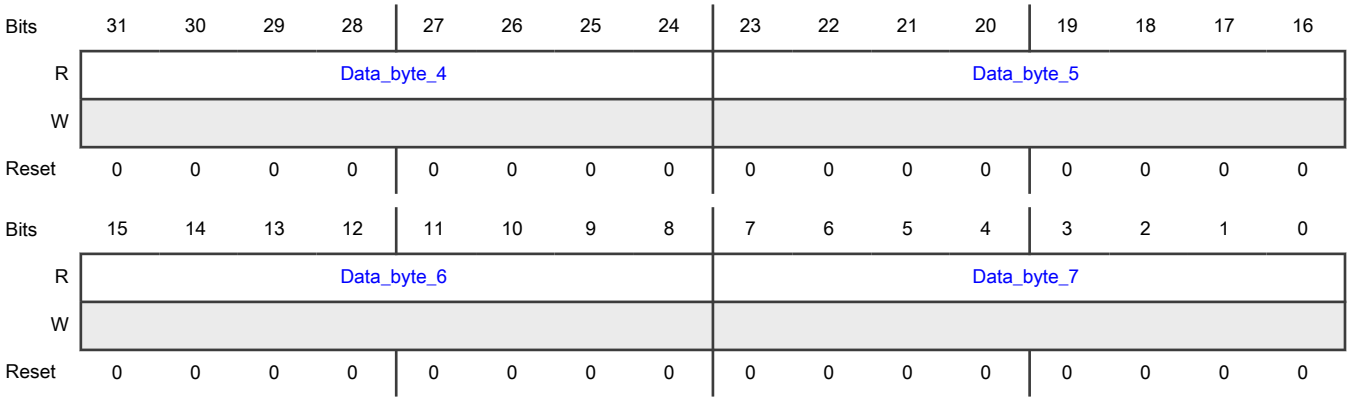
Offset

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



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

37.5.2.33 Enhanced CAN Bit Timing Prescalers (EPRS)

Offset

Register	Offset
EPRS	BF0h

Function

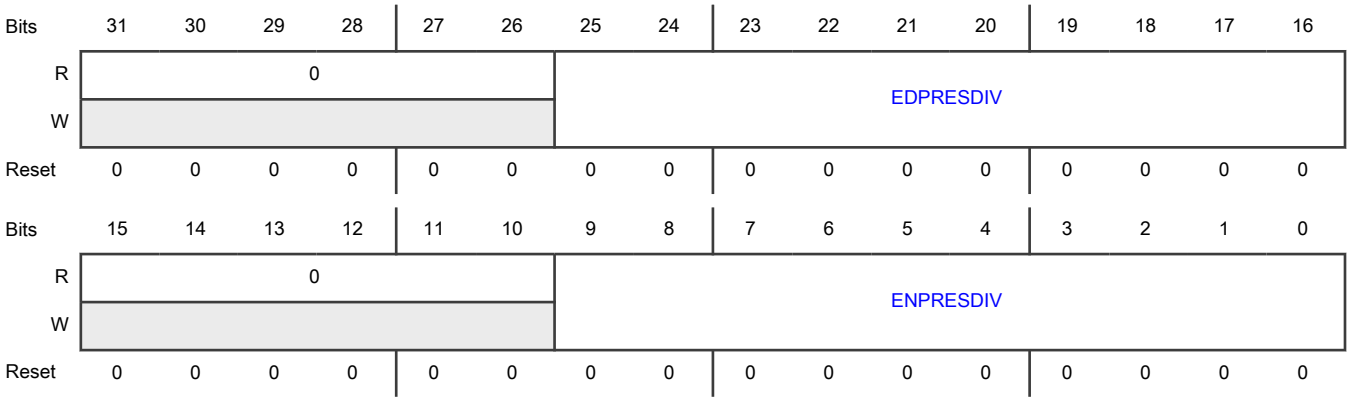
Defines the CAN bit timing prescalers for the nominal phase and data phase when CTRL2[BTE] = 1.

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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)

37.5.2.34 Enhanced Nominal CAN Bit Timing (ENCBT)

Offset

Register	Offset
ENCBT	BF4h

Function

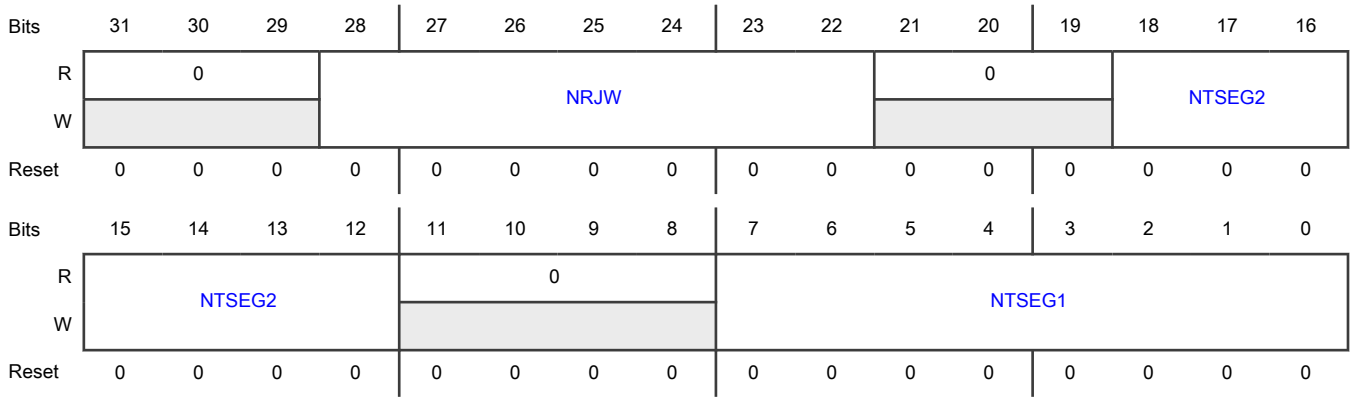
Provides an alternative way to store the CAN bit timing variables described in Control 1 (CTRL1) and CAN Bit Timing (CBT), to get higher CAN bit timing resolution.

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

37.5.2.35 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.

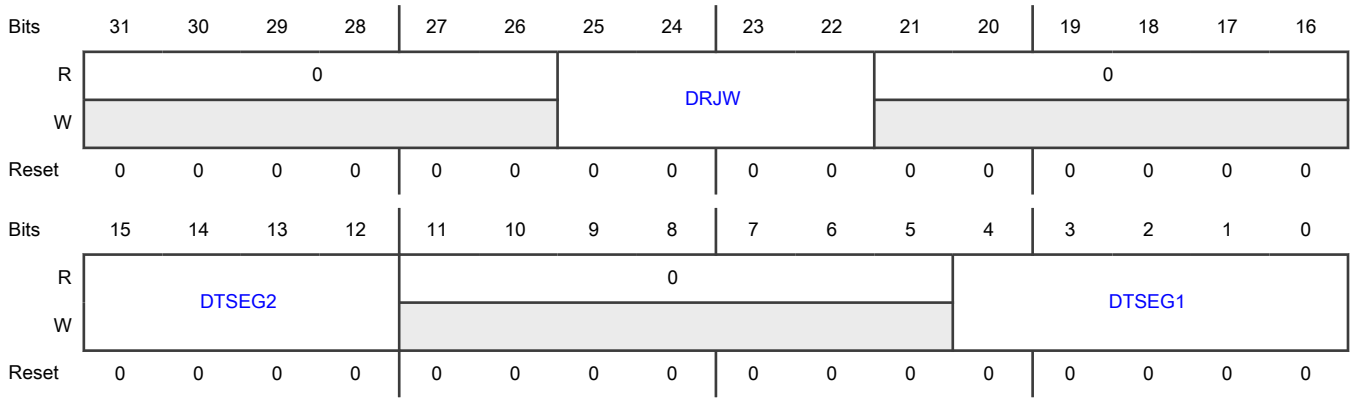
NOTE

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.
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.36 Enhanced Transceiver Delay Compensation (ETDC)

Offset

Register	Offset
ETDC	BFCh

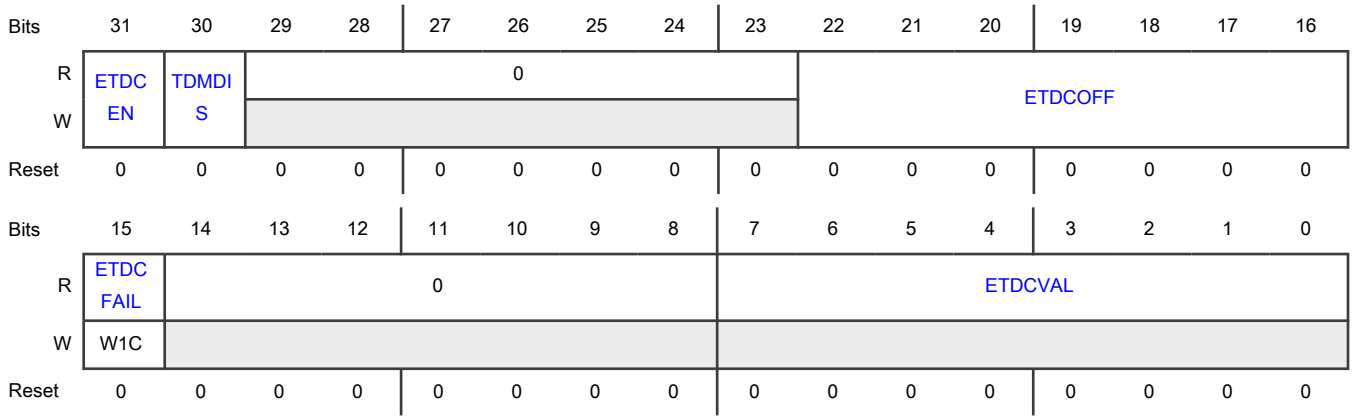
Function

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 ETDCOFF	Enhanced 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 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.37 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 253.

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.

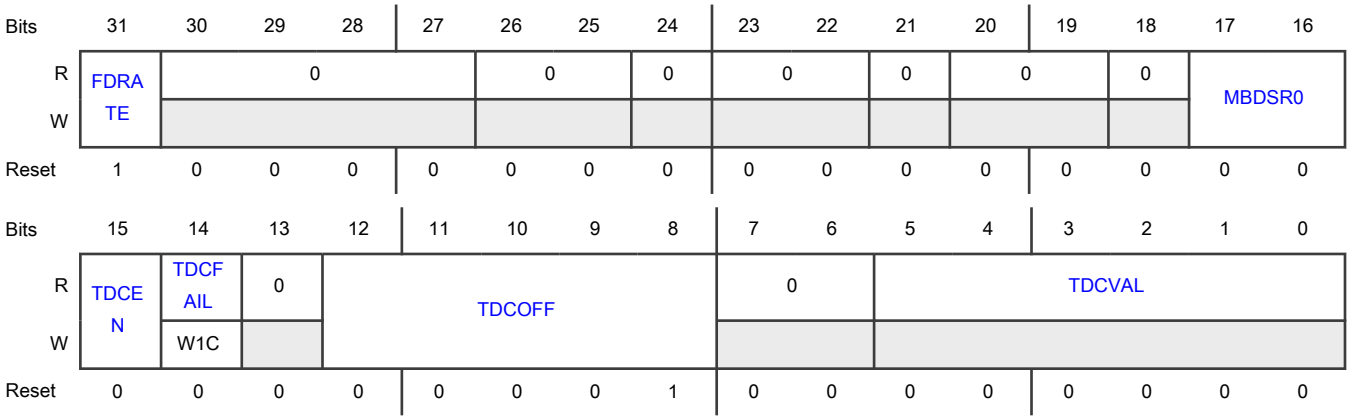
Table 253. Number of message buffers

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. • 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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]). **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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.

37.5.2.38 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.

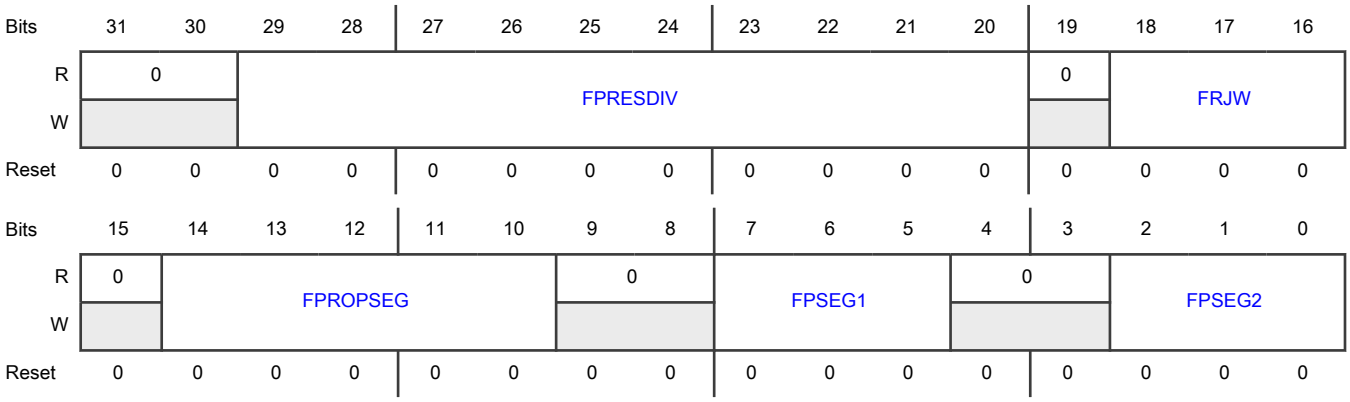
The sum of the Fast Propagation Segment (FPROPSEG) and Fast Phase Segment 1 (FPSEG1) must be at least two time quanta.

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. 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. Phase Segment 2 = (FPSEG2 + 1) × Time Quanta. One Time Quantum = one Sclock period.

37.5.2.39 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.

$$CRC\_15 = C599h: \ (x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1)$$

$$CRC\_17 = 3685Bh: \ (x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^{6} + x^{4} + x^{3} + x^{1} + 1)$$

$$CRC\_21 = 302899h: \ (x^{21} + x^{20} + x^{13} + x^{11} + x^{7} + x^{4} + x^{3} + 1)$$

Equation 21. CRC polynomial used on CAN frame

NOTE

See CRC sequence calculation 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	0	FD_MBCRC							0				FD_TXCRC			
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	FD_TXCRC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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).

37.5.2.40 Enhanced RX FIFO Control (ERFCR)

Offset

Register	Offset
ERFCR	C0Ch

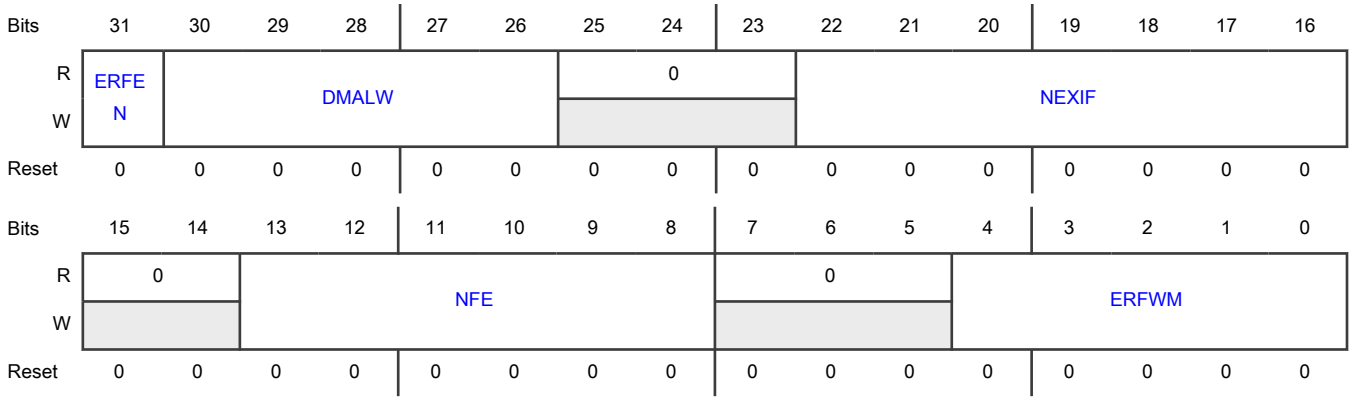
Function

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

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>30-26
DMALW</td><td>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.
<table>
<tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr>
<tr><td>0</td><td>1</td><td>2000h</td></tr>
<tr><td>1</td><td>2</td><td>2004h</td></tr>
<tr><td>2</td><td>3</td><td>2008h</td></tr>
<tr><td>3</td><td>4</td><td>200Ch</td></tr>
<tr><td>4</td><td>5</td><td>2010h</td></tr>
<tr><td>5</td><td>6</td><td>2014h</td></tr>
<tr><td>6</td><td>7</td><td>2018h</td></tr>
<tr><td>7</td><td>8</td><td>201Ch</td></tr>
<tr><td>8</td><td>9</td><td>2020h</td></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.</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>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>
<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>
</table></td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<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>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><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>
7-5 —	Reserved
4-0 ERFWM	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.

37.5.2.41 Enhanced RX FIFO Interrupt Enable (ERFIER)

Offset

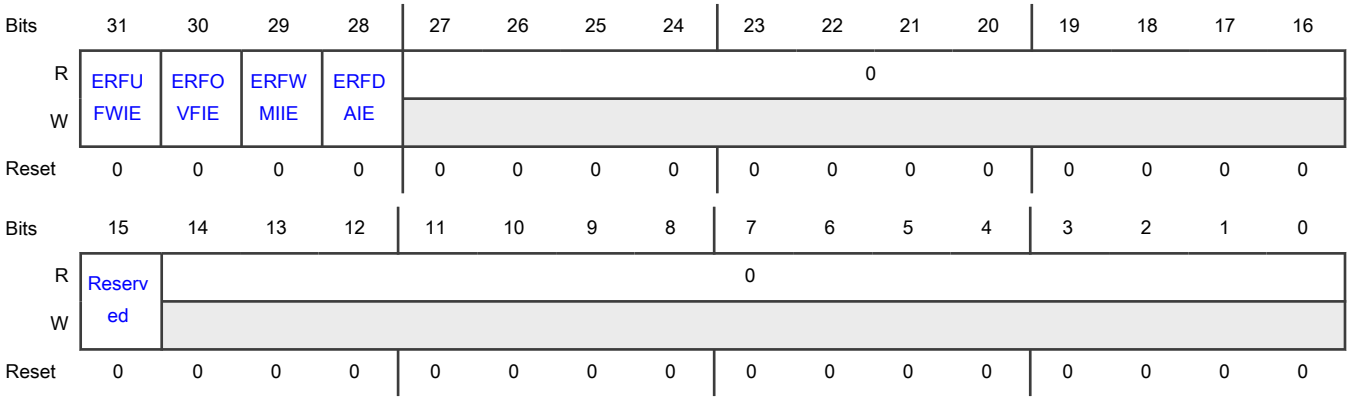
Register	Offset
ERFIER	C10h

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14-0 —	Reserved

37.5.2.42 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 ERFWMI	Enhanced RX FIFO Watermark Indication Flag 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not full 1b - Full
15 —	Reserved
14-6 —	Reserved
5-0 ERFEL	Enhanced RX FIFO Elements Indicates the number of CAN messages stored in the Enhanced RX FIFO.

37.5.2.43 Enhanced RX FIFO Filter Element (ERFFEL0 - ERFFEL31)

Offset

For n = 0 to 31:

Register	Offset
ERFFELn	3000h + (n × 4h)

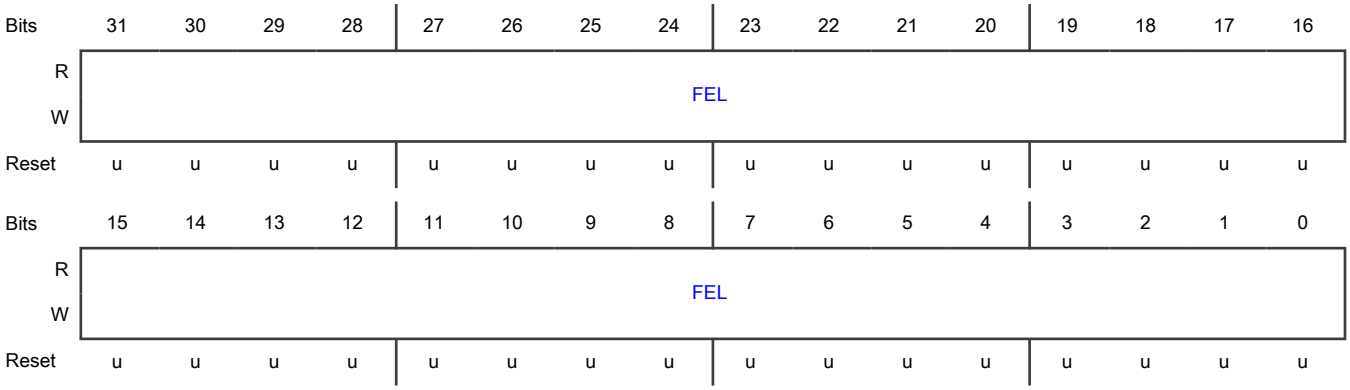
Function

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.

37.5.3 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 254. Message buffer structure example with 64-byte payload

	31	30	29	28	27	24	23	22	21	20	19	18	17	16	15	8	7	0
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 255. 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 256).
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 256 and Table 257. See Functional description.

Table 256. 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.

Table continues on the next page...

Table 256. 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
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 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 257. 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 258. 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 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 256, Table 257, 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 254). 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 259).
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 (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least

Table continues on the next page...

Table 258. RX and TX message buffer field descriptions (continued)

Mnemonic	Field	Description
		significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.
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 259.

Table 259. 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

37.5.4 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 260.

Table 260. 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

Table continues on the next page...

Table 260. 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
			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 261 indicates how the message buffers are arranged in RAM.

Table 261. 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

37.5.5 FlexCAN message buffer memory map

The FlexCAN memory buffers are allocated in memory according to the tables below.

Table 262. 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

Table continues on the next page...

Table 262. 8-byte message buffers (continued)

Address offset (hex)	MBDSR = b00 8-byte payload
0160	MB14
0170	MB15
0180	MB16
0190	MB17
01A0	MB18
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 263. 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

Table continues on the next page...

Table 263. 16-byte message buffers (continued)

Address offset (hex)	MBDSR = b01 16-byte payload
0158	MB9
0170	MB10
0188	MB11
01A0	MB12
01B8	MB13
01D0	MB14
01E8	MB15
0200	MB16
0218	MB17
0230	MB18
0248	MB19
0260	MB20

Table 264. 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 265. 64-byte message buffers

Address offset (hex)	MBDSR = b11 64-byte payload
0080	MB0
00C8	MB1
0110	MB2
0158	MB3
01A0	MB4
01E8	MB5
0230	MB6

37.5.6 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 266. Legacy RX FIFO structure

	31 28 24	23 22 21 20 19 18 17 16	15 8	7 0
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 267 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 267. ID table structure

Format	31	30	29 24	23 16	15	14	13 8	7 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 29–16)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std and ext = 31–24)		RXIDC_1 (std and ext = 23–16)		RXIDC_2 (std and ext = 15–8)			RXIDC_3 (std and ext = 7–0)	
		= Unimplemented or Reserved							

Table 268. 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 (29 to 19) 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) (29 to 19 and 13 to 3) 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.

37.5.7 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 269. Enhanced RX FIFO structure

	31	30	29	28	24	23	22	21	20	19	18	17	16	15	8	7	6	0
2000h	EDL	BRS	ESI	Reserved			SRR	IDE	RTR	DLC				TIMESTAMP LEGACY				
2004h	Reserved			ID (standard/extended)										ID (extended)				
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 270.

Table 270. ID HIT offset

Data Length Code (DLC)	ID HIT offset (IH_OFF)
0	2008h
1–4	200Ch

Table continues on the next page...

Table 270. ID HIT offset (continued)

Data Length Code (DLC)	ID HIT offset (IH_OFF)
5–8	2010h
9	2014h
10	2018h
11	201Ch
12	2020h
13	2028h
14	2038h
15	2048h

Table 271. 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

Table continues on the next page...

Table 271. Field descriptions (continued)

Mnemonic	Field	Description
		not include the data field, regardless of the DLC field. See Table 259 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 272 shows how FlexCAN writes IDHIT according to each filter element.

Table 272. 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.

Chapter 38
Low Power Inter-Integrated Circuit (LPI2C)

38.1 Chip-specific LPI2C information

Table 273. Reference links to related information

Topic	Related module	Reference
Full description	LPI2C	LPI2C
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

38.1.1 Module instances

This device supports four instances of LPI2C: LPI2C0, LPI2C1, LPI2C2 and LPI2C3.

38.2 Overview

LPI2C supports an efficient interface to an I2C bus as a controller and target:

- Implements logic support for Standard, Fast, Fast+, HS-mode (target only) and Ultra-Fast modes of operation
- Uses little CPU overhead, with DMA offloading of FIFO register accesses
- Continues operating in Deep Sleep modes if an appropriate clock is available

LPI2C 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.

The Inter-Integrated Circuit (I^2C) 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, as shown in Table 274.

Table 274. Updated terms

Updated term	Deprecated term
Controller	Master
Target	Slave

38.2.1 Block diagram

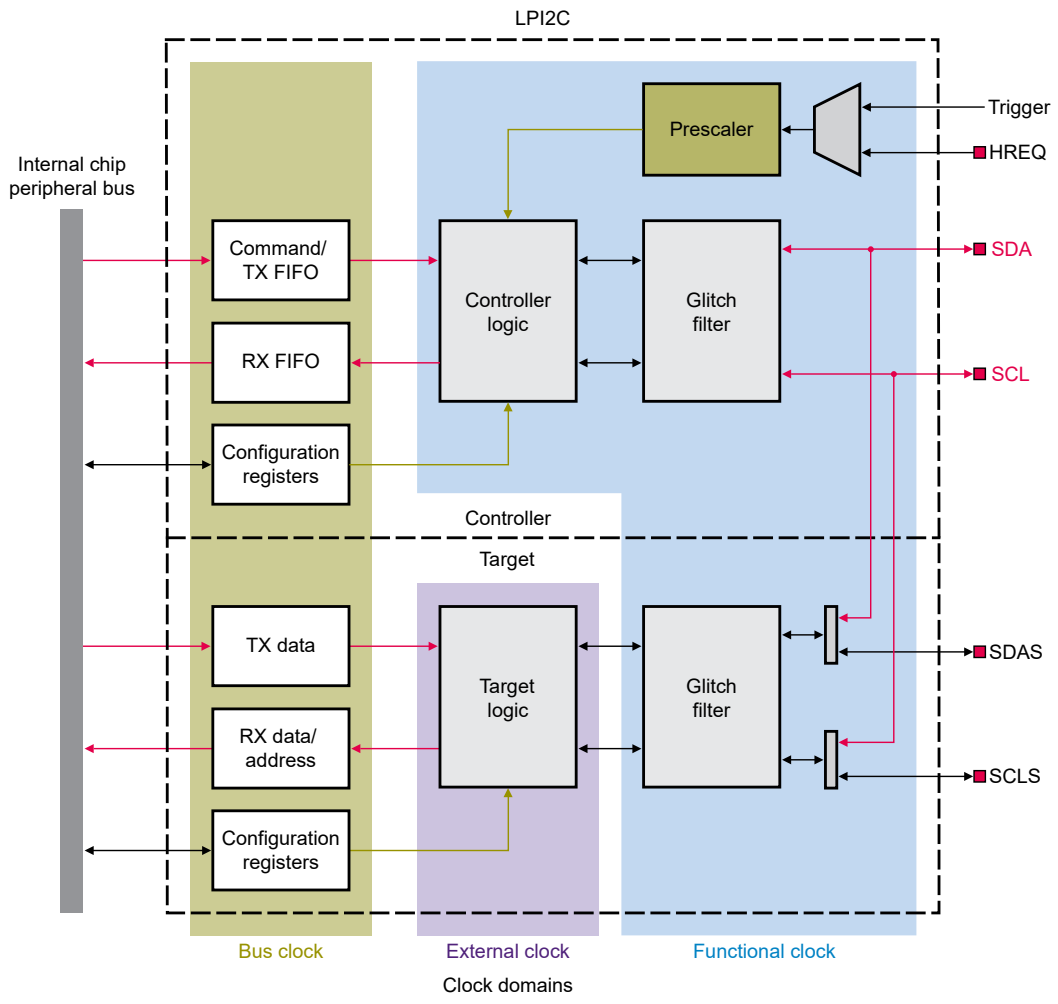


Figure 164. Block diagram

38.2.2 Features

LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes
- HS mode in target mode
- Multicontroller, including synchronization and arbitration, means that any number of controller nodes can be present. Also, controller and target roles can be changed between messages (after a Stop signal is sent).
- Clock stretching. Used on the SCL line, as an I2C flow control mechanism.
- Arbitration for when the system has more than one controller. When used on the SDA line, ensures that there is only one I2C 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 and transmit FIFO of 4 words (8-bit transmit data + 3-bit command)

- Receive FIFO of 4 words (8-bit receive data).
- Command FIFO waiting for an I2C idle bus before initiating a transfer
- Initiation of repeated Start and Stop conditions and one or more controller-receiver transfers by command FIFO
- Stop condition generation from command FIFO, or automatic generation of Stop condition when the transmit FIFO is empty
- Host request input to control the start time of an I2C bus transfer
- Interrupt generation on data match and unwanted data rejection, 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 or 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 or NACK field
- 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

38.3 Functional description

38.3.1 Controller mode

The LPI2C controller logic operates independently from the target logic to perform all controller-mode transfers on the I2C bus.

38.3.1.1 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. You can configure this operation to discard received data and not to store it in the receive FIFO.
- Stop condition. You can configure this operation to send a Stop condition when the transmit FIFO is empty.

Multiple transmit and receive commands can be inserted between the Start and Stop conditions. 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 via a (repeated) Start condition, followed by a transmit data byte containing the second address byte, followed by any number of data bytes with the controller transmit data.

A Start or repeated Start condition expecting a NACK (for example, HS mode controller code) must be followed by a Stop or (repeated) Start condition.

38.3.1.2 Controller operations

When 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 an HS 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 performed 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 or MCFGR1[AUTOSTOP]. A repeated Start can change which timing configuration register is used.

When 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 it is 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, 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:

- The 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.
- The 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.

38.3.1.3 Receive FIFO and data matching

The receive FIFO stores receive data during controller-receiver transfers. You can configure the LPI2C controller to discard received data instead of storing it in the receive FIFO. This option is configured via 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. You can configure the data match function 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.

You can configure the receiver match function to discard all received data until a data match is detected, using MCFGR0[RDMO]. Following a data match, write 0 to MCFGR0[RDMO] before writing 0 to MSR[DMF] to allow all subsequent data to be received.

38.3.1.4 Timing parameters

The LPI2C controller can configure the following timing parameters. Parameters are configured separately for HS mode (Controller Clock Configuration 1 (MCCR1)) and other modes (Controller Clock Configuration 0 (MCCR0)). This separation allows the HS mode controller code to be sent using regular timing parameters. Then it allows a switch to HS mode timing (following a repeated Start) until the next STOP condition.

Configure the LPI2C controller timing parameters, measured in LPI2C functional clock cycles, as shown in Table 275. You must configure these parameters to meet the I2C timing specification for the required mode.

Table 275. Timing parameters

I2C specification timing parameter	I2C specification timing symbol	LPI2C timing parameter (in LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) × (2 ^ PRESCALE)
Hold time (repeated) Start condition	tHD:STA	(SETHOLD +1) × (2 ^ PRESCALE)
Low period of the SCL clock	tLOW	(CLKLO + 1) × (2 ^ PRESCALE)
High period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Setup time for a repeated Start condition or Stop condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Data hold time	tHD:DAT	(DATAVD + 1) × (2 ^ PRESCALE)
Data setup time	tSU:DAT	(SDA_LATENCY + 1) × (2 ^ PRESCALE)
Bus free time between a Stop and Start condition	tBUF	(CLKLO + 1 + SDA_LATENCY) × (2 ^ PRESCALE)
Data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) × (2 ^ PRESCALE)

Table 276 defines the latency parameters. 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 pullup 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 276. 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 277. LPI2C timing parameter restrictions

Timing parameter	Minimum	Maximum	Comment
CLKLO	03h	—	CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	01h	—	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	02h	—	SETHOLD × (2 ^ PRESCALE) > SDA_LATENCY

Table continues on the next page...

Table 277. LPI2C timing parameter restrictions (continued)

Timing parameter	Minimum	Maximum	Comment
DATAVD	01h	CLKLO – SDA_LATENCY – 1	Configure DATAVD to meet the data hold requirement in the I2C specification
FILTSCL	00h	[CLKLO × (2 ^ PRESCALE)] – 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)
FILTSDA	FILTSCL	[CLKLO × (2 ^ PRESCALE)] – 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO + SETHOLD + 2) × 2	—	Must also be greater than (CLKHI + 1)

See the UM10204, *I2C-bus specification and user manual*.

See Application information for example LPI2C timing configurations.

38.3.1.5 Error conditions

The LPI2C controller monitors 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).

You must respond to MSR[PLTF] to terminate the existing command. You can terminate the command cleanly by writing 0 to MCR[MEN], or you can terminate it abruptly by writing 1 to MCR[RST].

You can use MCFGR0[RELAX] to attempt to recover 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 address FFh, then the Stop condition, should generate sufficient SCL clock edges to cause the target to release SDA.

You can use MCFGR2[BUSIDLE] 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 or SDA must be high for (BUSIDLE + 1) prescaler cycles before the I2C bus is considered idle.

38.3.1.6 Pin configuration

Configuration	Description
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.
HS mode support	Support for HS mode depends on the specific device. This mode requires the SCL pin to support the current source pullup 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 MCFGR1[IGNACK] 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 HS mode. A partial HS mode, not a full HS mode, because this configuration actively drives high rather than enabling a current service pull-up. This configuration sets the SCL pin as push-pull for every clock except the first clock pulse, to allow HS-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, configure the pin 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 cannot connect to separate I2C buses.

38.3.2 Target mode

To perform all target mode transfers on the I2C bus, the LPI2C target logic operates independently from the LPI2C controller logic.

38.3.2.1 Address matching

You can configure the LPI2C target:

- To match one of two addresses, using either 7-bit or 10-bit addressing modes for each address.
- To match a range of addresses in either 7-bit or 10-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 HS mode controller code address, 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 is matched, it never drives the SDA pin. Configure the LPI2C target to match only on 7-bit address mode when this feature is enabled. Software can support 10-bit address mode. This option is intended for debugging the contents of an I2C transfer between another controller

and target. Likewise, SCFGR1[SDCFG] and SCFGR1[RSCFG] configure the Stop or repeated Start flags to assert on all Stop or repeated Start conditions, even when there is no address match.

38.3.2.2 Transmit and receive data

Target Transmit Data (STDR) and Target Receive Data (SRDR) are double-buffered and only update during a target-transmit and target-receive transfer, respectively.

You can configure the target address that was received to be read from SRDR (for example, when using DMA to transfer data) or from Target Address Status (SASR).

You can configure STDR to request data only after a target-transmit transfer is detected. You can also configure it to request new data whenever STDR is empty.

Write to STDR only when SSR[TDF] is set.

Read SRDR only when SSR[RDF] is set, or when SSR[AVF] is set and SCFGR1[RXCFG] = 1.

Read SASR only when SSR[AVF] is set.

38.3.2.3 Read request

The LPI2C target generates a read request when the I2C bus is idle and you write 1 to SCFGR0[RDREQ]. This occurrence causes the LPI2C target to pull the SDA pin low until either the first SCL falling edge or SCFGR0[RDREQ] becomes 0. Following the next Stop condition or a software timeout, you can proceed as follows:

- If the LPI2C target is accessed at the correct address, the read request is acknowledged, and you must write 0 to SCFGR0[RDREQ].
- If SCFGR0[RDACK] is 1, the read request is acknowledged by a Start followed by one SCL pulse followed by a STOP condition. You must write 0 to SCFGR0[RDREQ].
- If neither of the first two cases occurred, the read request is not acknowledged. You must 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 is not acknowledged, and you should write 0 to SCFGR0[RDREQ]. Another request can be generated after an appropriate delay.

Writing to SCFGR0[RDREQ] when the I2C bus is busy results in a logic 0 being written. Write 1 to SCFGR0[RDREQ], then confirm that the register is written to correctly. If the register does not update correctly, the I2C bus is no longer idle and the read request must be attempted later.

38.3.2.4 Clock stretching

The LPI2C target supports many configurable options for clock stretching. You can configure these conditions to perform clock stretching:

- SSR[AVF] is set during the ninth clock pulse of the address byte.
- SSR[TDF] is set during the ninth clock pulse of a target-transmit transfer.
- SSR[RDF] is set during the ninth clock pulse of a target-receive transfer.
- SSR[TAF] is set during the eighth clock pulse of an address byte or a target-receive transfer. In HS mode, this option is disabled.
- Clock stretching can be extended for a number of cycles equal to the value of SCFGR2[CLKHOLD] cycles. This stretching allows additional setup time to sample the SDA pin externally. In HS 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 SCFGR2[CLKHOLD] configuration.

38.3.2.5 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 HS 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:

- You must configure SCFGR2[FILTSDA] to be greater than or equal to SCFGR2[FILTSCL] (unless compensating for board level skew between SDA and SCL).
- You must configure SCFGR2[DATAVD] to be less than the minimum SCL low period.

38.3.2.6 Error conditions

The LPI2C target can flag the following error conditions:

- SSR[BEF] is set when the LPI2C target is driving SDA but it samples a different value than what is expected.
- SSR[FEF] is set 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.
- SSR[FEF] is also set 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 the value of SCFGR1[RXCFG].

The LPI2C target does not implement a timeout due to SCL or SDA being stuck low. If this detection is required, use the LPI2C controller logic so you can reset the LPI2C target when this condition is detected.

38.3.3 Low-power modes

LPI2C remains functional during low-power modes, if MCR[DOZEN] = 0 and LPI2C uses an external or internal clock source that remains enabled. LPI2C can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPI2C to be disabled in low-power modes when MCR[DOZEN] = 1. In this case, LPI2C waits for the current transfer to complete any pending operation.

NOTE

See the chip-specific information for low-power modes available on your chip.

38.3.4 Debug mode

Table 278. Debug mode

Mode	LPI2C operation
Debug	If MCR[DBGEN] = 1, can continue operating in Debug mode.

38.3.5 Peripheral triggers

The connection of the LPI2C peripheral triggers to other peripherals depends upon the specific device being used.

Table 279. LPI2C triggers

Trigger	Description
Controller output trigger	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	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	To control the start of a LPI2C bus transfer, the LPI2C input trigger can be selected instead of the HREQ input. 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 the value of MCFGR1[PRESCALE]. When LPI2C is busy, the HREQ input (and therefore the input trigger) is ignored.

38.3.6 Clocking

Table 280. LPI2C clocks

Clock	Description
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. These pins are SCL and SDA, or SCLS and SDAS if the 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 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 chip-specific clocking information, see the Clocking chapter.

38.3.7 Reset

Table 281. LPI2C resets

Reset	Description
Chip reset	The logic and registers for the LPI2C controller and target are reset to their default states after a chip reset.
Software reset	The LPI2C controller implements a software reset field in its control register. MCR[RST] resets all controller logic and registers to their default states, except for Controller Control (MCR) itself.

Table continues on the next page...

Table 281. LPI2C resets (continued)

Reset	Description
	The LPI2C target implements a software reset field in its control register. SCR[RST] resets all target logic and registers to their default states, except for Target Control (SCR) itself.
FIFO reset	The LPI2C controller implements write-only control fields 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 fields 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.

38.3.8 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

38.3.8.1 Controller mode

Table 282 lists the Controller mode sources that can generate LPI2C controller interrupts and LPI2C controller transmit and receive DMA requests.

Table 282. Controller interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (MSR[TDF])	Data can be written to transmit FIFO, as configured by MFCR[TXWATER].	Y	TX	Y
Receive Data Flag (MSR[RDF])	Data can be read from the receive FIFO, as configured by MFCR[RXWATER].	Y	RX	Y
End Packet Flag (MSR[EPF])	Controller has transmitted a repeated Start or Stop condition.	Y	N	Y
Stop Detect Flag (MSR[SDF])	Controller has transmitted a Stop condition (and the LPI2C controller is idle, if MCFGR1[STOPCFG] = 1) .	Y	N	Y
NACK Detect Flag (MSR[NDF])	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
Arbitration Lost Flag (MSR[ALF])	The controller lost arbitration due to a Start or Stop condition detected at the wrong time, or the controller was transmitting data	Y	N	Y

Table continues on the next page...

Table 282. Controller interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
	but received data different from the data that was transmitted.			
FIFO Error Flag (MSR[FEF])	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
Pin Low Timeout Flag (MSR[PLTF])	Pin low timeout is enabled and SCL (or SDA, if configured) is low for longer than the configured timeout.	Y	N	Y
Data Match Flag (MSR[DMF])	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry.	Y	N	Y
Start Detect Flag (MSR[STF])	A Start condition is detected on the I2C bus (and the LPI2C controller is idle, if MCFGR1[STARTCFG] = 1).	Y	N	Y
Controller Busy Flag (MSR[MBF])	LPI2C controller is busy transmitting or receiving data.	N	N	N
Bus Busy Flag (MSR[BBF])	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

38.3.8.2 Target mode

Table 283 lists the target mode sources that can generate LPI2C target interrupts and LPI2C target transmit and receive DMA requests.

Table 283. Target interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (SSR[TDF])	Data can be written to Target Transmit Data (STDR).	Y	TX	Y
Receive Data Flag (SSR[RDF])	Data can be read from Target Receive Data (SRDR).	Y	RX	Y
Address Valid Flag (SSR[AVF])	Address can be read from Target Address Status (SASR).	Y	RX	Y
Transmit ACK Flag (SSR[TAF])	ACK or NACK can be written to Target Transmit ACK (STAR).	Y	N	Y
Repeated Start Flag (SSR[RSF])	Target has detected an address match followed by a repeated Start condition.	Y	RX	Y

Table continues on the next page...

Table 283. Target interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Stop Detect Flag (SSR[SDF])	Target has detected an address match followed by a Stop condition.	Y	RX	Y
Bit Error Flag (SSR[BEF])	Target was transmitting data, but received data is different from what was transmitted.	Y	N	Y
FIFO Error Flag (SSR[FEF])	This flag is set by: • Transmit data underrun • Receive data overrun • Address status overrun when SCFGR1[RXCFG] = 1 This flag can only be set when clock stretching is disabled.	Y	N	Y
Address Match 0 Flag (SSR[AM0F])	Target detected an address match SAMR[ADDR0].	Y	N	N
Address Match 1 Flag (SSR[AM1F])	Target detected an address match with SAMR[ADDR1] or using an address range.	Y	N	N
General Call Flag (SSR[GCF])	Target detected an address match with the general call address.	Y	N	N
SMBus Alert Response Flag (SSR[SARF])	Target detected an address match with the SMBus alert address.	Y	N	N
Target Busy Flag (SSR[SBF])	LPI2C target is busy receiving an address byte or is transmitting or receiving data.	N	N	N
Bus Busy Flag (SSR[BBF])	LPI2C target is enabled and a Start condition is detected on I2C bus, but no Stop condition detected.	N	N	N

38.3.8.3 End-of-packet DMA transfer

End-of-packet functionality serves serial interfaces where the transfer size may not be known 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 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 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 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. You can perform this reconfiguration through software or hardware (for example, channel linking or scatter-gather). Minimize this delay 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, increase the UART idle line length as needed.

38.4 External signals

Table 284. External signals

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
SCL	LPI2C clock line	SCL	In Four-Wire mode, this pin is the SCL input pin.	Input or output
SDA	LPI2C data line	SDA	In Four-Wire mode, this pin is the SDA input pin.	Input or output
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.	Input or output
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.	Input or output
HREQ	Host request	When configured for controller (input), if host request is asserted and the I2C bus is idle, it initiates a transfer. When configured for target (output), it asserts while valid data is present in Target Transmit Data (STDR). HREQ is an additional pin separate from the two-wire or four-wire scheme.		Input or output

Figure 165 shows the two-signal connection.

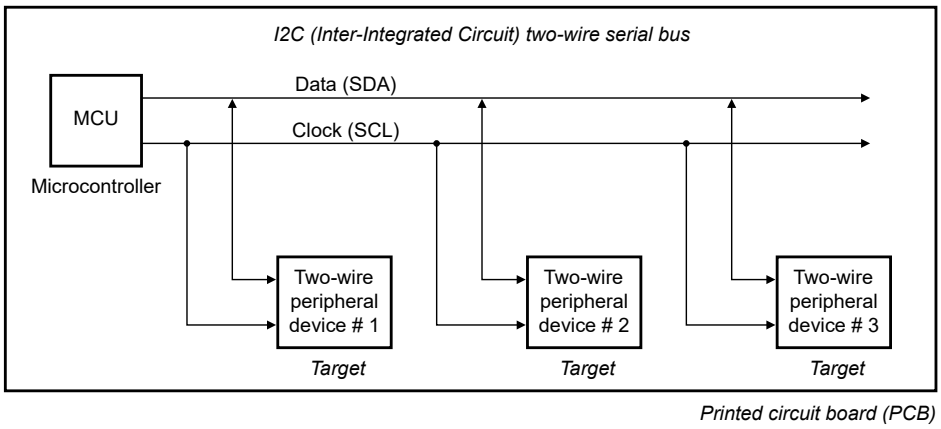


Figure 165. I²C two-wire serial bus

Figure 166 shows a possible four-signal connection.

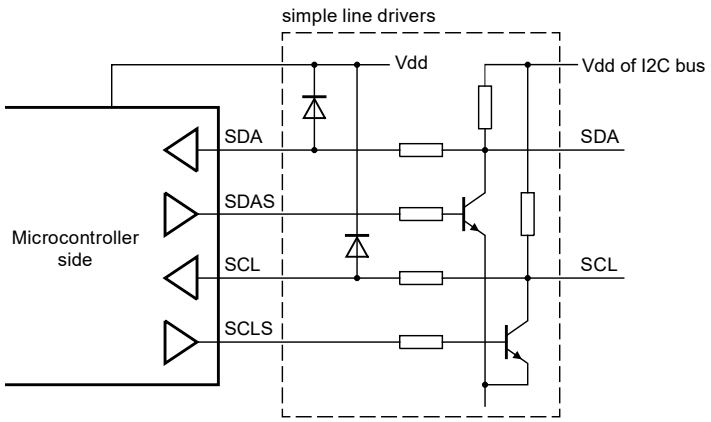


Figure 166. I²C four-wire serial bus

38.5 Initialization

To initialize the LPI2C controller:

1. Configure Controller Configuration 0 (MCFGR0)–Controller Configuration 3 (MCFGR3) as required by the application.
2. Configure Controller Clock Configuration 0 (MCCR0) and Controller Clock Configuration 1 (MCCR1) to satisfy the timing requirements of the I2C mode supported by the application.
3. Enable controller interrupts and DMA requests as required by the application.
4. Enable the LPI2C controller by writing 1 to MCR[MEN].

To initialize the LPI2C target:

1. Configure Target Address Match (SAMR) with the I2C address of the target location on the I2C bus.
2. Configure Target Configuration 0 (SCFGR0) and Target Configuration 1 (SCFGR1) as required by the application.
3. Configure Target Configuration 2 (SCFGR2) to satisfy the timing requirements of the I2C mode supported by the application.
4. Enable target interrupts and DMA requests as required by the application.
5. Enable the LPI2C target by writing 1 to SCR[SEN].

38.6 Application information

Configure the I2C timing parameters to meet the requirements of the I2C specification. This configuration depends on the supported mode and LPI2C functional clock frequency. When switching between two modes using different clock configuration registers (for example, Fast mode and HS mode), MCFGR1[PRESCALE] must remain constant between the modes.

Table 285. Example timing configurations

I2C mode	Clock frequency	Baud rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Standard	8 MHz	100 kbit/s	0h	0h/0h	24h	28h	24h	02h
Standard	48 MHz	100 kbit/s	2h	1h/1h	37h	3Fh	37h	03h
Standard	60 MHz	100 kbit/s	2h	1h/1h	45h	50h	44h	04h
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	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	60 MHz	3.33 Mbit/s	1h	0h/0h	04h	04h	02h	01h

38.7 Memory map and registers

38.7.1 LPI2C register descriptions

Writing to a read-only register or reading from a write-only register can cause bus errors. This module does not check whether programmed values in the registers are correct; you must ensure that valid programmed values are written to the registers.

38.7.1.1 LPI2C memory map

LPI2C0 base address: 4009_A000h

LPI2C1 base address: 4009_B000h

LPI2C2 base address: 400D_4000h

LPI2C3 base address: 400D_5000h

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

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
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
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

38.7.1.2 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	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 design specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0010b - Controller only, with standard feature set 0000_0000_0000_0011b - Controller and target, with standard feature set

38.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function

Contains parameter values 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
3-0 MTXFIFO	Controller Transmit FIFO Size Configures the number of words in the controller transmit FIFO to $2^{MTXFIFO}$.

38.7.1.4 Controller Control (MCR)

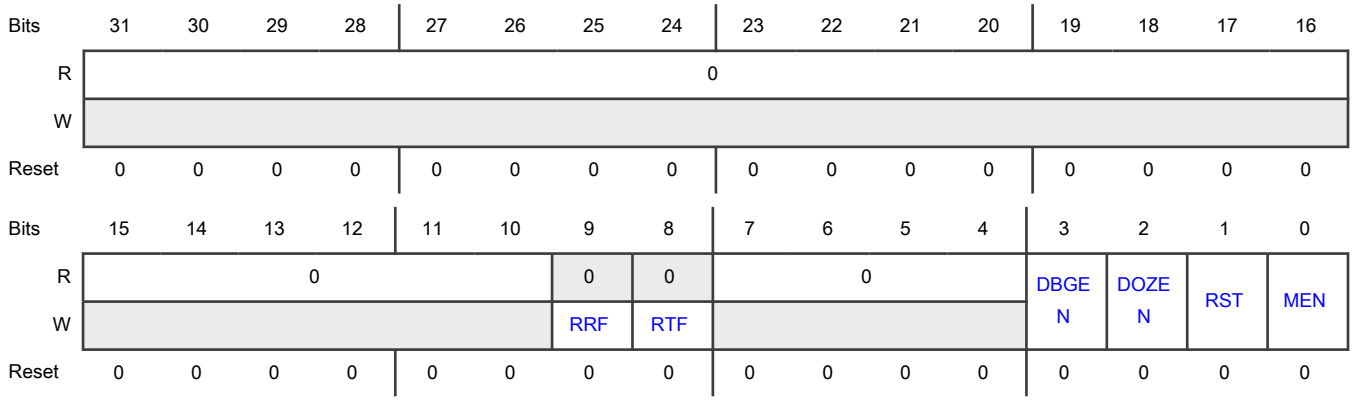
Offset

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 - Reset receive FIFO
8 RTF	Reset Transmit FIFO Resets the transmit FIFO. 0b - No effect 1b - Reset transmit FIFO
7-4 —	Reserved
3 DBGEN	Debug Enable Enables the controller in Debug mode. 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables the controller in Doze mode. 0b - Enable 1b - Disable
1	Software Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
RST	Resets all internal controller logic and registers except Controller Control (MCR). This field remains 1 (enabled) until you write 0 to it. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - No effect 1b - Reset
0 MEN	Controller Enable Enables the controller logic. 0b - Disable 1b - Enable

38.7.1.5 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 BBF	Bus Busy Flag Specifies whether the I2C bus is busy. 0b - Idle 1b - Busy
24 MBF	Controller Busy Flag Specifies whether the I2C controller is busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 STF	Start Flag Specifies whether a Start condition is detected on the bus when the bus is idle. When MCFGR1[STARTCFG] is 0, this field becomes 1 only if the LPI2C controller is also idle. When MCFGR1[STARTCFG] is 1, STF becomes 1 for any Start condition when the I2C bus is idle. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Start condition not detected 1b - Start condition detected When writing 0b - No effect 1b - Clear the flag
14 DMF	Data Match Flag Indicates whether the received data matches MDMR[MATCH0] or MDMR[MATCH1] (as configured by MCFGR1[MATCFG]). Received data discarded due to MTDR[CMD] does not cause this flag to set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Matching data not received 1b - Matching data received When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 PLTF	Pin Low Timeout Flag Indicates whether pin low timeout has occurred. Sets when the SCL or SDA input is low for more than the number of PINLOW cycles defined by MCFGR3[PINLOW], even when the LPI2C controller is idle. You must resolve the pin low condition via software. PLTF cannot be cleared as long as the pin low timeout continues. Before LPI2C can initiate a Start condition, you must clear this flag. See MCFGR1[TIMECFG] for the SCL and/or SDA timeout settings. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Pin low timeout did not occur 1b - Pin low timeout occurred When writing 0b - No effect 1b - Clear the flag
12 FEF	FIFO Error Flag Detects the LPI2C controller's attempt to send or receive data without first generating a (repeated) Start condition. This error can occur when the transmit FIFO underflows when MCFGR1[AUTOSTOP] = 1. When this flag is set, the LPI2C controller sends a Stop condition (if busy). The controller does not initiate a new Start condition until the flag is cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
11 ALF	Arbitration Lost Flag Indicates whether arbitration is lost. Either of these conditions sets this flag: • 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 when the LPI2C controller is transmitting data. When ALF is set, the LPI2C controller releases the I2C bus (goes idle), and the LPI2C controller does not initiate a new Start condition until the ALF is cleared.

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 - Controller did not lose arbitration 1b - Controller lost arbitration When writing 0b - No effect 1b - Clear the flag
10 NDF	NACK Detect Flag Indicates whether an unexpected NACK has been detected. This flag is set when the LPI2C controller detects a NACK it was not expecting when transmitting an address or data. When set, the controller does not initiate a new Start condition until this flag is cleared. If a NACK is expected for a given address (as configured by the command word), this flag is set if a NACK is not generated. When this flag is set, the LPI2C controller automatically transmits a Stop condition if MCFGR1[AUTOSTOP] = 1, or if the transmit FIFO is not empty. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No unexpected NACK detected 1b - Unexpected NACK detected When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C controller has generated a Stop condition. When MCFGR1[STOPCFG] = 0, this flag is set for any Stop condition. When MCFGR1[STOPCFG] = 1, this flag is set only when the LPI2C controller is also idle (transmit FIFO is empty). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop condition generated 1b - Stop condition generated When writing 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clear the flag
8 EPF	End Packet Flag Indicates whether the LPI2C controller has generated a repeated Start condition or a Stop condition. When the controller first generates a Start condition, this flag is not set. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop or repeated Start generated 1b - Stop or repeated Start generated When writing 0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the receive data is ready. This flag is set when the number of words in the receive FIFO is greater than MFCR[RXWATER]. 0b - Receive data not ready 1b - Receive data ready
0 TDF	Transmit Data Flag Indicates whether transmit data is requested. This flag is set when the number of words in the transmit FIFO is equal or less than MFCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data requested

38.7.1.6 Controller Interrupt Enable (MIER)

Offset

Register	Offset
MIER	18h

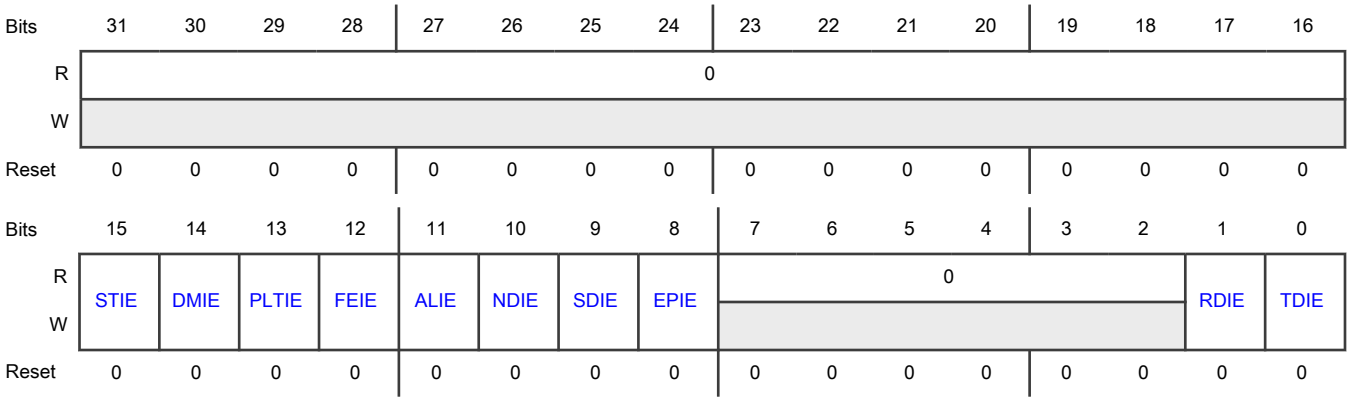
Function

Contains enables for:

- Transmit and receive data interrupts

- Start, Stop, and NACK detection interrupts
- DMA interrupts

Diagram



Fields

Field	Function
31-16 —	Reserved
15 STIE	Start Interrupt Enable Enables interrupt for Start condition. 0b - Disable 1b - Enable
14 DMIE	Data Match Interrupt Enable Enables interrupt for data match. 0b - Disable 1b - Enable
13 PLTIE	Pin Low Timeout Interrupt Enable Enables interrupt for pin-low timeout. 0b - Disable 1b - Enable
12 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable
11	Arbitration Lost Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ALIE	Enables interrupt for arbitration lost. 0b - Disable 1b - Enable
10 NDIE	NACK Detect Interrupt Enable Enables interrupt for NACK detection. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 EPIE	End Packet Interrupt Enable Enables interrupt for end packet. 0b - Disable 1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

38.7.1.7 Controller DMA Enable (MDER)

Offset

Register	Offset
MDER	1Ch

Function

Contains DMA transmit, request, and receive enables.

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														RDDE	TDDE
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 RDDE	Receive Data DMA Enable Enables DMA receive data. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables DMA transmit data. 0b - Disable 1b - Enable

38.7.1.8 Controller Configuration 0 (MCFGR0)

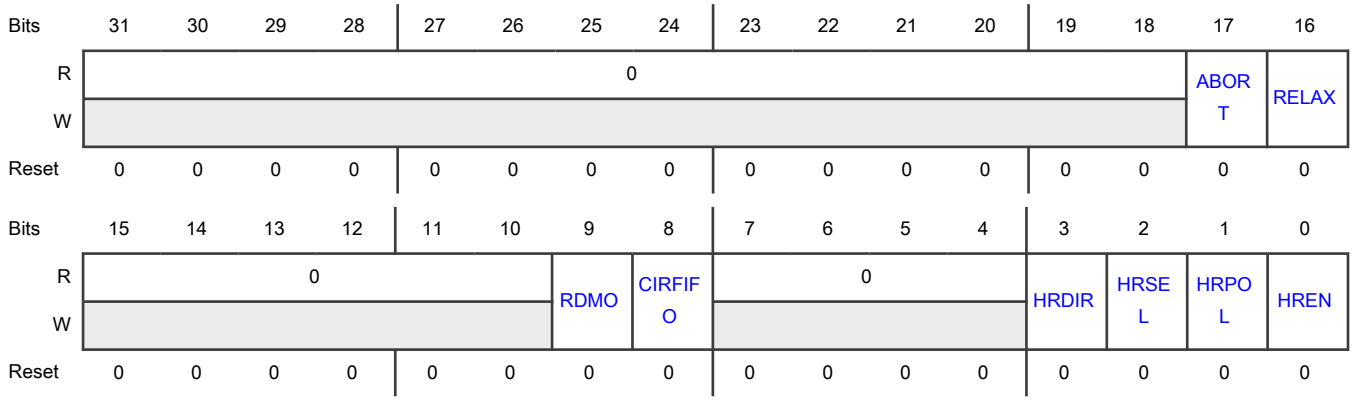
Offset

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 Determines whether the LPI2C controller completes the existing byte and then sends a Stop condition. A new transfer does not start while this field is 1. 0b - Normal transfer 1b - Abort existing transfer and do not start a new one
16 RELAX	Relaxed Mode Specifies the type of transfer. If this field is 1, the LPI2C controller relaxes several transfer restrictions. Transfers start when the bus is busy and FIFO commands are not checked for errors. You can use this mode to attempt recovery of a stuck I2C target by toggling the SCL pin. 0b - Normal transfer 1b - Relaxed transfer
15-10 —	Reserved
9 RDMO	Receive Data Match Only Determines whether all received data that does not set MSR[DMF] is discarded. After MSR[DMF] is set, the RDMO configuration is ignored. When disabling RDMO, write 0 to this field 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 MSR[DMF] is set
8 CIRFIFO	Circular FIFO Enable Enables the transmit FIFO read pointer to be saved to a temporary register. The transmit FIFO empties as normal. After the LPI2C controller is idle and the transmit FIFO is empty, the read pointer value is

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 - Disable 1b - Enable
7-4 —	Reserved
3 HRDIR	Host Request Direction Configures the direction of the HREQ pin. When this field is 1, the LPI2C target drives the HREQ output pin and the pin becomes 1 when there is data in the target transmit data register. For proper operation when this field is 1, write 1 to SCFGR1[TXCFG]. 0b - HREQ pin is input (for LPI2C controller) 1b - HREQ pin is output (for LPI2C target)
2 HRSEL	Host Request Select Selects the source of the host request input. When host request is enabled, this field must not change. 0b - Host request input is pin HREQ 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 field must not change. HRPOL sets the polarity for both the HREQ pin and the input trigger. • When HRPOL=0, the polarity is configured for active low, so host request is asserted if the HREQ pin or input trigger are logic 0. • When HRPOL=1, the polarity is configured for active high, so host request is asserted if the HREQ pin or input trigger are logic 1. 0b - Active low 1b - Active high
0 HREN	Host Request Enable Enables host request. 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 condition is not affected by the host request. 0b - Disable 1b - Enable

38.7.1.9 Controller Configuration 1 (MCFGR1)

Offset

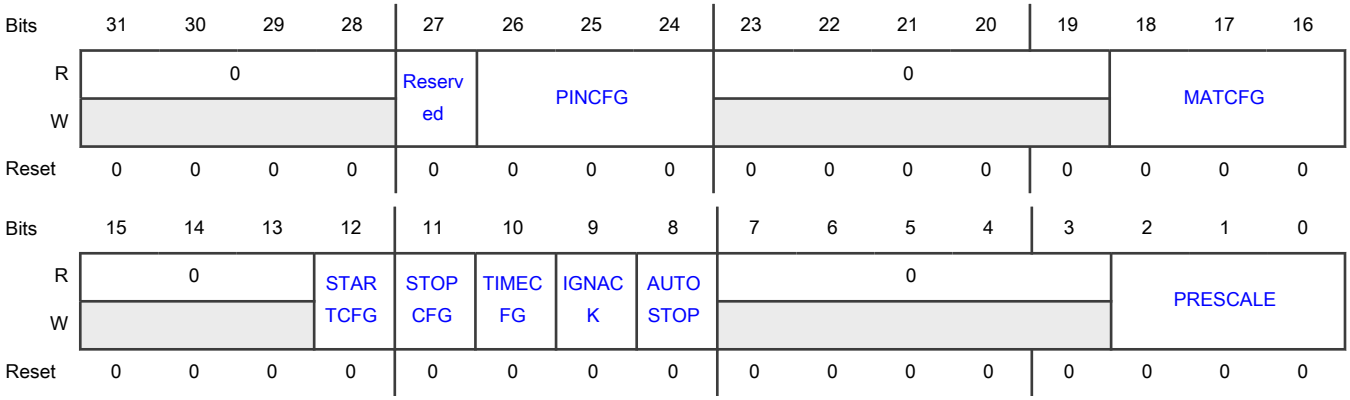
Register	Offset
MCFGR1	24h

Function

Contains controls for pin configuration, clock prescaler, and various other control settings.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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.
23-19 —	Reserved
18-16 MATCFG	Match Configuration Configures the condition that sets MSR[DMF]. 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 conditions that set MSR[STF] when a Start condition occurs. When this field is 0, MSR[STF] is set on a Start condition when both the I2C bus and LPI2C are idle. In other words, any nonrepeated Start condition is initiated by any controller on the bus except the LPI2C controller. When this field is 1, MSR[STF] is set on a Start condition when the I2C bus is idle. In other words, any nonrepeated Start condition is initiated by any controller on the bus including the LPI2C controller. 0b - Sets when both I2C bus and LPI2C controller are idle 1b - Sets when I2C bus is idle

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 STOPCFG	Stop Configuration Configures the conditions that set MSR[SDF]. When this field is 0, any Stop condition generated by the LPI2C controller sets SDF. When this field is 1, the last Stop condition before the LPI2C controller is idle sets SDF. In this case, the transmit FIFO is empty at the time of the Stop condition. 0b - Any Stop condition 1b - Last Stop condition
10 TIMECFG	Timeout Configuration Configures which signals must be low for longer than the configured timeout to set MSR[PLTF]. When this field is 0, MSR[PLTF] is set when SCL is low for longer than the configured timeout. 0b - SCL 1b - SCL or SDA
9 IGNACK	Ignore NACK Determines whether the LPI2C controller ignores a received NACK and treats it as an ACK. In this case, MSR[NDF] is never set. This field must be 1 in Ultra-Fast mode. 0b - No effect 1b - Treat a received NACK as an ACK
8 AUTOSTOP	Automatic Stop Generation Determines whether a Stop condition is generated when the LPI2C controller is busy and the transmit FIFO is empty. A Stop condition can also be generated using a transmit FIFO command. When this field is 1, a Stop condition is automatically generated when the transmit FIFO is empty and the LPI2C controller is busy. 0b - No effect 1b - Stop automatically generated
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C controller logic except the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16

Table continues on the next page...

Table continued from the previous page...

Field	Function
	101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

38.7.1.10 Controller Configuration 2 (MCFGR2)

Offset

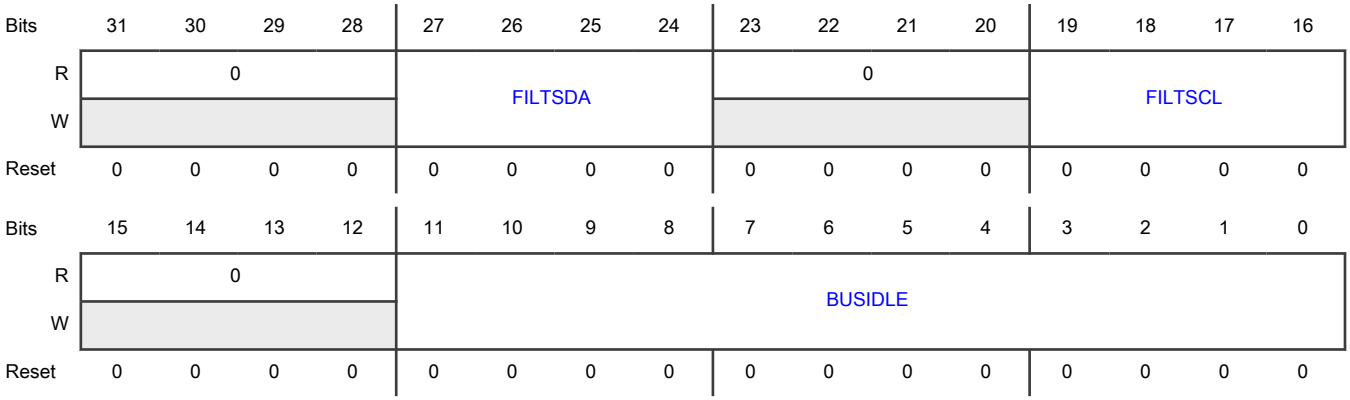
Register	Offset
MCFGR2	28h

Function

Contains the configuration for the bus idle timeout and glitch filters for SDA and SCL.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C controller digital glitch filters for the SDA input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C controller digital glitch filters for SCL input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. These cycles are based on the functional clock. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
15-12 —	Reserved
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 the number of cycles defined by this field, the I2C bus is assumed to be idle and the controller can generate a Start condition. Writing 0 to this field disables the bus idle timeout.

38.7.1.11 Controller Configuration 3 (MCFGR3)

Offset

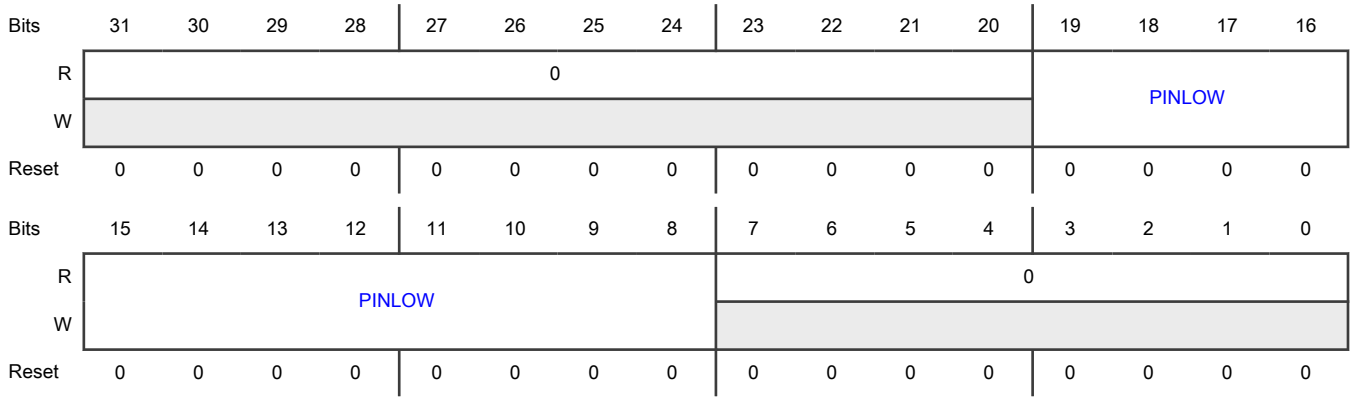
Register	Offset
MCFGR3	2Ch

Function

Configures the threshold value for the pin low timeout flag.

Write to this register only when the I2C controller is disabled.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-8 PINLOW	Pin Low Timeout Configures the threshold value, in clock cycles, that sets MSR[PLTF]. If SCL or SDA (selected by MCFGR1[TIMECFG]) is low for longer than (PINLOW × 256) cycles, MSR[PLTF] is set. When this field is 0, the pin low timeout feature is disabled.
7-0 —	Reserved

38.7.1.12 Controller Data Match (MDMR)

Offset

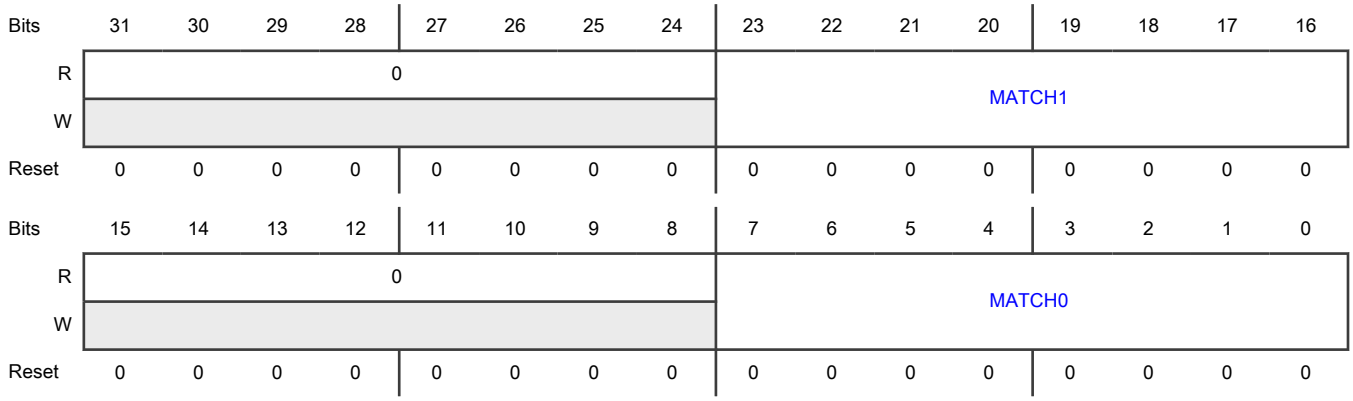
Register	Offset
MDMR	40h

Function

Contains data match values.

Write to this register only when the I2C controller is disabled or idle.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Specifies match 1 value that is compared to the received data when receive data match is enabled.
15-8 —	Reserved
7-0 MATCH0	Match 0 Value Specifies match 0 value that is compared to the received data when receive data match is enabled.

38.7.1.13 Controller Clock Configuration 0 (MCCR0)

Offset

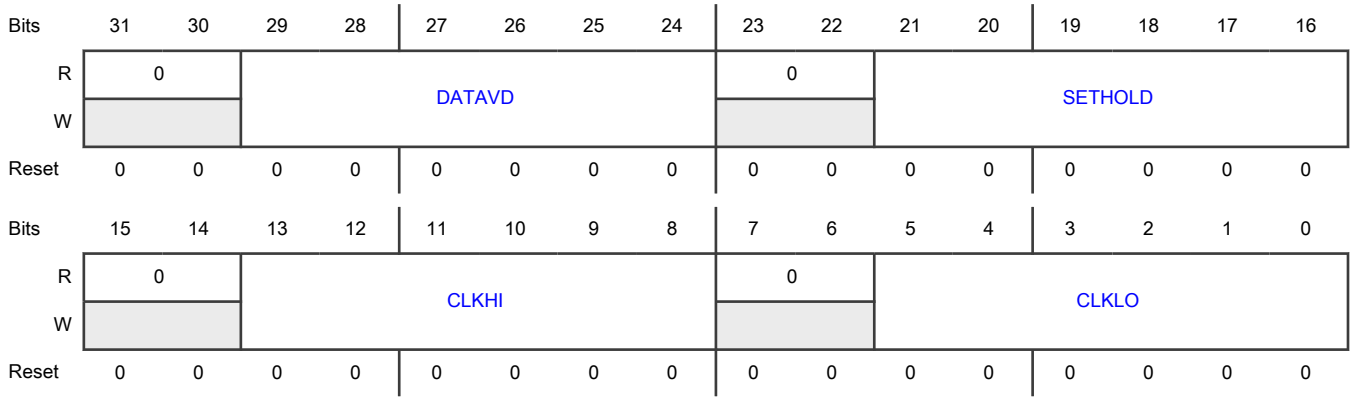
Register	Offset
MCCR0	48h

Function

Configures various clock controls.

You cannot make changes to this register 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 Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) 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 Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed 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 Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed 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.

38.7.1.14 Controller Clock Configuration 1 (MCCR1)

Offset

Register	Offset
MCCR1	50h

Function

Configures various clock controls.

You cannot makes changes to this register when the I2C controller is enabled and is used for HS mode transfers. The separate clock configuration for HS mode allows arbitration to take place in Fast mode (with timing configured by Controller Clock Configuration 0 (MCCR0)), before switching to HS mode (with timing configured by MCCR1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DATAVD						0		SETHOLD					
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		CLKHI						0		CLKLO					
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 DATAVD	Data Valid Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) 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 needed 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 Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed 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 Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed 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.

38.7.1.15 Controller FIFO Control (MFCR)

Offset

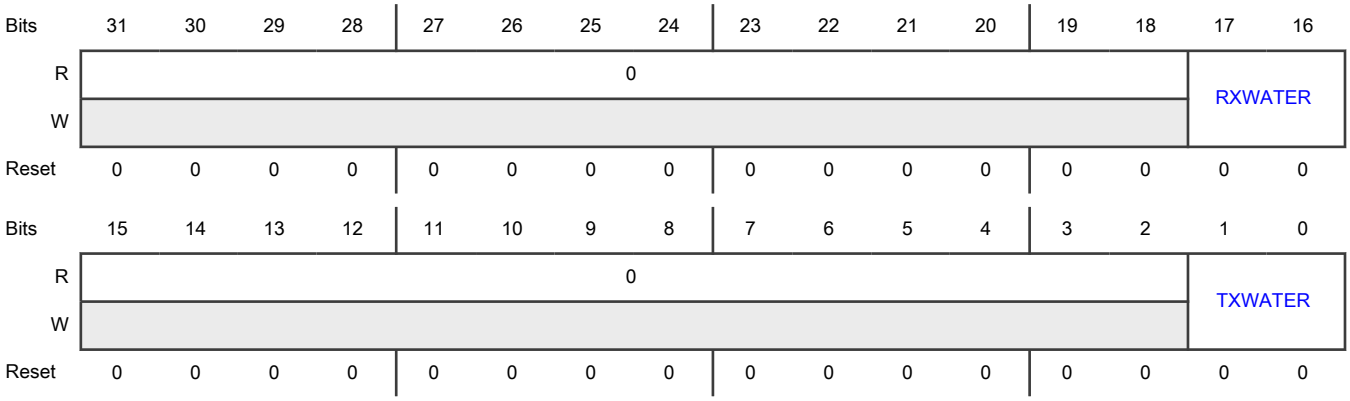
Register	Offset
MFCR	58h

Function

Controls the receive and transmit FIFO watermark values.

This register is used only in Stop mode, when this register is static (not changing).

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark Determines the watermark for setting SSR[RDF]. That flag is set when the number of words in the receive FIFO is greater than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.
15-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark Determines the watermark for setting SSR[TDF]. That flag is set when the number of words in the transmit FIFO is equal or less than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.

38.7.1.16 Controller FIFO Status (MFSR)

Offset

Register	Offset
MFSR	5Ch

Function

Specifies the number of words in the transmit and receive FIFOs.

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 Specifies the number of words in the receive FIFO.
15-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Specifies the number of words in the transmit FIFO.

38.7.1.17 Controller Transmit Data (MTDR)

Offset

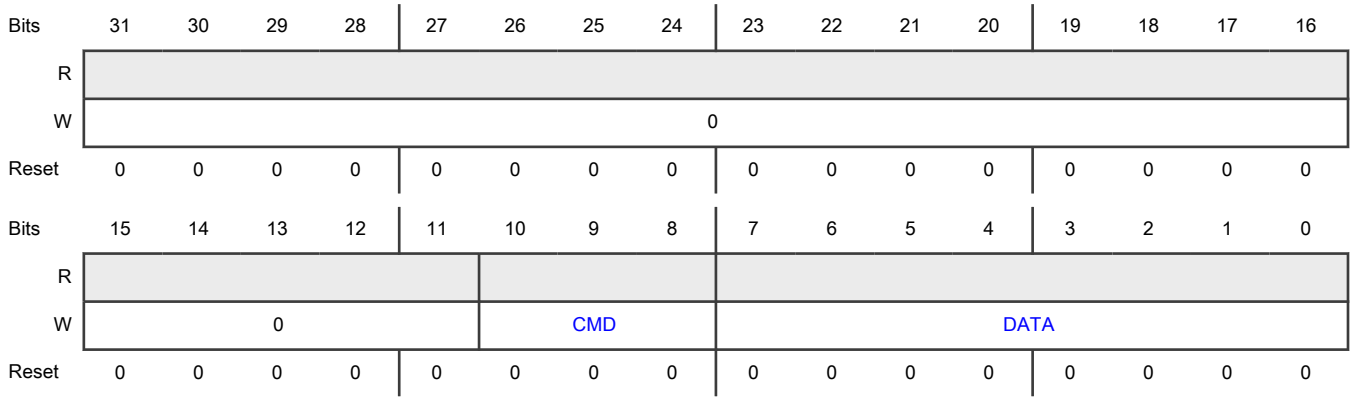
Register	Offset
MTDR	60h

Function

Configures transmit data:

- An 8-bit write to MTDR[CMD] is ignored and does not increment the FIFO write pointer.
- An 8-bit write to MTDR[DATA] zero-extends the value of MTDR[CMD] and increments the FIFO write pointer.
- A 16-bit or 32-bit write operation writes to both MTDR[CMD] and MTDR[DATA] and increments the FIFO write pointer.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Data Selects command transmitted by controller. 000b - Transmit the value in DATA[7:0] 001b - Receive (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes and check each for a data match (if configured) before storing the received data in the receive FIFO. 010b - Generate Stop condition on I2C bus 011b - Receive and discard (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes but do not check for a data match or store those bytes in the receive FIFO. 100b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] 101b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] (this transfer expects a NACK to be returned) 110b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode 111b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode (this transfer expects a NACK to be returned)
7-0 DATA	Transmit Data Contains data used by the commands listed in MTDR[CMD]. Performing an 8-bit write to this field zero-extends the value of MTDR[CMD].

38.7.1.18 Controller Receive Data (MRDR)

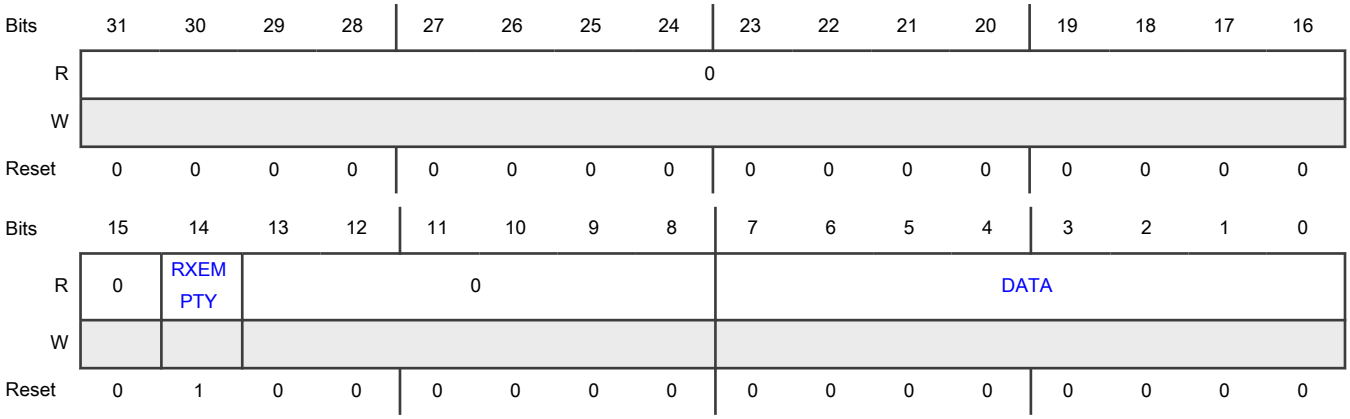
Offset

Register	Offset
MRDR	70h

Function

Contains the status of the receive FIFO and the data received by the I2C controller that has not been discarded.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	Receive Empty Indicates whether the controller receive data FIFO is empty. 0b - Not empty 1b - Empty
13-8 —	Reserved
7-0 DATA	Receive Data Contains data received by the I2C controller that has not been discarded. Received data can be discarded due to the command in MTDR[CMD], or the controller can be configured to discard nonmatching data.

38.7.1.19 Controller Receive Data Read Only (MRDROR)

Offset

Register	Offset
MRDROR	78h

Function

Contains the status of the receive FIFO and returns the data received by the I2C controller, but does not pull the data from the FIFO.

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	RXEMPTY	0						DATA							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	RX Empty Indicates whether the receive data FIFO is empty. 0b - Not empty 1b - Empty
13-8 —	Reserved
7-0 DATA	Receive Data

38.7.1.20 Target Control (SCR)

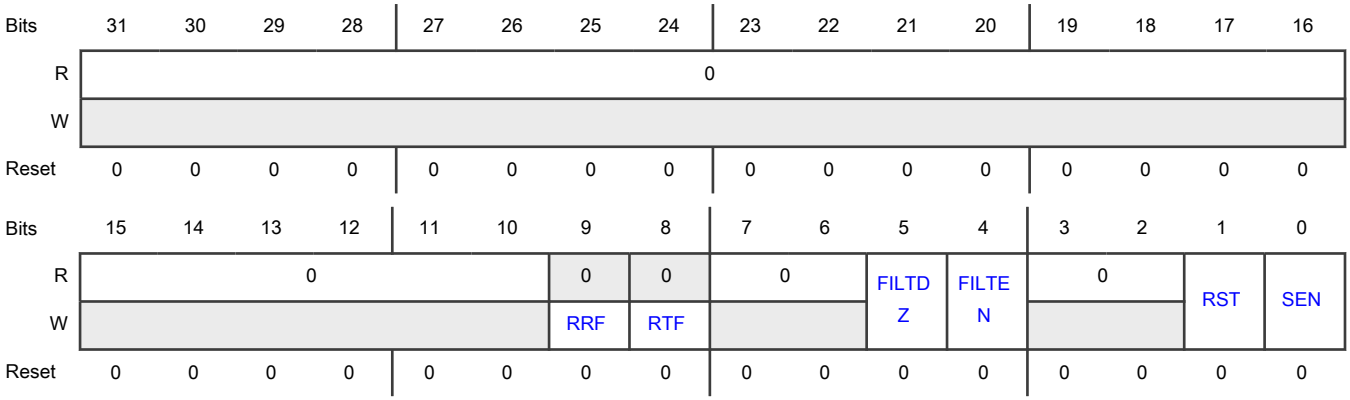
Offset

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 in Target Receive Data (SRDR). 0b - No effect 1b - SRDR is now empty
8 RTF	Reset Transmit FIFO Empties the transmit FIFO in Target Transmit Data (STDR). 0b - No effect 1b - STDR is now empty
7-6 —	Reserved
5	Filter Doze Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FILTDZ	Enables filter in Doze mode. Update this field only when the I2C target is disabled. 0b - Enable 1b - Disable
4 FILTEN	Filter Enable Enables digital filter and output delay counter for target mode. Update this field only when the I2C target is disabled. 0b - Disable 1b - Enable
3-2 —	Reserved
1 RST	Software Reset Resets target mode logic. The reset takes effect immediately. The value of this field remains 1 until you write 0 to it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 SEN	Target Enable Enables I2C Target mode. 0b - Disable 1b - Enable

38.7.1.21 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 whether an I2C bus is idle or busy. 0b - Idle 1b - Busy
24 SBF	Target Busy Flag Indicates whether an I2C target is idle or busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 SARF	SMBus Alert Response Flag Indicates whether an SMBus alert response has been detected. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Disabled or not detected 1b - Enabled and detected
14 GCF	General Call Flag Indicates whether a target has detected the general call address. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - General call address disabled or not detected 1b - General call address detected
13 AM1F	Address Match 1 Flag Indicates whether the received address matches the value in ADDR1, or it falls within the ADDR0 to ADDR1 range as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Matching address not received 1b - Matching address received
12 AM0F	Address Match 0 Flag Indicates whether the received address matches the ADDR0 field, as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - ADDR0 matching address not received 1b - ADDR0 matching address received
11 FEF	FIFO Error Flag Indicates whether there is a FIFO error. This flag can only be set when clock stretching is disabled. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
10 BEF	Bit Error Flag Indicates whether the LPI2C target has transmitted 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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No bit error occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Bit error occurred When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C target detects a Stop condition, and if the LPI2C target matched the last address byte. When SCFGR1[SDCFG] = 1, this flag is set on any Stop condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stop detected 1b - Stop detected When writing 0b - No effect 1b - Clear the flag
8 RSF	Repeated Start Flag Indicates whether the LPI2C target detects a repeated Start condition and if the LPI2C target matched the last address byte. When SCFGR1[RSCFG] = 1, this flag is set on any repeated Start condition. This flag is not set when the target first detects a Start condition. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No repeated Start detected 1b - Repeated Start detected When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3 TAF	Transmit ACK Flag Indicates whether a transmit ACK or NACK is required. You can clear this flag by writing to Target Transmit ACK (STAR). 0b - Not required

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Required
2 AVF	Address Valid Flag Indicates whether the contents of Target Address Status (SASR) are valid. You can clear this flag by reading SASR. When SCFGR1[RXCFG] = 1, this flag is also cleared by reading Target Receive Data (SRDR). 0b - Not valid 1b - Valid
1 RDF	Receive Data Flag Indicates whether receive data is ready. You can clear this flag by reading Target Receive Data (SRDR). When SCFGR1[RXCFG] = 1, this flag is not cleared when reading Target Receive Data (SRDR) if SSR[AVF] = 1. 0b - Not ready 1b - Ready
0 TDF	Transmit Data Flag Indicates whether transmit data has been requested. This flag is cleared by writing to Target Transmit Data (STDR). When SCFGR1[TXCFG] = 0, if a NACK, repeated Start, or Stop condition is detected, this flag is also cleared. 0b - Transmit data not requested 1b - Transmit data is requested

38.7.1.22 Target Interrupt Enable (SIER)

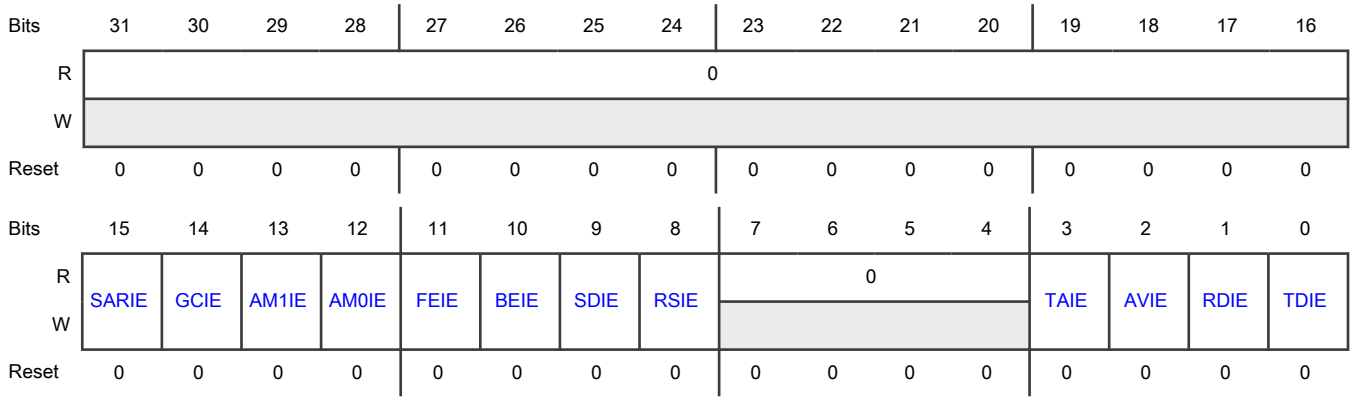
Offset

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 - Disable 1b - Enable
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. 0b - Disable 1b - Enable
12 AM0IE	Address Match 0 Interrupt Enable Enables interrupt for address match 0. 0b - Disable 1b - Enable
11 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 BEIE	Bit Error Interrupt Enable Enables interrupt for bit error. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 RSIE	Repeated Start Interrupt Enable Enables interrupt for repeated start. 0b - Disable 1b - Enable
7-4 —	Reserved
3 TAIE	Transmit ACK Interrupt Enable Enables interrupt for transmit ACK. 0b - Disable 1b - Enable
2 AVIE	Address Valid Interrupt Enable Enables interrupt for valid address. 0b - Disable 1b - Enable
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

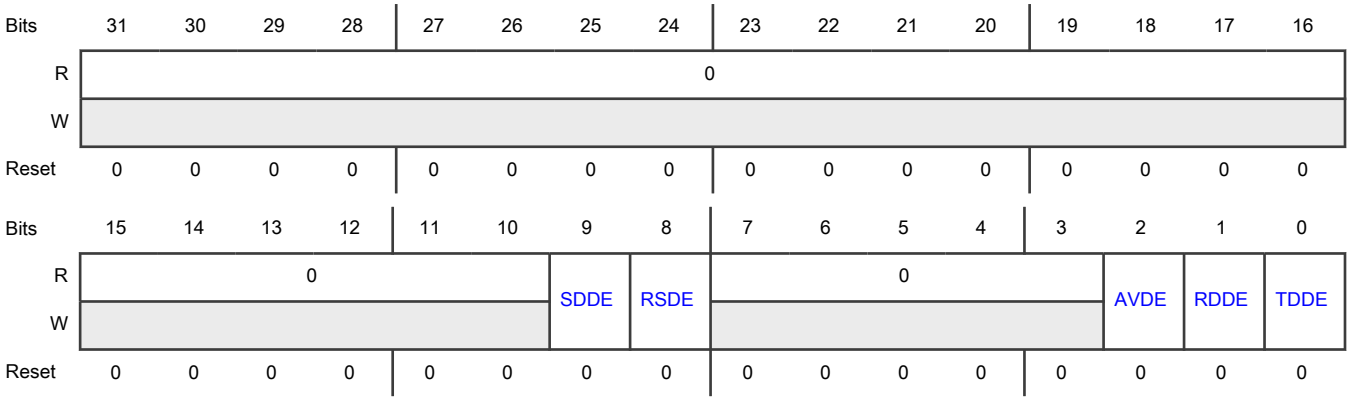
38.7.1.23 Target DMA Enable (SDER)

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 Target Receive Data (SRDR) when it is empty: • Generates a DMA end-of-packet response. • Returns 0000_40FFh. • Writes 0 to MSR[SDF]. 0b - Disable 1b - Enable
8 RSDE	Repeated Start DMA Enable Enables DMA end-of-packet processing on repeated start. Reading Target Receive Data (SRDR) when it is empty: • Generates a DMA end-of-packet response.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Returns 0000_40FFh. • Writes 0 to MSR[RDF]. 0b - Disable 1b - Enable
7-3 —	Reserved
2 AVDE	Address Valid DMA Enable Enables address valid DMA request. The address valid DMA request is shared with the receive data DMA request. If both are enabled, write 1 to SCFGR1[RXCFG] to allow the DMA to read the address from Target Receive Data (SRDR). 0b - Disable 1b - Enable
1 RDDE	Receive Data DMA Enable Enables receive data for DMA. 0b - Disable DMA request 1b - Enable DMA request
0 TDDE	Transmit Data DMA Enable Enables transmit data for DMA. 0b - Disable 1b - Enable

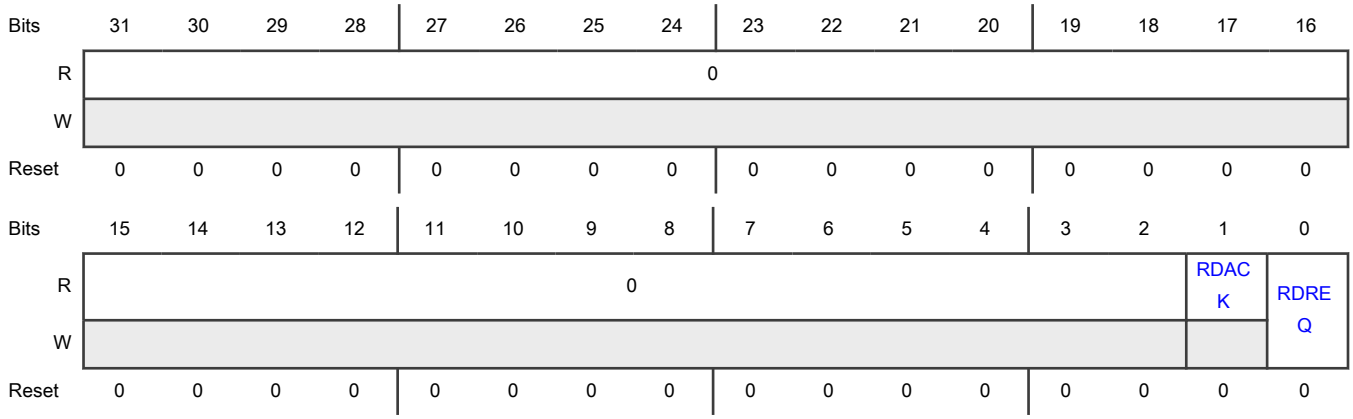
38.7.1.24 Target Configuration 0 (SCFGR0)

Offset

Register	Offset
SCFGR0	120h

Function
Configures the read request feature.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 RDACK	Read Acknowledge Flag Indicates whether a Start then Stop sequence with one SCL pulse between them acknowledged the read request while SCFGR0[RDREQ] = 1. You can clear this flag by writing 0 to SCFGR0[RDREQ]. 0b - Read Request not acknowledged 1b - Read Request acknowledged
0 RDREQ	Read Request Enables read request. When the I2C bus is idle, writing 1 to this field 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 when you write 0 to this field. To initiate a second read request (for example, following I2C bus activity that did not acknowledge the request), write 0 then 1 to this field. When the I2C bus is busy, writing to this register always writes 0 to this field. 0b - Disable 1b - Enable

38.7.1.25 Target Configuration 1 (SCFGR1)

Offset

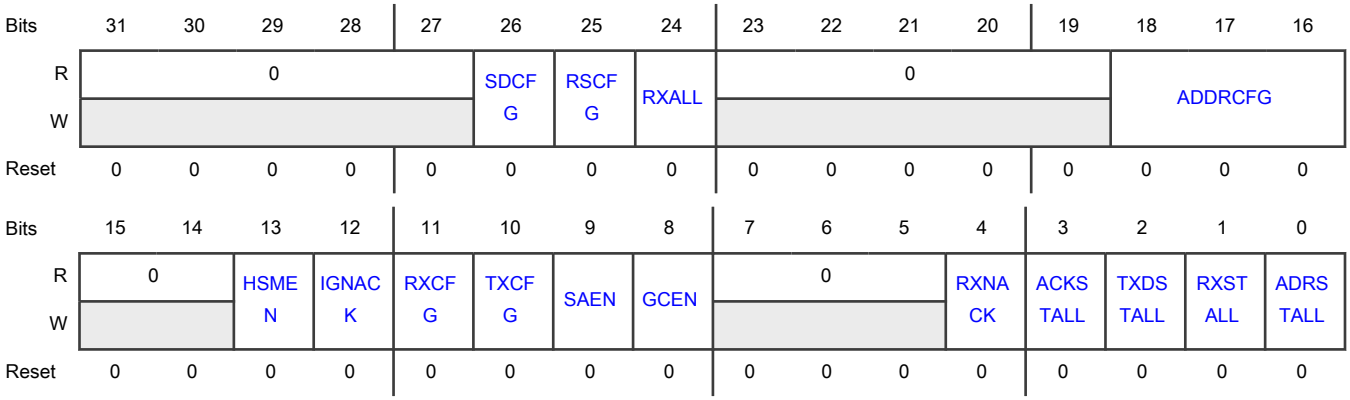
Register	Offset
SCFGR1	124h

Function

Configures various aspects of the target.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26 SDCFG	Stop Detect Configuration Configures the conditions that set MSR[SDF]. 0b - Any Stop condition following an address match 1b - Any Stop condition
25 RSCFG	Repeated Start Configuration Configures the conditions that set MSR[STF]. 0b - Any repeated Start condition following an address match 1b - Any repeated Start condition
24 RXALL	Receive All Enables receive-all functionality. When enabled, the LPI2C target stores all addresses on the I2C bus in Target Address Status (SASR) and all data on the I2C bus in Target Receive Data (SRDR). However, the LPI2C target does not drive SDA (for ACK or target-transmit transfer) unless there is an address match. The LPI2C target can drive SCL if clock stretching is enabled. When this field is 1, the LPI2C target only supports 7-bit addressing modes (you must configure SCFGR1[ADDRCFG] for 7-bit address match). Software can support 10-bit addresses, however. The first byte of a 10-bit address is saved to Target Address Match (SAMR). The second byte is saved as the first data byte in Target Receive Data (SRDR).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Use this field to aid debugging of the I2C bus by saving all data on the bus without altering the state of the bus. When this field is 1, it is recommended to also write 1 to SCFGR1[RSCFG] and SCFGR1[SDCFG] so you can track the full state of the I2C bus. 0b - Disable 1b - Enable
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
13 HSMEN	HS Mode Enable Enables detection of the HS 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 HS mode controller code is detected, SCR[FILTEN] and SCFGR1[ACKSTALL] are ignored until the next Stop condition is detected. 0b - Disable 1b - Enable
12 IGNACK	Ignore NACK Determines whether the target ends transfer when a NACK condition is detected. When this field is 1, the LPI2C target continues transfers after a NACK is detected. This field is required to be 1 in Ultra-Fast mode. 0b - End transfer on NACK 1b - Do not end transfer on NACK
11 RXCFG	Receive Data Configuration Configures which data is returned and which flags are cleared when reading Target Receive Data (SRDR).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When this field is 0, reading SRDR returns received data and clears MSR[RDF]. When this field is 1, reading SRDR: • Returns the value of Target Address Status (SASR) and clears SSR[AVF] when SSR[AVF] is set. • Returns received data and clears MSR[RDF] when SSR[AVF] is not set. 0b - Return received data, clear MSR[RDF] 1b - Return SASR and clear SSR[AVF] when SSR[AVF] is set, return received data and clear MSR[RDF] when SSR[AFV] is not set
10 TXCFG	Transmit Flag Configuration Determines which conditions set MSR[TDF]. This field 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 this field is 0, Target Transmit Data (STDR) is automatically emptied when a target-transmit transfer is detected. MSR[TDF] is set when a target-transmit transfer is detected, and MSR[TDF] is cleared at the end of the target-transmit transfer. When this field is 1, MSR[TDF] is set when STDR is empty, and MSR[TDF] is cleared when STDR is full. This setting allows STDR to be filled before a target-transmit transfer is detected. However, it can cause STDR to be written before a NACK is detected on the last byte of a target-transmit transfer. 0b - MSR[TDF] is set only during a target-transmit transfer when STDR is empty 1b - MSR[TDF] is set whenever STDR is empty
9 SAEN	SMBus Alert Enable Enables a match on an SMBus alert. 0b - Disable 1b - Enable
8 GCEN	General Call Enable Enables a general call address. 0b - Disable 1b - Enable
7-5 —	Reserved
4 RXNACK	Receive NACK Determines whether to override the setting of STAR[TXNACK] when the LPI2C receives a matching address during an overrun. When this field is 1, the LPI2C target responds with a NACK under the following conditions:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• SSR[AVF] would be set due to matching an address, but that flag is already 1 (address overrun). • SSR[RDF] would be set due to receiving data, but that flag is already 1 (receive data overrun). 0b - ACK or NACK always determined by STAR[TXNACK] 1b - NACK always generated on address overrun or receive data overrun, otherwise ACK or NACK is determined by STAR[TXNACK]
3 ACKSTALL	ACK SCL Stall Enables SCL clock stretching during target-transmit address bytes and target-receiver address and data bytes, so you can write to Target Transmit ACK (STAR) before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the ninth bit, and is therefore not compatible with HS mode. If this field is 1: • You do not need to write 1 to SCFGR1[RXSTALL] or SCFGR1[ADRSTALL]. • When there is an address match on the first byte of a 10-bit address, SSR[AVF] is set, allowing you to read the received address before writing to Target Transmit ACK (STAR). 0b - Disable 1b - Enable
2 TXDSTALL	Transmit Data SCL Stall Enables SCL clock stretching when SSR[TDF] = 1 during a target-transmit transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
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 HS mode. 0b - Disable 1b - Enable
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 HS mode. 0b - Disable 1b - Enable

38.7.1.26 Target Configuration 2 (SCFGR2)

Offset

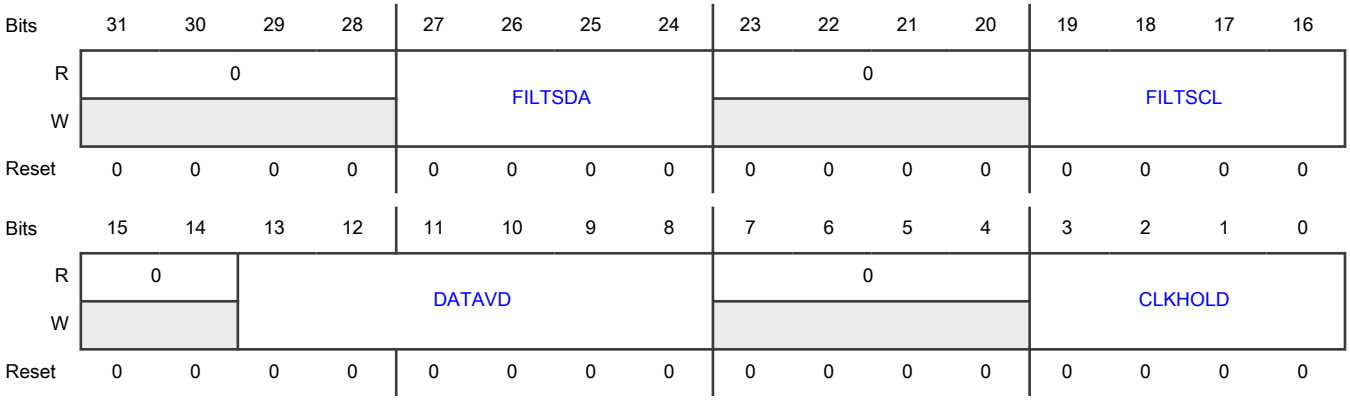
Register	Offset
SCFGR2	128h

Function

Configures data valid delay, clock hold time, and glitch filters for SDA and SCL.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C target digital glitch filters for SDA input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
23-20 —	Reserved
19-16	Glitch Filter SCL

Table continues on the next page...

Table continued from the previous page...

Field	Function
FILTSCL	Configures the I2C target digital glitch filters for SCL input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS 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. MCFGR1[PRESCALE] does not affect the I2C target data valid delay time, and the I2C target data valid delay time is disabled in HS 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 the number of cycles defined by this field + 3. MCFGR1[PRESCALE] does not affect the I2C target clock hold time, and the I2C target clock hold time is disabled in HS mode.

38.7.1.27 Target Address Match (SAMR)

Offset

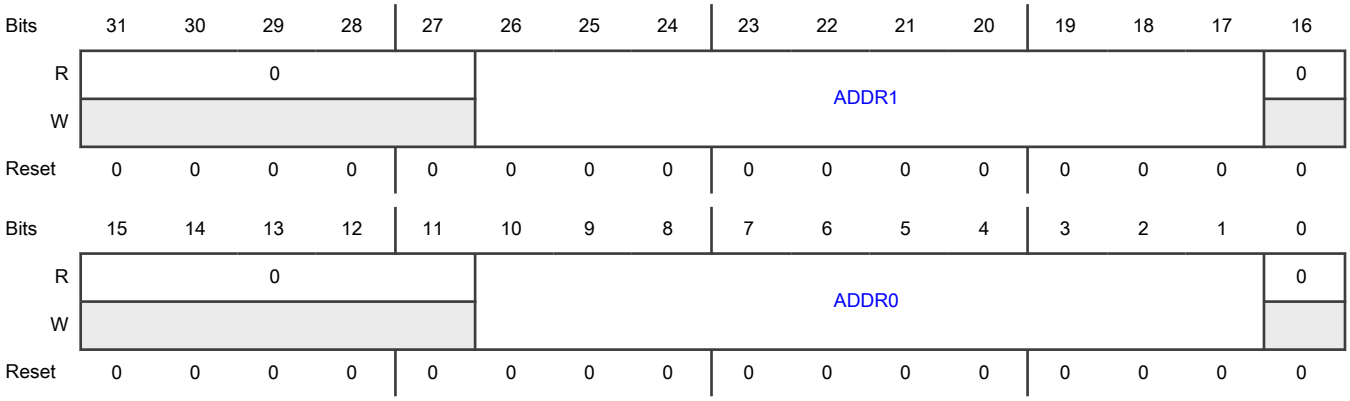
Register	Offset
SAMR	140h

Function

Contains address values for received target match comparison.

Write to this register only when the I2C target is disabled.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 Value Contains the value of address 1, which is compared to 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 Contains the value of address 0, which is compared to 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].
0 —	Reserved

38.7.1.28 Target Address Status (SASR)

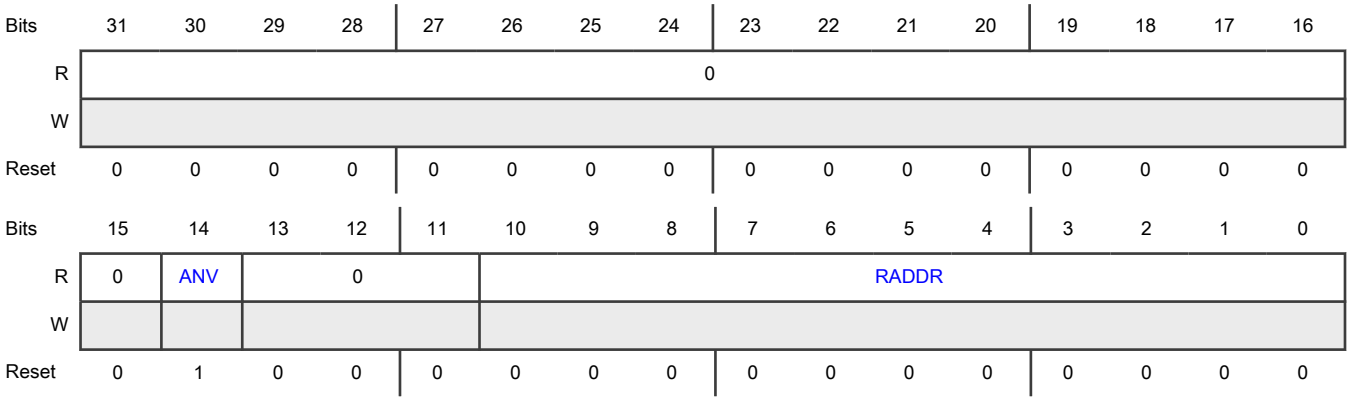
Offset

Register	Offset
SASR	150h

Function

Contains the received address and its validity.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 ANV	Address Not Valid Indicates whether SASR[RADDR] is valid. 0b - Valid 1b - Not valid
13-11 —	Reserved
10-0 RADDR	Received Address Contains the received address. Updates whenever SSR[AM0F] or SSR[AM1F] is set. Reading Target Address Status (SASR) clears SSR[AM0F] and SSR[AM1F]. 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 Read-or-Write bit is therefore always stored in RADDR[0]. When SCFGR1[ACKSTALL] = 1, if the first address byte matches in 10-bit mode, the first address byte is stored in RADDR[7:0] so you can read this field before writing the Transmit ACK. If the second address byte matches, this field is then updated with the full 10-bit address.

38.7.1.29 Target Transmit ACK (STAR)

Offset

Register	Offset
STAR	154h

Function

Configures choice of ACK or NACK on each received word.

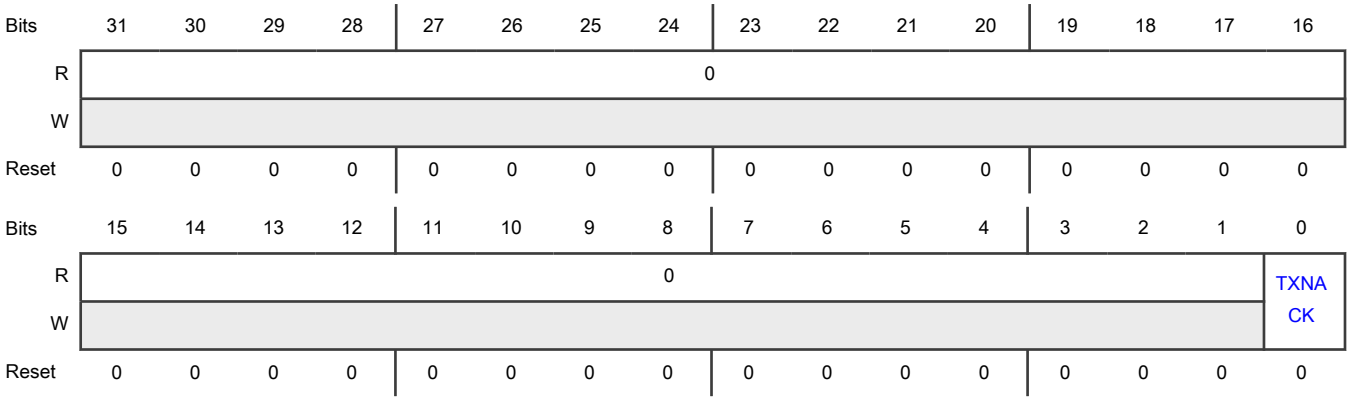
You can write to this register only when SCFGR1[ACKSTALL] = 1.

SCFGR1[ACKSTALL] enables clock stretching during the ACK-or-NACK bit slot. During this time, you can write to this register.

The logic ensures that the clock stretching continues for at least one bus clock cycle after this register is updated.

This clock stretching time can be extended via SCFGR2[CLKHOLD].

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK Selects whether transmit ACK (logic 0) or NACK (logic 1) is returned on the bus by the I2C target after receiving each word. • When SCFGR1[ACKSTALL] = 1, a transmit NACK signal must be written once for each matching address byte and each received word. SCFGR1[ACKSTALL] must be 1, because that setting stalls the data transfer until software reads the received word (and determines whether to respond with an ACK or NACK). • To configure the default (ACK or NACK), you can write to this field when LPI2C target is disabled or idle. 0b - Transmit ACK 1b - Transmit NACK

38.7.1.30 Target Transmit Data (STDR)

Offset

Register	Offset
STDR	160h

Function

Contains the I2C target data to transmit.

Clock stretching (enabled or disabled) affects when the transmit data is transferred. SCFGR1[TXDSTALL] enables clock stretching during the first data bit of a target-transmit transfer.

If clock stretching is enabled (SCFGR1[TXDSTALL] = 1), the transmit data transfer is stalled until this register is updated. Clock stretching is extended by at least 1 bus clock cycle after this register is updated. Clock stretching can be delayed further by using SCFGR2[CLKHOLD].

If clock stretching is disabled (SCFGR1[TXDSTALL] = 0), the transmit data must be written before the start of the target-transmit transfer, otherwise SSR[FEF] is set.

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 Contains the I2C target data to transmit. Writing data to this register stores I2C target transmit data in this register.

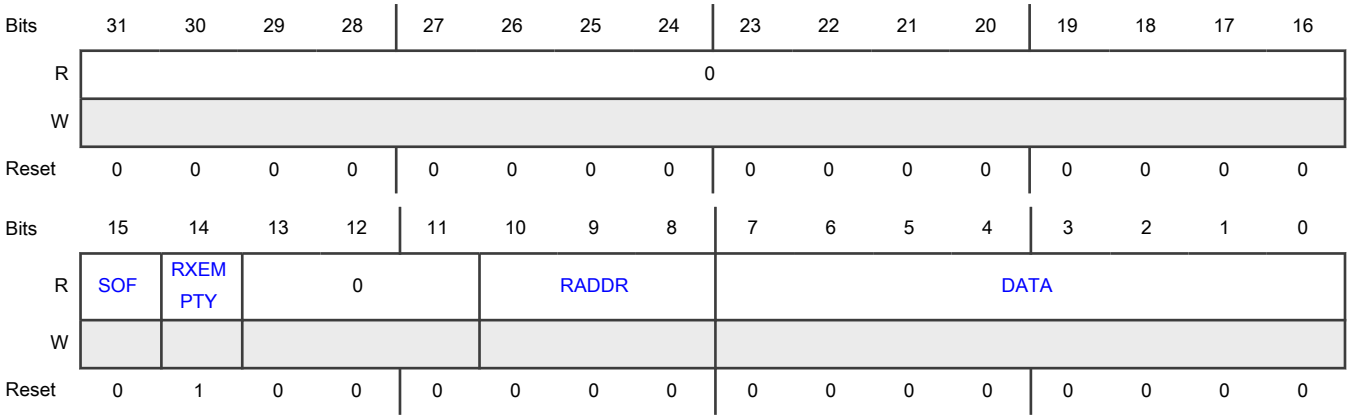
38.7.1.31 Target Receive Data (SRDR)

Offset

Register	Offset
SRDR	170h

Function
Contains status of target receive data transfer.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 SOF	Start of Frame Indicates whether this data word is the first data word since a (repeated) Start or Stop condition. 0b - Not first 1b - First
14 RXEMPTY	Receive Empty Indicates whether this register is empty. 0b - Not empty 1b - Empty
13-11 —	Reserved
10-8	Received Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
RADDR	Contains the address received by the IC2 target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [10:8] of SASR[RADDR] are returned. Otherwise, this field returns zero.
7-0 DATA	Received Data Contains the data received by the I2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [7:0] of SASR[RADDR] are returned.

38.7.1.32 Target Receive Data Read Only (SRDROR)

Offset

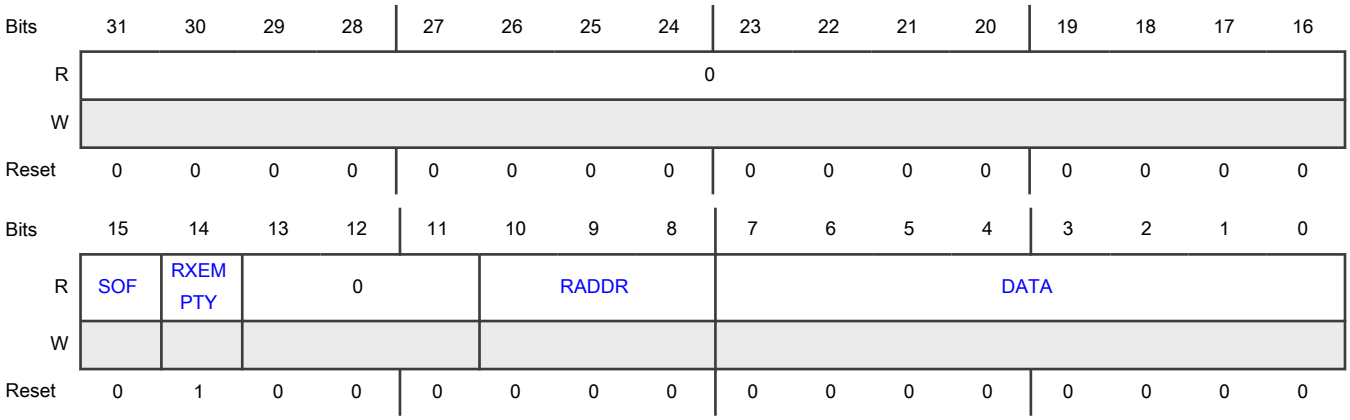
Register	Offset
SRDROR	178h

Function

Contains the data received by the I2C target.

Reading this 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 Indicates whether this data word is the first data word since a (repeated) Start or Stop condition.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not the first 1b - First
14 RXEMPTY	Receive Empty Indicates whether Target Receive Data (SRDR) is empty. 0b - Not empty 1b - Empty
13-11 —	Reserved
10-8 RADDR	Received Address Contains address received by the LPI2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [10:8] of SASR[RADDR] are returned. Otherwise, this field returns zero.
7-0 DATA	Receive Data Contains data received by the LPI2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [7:0] of SASR[RADDR] are returned.

Chapter 39
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

39.1 Chip-specific LPUART information

Table 286. Reference links to related information

Topic	Related module	Reference
Full description	LPUART	LPUART
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

39.1.1 Module instances

This device supports five instances of LPUART: LPUART0, LPUART1, LPUART2, LPUART3 and LPUART4.

39.2 Overview

LPUART provides asynchronous, serial communication capabilities with external devices. It supports the non-return-to-zero (NRZ) encoding format and infrared data association (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.

39.2.1 Block diagram

Figure 167 shows the transmitter portion of LPUART.

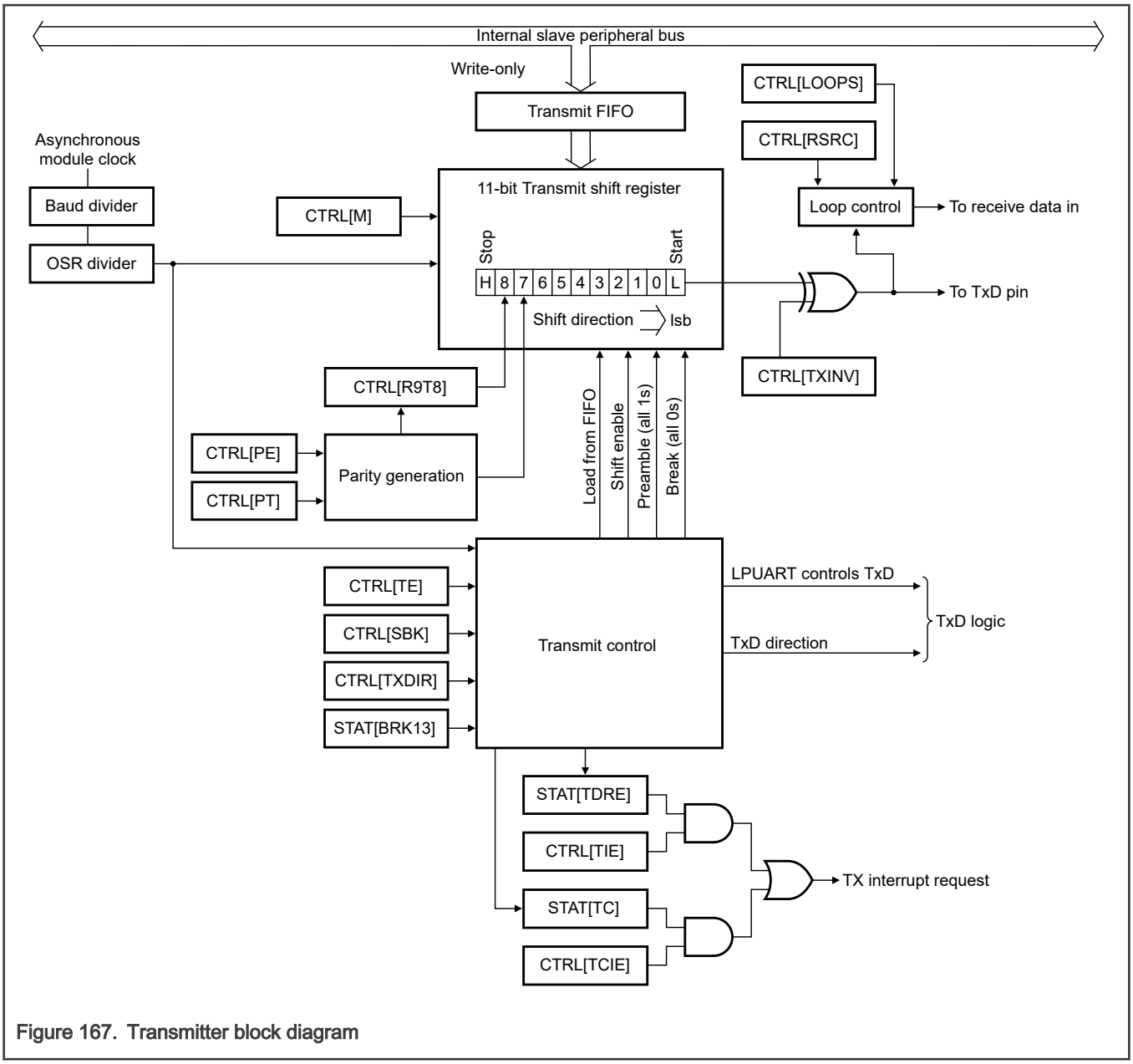


Figure 167. Transmitter block diagram

Figure 168 shows the receiver portion of LPUART.

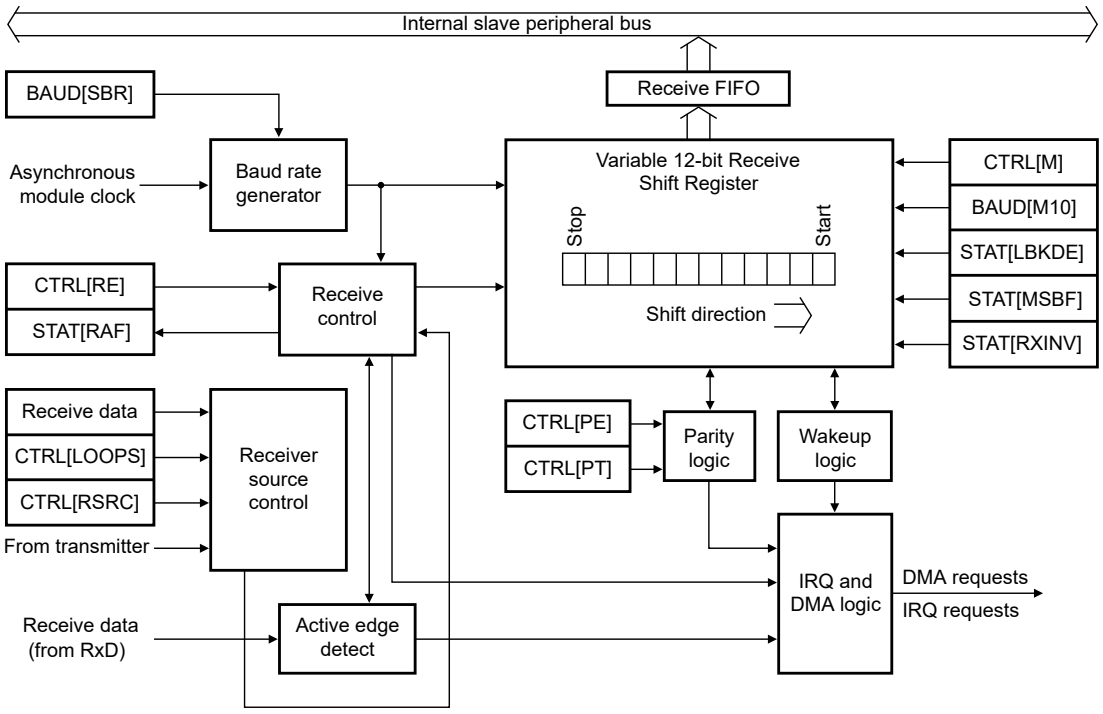


Figure 168. Receiver block diagram

39.2.2 Features

- Full-duplex, standard NRZ format
- Programmable baud rates (13-bit modulo divider) with a configurable oversampling ratio (OSR) from 4× to 32×
- Asynchronous operation 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 wake-up methods:
  - Idle line wake-up

  - Address mark wake-up
  - 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

39.3 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.

39.3.1 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.

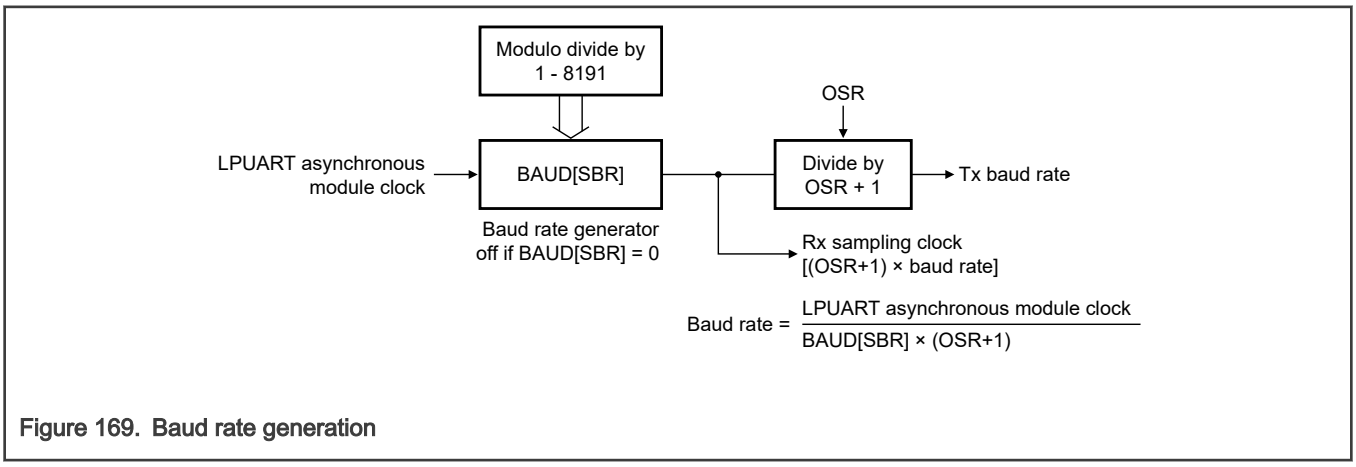


Figure 169. Baud rate generation

Baud rate generation is subject to these 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 (see Data sampling technique) are also closer together, which may impact noise sensitivity.

39.3.2 Baud rate tolerance

A transmitting device may operate at a baud rate below or above that of the receiver.

Accumulated bit time misalignment can cause one of the three stop bit data samples to fall outside the actual stop bit. A noise error will occur if the three samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the three stop bit samples are a logic zero.

As the receiver samples an incoming frame, it may re-synchronize the oversampling clock on any valid falling edge within the frame. Resynchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

In general, increasing the number of samples per bit will increase the baud rate tolerance and decreasing the number of samples per bit will reduce the baud rate tolerance. Note that since LPUART implements triple voting on consecutive receive data samples, increasing the number of samples per bit will move those samples closer together which would reduce the width of noise that can be filtered by the triple voting logic.

39.3.3 Calculating baud rate tolerance

Using the following definitions:

- SAM is the number of sample points per bit (valid range from 8 to 32; equal to (OSR + 1) × (BOTHEDGE + 1)).
- BIT is the number of bits in a character including start, data and stop bits (valid range from 9 to 13).

The ideal baud rate tolerance can be calculated as follows:

- Slow data rate tolerance = ((SAM ÷ 2) - 1) ÷ ((SAM × BIT) - (SAM ÷ 2) + 2)
- Fast data rate tolerance = ((SAM ÷ 2) - 2) ÷ (SAM × BIT)

As an example, if configured for 8-bit data, 1 stop bit (BIT = 10) and with OSR=0x7 and BOTHEDGE = 1 (SAM = 16):

- Slow data rate tolerance = (8 - 1) ÷ (160 - 8 + 2) = 7 ÷ 154 = 4.54%
- Fast data rate tolerance = (8 - 2) ÷ 160 = 6 ÷ 160 = 3.75%

If configured for 9-bit data with 1 stop bit (BIT = 11) with same oversampling configuration, then:

- Slow data rate tolerance = (8 - 1) ÷ (176 - 8 + 2) = 7 ÷ 170 = 4.12%
- Fast data rate tolerance = (8 - 2) ÷ 176 = 6 ÷ 176 = 3.41%

NOTE

Additional factors can contribute to a lower baud rate tolerance than the ideal. These include clock uncertainty or jitter on the LPUART functional clock source, differences in rise and fall times on the transmitter output and synchronization of the external receive pin to the local LPUART functional clock.

39.3.4 Transmitter functional description

This section describes 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 and 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 transmit FIFO 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.

39.3.4.1 Break character length

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 wake-up 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, write 0 and then write 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 287. 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

Table continues on the next page...

Table 287. Break character length (continued)

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
1	0	0	1	1	12
1	1	0	—	0	14
1	1	0	—	1	14
1	—	1	—	0	15
1	—	1	—	1	15

39.3.4.2 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 CTS.

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.

39.3.4.3 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.

39.3.4.4 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 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.

The following figure shows the receiver enable signal asserted. This connection can also connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled when driving. A pullup can pull RXD to a nonfloating value during this time. You can refine this option further by operating LPUART in Single-Wire mode, freeing the RXD pin for other uses.

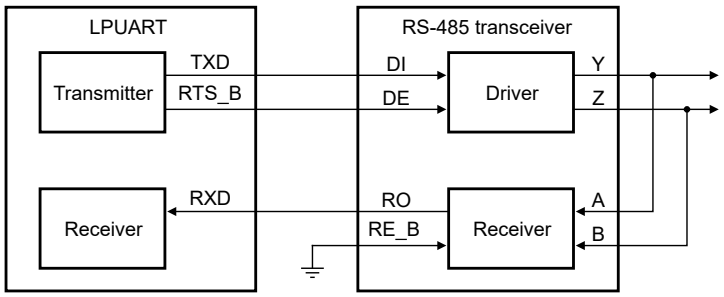


Figure 170. Transceiver driver enable using RTS_B

39.3.5 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 wake-up 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, along with N (7, 8, 9, 10) 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 the receive data register is not already full (STAT[RDRF] = 0), the data character is transferred to the receive FIFO, resulting in STAT[RDRF] 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 the receive data register is full (STAT[RDRF] = 1), it gets the data from the FIFO by reading Data (DATA). See Interrupts for details about flag clearing.

39.3.5.1 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 they are 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, noise flag (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" (one to zero) 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 synchronization to the start bit is termed as resynchronizarion.

When BAUD[RESYNCDIS] is 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] is 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 288. 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

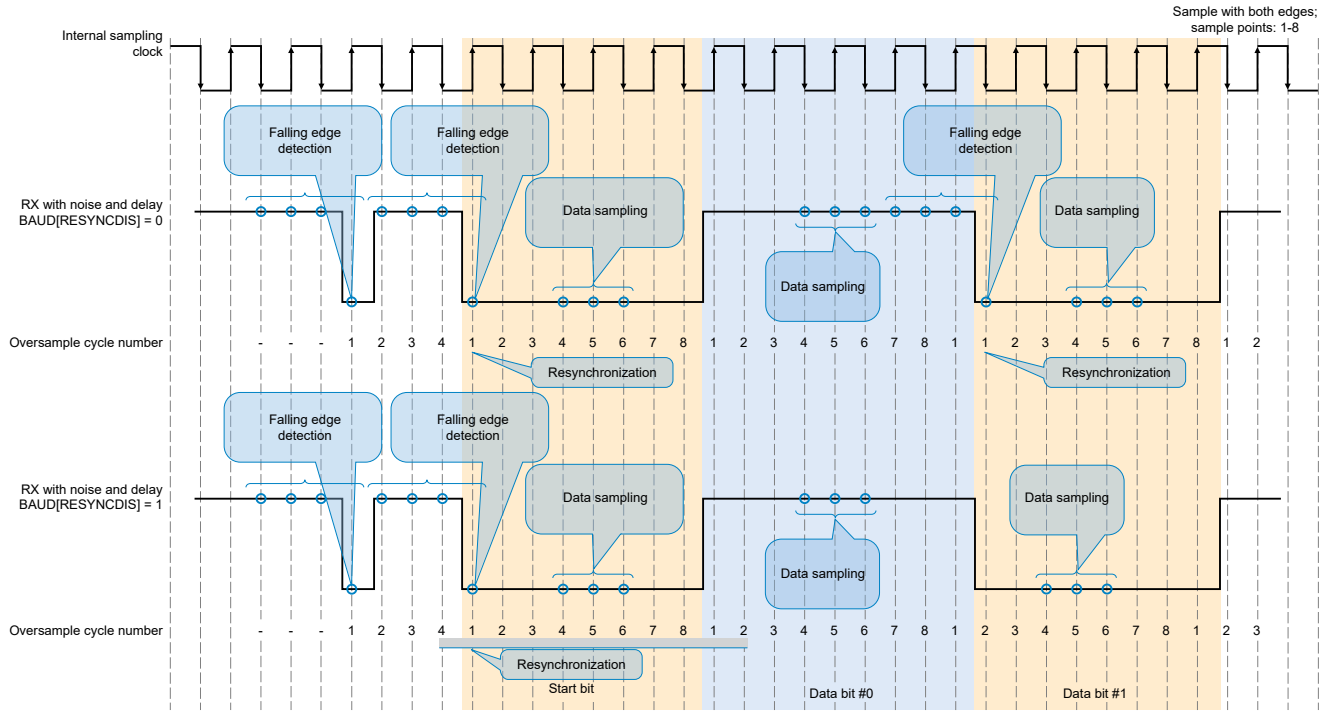


Figure 171. LPUART resynchronization diagram

39.3.5.2 Receiver wake-up operation

Receiver wake-up 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 wake-up, 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 289. Receiver wake-up options

CTRL[RWU]	BAUD[MAEN1] | BAUD[MAEN2]	BAUD[MATCFG]	CTRL[WAKE]: STAT[RWUID]	Receiver wake-up
0	0	X	X	Normal operation
1	0	00	00	Receiver wake-up on idle line; STAT[IDLE] = 0
1	0	00	01	Receiver wake-up on idle line; STAT[IDLE] = 1
1	0	00	10	Receiver wake-up on address mark
1	1	11	10	Receiver wake-up 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

39.3.5.2.1 Idle line wake-up

When CTRL[WAKE] is 0, you can configure the receiver for an idle line wake-up. 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] is 1 and STAT[RWUID] is 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] is 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.

These are the ways to detect an idle line:

- When CTRL[ILT] is 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] is 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.

39.3.5.2.2 Address mark wake-up

When CTRL[WAKE] is 1, you can configure the receiver for an address mark wake-up. 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 wake-up.

Address mark wake-up 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.

39.3.5.2.3 Data match wake-up

When CTRL[RWU] and CTRL[WAKE] are 1, and BAUD[MATCFG] equals 11, the receiver is configured for a data match wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a character that matches MATCH[MA1] when BAUD[MAEN1] is 1, or that matches MATCH[MA2] when BAUD[MAEN2] is 1.

39.3.5.2.4 Address match operation

You can enable the address match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 0. In this function, a character that the RXD pin receives with a logic 1 in the most significant bit (or the second most significant bit when parity is enabled) is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. 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 the 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) 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.

39.3.5.2.5 Idle match operation

You can enable the idle match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 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]. 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) 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.

39.3.5.2.6 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.

39.3.5.3 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] is 1 or a start bit is detected that causes STAT[RDRF] to become 1.
- The receiver asserts RTS_B when STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. There is no impact if STAT[RDRF] is 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.

39.3.6 Additional LPUART functions

39.3.6.1 Data modes

You can configure the LPUART transmitter and receiver 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 you must read them 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 wake-up so that the ninth data bit can serve as the wake-up bit. The 10-bit Data mode is typically used with parity and address mark wake-up so that the ninth data bit can serve as the wake-up 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.

39.3.6.2 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 wake-up 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.

39.3.6.3 Loop mode

Enable Loop mode by setting CTRL[LOOPS] = 1 and CTRL[RSRC] = 0. 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.

Loop mode also internally connects the RTS_B output to the CTS_B input.

39.3.6.4 Single-Wire mode

Enable Single-Wire mode by setting CTRL[LOOPS] = 1 and CTRL[RSRC] = 1. Single-Wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and 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.

39.3.7 Peripheral triggers

The connection of the LPUART peripheral triggers with other peripherals is chip-specific.

39.3.7.1 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, and negates when the next valid start bit is detected.

39.3.7.2 Input trigger

LPUART supports a 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.

39.3.8 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.

39.3.8.1 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] is 0, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] is 1.

39.3.8.2 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] is 0, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] is 1. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

39.3.8.3 Start-bit detection

When STAT[RXINV] is 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.

39.3.8.4 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.

39.3.8.5 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.

39.3.8.6 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.

39.3.9 Modes of operation

39.3.9.1 Low-Power modes

LPUART remains functional during low-power modes, provided CTRL[DOZEEN] is 0 and the LPUART functional clock is enabled. LPUART can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPUART to be disabled in low-power modes, when CTRL[DOZEEN] is 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 wake-up 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.

39.3.9.2 Debug mode

LPUART remains functional in Debug mode.

39.3.10 Clocking

Table 290. Types of clocks

Clock	Description
Functional	Is asynchronous to the bus clock and can remain enabled in Low-Power modes to support transmit and/or receive functions, including low-power wake-up.
Bus	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.

39.3.11 Reset

Table 291. Types of resets

Reset	Description
Chip	Enables the logic and registers for the LPUART transmitter and receiver to reset to their default states.
Software	Resets the LPUART logic and registers to their default states, except for Global (GLOBAL). GLOBAL[RST] controls the LPUART software reset.
FIFO	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.

39.3.12 Interrupts

The LPUART transmitter has two status fields that can optionally generate hardware interrupt requests. If STAT[TDRE] is 1, it indicates that there is room in the transmit FIFO to write another transmit character to Data (DATA). If CTRL[TIE] is 1, a hardware interrupt is requested when STAT[TDRE] is 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] is 1, a hardware interrupt is requested when STAT[TC] is 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] field is 0.

When a program detects that STAT[RDRF] is 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 1 as the value of STAT[RDRF].

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 getting 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 (the value of CTRL[RE] being 1).

39.3.13 DMA

39.3.13.1 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. 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 the 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.

39.4 External signals

Table 292. 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 the 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

39.5 Initialization

This module does not require initialization.

39.6 LPUART register descriptions

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. LPUART does not verify whether programmed values in the registers are correct; you must write valid values to them.

39.6.1 LPUART memory map

LPUART0 base address: 4009_F000h

LPUART1 base address: 400A_0000h

LPUART2 base address: 400A_1000h

LPUART3 base address: 400A_2000h

LPUART4 base address: 400A_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0404_0003h
4h	Parameter (PARAM)	32	R	0000_0202h
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

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1Ch	Data (DATA)	32	RW	0000_1000h
20h	Match Address (MATCH)	32	RW	0000_0000h
24h	MODEM IrDA (MODIR)	32	RW	0000_0000h
28h	FIFO (FIFO)	32	RW	00C0_0011h
2Ch	Watermark (WATER)	32	RW	0000_0000h
30h	Data Read-Only (DATARO)	32	R	0000_1000h

39.6.2 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	1	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	1	1

Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number Indicates the feature set number. 0000_0000_0000_0001b - Standard feature set 0000_0000_0000_0011b - Standard feature set with MODEM and IrDA support

39.6.3 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	0	0	0	0	0	0	0	1	0

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 TXFIFO	Transmit FIFO Size Indicates the number of characters in the transmit FIFO, which is 2^TXFIFO.

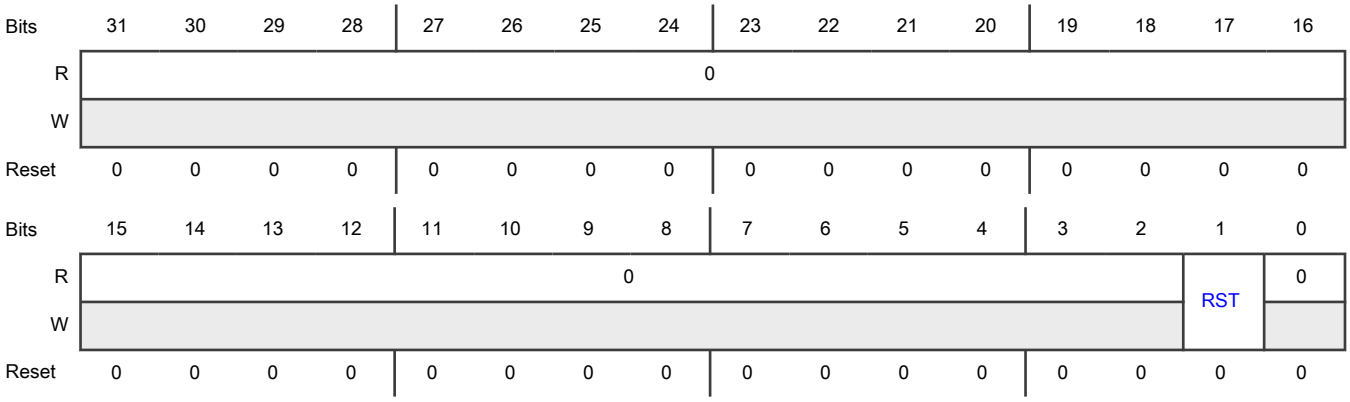
39.6.4 Global (GLOBAL)

Offset

Register	Offset
GLOBAL	8h

Function
Performs global functions.

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

39.6.5 Pin Configuration (PINCFG)

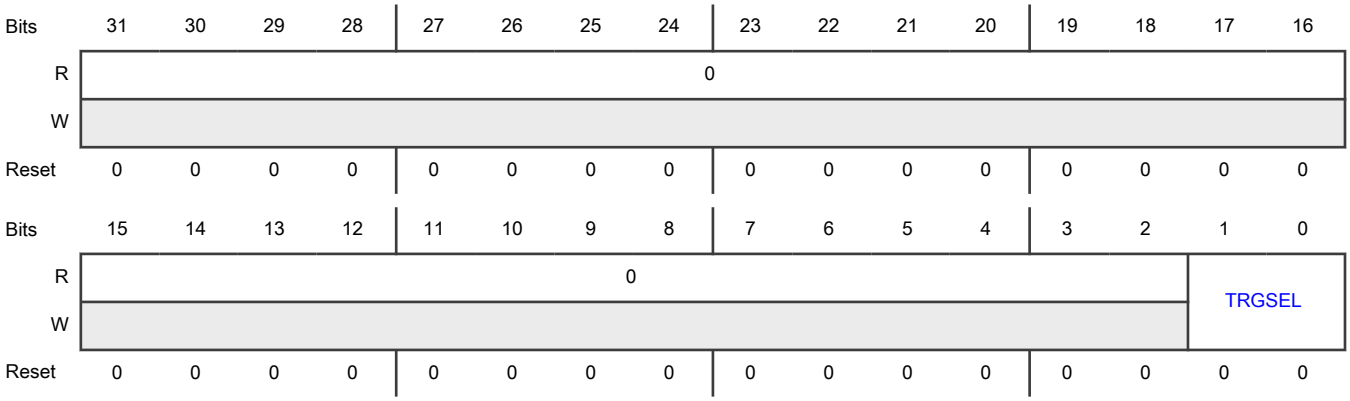
Offset

Register	Offset
PINCFG	Ch

Function

Enables the selection of input pins.

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

39.6.6 Baud Rate (BAUD)

Offset

Register	Offset
BAUD	10h

Function

Configures the baud rate.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAEN1	MAEN2	M10	OSR					TDMAE	0	RDMAE	RIDMAE	MATCFG		BOTHEDGE	RESYNCD...
W																
Reset	0	0	0	0	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	LBKDIE	RXEDGIE	SBNS	SBR												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 Enables automatic address matching or data matching mode for MATCH[MA1]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
30 MAEN2	Match Address Mode Enable 2 Enables automatic address matching or data matching mode for MATCH[MA2]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
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

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 OSR	Oversampling Ratio 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] results in an OSR of BAUD[OSR] + 1, for example, BAUD[OSR] = 0_0101b results in a final division by 6. 0_0000b - Results in an OSR of 16 0_0001b - Reserved 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 Enables STAT[TDRE] to generate a DMA request. 0b - Disable 1b - Enable
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable Enables STAT[RDRF] to generate a DMA request. 0b - Disable 1b - Enable
20 RIDMAE	Receiver Idle DMA Enable Enables STAT[IDLE] to generate a DMA request. If this field is 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 - Disable 1b - Enable
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 wake-up 01b - Idle match wake-up 10b - Match on and match off 11b - Enables RWU on data match and match on or off for the transmitter CTS input
17 BOTHEDGE	Both Edge Sampling

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. If this field is 0, the receiver samples input data using the rising edge of the baud rate clock. If this field is 1, the receiver samples input data using the rising and falling edges of the baud rate clock. 0b - Rising edge 1b - Both rising and falling edges
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. 0b - Enable 1b - Disable
15 LBKDIE	LIN Break Detect Interrupt Enable Enables STAT[LBKDIF] to generate hardware interrupt requests. If this field is 0, hardware interrupts from STAT[LBKDIF] (uses polling) are disabled. If this field is 1, hardware interrupts are requested when STAT[LBKDIF] is 1. 0b - Disable 1b - Enable
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables STAT[RXEDGIF] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[RXEDGIF] are disabled. If this field is 1, hardware interrupts are requested when STAT[RXEDGIF] is 1. Changing the value of CTRL[LOOPS] or CTRL[RSRC] when this field (RXEDGIE) is 1 can cause STAT[RXEDGIF] to become 1. 0b - Disable 1b - Enable
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	Baud Rate Modulo Divisor

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBR	Sets the modulo divide rate for the baud rate generator. • If SBR is 0, baud rate generator is disabled. • 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).

39.6.7 Status (STAT)

Offset

Register	Offset
STAT	14h

Function
Provides the module status.

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 Indicates 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. **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
	0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag Indicates 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] is 0; rising if STAT[RXINV] is 1) on the RXD pin occurs. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not occurred 1b - Occurred When writing 0b - No effect 1b - Clear the flag
29 MSBF	MSB First Specifies the first bit that is transmitted after the start bit. If this field is 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 this field is 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 RXINV	Receive Data Inversion 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 wake-up, 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 this field is 0, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] does not become 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] does not become 1 when an address does not match. If this field is 1, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] becomes 1 upon detection of an idle character. During address match wake-up, 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] is 1 and DATA[R9T9] is 0. 0b - 9 to 13 bit times 1b - 12 to 15 bit times
25 LBKDE	LIN Break Detection Enable Enables LIN break detection. If this field is 0, LIN break detect is disabled, and only a normal break character can be detected. If 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] is 0), 12 bit times (if CTRL[M] is 1), or 13 bit times (if BAUD[M10] is 1). This field selects a longer break character detection length. When the field is 1, receive data is not stored in the receive FIFO. **NOTE** This field enables the LIN break detect circuit and disables writing receive data to FIFO. Therefore, it ignores all characters except a LIN break.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
24 RAF	Receiver Active Flag Indicates 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 Indicates 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 again, 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 Indicates whether the transmitter is active. This field becomes 0 when a transmission is in progress or a preamble or break character is loaded; in other words, when the transmitter is active (sending data, a preamble, or a break). The field becomes 1 when the transmit buffer is empty and no data, preamble, or break character is being transmitted; in other words, when the transmission activity is complete. 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 1b - Transmitter idle
21 RDRF	Receive Data Register Full Flag Indicates 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 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).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 - Equal to or less than watermark 1b - Greater than watermark
20 IDLE	Idle Line Flag Indicates 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] is 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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Idle line detected 1b - Idle line not detected When writing 0b - No effect 1b - Clear the flag
19 OR	Receiver Overrun Flag Indicates 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. **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
	0b - No overrun 1b - Receive overrun (new LPUART data is lost) When writing 0b - No effect 1b - Clear the flag
18 NF	Noise Flag Indicates 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 some of these samples disagree 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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No noise detected 1b - Noise detected When writing 0b - No effect 1b - Clear the flag
17 FE	Framing Error Flag Indicates 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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No framing error detected (this does not guarantee that the framing is correct) 1b - Framing error detected When writing 0b - No effect 1b - Clear the flag
16 PF	Parity Error Flag Indicates whether a parity error is detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field becomes 1 whenever the next character to be read from Data (DATA) is received when parity is enabled (CTRL[PE] is 1) and the parity bit in the received character does not agree with the expected parity value. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No parity error detected 1b - Parity error detected When writing 0b - No effect 1b - Clear the flag
15 MA1F	Match 1 Flag Indicates 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]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA1 1b - Equal to MA1 When writing 0b - No effect 1b - Clear the flag
14 MA2F	Match 2 Flag Indicates 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]. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA2 1b - Equal to MA2 When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
13-10 —	Reserved
9-8 —	Reserved
7-2 —	Reserved
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 this field is 0, address mark in character is MSB. If this field is 1, the address mark is stored in Data (DATA) at MSB (or MSB-1 when the parity bit is enabled). In other words, the address mark in character is the last bit before the stop bit (or parity bit when enabled). 0b - Disable 1b - Enable
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] as 1. The character following the LIN break has DATA[LINBRK] as 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 this field is 1, the LIN break character is detected at a length of 11-bit times (if CTRL[M] is 0), 12 (if CTRL[M] is 1), or 13 (if BAUD[M10] is 1). 0b - Disable 1b - Enable

39.6.8 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	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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29 TXDIR	TXD Pin Direction in Single-Wire Mode Determines the direction of data at the TXD pin, in Single-Wire mode, when LPUART is configured for a single-wire half-duplex operation (CTRL[LOOPS] and CTRL[RSRC] are 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. 0b - Input 1b - Output
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 - Disable 1b - Enable
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 - Disable 1b - Enable
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 - Disable 1b - Enable
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 - Disable 1b - Enable
23	Transmit Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TIE	Enables STAT[TDRE] to generate interrupt requests if STAT[TDRE] is 1. 0b - Disable 1b - Enable
22 TCIE	Transmission Complete Interrupt Enable Enables STAT[TC] to generate interrupt requests if STAT[TC] is 1. 0b - Disable 1b - Enable
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate hardware interrupt requests if STAT[RDRF] is 1. 0b - Disable 1b - Enable
20 ILIE	Idle Line Interrupt Enable Enables hardware interrupts. This field enables STAT[IDLE] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[IDLE] are disabled and polling is used, and if this field is 1, hardware interrupts are enabled when STAT[IDLE] is 1. 0b - Disable 1b - Enable
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] and DATA[R9T9] = 1. 0b - Disable 1b - Enable
18 RE	Receiver Enable Enables the LPUART receiver. After you write 0 to this field, this field remains 1 until the receiver finishes receiving the current character (if any). 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 RWU	Receiver Wake-Up Control Specifies whether the LPUART receiver in standby is waiting for a wake-up 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] is 0 or an address match when CTRL[WAKE] is 1 and STAT[RWUID] is 0. **NOTE** You must write 1 to this field only when CTRL[WAKE] is 0 (wake-up 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 condition is detected before the IDLE flag is allowed to be reasserted. 0b - Normal receiver operation 1b - LPUART receiver in standby, waiting for a wake-up condition
16 SBK	Send Break Specifies whether queue break character(s) are to be sent. Writing 1 and then 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] is 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] is 1 and DATA[R9T9] is 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 - Disable 1b - Enable
14 MA2IE	Match 2 (MA2F) Interrupt Enable Enables the MA2F interrupt. 0b - Disable 1b - Enable
13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
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 1b - 7-bit
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	Doze Mode Enables LPUART in Doze mode. If this field is 1, LPUART remains active when not in Doze mode. 0b - Enable 1b - Disable
5 RSRC	Receiver Source Select Determines the source of the receiver shift register input if CTRL[LOOPS] is 1. This field has no effect unless CTRL[LOOPS] is 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 0, internal Loopback mode is selected. LPUART does not use the RXD pin. Additionally, the CTS_B pin is not used and internally driven by the RTS_B output. If 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 data characters that the receiver and transmitter use. 0b - 8-bit 1b - 9-bit
3 WAKE	Receiver Wake-Up 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 If this field is 0, CTRL[RWU] is configured for idle line wake-up, and if this field is 1, CTRL[RWU] is configured with address mark wake-up. 0b - Idle 1b - Mark
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 - Disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enable
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

39.6.9 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 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 Specifies 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 - Valid data 1b - Invalid data and empty
11 IDLINE	Idle Line Indicates whether the receiver line was idle before receiving the character in DATA[9:0]. It can be read as “1” on the first character when the receiver is first enabled. The difference between this field and STAT[IDLE] is that, STAT[IDLE] flag does not set on an idle line after the receiver is first enabled, it needs to receive a character before it can become set, whereas this field does not have this limitation and can be set on the first character received if an idle line is detected beforehand. 0b - Not idle 1b - Idle
10	LIN Break

Table continues on the next page...

Table continued from the previous page...

Field	Function
LINBRK	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. If this field is 0, the LIN break detect circuitry is disabled. 0b - Not detected 1b - 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 R7T7	Read receive FIFO bit 7 or write transmit FIFO bit 7
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

39.6.10 Match Address (MATCH)

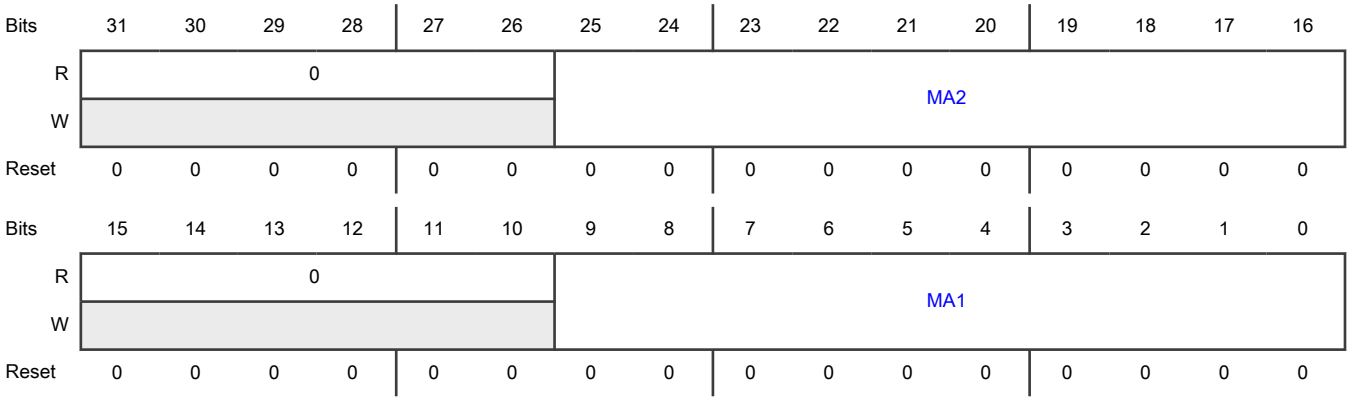
Offset

Register	Offset
MATCH	20h

Function

Provides addresses for address matching during the receiver operation.

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 field in Baud Rate (BAUD) is 0.

39.6.11 MODEM IrDA (MODIR)

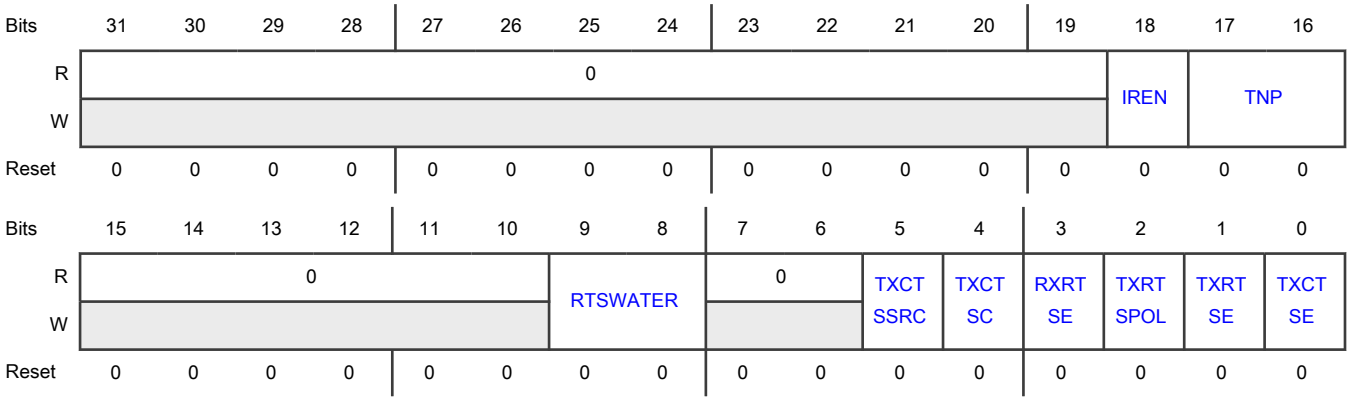
Offset

Register	Offset
MODIR	24h

Function

Controls options for setting the MODEM configuration.

Diagram



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 - Disable 1b - Enable
17-16 TNP	Transmitter Narrow Pulse Specifies 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
15-10 —	Reserved
9-8	Receive RTS Configuration

Table continues on the next page...

Table continued from the previous page...

Field	Function
RTSWATER	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
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 this field is 0, the receiver has no effect on RTS. If this field is 1, RTS is deasserted if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1. RTS is asserted if STAT[RDRF] is 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 - Disable 1b - Enable
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] is 1. You must change the value of this field only when the transmitter is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Active low 1b - 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. If this field is 0, the transmitter has no effect on RTS, and if this field is 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. 0b - Disable 1b - Enable
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 this field is 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 TXD signal remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS, when a character is being sent, do not affect its transmission. 0b - Disable 1b - Enable

39.6.12 FIFO (FIFO)

Offset

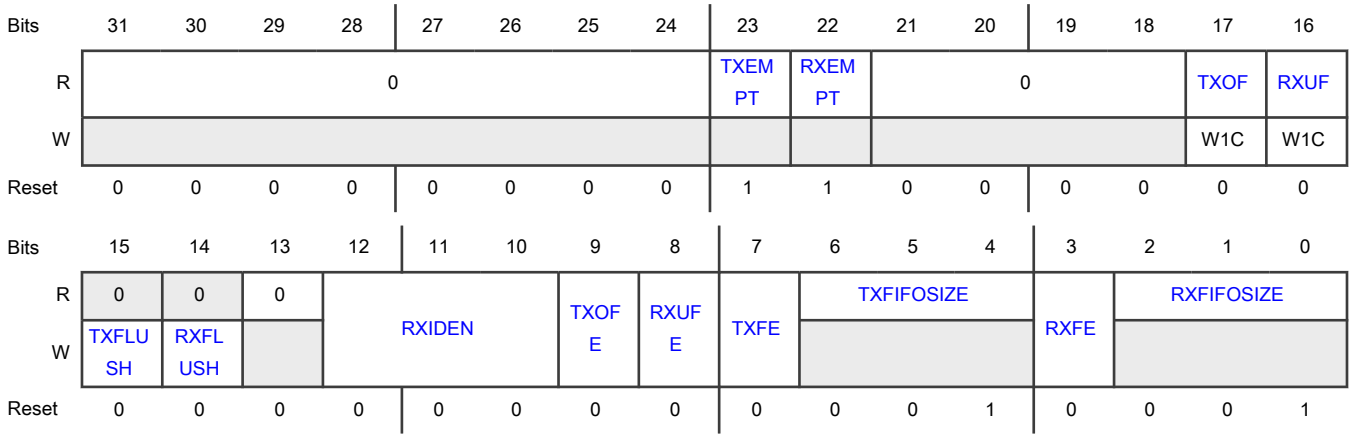
Register	Offset
FIFO	28h

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



Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit FIFO Or Buffer Empty Indicates whether the transmit buffer is empty. 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 Indicates 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 Indicates whether more data has been written to the transmit FIFO than it can hold. If this field is 0, no transmit FIFO overflow has occurred since the last time the field was cleared, and if this field is 1, at least one 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] is 1.

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 - No overflow 1b - Overflow When writing 0b - No effect 1b - Clear the flag
16 RXUF	Receiver FIFO Underflow Flag Indicates whether more data has been read from the receive FIFO than was present. If this field is 0, no receive FIFO underflow has occurred since the last time the field was cleared, and if this field is 1, at least one 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] is 1. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underflow 1b - Underflow When writing 0b - No effect 1b - Clear the flag
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. 0b - No effect 1b - All data 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This does not affect data in the receive shift register. 0b - No effect 1b - All data 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 - Disable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle 001b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for one character 010b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for two characters 011b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for four characters 100b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for eight characters 101b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 16 characters 110b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 32 characters 111b - Enable 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 - Disable 1b - Enable
8 RXUFE	Receive FIFO Underflow Interrupt Enable Enables FIFO[RXUF] to generate an interrupt to the host. 0b - Disable 1b - Enable
7 TXFE	Transmit FIFO Enable Enables the transmit FIFO. If this field is 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is 1, the built-in FIFO structure for the transmit buffer is enabled. FIFO[TXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth Indicates the maximum number of transmit datawords (transmit FIFO buffer depth) that can be stored in the transmit buffer. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256
3 RXFE	Receive FIFO Enable Enables the receive FIFO. If this field 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. If this field is 1, the built-in FIFO structure for the receive buffer is enabled. FIFO[RXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth Indicates the maximum number of receive datawords (receive FIFO buffer depth) that can be stored in the receive buffer before an overrun occurs. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256

39.6.13 Watermark (WATER)

Offset

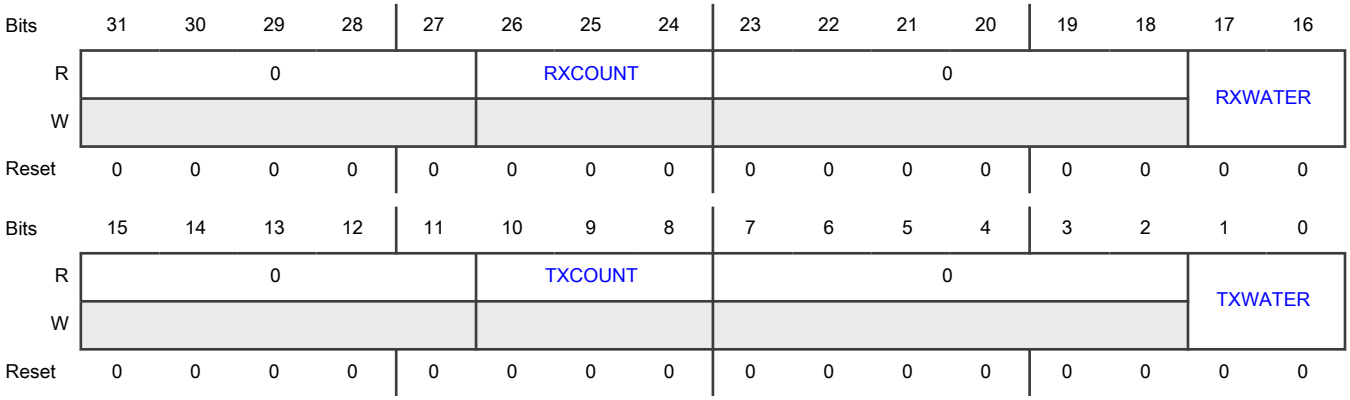
Register	Offset
WATER	2Ch

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] is 0.

Diagram



Fields

Field	Function
31-27 —	Reserved
26-24 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.
23-18 —	Reserved
17-16 RXWATER	Receive Watermark Generates an interrupt or 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].

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-11 —	Reserved
10-8 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.
7-2 —	Reserved
1-0 TXWATER	Transmit Watermark Generates an interrupt or 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].

39.6.14 Data Read-Only (DATARO)

Offset

Register	Offset
DATARO	30h

Function

Indicates the first entry in the receive FIFO, but does not pull data from the FIFO.

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	DATA															
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Receive Data Indicates the first entry from the receive FIFO. This register has the same functionality as that of Data (DATA).

Chapter 40
Low Power Serial Peripheral Interface (LPSPI)

40.1 Chip-specific LPSPI information

Table 293. Reference links to related information

Topic	Related module	Reference
Full description	LPSPI	LPSPI
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

40.1.1 Module instances

This device supports two instances of LPSPI: LPSPI0 and LPSPI1.

40.2 Overview

LPSPI provides an efficient interface (either as a controller or peripheral) to an SPI bus, which 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 with secure digital cards and LCD displays.

NOTE

The terminology in this chapter has been updated to align with NXP's inclusive language standards, as shown in the table below.

Table 294. Updated terms

Updated term	Deprecated term
Controller	Master
Peripheral	Slave

40.2.1 Block diagram

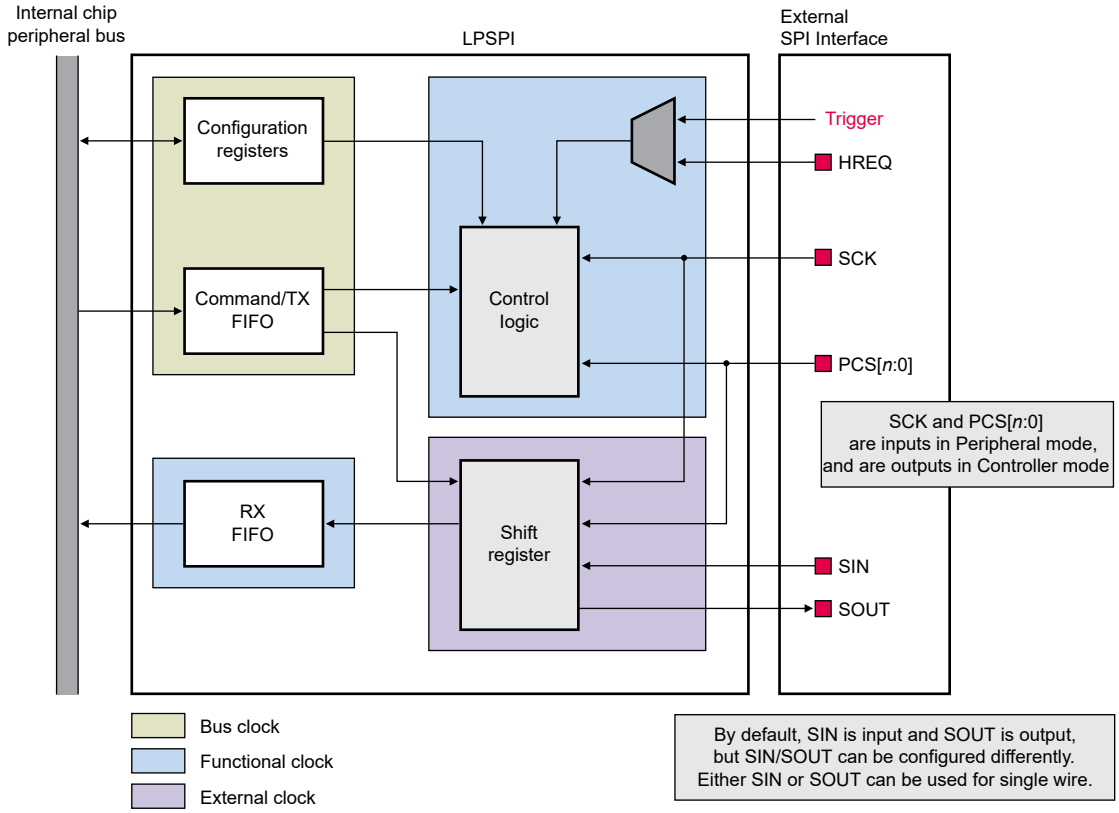


Figure 172. Block diagram

40.2.2 Features

- Minimal CPU overhead, with DMA transmit and receive requests supporting FIFO register accesses
- Operation continues in Deep Sleep mode, if configured to do so and an appropriate clock is available
- Support available for 32-bit word size
- Configurable clock polarity and phase
- Support available for 4 peripheral chip selects in Controller mode
- Support available for Peripheral mode
- 4-word transmit and command FIFO
- 4-word receive FIFO
- Flexible timing parameters in Controller 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 that support 1-bit transmit and receive on each clock edge
- Half-duplex transfers that 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

- Option to use host request to control the start of an SPI bus transfer
- Receive data match logic that discards nonmatching data and interrupt on data match

40.3 Functional description

40.3.1 Controller mode

40.3.1.1 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 in the transmit and command FIFO, by writing to Transmit Data (TDR).
- Command words in 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 on whether LPSPI is busy or between frames (see TCR[CONT] and TCR[CONTC]). See Table 295 for conditions and possible corresponding actions when a command word is at the top of the transmit and command FIFO.

Table 295. 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 this command word controls all subsequent transfers.
LPSPI is busy and 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 value of FRAMESZ), 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.

TCR[CONT] = 1 keeps PCS asserted at end of frame, allowing the transfer to continue.

TCR[CONTC] = 1 specifies that this command word must 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 three LPSPI functional clock cycles for Transmit Command (TCR) to update after you write to it (assuming an empty FIFO), and LPSPI must be enabled (CR[MEN] = 1).

Writing to Transmit Command (TCR) does not initiate an SPI bus transfer, unless TCR[TXMSK] = 1. When TCR[TXMSK] = 1, a new command word is not loaded until the end of the existing frame (based on the value of TCR[FRAMESZ]); at the end of the transfer, TCR[TXMSK] transitions to 0.

In Controller mode, the LPSPI command word in Transmit Command (TCR) controls SPI attributes based on the selections in register fields. See Table 296 for TCR fields and associated functionality related to data transfer.

Table 296. Command word in Controller 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 LPSPI to connect to different peripheral 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 with 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, this field supports changing the command word at 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 in 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 with 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 296. Command word in Controller 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 three words: the first and second words are 32 bits, and the third word is 8 bits.	

40.3.1.1.1 SPI bus transfers

LPSPI initiates an SPI bus transfer when all these 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 the timing parameters defined in Clock Configuration (CCR).

The SPI bus transfer ends after the number of bits indicated by the value of FRAMESZ 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.

40.3.1.1.2 Circular FIFO

The transmit and command FIFO supports a circular FIFO feature. This feature enables the LPSPI controller 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 are not updated. 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 when Circular FIFO mode is enabled.

40.3.1.2 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 the two words in DMR0 and DMR1, or against a masked data word. You can also configure the received data match function to compare only the first one or two received data words since the start of the frame:

- Received data that is already discarded because of TCR[RXMSK] 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 CFGR0[RDMO], and then write 0 to SR[DMF].

40.3.1.3 Timing parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the selection specified in TCR[PRESCALE]. Although you cannot change Clock Configuration (CCR) when LPSPI is busy, to support interfacing with different peripheral devices at different frequencies, you can change the TCR[PRESCALE] selection between SPI bus transfers by using Transmit Command (TCR).

NOTE

The minimum value shown in Table 297 is the minimum counter value, but the values of Clock Configuration (CCR) must also satisfy the data sheet specs based on the LPSPI functional clock frequency and prescaler value.

Table 297. Timing parameters

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Minimum value	Maximum value
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 peripheral 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 297. Timing parameters (continued)

Clock Configuration (CCR) Clock Configuration 1 (CCR1)		Description	Minimum value	Maximum value
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)

Figure 173 shows the timing settings controlled by:

- TCR[CPHA]
- TCR[CPOL]
- CCR[SCKPCS]
- CCR[PCSSCK]
- CCR1[SCKSET]
- CCR1[SCKHLD]

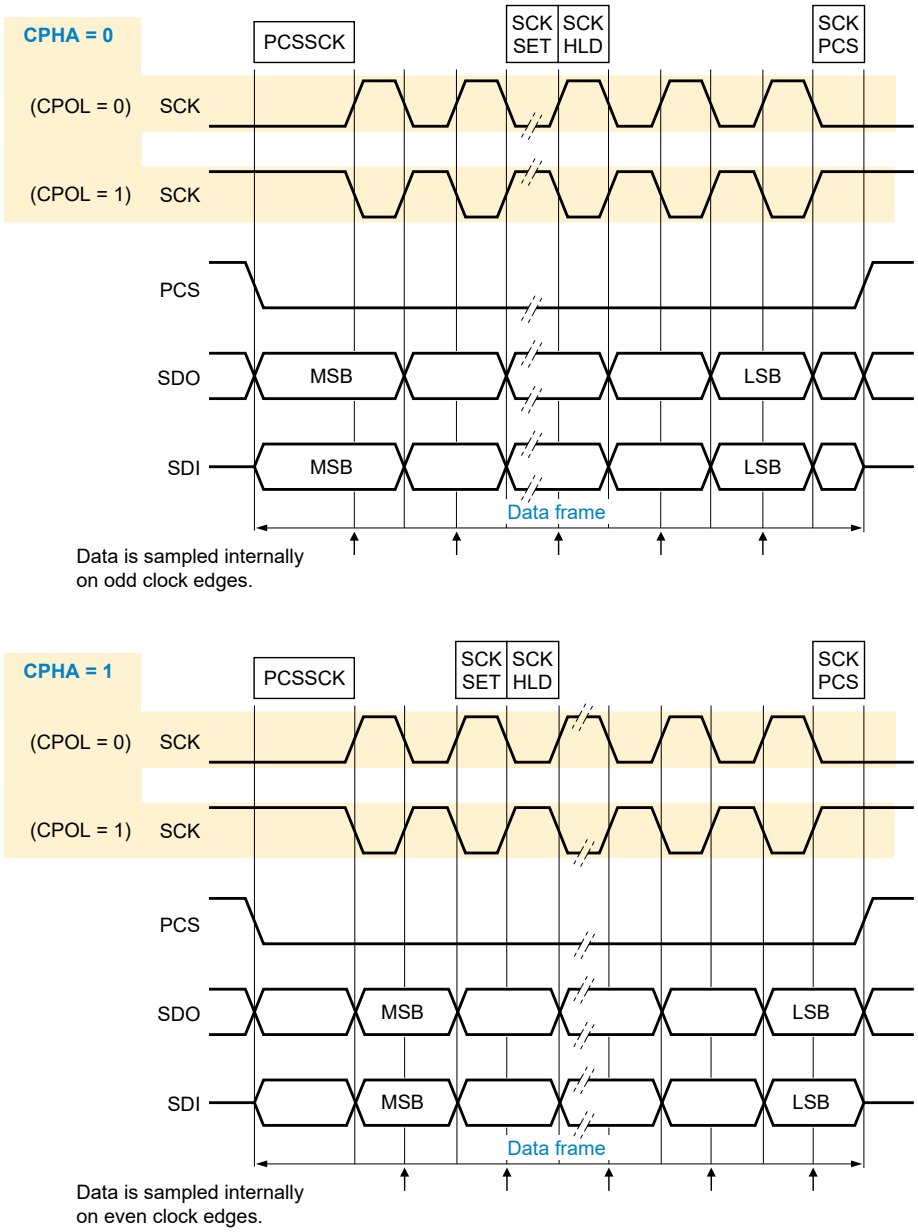


Figure 173. Clock phase (TCR[CPHA]) timing diagram example

To configure for a baud rate of 10 MHz with 50/50 duty cycle and with a functional clock frequency of 100 MHz, use the following settings:

- CCR1[SCKSET] = 0x4 (5 cycles)
- CCR1[SCKHLD] = 0x4 (5 cycles)
- CCR1[PCSPCS] = 0x8 (10 cycles)
- CCR1[SCKSCK] = 0x4 (5 cycles)
- CRR[PCSSCK] = 0x4 (5 cycles)
- CRR[SCKPCS] = 0x4 (5 cycles)
- TCR[PRESCALE] = 0x0 (divide by 1)

40.3.1.4 Pin configuration

Following are the pin configuration settings for half-duplex transfers:

- To swap directions or to 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 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 write any value to CFGR1[PCSCFG].
- When performing half-duplex 4-bit transfers, you must write 1h to CFGR1[PCSCFG].

40.3.1.5 Clock loopback

Configure the LPSPI controller 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 pin can improve the setup time of the input data from the peripheral.

See the chip data sheet for the specific input setup time in Controller Loopback mode.

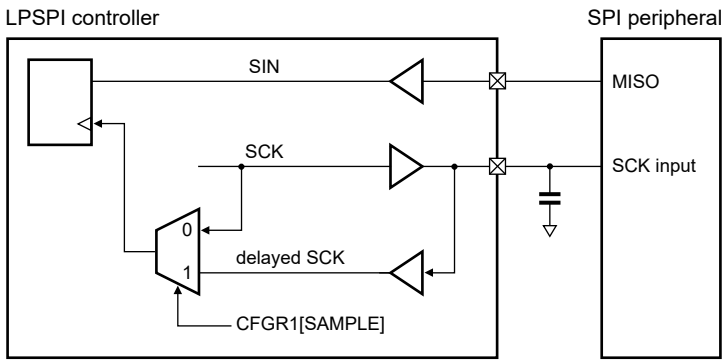


Figure 174. Clock loopback

40.3.2 Peripheral mode

LPSPI Peripheral mode:

- Uses the same shift register and logic that Controller mode uses.
- Does not use Clock Configuration (CCR).
- Requires Transmit Command (TCR) to remain static (unchanged) during SPI bus transfers.

40.3.2.1 Transmit and command FIFO commands

You must initialize Transmit Command (TCR) before enabling LPSPI in Peripheral mode, although this register is not updated until after LPSPI is enabled. After LPSPI is enabled, you must make changes to this register only when LPSPI is idle. In Peripheral mode, the LPSPI command word in this register controls SPI attributes. Before the PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag sets.

Table 298. Command word in Peripheral 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.
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 with devices that organize data as big-endian.
CONT	Continuous transfer	When continuous transfer is selected in Peripheral mode, after the number of bits indicated by FRAMESZ are transferred, LPSPI passes through and transmits the received data until the next PCS negation. Whatever is shifted in on the receive data is shifted out as transmit data considering that there is a 32-bit shift register.
CONTC	Continuing command	When the continuing command is enabled in Peripheral mode, after the number of bits indicated by FRAMESZ are transferred, 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 in 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 in 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 with QuadSPI memory devices, and at least either 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 three words: the first and second words are 32 bits, and the third word is 8 bits.

40.3.2.2 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 the two words in DMR0 and DMR1 or against a masked data word. You can also configure the data match function 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.
- By using CFGR0[RDMO], you can also configure the receiver match function to discard all received data until a data match is detected.
- After a receive data match, to allow all subsequent data to be received, first write 0 to CFGR0[RDMO], then clear SR[DMF].

40.3.2.3 Partial received word

When the PCS pin deasserts and the receive shift register shifts in a partial word, you can configure the receive shift register to either discard the partial word or to store it in the receive FIFO. You must specify this using CFGR1[PARTIAL].

A partial word is defined as less than TCR[FRAMESZ] bits (when TCR[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 TCR[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 supported, but a partial received frame of 1 bit is not supported.

40.3.2.4 Clocked interface

LPSPI supports interfacing with external controllers 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[PCSPOL*n*] = 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); you could use CFGR1[PCSPOL*n*] 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 selection specified in TCR[PRESCALE]) is required between the last SCK edge of one word and the first SCK edge of the next word.

40.3.3 Debug mode

Table 299. Debug mode

Chip mode	LPSPI operation
Debug (the core is in Debug or Halted mode)	Can continue operating in Debug mode, if CR[DBGEN] = 1

40.3.4 Clocking

Table 300. LPSPI clocks

Type of clock	Description
Functional	• 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 Controller and Peripheral modes. • LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least two times faster than the SPI external clock frequency (SCK).
External	• The LPSPI shift register is clocked directly by the SCK clock. • How the SCK clock is generated or supplied depends on the mode (Controller or Peripheral): — In Controller mode, the SCK clock is generated internally. — In Peripheral mode, the SCK clock is supplied externally.
Bus	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 for more.

40.3.5 Reset

Table 301. LPSPI resets

Type of reset	Description
Chip	Resets the LPSPI logic and registers to their default states.
Software	• Resets the LPSPI logic and registers to their default states, except for the Control register. • The LPSPI software reset is controlled using CR[RST].
FIFO	• Resets the transmit and command FIFO and the receive FIFO. • CR[RTF] and CR[RRF] are write-only. • After being reset, FIFO is empty.

40.3.6 Interrupts and DMA requests

The following table lists sources (status flags) that can generate LPSPI interrupts and LPSPI transmit and receive DMA requests.

Table 302. Interrupts and DMA requests

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit data flag	Indicates that data can be written to transmit FIFO, as configured by the transmit FIFO watermark, FCR[TXWATER].	Y	TX	Y

Table continues on the next page...

Table 302. Interrupts and DMA requests (continued)

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
RDF	Receive data flag	Indicates that data can be read from the receive FIFO, as configured by the receive FIFO watermark, FCR[RXWATER].	Y	RX	Y
WCF	Word complete flag	Indicates that the word is complete and the last bit of the word has been sampled.	Y	N	Y
FCF	Frame complete flag	Indicates that the frame is complete and PCS is deasserted.	Y	RX	Y
TCF	Transfer complete flag	Indicates that transfer is complete, PCS is deasserted, and the transmit and command FIFO is empty.	Y	N	Y
TEF	Transmit error flag	Indicates a transmit and command FIFO underrun. In Controller mode, when CFGR1[NOSTALL] = 0 (transfers stall when transmit FIFO is empty), TEF cannot be set.	Y	N	Y
REF	Receive error flag	Indicates a receive FIFO overflow. In Controller mode, when CFGR1[NOSTALL] = 0 (transfers stall when receive FIFO is full), REF cannot be set.	Y	N	Y
DMF	Data match flag	Indicates that the received data matches the configured data match value.	Y	N	Y
MBF	Module busy flag	Indicates that LPSPI is busy performing an SPI bus transfer.	N	N	N

40.3.6.1 End-of-packet DMA transfer

The end-of-packet functionality is designed for serial interfaces where you may not be aware of the size of the transfer in advance and the data is pushed by an external device. Examples include UART receive, I2C Peripheral mode, and SPI Peripheral mode. The end-of-packet processing is intended to ensure that data is not 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 the serial peripheral and DMA process it in the same way. For example, UART end of packet is signaled by an idle line condition, I2C end of packet by a stop and/or repeated start condition, and SPI end of packet by PCS negation.

If you configure the serial peripheral 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, 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 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 is complete 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 the 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 is complete, including final offsets and optional interrupts, channel linking and scatter/gather. The final destination address can optionally be saved in the system memory or is available in the destination address register.

After the DMA terminates the major loop, no servicing of the receive FIFO occurs until either software or hardware reconfigures the DMA (for example, channel linking or scatter/gather). This delay must be minimized to avoid receiver FIFO overrun. NXP does not recommend automatic DMA end-of-packet processing when there are only a few words transferred between end-of-packet conditions. That 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.

40.3.6.2 DMA support registers

To support efficient DMA transfers to the transmit and control FIFO, an alias register supports 32-bit write access to the Transmit Command (TCR) and an alias region supports incrementing 32-bit write accesses to the Transmit Data (TDR).

- Transmit Command Burst (TCBR) is a 32-bit alias register for writing to TCR.
- Transmit Data Burst (TDBR0 - TDBR127) is a 512-byte alias region that supports writing to the TDR.

The burst alias locations are contiguous. A DMA transfer can start by writing to the TCBR register to initialize the transfer and then increment into the TDBR region without changing the DMA transfer size. The alias registers can also be used by the DMA to perform burst transfers when accessing the transmit FIFO.

The transmit FIFO prevents writes that overflow the FIFO, but this prevention does not signal an error. Do not perform writes to the TDBR unless there is sufficient room in the transmit FIFO.

Receive Data Burst (RDBR0 - RDBR127) is a 512-byte alias region that supports reading the Receive Data. This can be used by the DMA to perform burst transfers when accessing the receive FIFO.

40.3.7 Peripheral triggers

The connection of the LPSPI peripheral triggers with other peripherals depends on the device that is used.

Table 303. Peripheral triggers

Type of trigger	Description	Additional information
Frame output	The frame output trigger: • Asserts at the end of each frame (when PCS deasserts). • Remains asserted for one cycle of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE].	LPSPI generates two output triggers that can be connected to other peripherals on the chip.
Word output	The word output trigger: • Asserts at the end of each received word. • Remains asserted for one cycle of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE].	
Input	To control the start of an LPSPI bus transfer, the LPSPI input trigger can be selected instead of the HREQ input:	

Table continues on the next page...

Table 303. Peripheral triggers (continued)

Type of trigger	Description	Additional information
	• The LPSPI input trigger is synchronized, and must assert for at least two cycles of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE] so that the input trigger can be detected. • When LPSPI is busy, the HREQ input (and therefore the LPSPI input trigger) is ignored. • When LPSPI is busy, both the HREQ and LPSPI input triggers are ignored. They are used to start a new transfer when LPSPI is idle.	

40.4 External signals

Table 304. External signals

Signal	Name	Description	I/O
SCK	Serial clock	• Input in Peripheral mode • Output in Controller mode	I/O
PCS[0]	Peripheral chip select	• Input in Peripheral mode • Output in Controller 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 Peripheral mode or when used as controller host request • Output in either Controller mode or when used as peripheral 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 Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	I/O
PCS[3]/DATA[3]	Peripheral chip select or data pin 3 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Peripheral mode • Output in Controller mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers	I/O

Table continues on the next page...

Table 304. External signals (continued)

Signal	Name	Description	I/O
		• Output in half-duplex parallel data transmit transfers	
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

40.5 Initialization

This module does not require initialization.

40.6 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.

40.6.1 LPSPI register descriptions

40.6.1.1 LPSPI memory map

LPSPI0 base address: 4009_C000h

LPSPI1 base address: 4009_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0200_0004h
4h	Parameter (PARAM)	32	R	0004_0202h
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

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
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

40.6.1.2 Version ID (VERID)

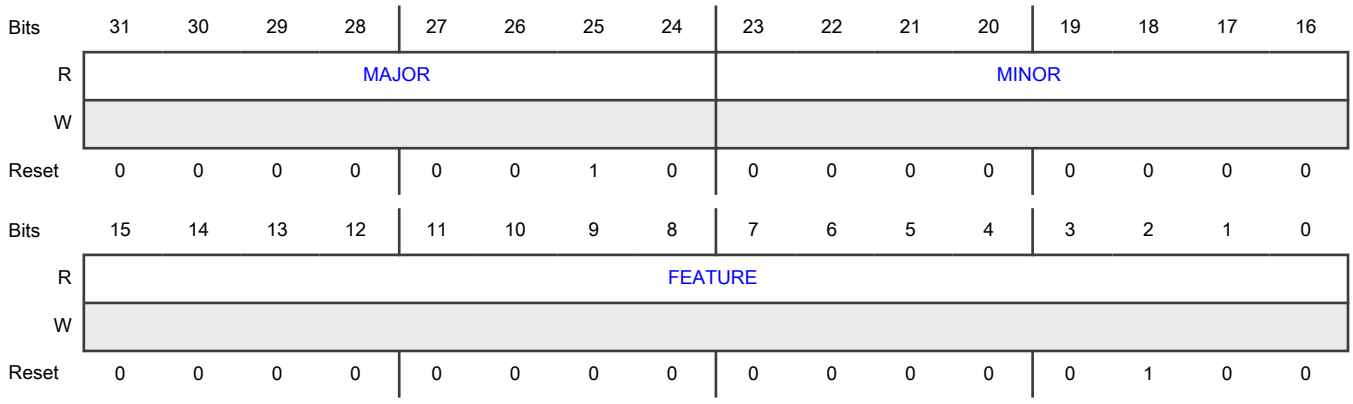
Offset

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 Indicates the major version number of the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number of 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.

40.6.1.3 Parameter (PARAM)

Offset

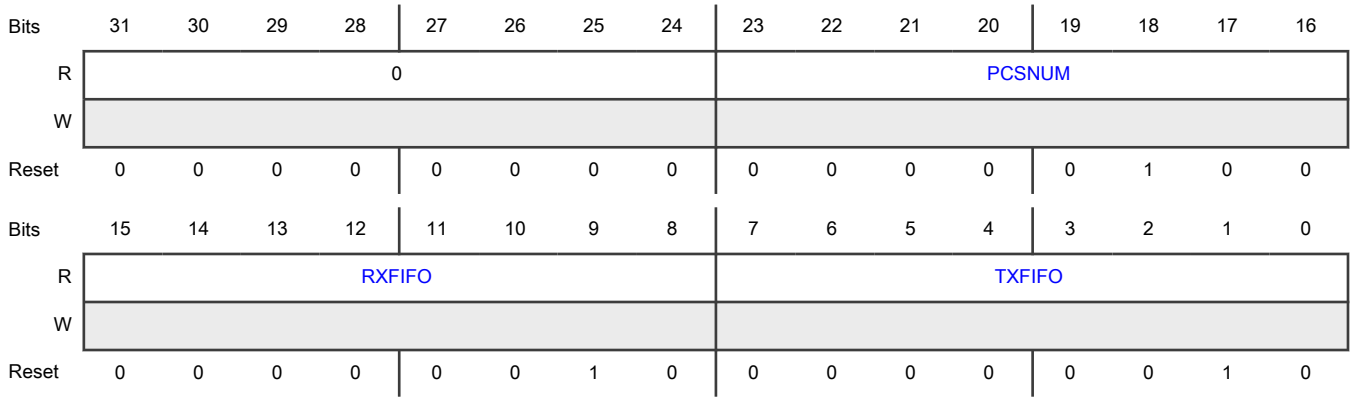
Register	Offset
PARAM	4h

Function

Contains:

- Number of PCS pins.
- Receive FIFO size.
- Transmit FIFO size.

Diagram



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}.

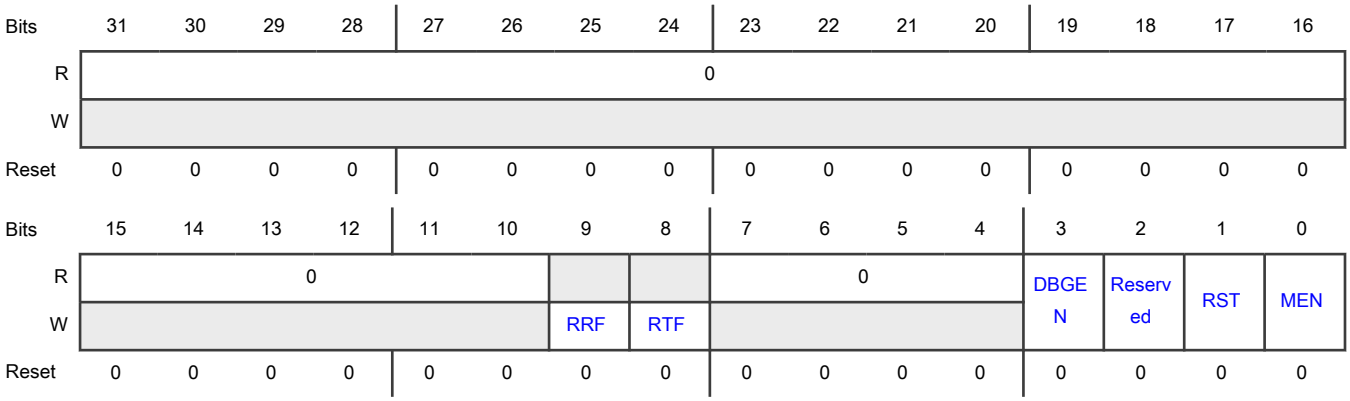
40.6.1.4 Control (CR)

Offset

Register	Offset
CR	10h

Function
Contains fields that control the module operation.

Diagram



Fields

Field	Function
31-10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 RRF	Reset Receive FIFO Deletes all entries in the receive FIFO. This field always reads 0. 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 when the CPU is in Debug mode. If 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). You must update this field only when LPSPI is disabled (MEN = 0). 0b - Disable 1b - Enable
2 —	Reserved
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 Enables the module. After writing 0, MEN remains set until LPSPI has completed the current transfer and is idle. 0b - Disable 1b - Enable

40.6.1.5 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 Indicates, in Controller mode, whether there is data to transmit and LPSPI is able to transmit (for example, the HREQ pin is asserted). The HREQ pin deasserts after the PCS pin deasserts and the LPSPI controller has waited for half the time specified in CCR[DBT] with no new data to transmit. Peripheral mode sets 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 whether the received data matches DMR0[MATCH0] and/or DMR1[MATCH1] (as configured by CFGR1[MATCFG]).

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 - No match 1b - Match When writing 0b - No effect 1b - Clear the flag
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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overflow 1b - Overflow When writing 0b - No effect 1b - Clear the flag
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. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underrun 1b - Underrun

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When writing 0b - No effect 1b - Clear the flag
10 TCF	Transfer Complete Flag Indicates, in Controller mode, whether all transfers are complete and LPSPI has returned to the Idle state and the transmit FIFO is empty. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
9 FCF	Frame Complete Flag Indicates whether a frame transfer is complete after PCS deasserts. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
8 WCF	Word Complete Flag Indicates whether the last bit of a received word is sampled. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the number of words in the receive FIFO is greater than the value in FCR[RXWATER]. 0b - Receive data not ready 1b - Receive data ready
0 TDF	Transmit Data Flag Indicates whether 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 requested

40.6.1.6 Interrupt Enable (IER)

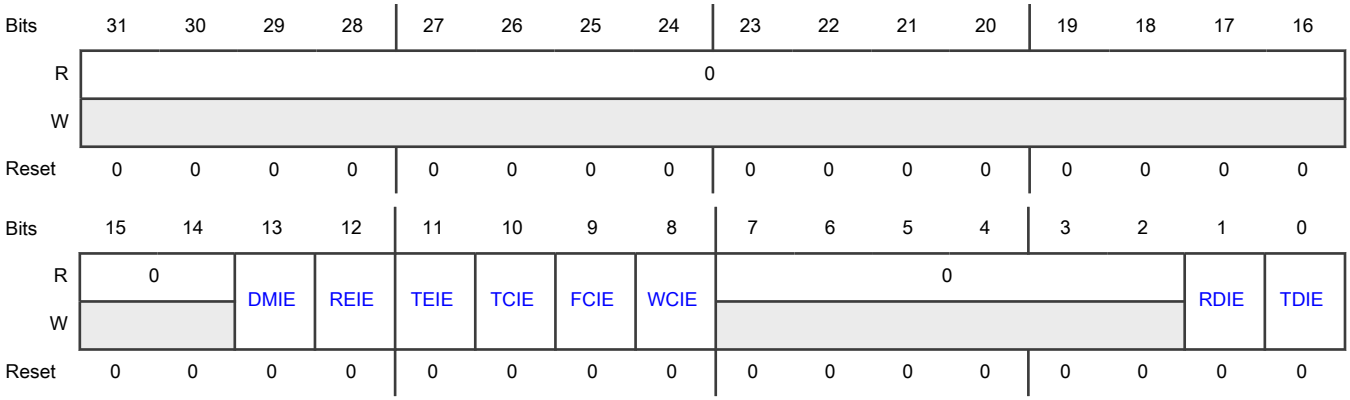
Offset

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 Enables the data match interrupt. 0b - Disable 1b - Enable
12 REIE	Receive Error Interrupt Enable Enables the receive error interrupt. 0b - Disable 1b - Enable
11 TEIE	Transmit Error Interrupt Enable Enables the transmit error interrupt. 0b - Disable 1b - Enable
10 TCIE	Transfer Complete Interrupt Enable Enables the transfer complete interrupt. 0b - Disable 1b - Enable
9 FCIE	Frame Complete Interrupt Enable Enables the frame complete interrupt. 0b - Disable 1b - Enable
8 WCIE	Word Complete Interrupt Enable Enables the word complete interrupt. 0b - Disable 1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables the receive data interrupt.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables the transmit data interrupt. 0b - Disable 1b - Enable

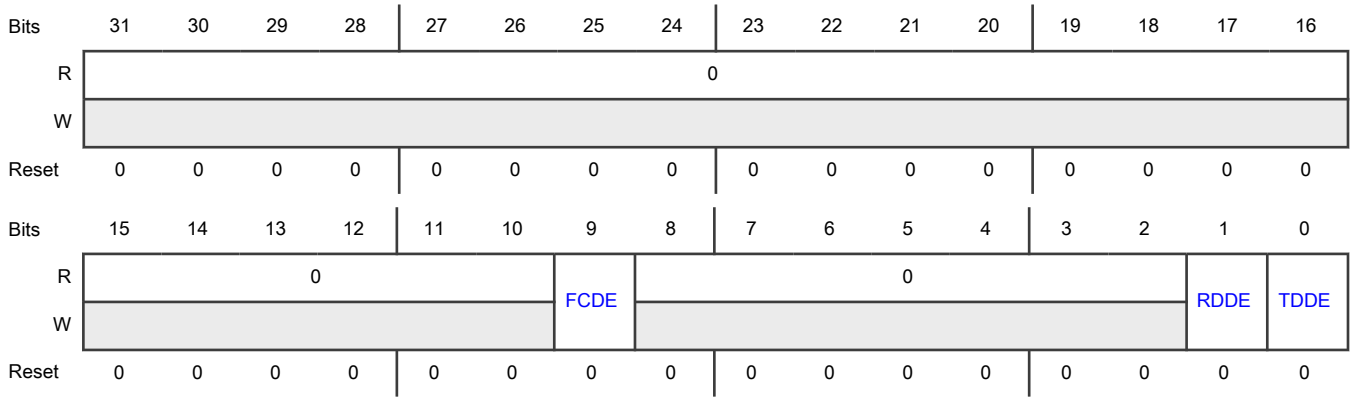
40.6.1.7 DMA Enable (DER)

Offset

Register	Offset
DER	1Ch

Function
Enables the DMA data flow.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 FCDE	Frame Complete DMA Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 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 Enables the receive data DMA. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables the transmit data DMA. 0b - Disable 1b - Enable

40.6.1.8 Configuration 0 (CFGR0)

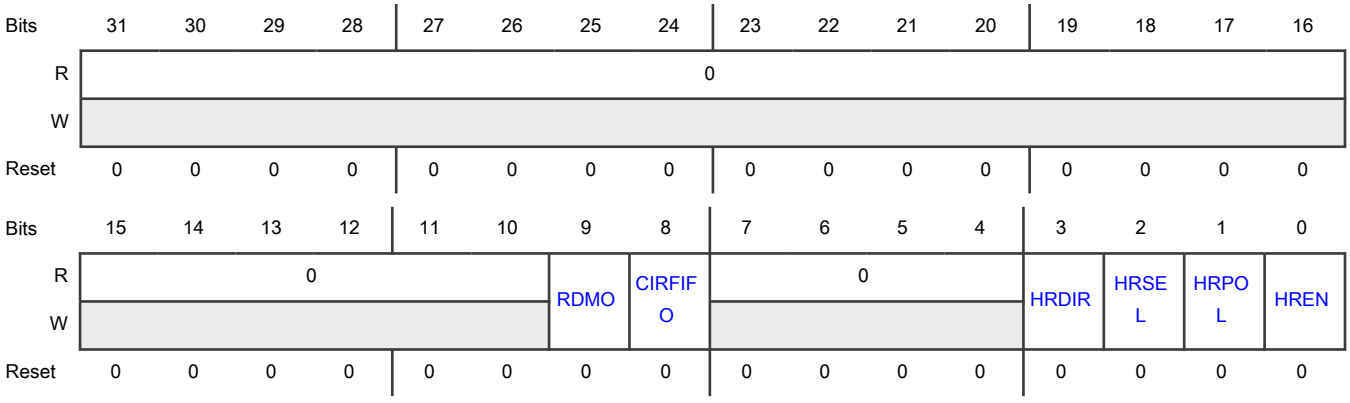
Offset

Register	Offset
CFGR0	20h

Function

Includes fields to configure LPSPI.

Diagram



Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only Enables receive data match. When enabled, all received data that does not cause 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 Enables circular FIFO. 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. You must configure the HREQ pin only as an output when LPSPI is in Peripheral mode. The HREQ pin direction must be an input for Controller 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - HREQ pin 1b - Input trigger
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 Enables LPSPI, in Controller mode, 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 Peripheral mode, causes the HREQ output pin to assert when data is available to be transmitted. 0b - Disable 1b - Enable

40.6.1.9 Configuration 1 (CFGR1)

Offset

Register	Offset
CFGR1	24h

Function

Includes fields to configure LPSPI. You must write to this register only when LPSPI is disabled.

In addition to pin and output configurations, this register contains match configuration details; the following table shows match conditions specified in MATCFG.

Table 305. Match conditions for CFGR1[MATCFG]

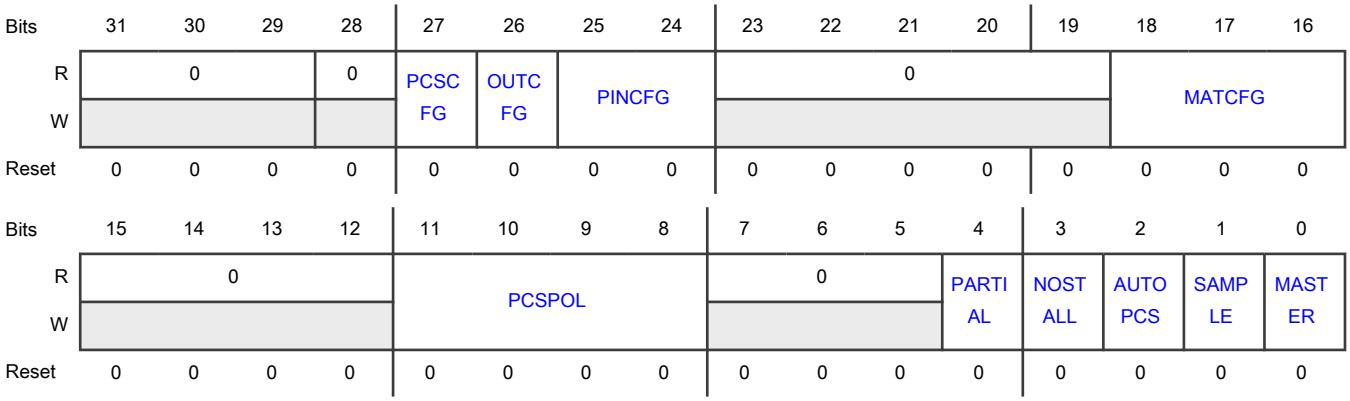
Condition	Description
Match first data word with compare word	Match if first data word equals MATCH0 logically ORed with MATCH1 first_data_word == (MATCH0 || MATCH1)
Match any data word with compare word	Match if any data word equals MATCH0 logically ORed with MATCH1 any_data_word == (MATCH0 || MATCH1)
Sequential match, first data word	Match if first data word equals MATCH0, and second data word equals MATCH1 (first_data_word == MATCH0) && (second_data_word == MATCH1)
Sequential match, any data word	Match if any data word equals MATCH0, and the next data word equals MATCH1

Table continues on the next page...

Table 305. Match conditions for CFGR1[MATCFG] (continued)

Condition	Description
	(any_data_word == MATCH0) && (next_data_word == MATCH1)
Match first data word (masked) with compare word (masked)	Match if first data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1 (first_data_word && MATCH1) == (MATCH0 && MATCH1)
Match any data word (masked) with compare word (masked)	Match if any data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1 (any_data_word && MATCH1) == (MATCH0 && MATCH1)

Diagram



Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 PCSCFG	Peripheral Chip Select Configuration Specifies PCS pin configuration. When performing parallel transfers, you must configure this field to enable the desired transfer. 0b - PCS[3:2] configured for chip select function 1b - PCS[3:2] 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Retain last value 1b - 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. See the match conditions listed in Table 1 for more information. **NOTE** When writing to this field, either the old value or new value must be in the disabled state (0). You cannot transition from a nonzero value to another nonzero value. 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)
15-12 —	Reserved
11-8 PCSPOL	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*]. 0000b - Active low 0001b - Active high
7-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 PARTIAL	Partial Enable Specifies whether LPSPI, when in Peripheral mode, stores a partial received word in the receive FIFO, or discards it, when PCS deasserts. See Partial received word for more information. 0b - Discard 1b - Store
3 NOSTALL	No Stall Disables a normal operating feature that causes LPSPI, when in Controller 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 Peripheral 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 four LPSPI functional clock cycles, divided by the prescaler (see TCR[PRESCALE]) selected between each word to ensure correct operation. This field is ignored in Controller mode. 0b - Disable 1b - Enable
1 SAMPLE	Sample Point Specifies the SCK clock edge on which LPSPI, when in Controller mode, samples input data. Writing 1 to this field causes LPSPI 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 Controller mode is equal to the input data setup time in Peripheral mode. In Peripheral mode, this field is ignored. **NOTE** When SAMPLE = 1, both the input and output buffers must be enabled for the SCK pin. Buffers are configured at the chip level. 0b - SCK edge 1b - Delayed SCK edge
0 MASTER	Controller Mode Specifies the LPSPI operating mode, Controller or Peripheral. This field directly controls the direction of the SCK and PCS pins.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Peripheral mode 1b - Controller mode

40.6.1.10 Data Match 0 (DMR0)

Offset

Register	Offset
DMR0	30h

Function

Specifies the 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 when CFGR1[MATCFG] > 0.

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 Specifies the MATCH0 value to be compared against received data.

40.6.1.11 Data Match 1 (DMR1)

Offset

Register	Offset
DMR1	34h

Function

Specifies the 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 CFGR1[MATCFG] > 0.

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 Specifies the MATCH1 value to be compared against received data.

40.6.1.12 Clock Configuration (CCR)

Offset

Register	Offset
CCR	40h

Function

Contains clock configuration fields that are used only in Controller mode; 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, do one of the following:

- Write to all four fields in Clock Configuration (CCR) simultaneously and only once in a 32-bit data.
- Modify the values of CCR[SCKPCS] and/or CCR[PCSSCK]; write only these two upper bytes in a 16-bit data or one of them in an 8-bit data.
- Modify CCR1[PCSPCS] and CCR1[SCKSCK] only or CCR1[SCKSET] and CCR1[SCKHLD] only, write respectively to 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 Configures SCK-to-PCS delay. In Controller mode, this field helps you configure the delay from the last SCK edge to PCS negation: • The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. See Figure 173 for more information.
23-16 PCSSCK	PCS-to-SCK Delay Configures PCS-to-SCK delay. In Controller mode, this field helps you configure the delay from PCS assertion to the first SCK edge: • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. See Figure 173 for more information.
15-8 DBT	Delay Between Transfers Configures the delay between transfers. Writing to this field updates the contents of CCR1[PCSPCS] and CCR1[SCKSCK].

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 SCKDIV	SCK Divider Updates the contents of CCR1[SCKSET] and CCR1[SCKHLD]. Baud rate = function clock ÷ (2^PRESCALE × (SCKSET + SCKHLD + 2))

40.6.1.13 Clock Configuration 1 (CCR1)

Offset

Register	Offset
CCR1	44h

Function

Contains clock configuration fields, which are used only in Controller mode. You can change them only 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 Configures SCK inter-frame delay in Controller mode: • This field helps you configure 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 selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE For backward compatibility, writing to CCR[DBT] updates this field with the value written.
23-16 PCSPCS	PCS to PCS Delay Configures PCS to PCS delay in Controller mode: • This field helps you configure 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 selected prescaler (see TCR[PRESCALE]). • The minimum delay is two 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 two 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 selected prescaler. **NOTE** For backward compatibility, writing to CCR[DBT] updates this field with (DBT÷2) rounded up.
15-8 SCKHLD	SCK Hold Configures the hold phase of the SCK pin in Controller mode: • The hold phase is the delay between the SCK edge that samples the receive data and the SCK edge that drives the transmit data. • It is the SCK low period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. It 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 selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD. You must configure both these fields to the same value for a 50/50 duty cycle. See Figure 173 for more information. **NOTE** For backward compatibility, writing to CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded down.
7-0 SCKSET	SCK Setup Configures the setup phase of the SCK pin in Controller mode:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• The setup phase is the delay between the SCK edge that drives the transmit data and the SCK edge that samples the receive data. • It is the SCK high period when CPHA = 0 and CPOL = 1, or CPHA = 1 and CPOL = 0. It 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 selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. • The SCK period is equal to (SCKSET + SCKHLD + 2) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The SCK duty cycle is based on the difference between SCKSET and SCKHLD. You must configure both these fields to the same value for a 50/50 duty cycle. See Figure 173 for more information. **NOTE** For backward compatibility, writing to CCR[SCKDIV] updates this field with (SCKDIV ÷ 2) rounded up.

40.6.1.14 FIFO Control (FCR)

Offset

Register	Offset
FCR	58h

Function

Contains the receive FIFO and transmit FIFO watermark values.

Diagram

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

Fields

Field	Function
31-24 —	Reserved
23-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark Causes LPSPI to set SR[RDF] when 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 written value.
15-8 —	Reserved
7-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark Causes LPSPI to set SR[TDF] when the number of words in the transmit FIFO is equal to or less than TXWATER. Writing a value equal to or greater than the FIFO size truncates the written value.

40.6.1.15 FIFO Status (FSR)

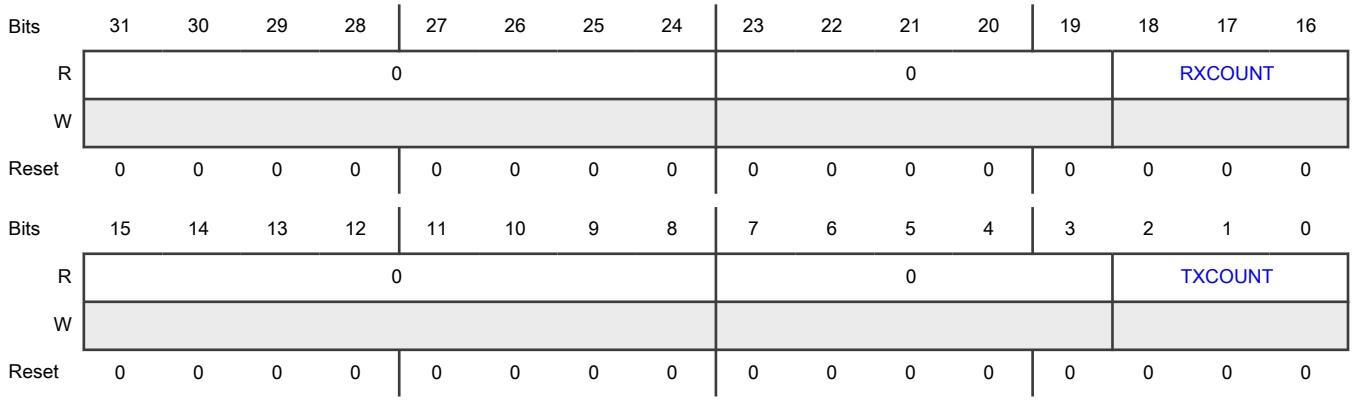
Offset

Register	Offset
FSR	5Ch

Function

Contains fields that indicate the number of words currently stored in the receive and transmit FIFOs.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Indicates the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Indicates the number of words currently stored in the transmit FIFO.

40.6.1.16 Transmit Command (TCR)

Offset

Register	Offset
TCR	60h

Function

Pushes the data into the transmit FIFO, in the same order as written.

When you write to either this register or to Transmit Data (TDR), each write pushes data into the transmit FIFO. You must write to this register only using 32-bit writes, which are tagged and cause the command register to update; after that the entry reaches the top of the FIFO and LPSPI is enabled. This allows changes to the command word and the transmit data itself to be interleaved. That is, 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 Controller 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 PCS deasserts.
- In Controller mode, if the command word is changed before an existing frame has completed, then the existing frame terminates and the command word 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 Peripheral mode, the command word must be changed only when LPSPI is idle and there is no SPI bus transfer.

Avoid resetting the transmit FIFO after writing to this register; wait for the command register to update from the FIFO first.

Avoid register reading problems: Reading this register returns the current state of the register. Reading this register at the same time that it is loaded from the transmit FIFO can return an incorrect register value. It is recommended to:

- Read this register when the transmit FIFO is empty.
- Read this 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	CPOL	CPHA	PRESCALE			Reserved	PCS		LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
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				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. You can update this field only when PCS is deasserted. See Figure 173 for more information. 0b - Inactive low 1b - Inactive high
30 CPHA	Clock Phase Indicates whether data is captured or changed on the leading edge of SCK and captured or changed on the following edge of SCK. You can update this field only when PCS is deasserted. See Figure 173 for more information. 0b - Captured 1b - Changed
29-27 PRESCALE	Prescaler Value Specifies the division of the LPSPI functional clock. For all SPI bus transfers, this value is applied to Clock Configuration (CCR). You can update this field only when PCS is 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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. This field is updated only when PCS is deasserted. **NOTE** This entire 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] 10b - Transfer using PCS[2] 11b - Transfer using PCS[3]
23 LSBF	LSB First Indicates whether data is transferred with MSB first or LSB first. 0b - MSB first 1b - 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 - Disable byte swap 1b - Enable byte swap
21 CONT	Continuous Transfer Enables continuous transfer: • In Controller mode, this field keeps PCS asserted at the end of the frame size until a command word is received that starts a new frame. • In Peripheral mode, when this field 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 - Disable 1b - Enable
20 CONTC	Continuing Command Enables the command word to be changed within a continuous transfer in Controller mode:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 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 Peripheral mode, this field modifies the internal RXMSK and TXMSK configuration after the first FRAMESZ bits and until 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 the receive FIFO). 0b - Normal transfer 1b - Mask receive data
18 TXMSK	Transmit Data Mask Masks transmit data (no data is loaded from the transmit FIFO and the output pin is 3-stated). In Controller mode, TXMSK initiates a new transfer that 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 RXMSK or 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):

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 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 three words: the first and second words are 32 bits, and the third word is 8 bits.

40.6.1.17 Transmit Data (TDR)

Offset

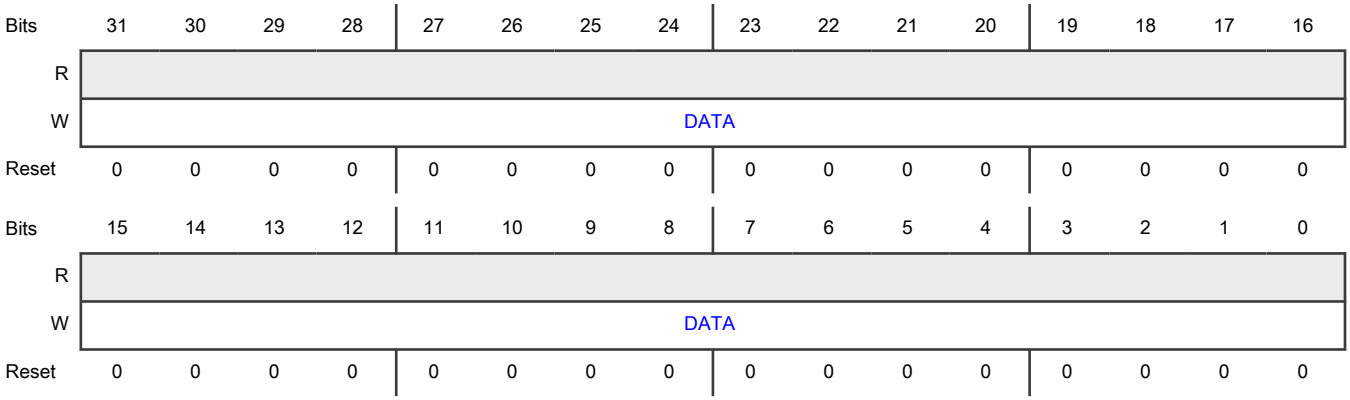
Register	Offset
TDR	64h

Function

Pushes the data into the transmit FIFO, in the same order that the data is written. You can write to this register using 32-, 16-, or 8-bit writes.

When you write to this register or to Transmit Command (TCR), each write pushes data into the FIFO with zero pushed in unwritten bytes.

Diagram



Fields

Field	Function
31-0 DATA	Transmit Data Indicates transmit data. Both 8-bit and 16-bit writes of transmit data zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8-bit 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.

40.6.1.18 Receive Status (RSR)

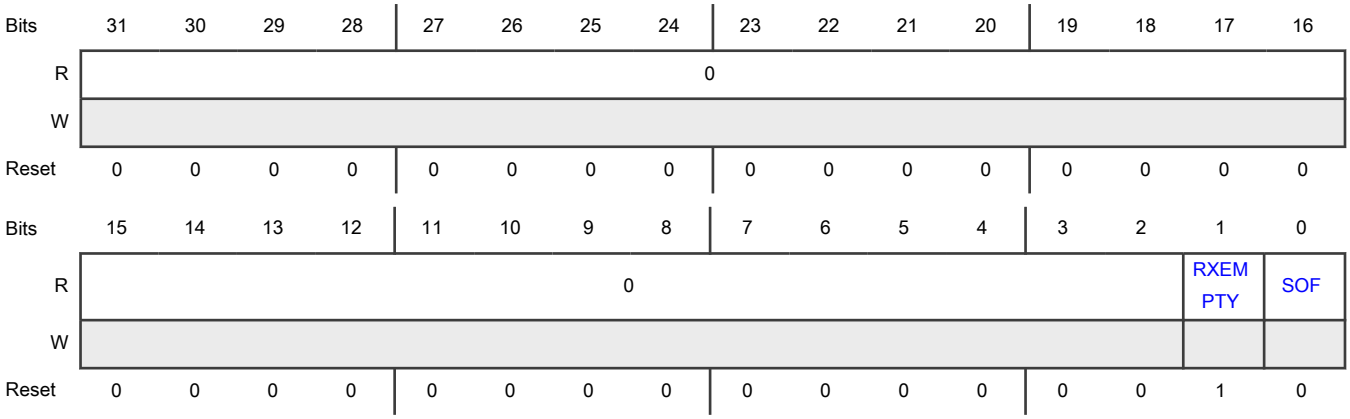
Offset

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 Indicates whether the receive FIFO is empty. 0b - Not empty 1b - Empty
0 SOF	Start of Frame Indicates whether this is the first data word received after PCS assertion. 0b - Subsequent data word or RX FIFO is empty (RXEMPTY=1). 1b - First data word

40.6.1.19 Receive Data (RDR)

Offset

Register	Offset
RDR	74h

Function
Pulls the first entry from the receive 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

40.6.1.20 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

40.6.1.21 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.

See DMA support registers.

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 Writes data to Transmit Command (TCR).

40.6.1.22 Transmit Data Burst (TDBR0 - TDBR127)

Offset

For n = 0 to 127:

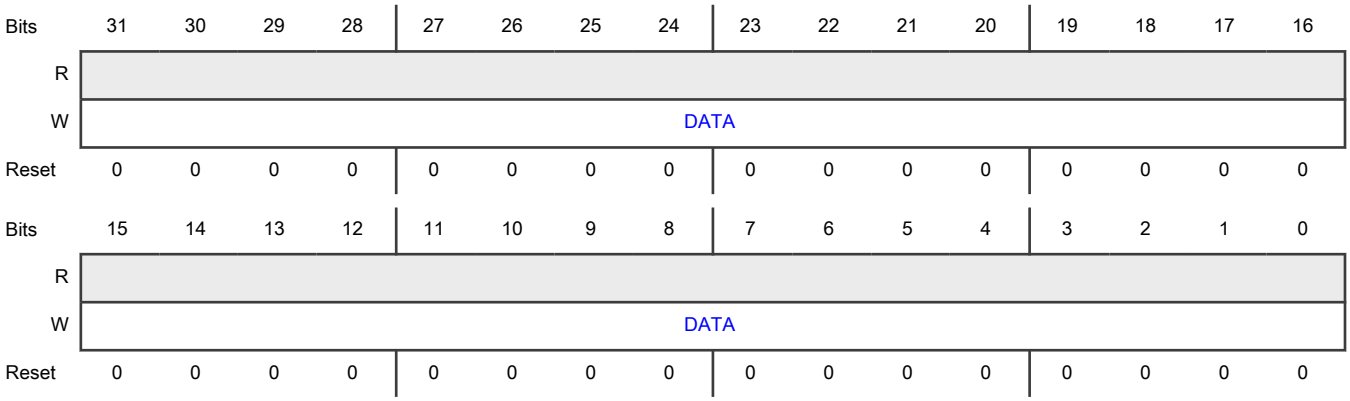
Register	Offset
TDBRn	400h + (n × 4h)

Function

Supports burst transfers of data to the transmit FIFO for use with the DMA controller. The size of this register is 512 bytes.

See DMA support registers.

Diagram



Fields

Field	Function
31-0 DATA	Data Writes data to Transmit Data (TDR).

40.6.1.23 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.

See DMA support registers.

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	Data Reads data from Receive Data (RDR).

Chapter 41
Improved Inter-Integrated Circuit (I3C)

41.1 Chip-specific MIPI I3C information

Table 306. Reference links to related information

Topic	Related module	Reference
Full description	I3C	I3C
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

41.1.1 Module instances

This device has one instances of the MIPI I3C module: I3C0.

41.2 Overview

I3C is a communications processor that improves upon the use and power of I2C, and provides an alternative to SPI for mid-speed applications.

The I3C bus protocol supports:

- In-band interrupts (IBI): these interrupts move from target to controller without extra wires, and the controller knows which target sent the interrupt
- Common command codes (CCC)
- Dynamic addressing
- Multi-controller and multi-drop
- Hot-Join (HJ)
- I2C compatibility

I3C supports all required and many optional features of the MIPI Alliance Specification for I3C, v1.0 and v1.1, except for ternary data rates (HDR-TSP and HDR-TSL). See Features, Target Capabilities (SCAPABILITIES), and Target Capabilities 2 (SCAPABILITIES2) for more information.

41.2.1 Block diagram

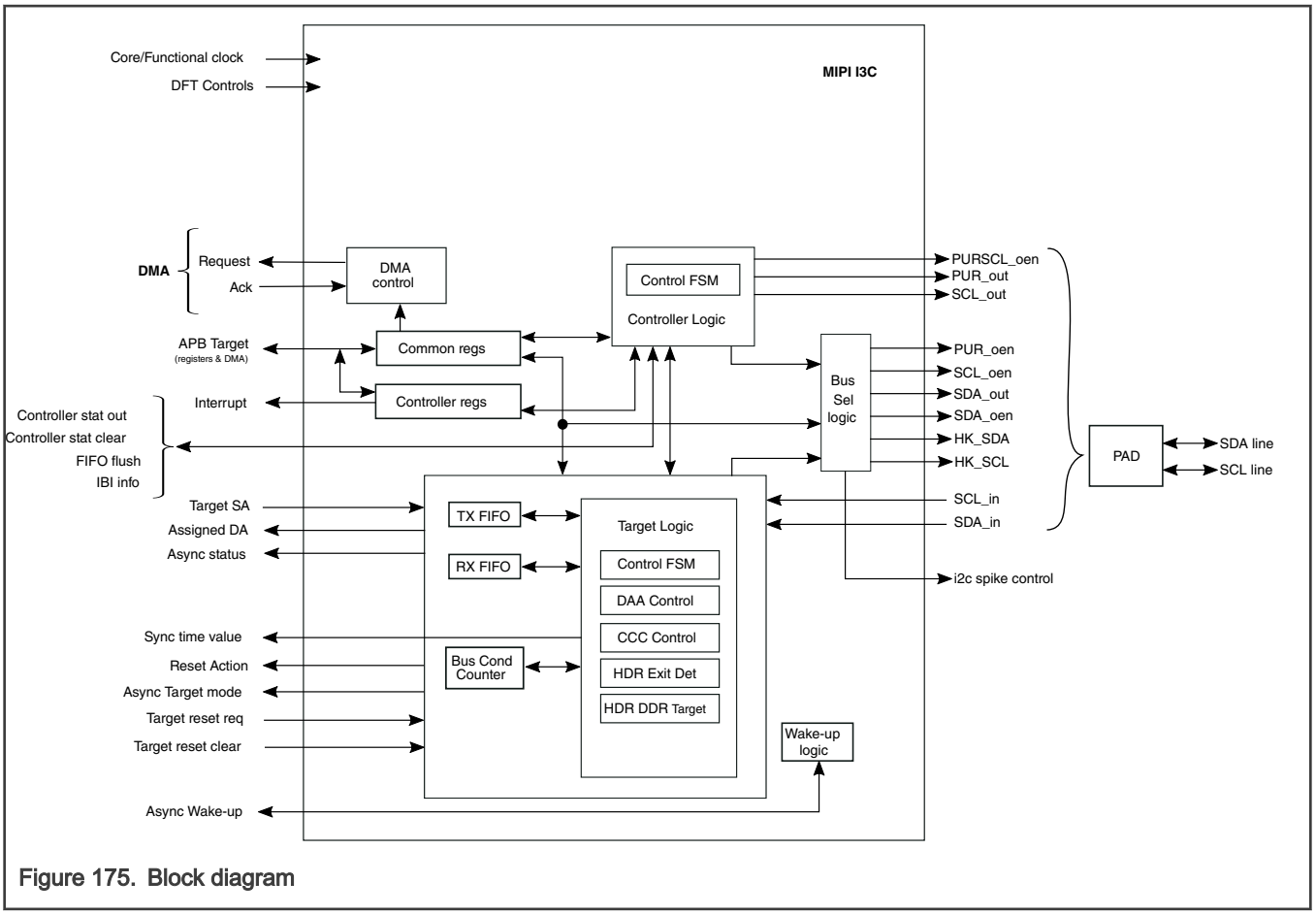


Figure 175. Block diagram

41.2.2 Features

- Two-wire multidrop bus that is capable of up to 12.5 MHz clock speeds with up to 11 devices:
  - Uses standard pads with 4 mA drive.
  - Dynamically assigns target addresses, and targets do not require static addresses (SAs). However, targets can have an I2C static address assigned at startup, so the target can operate on an I2C bus. By default, I3C supports 7-bit I2C-style addresses.
  - Allows targets to use the inbound SCL clock as the peripheral clock (instead of the clock from the controller) so that devices can have slow or inaccurate clocks internally.
  - Allows simple targets, such as temperature sensors, to have no internal clock.
  - I3C controller supports handoff from Open-Drain to Push-Pull mode for ACK to data transfer.
  - Generally, the controller terminates the read, but for I3C, the target can also end the read.
- IBIs that allow targets to send notifications to a controller:
  - Can be equivalent to a separate GPIO, but can also be directly data-bearing.
  - Can be prioritized. When multiple targets send interrupts to a controller at the same time, the order is resolved. Dynamic addresses (DAs) establish the priority of the targets, so the controller controls the priority of the targets.

    Targets with lower-value DAs are higher-priority level IBIs.

  - Can start interrupts even when the controller is not active on the bus. A free-running clock is not required, but starting an interrupt requires a Bus Available condition.
  - Can resolve an initial event via a time-stamping option, not requiring an interrupt.
- Built-in commands for applications created in software or firmware by using the register interface maintained in a separate space. These commands do not collide with normal controller-to-target messages and:
  - Control bus behavior, modes and states, low-power state, inquiries, and more.
  - Have an extra room for new built-in commands for other groups.
- Organized forms of multicontroller modes that are secondary controllers using clean handoffs between different controllers.
- Hot-Join onto I3C bus allows devices to connect to the bus later than when the bus starts. This:
  - Enables a device or module to access the I3C bus after power up or when physically inserted onto the I3C bus.
  - Provides a clean method for notification when new devices or modules access the I3C bus.
- I3C can use both I2C and I3C buses:
  - I3C supports specific legacy I2C devices on the bus.
  - I3C target devices can operate on I2C buses.
  - In I2C controller mode, it supports only Push-Pull mode on the SCL line.
  - A reset output is available for I2C mode SW reset for application usage. Asserted when the module is an I2C target and gets a I2C software General command from a controller. See the chip-specific section, for usage if this is available.
- Higher data rate modes are available:
  - I3C has a high data rate: double data rate (HDR-DDR) mode, which is double the data rate of SDR .
  - Only the controller and the specific targets, which are capable and configured to work with HDR-DDR, support the higher data rate (the other targets can ignore it).

I3C supports most of the I3C features (see Target Capabilities (SCAPABILITIES) and Target Capabilities 2 (SCAPABILITIES2)), except for the ternary data rates (HDR-TSP and HDR-TSL), HDR-BT modes, and peer-to-peer messaging.

41.3 Functional description

41.3.1 Operating modes

This section describes all functional operation modes of the I3C module.

41.3.1.1 Target and controller roles for I3C

The I3C protocol defines these roles for devices on the I3C bus:

- Main controller: initially configures the I3C bus and serves as the first active controller
- Secondary controller: accepts the controller role from any active controller to become the new active controller (the secondary controller may then pass the controller role along: either back to the previous active controller, or on to any other controller-capable device)
- Target: responds to commands from any I3C controller
- I2C target: responds to compatible commands from any I3C controller

The I3C peripheral contains both controller and target components and can be configured to be either controller or target, or as target with secondary controller capability. However, if I3C is chosen to be a controller, then the I3C peripheral supports Target mode to facilitate handoffs to multiple controllers.

In general, any I3C controller can be a controller or a target because a controller becomes a target when giving over control to another controller. The only exceptions to this rule occur when using point-to-point communication or when using a controller that never shares controllership.

41.3.1.1.1 Controller requirements

Controller mode uses software that supports the requirements of the controller (including as a secondary controller):

- Managing the enter dynamic address assignment (ENTDAA) assigning dynamic addresses (DAs) to each target. The I3C peripheral supports this process but requires you to make choices.
- Driving the serial data (SDA) signal in both Open-Drain and Push-Pull modes. After START command, SDA is in Open-Drain mode and after Repeated START command, it is in Push-Pull mode:
  - SDA is subject to arbitration because both controller and target can drive the SDA line (arbitration is also useful for handoff from controller to target).
  - The ninth bit of SDA controller write data is an odd parity bit.
- Building a table of targets and their capabilities to control which actions and commands may be sent to the targets.
- Managing requests such as IBI and controller requests (handoff).
- Adjusting clock speed or write-to-read timing to match the limitations of the target. This adjustment can be done in hardware using dividers and uneven duty cycles, but you must decide how.
- Adjusting the maximum data length.
- Being a target after the controller role handoff to another controller-capable device.

The controller needs an accurate clock capable of running at a frequency that is the multiple of a frequency between 11 MHz and 12.5 MHz. Following are the possible clock frequencies:

- 24 MHz (multiple of 12 MHz)
- 25 MHz (multiple of 12.5 MHz)

If required to use a lower I3C SCL value, for example as small as 5 MHz (although the lower value does not support a mixed bus system):

- The controller clock can be a multiplier of that value.
- A lower SCL can also be achieved using a higher PPBAUD value.

41.3.1.1.2 I3C target acts like I2C target on I3C buses

If the target is assigned an I2C static address, the I3C target acts like an I2C device when it first powers on. If the I3C target is placed on an I2C bus with an I2C controller, then the I3C target stays in I2C mode and operates normally. The software is aware that the I3C target is in I2C mode because:

- There has not been an interrupt (see SSTATUS[DACHG]) indicating that a dynamic address was assigned.

For full I2C support of Fast mode (Fm) and Fast-mode Plus (Fm+), the pads must support a 50-ns spike filter. This filter must be turned off when the I3C 7Eh broadcast address is received (indicating an I3C controller). You can turn off the spike filters via hardware using the raw net indicating that the address was received, or via software. I3C does not need a 50-ns spike filter to operate on an I2C bus. Therefore, the spike filter is not a requirement for the I3C peripheral.

Depending on the configuration, the module supports most I2C features. See Target Capabilities (SCAPABILITIES) and Target Capabilities 2 (SCAPABILITIES2) for more information.

41.3.1.1.3 How a target rejoins the I3C bus

When a target tries to rejoin the I3C bus following a power-up or hard reset, the target needs a new dynamic address (DA). The target can rejoin the I3C bus in these ways:

- If the DA is lost, Hot-Join is used.

- If the DA is retained in the peripheral (for example, with state retention flip-flops), SCONFIG[OFFLINE] is used.

When SCONFIG[SLVENA] is 1, SCONFIG[OFFLINE] may become 1. This setting causes the peripheral to rejoin the bus safely. It does so by ensuring that the I3C bus is not in HDR mode, using the same approach as TE0 or TE1 exit. The peripheral waits for the HDR exit pattern from the controller, or for 60 µs of SCL and SDA lines not changing, whichever occurs first.

After using SCONFIG[OFFLINE], the I3C peripheral cannot safely use IBI until SSTATUS[STOP] becomes 1. This status ensures that the next START is safe to use for IBI.

If the application has to perform an IBI, it must wait for SSTATUS[STOP] to become 1, or for SCL and SDA lines to remain high for 200 µs. You can perform this process using the SSTATUS and SINTSET controls:

- If Target Status (SSTATUS) indicates that the bus is not busy and the peripheral interrupts on START or STOP, use a timer to measure 200 µs. If the timer finishes with no START or STOP, it is safe to use IBI.
- If a START causes an interrupt, then the timer must be turned off and the application must wait for a STOP.
- If a STOP causes an interrupt, it is safe to use IBI.

41.3.1.2 Using the I3C controller for I2C and I3C

The I3C controller operates with the following set of built-in capabilities with the application flow:

- Built-in enter dynamic address assignment (ENTDAA) mechanism to simplify the assignment of dynamic addresses to targets. This feature is used when set dynamic address from static address (SETDASA) is not available.

  NOTE

  When SETDASA CCC is available, use SETDASA CCC before using ENTDAA CCC; all targets without assigned dynamic addresses respond to ENTDAA CCC.

- 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 or NACK detection 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 the previous request, including both I2C and I3C.
- Auto-IBI mode, which responds immediately to a target-initiated IBI and can be used when in Sleep or Deep Sleep mode.
- Special message mode for SDR and DDR to simplify DMA use.
- Automated SCL, SDA, pullup, and High-Keeper controls.
- I2C and I3C frequency and duty cycle configurations from functional clock.

  NOTE

  You must configure Controller Configuration (MCONFIG) and Controller In-band Interrupt Registry and Rules (MIBIRULES) before using the controller.

41.3.1.3 Protocol modes and states

The following modes are activated in I3C:

- I2C mode is the default mode on startup:
  - If no I2C static address is used, this mode has no effect other than to track frames looking for I3C transitional frames.
  - If an I2C static address is used, the device can interact with the I2C bus controller using the address.
- When in I2C mode, I3C Transitory Frame mode occurs whenever the I3C broadcast address is received (7Eh). In particular:
  - This mode occurs when 7Eh is broadcast followed by SETDASA from the controller. If there is a static address match, the target is assigned a dynamic address and it enters I3C SDR mode.

— This mode occurs when 7Eh is broadcast followed by ENTDAA from the controller. If the target can send a 48-bit provisioned ID (48b ID), a dynamic address is assigned, and the target enters I3C SDR mode.

— If a Hot-Join arbitrated event occurs (sending 02h when the controller generates another address such as 7Eh), there is a conceptual Hot-Join mode. ENTDAA or SETDASA sets a dynamic address to resolve the event, and the target enters I3C SDR mode.

NOTE

An I3C target can operate as a normal I2C target only when it has a static address. Otherwise, it matches nothing in I2C and waits for the aforementioned events.

- I3C SDR mode is the standard mode after I3C activates, which is defined by a dynamic address being assigned. This mode is the resting mode of I3C, and all devices are normally in SDR mode:
  — RSTDAA CCC causes an exit from SDR mode and a return to I2C mode. This transition is not normally used.

    RSTDAA can be issued to ensure that all I3C targets participate in an upcoming dynamic address assignment process, before the I3C controller starts assigning the target address.

  — SETNEWDA does not affect the SDR mode; it only changes the dynamic address of an I3C device that already had a dynamic address assigned.

  — When in SDR mode, the device ignores messages to or from its original I2C static address.
- The I3C CCC command submode of type Direct or Broadcast. Following are the options to exit the CCC:
  — Exit the Broadcast CCC submode by a Repeated START or by a STOP.

  — Exit the Direct CCC submode by a 7Eh broadcast address after a Repeated START or by a STOP.

  — I3C dynamic address assignment (DAA) CCC command enters DAA mode. This mode is a special mode for dynamic addressing. Use STOP to exit. If a target has a dynamic address, the command is ignored.
- I3C SETDASA CCC command enters Static Address Match submode, which allows matching the static address of the device (if any). The command is ignored when a target has a dynamic address, or when the target in I2C mode has no static address.

NOTE

The special point-to-point address is also matched.

- I3C HDR modes are activated by ENTHDR*n* CCC commands, where *n* represents the type of HDR (0 to 7). This mode is valid until an HDR exit pattern is detected, irrespective of whether HDR mode for a target is supported.

  Exit-pattern detection must always be on to prevent errors. Alternatively, you could set it to be activated only on ENTHDR.
- A special HDR-DDR syntax is used for transmitting CCC in the HDR-DDR protocol.
- Internal states such as IBI or no-IBI and Low-Power mode are flagged internally (see SSTATUS[EVDET] for more information). These states are exported to the application and affect the engine to restrain it from performing a prohibited operation.

41.3.1.4 Address match

For an address match to occur, the target matches the I3C broadcast address and either the I2C-style static address or I3C dynamic address (see Table 307).

The address match is inactive (just listening) for all Repeated START commands and also for the START command, unless an IBI, CR, or Hot-Join has been activated. If inactive, it simply tries to match. If not matched, it waits for a Repeated START or STOP command. If an IBI, CR, or Hot-Join has been activated, the arbitration mechanism is used for START (but never for a Repeated START).

When matching the 7Eh broadcast address, it looks for a CCC until the next Repeated START or STOP command.

Table 307. Address match conditions

For	Match condition occurs when	Notes
I3C broadcast address	The address is 111 1110b (written as 7Eh).	
I2C-style static address	• The device has a static address. • The device is not in I3C mode.	When in I3C mode, the address matches only until the ENTDAA, SETDASA, or SETAASA command has a dynamic address assigned. SETDASA matches the static address or the special one-controller-to-one-target point-to-point address.
I3C dynamic address	ENTDAA or SETDASA assigns it, or SETNEWDA modifies it.	

41.3.2 Operations

This section describes the operations of the module.

41.3.2.1 Reading and writing I2C messages using the normal method

1. Set up interrupts using the following fields of Controller Interrupt Set (MINTSET):
   - MCTRLDONE: Indicates when the module completes an MCTRL request.
   - COMPLETE: Indicates when data is finished sending or being received.
   - RXPEND: Used for read operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use this field to allow DMA to read out data.
   - TXNOTFULL: Used for write operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use this field to allow DMA to supply data.
   - IBIWON: Indicates that an IBI, CR, or HJ has won the arbitration on a header address.
   - Additionally, Controller Errors and Warnings (MERRWARN) indicates the causes of errors and warnings.
2. Configure the following fields in Controller Control (MCTRL) simultaneously:
   a. Write 1 to REQUEST (EmitStartAddr).
   b. Write 1 to TYPE (I2C).
   c. Configure IBIRESP to respond to IBIs in your chosen manner.
   d. Write 1 to DIR for read, or write 0 for write.
   e. Write the static address of the I2C target to ADDR.
   f. Configure RDTERM to the maximum length for read operations, to auto-terminate, or set it to STOP as the data is read out. For example, write 1 (with MCTRL[REQUEST] = 0) to stop after the next character.
3. Write or read the data.
   - For write operations, write to Controller Write Data Byte (MWDATAB) for each byte before the last byte, and then write to Controller Write Data Byte End (MWDATABE) for the last byte:
     — You can perform or start this operation before setting up interrupts.
     — If there is more data than the FIFO can hold, use the TXNOTFULL interrupt based on the trigger level. This interrupt allows the application to provide more data or to use DMA.

- For read operations, wait for RXPEND, and then read out data via Controller Read Data Byte (MRDATAB). DMA may also be used.

4. After message transfer is complete (MSTATUS[COMPLETE] = 1), the message may end with a STOP condition, or a new message may start with a Repeated START condition:
   a. Write 2 to MCTRL[REQUEST] (EmitStop to STOP). Then wait for the MCTRLDONE status to be asserted for its completion. (When sending STOP in I2C mode, MCONFIG[ODSTOP] and MCTRL[TYPE] must be 1.)
   b. Write 1 to MCTRL[REQUEST] (EmitStartAddr to restart). IBI is not possible in this case.

NOTE

I2C Fm and I2C Fm+ modes are supported for legacy I2C devices as per the I3C standard specifications. See I2C configuration to meet timing requirements for Fm and Fm+ modes for more information.

41.3.2.2 Reading and writing I3C messages using the normal methods (SDR and HDR-DDR)

The normal method for I3C is the same as the method for I2C with a few differences:

1. Set up interrupts using the following fields of Controller Interrupt Set (MINTSET):
   - MCTRLDONE: Indicates when the I3C module completes an MCTRL request.
   - COMPLETE: Indicates when data is finished sending or being received.
   - RXPEND: Used for read operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use it to allow DMA to read out bytes.
   - TXNOTFULL: Used for write operations. Works with Controller Data Control (MDATACTRL) to set the FIFO trigger. You can also use it to allow DMA to supply bytes.
   - IBIWON: Indicates that an IBI, CR, or HJ has won the arbitration on a header address.
   - Additionally, Controller Errors and Warnings (MERRWARN) indicates the causes of errors and warnings.
2. Set up Controller Control (MCTRL).
   - Option 1: Configure the following fields of Controller Control (MCTRL) simultaneously in this way:
     a. Write 1 to REQUEST (EmitStartAddr).
     b. Write 0 to TYPE for I3C Single Data Rate (SDR) mode or write 2 for Double Data Rate (DDR) mode.
     c. Configure IBIRESP to respond to IBIs in your chosen manner.
     d. Write 1 to DIR for read, or write 0 for write.
     e. Write the dynamic address of the I3C target to ADDR.
     f. For read operations, you can configure RDTERM to the maximum length to terminate automatically.
     g. For write operations, prewriting the data (MWDATAB or MWDATAH) is preferred to ensure that there are no time delays waiting for the data.

        For DMA with MCTRL, use either Controller Write Byte Data 1 (to Bus) (MWDATAB1) or Controller Write Halfword Data (to Bus) (MWDATAH1).

     NOTE

     HDR-DDR mode requires writing an 8-bit command value for read or write. This value must be written into the TX FIFO via Controller Write Data Byte (MWDATAB). The END field is not used for this byte.

   - Option 2: This option is preferred when stopped (bus free condition) in SDR mode, and not in HDR-DDR mode. It allows any target to issue an IBI and avoids collisions with an IBI address. Also, it is faster (when the MSB of the dynamic address is always 0).

     Configure the following fields of Controller Control (MCTRL) in this way:

a. Write 1 to REQUEST (EmitStartAddr). No prewritten transmit data can be in the FIFO.
b. Write 0 to TYPE (I3C).
c. Configure IBIRESP to respond to IBIs in your chosen manner.
d. Write 0 to DIR.
e. Write 7Eh to ADDR.
f. Wait for MCTRLDONE (via interrupt, for example), then proceed as in option 1. The option 2 method has advantages for IBIs when 7Eh is sent on START (but not on Repeated START conditions).

3. Write or read the data.
   - For write operations, write to MWDATAB for each byte before the last byte, and then write to Controller Write Data Byte End (MWDATABE) for the last byte. For HDR-DDR, the byte with END must be even (second, fourth, sixth, and so on) because DDR uses byte pairs:
     — You can perform or start this operation before step 1 (REQUEST = 1) of the option 1 method, but not before the option 2 method.
     — If there is more data than the FIFO can hold, use the TXNOTFULL interrupt based on the trigger level. This interrupt allows the application to provide more data or to use DMA.
   - For read operations, wait for RXPEND, and then read out data via Controller Read Data Byte (MRDATAB) or Controller Read Data Halfword (MRDATAH). DMA may also be used. When using DMA, read using the same registers.
4. After message transfer is complete (MSTATUS[COMPLETE] = 1), the message may be ended with STOP (or EXIT in HDR mode). Alternatively, a new message may be started with a Repeated START (or HDR-Restart in HDR mode):
   - In SDR mode, write 2 to MCTRL[REQUEST] (EmitStop to STOP). Then wait for the MCTRLDONE status to be asserted for its completion.
   - For HDR mode, write 6 to MCTRL[REQUEST] (ForceExit) to end HDR mode. Then wait for the MCTRLDONE status to be asserted for its completion. When sending the HDR exit pattern, MCONFIG[ODSTOP] must be 0.
   - Write 1 to MCTRL[REQUEST] (EmitStartAddr to start another message). IBI is not possible in this case.

41.3.2.3 Determining bus types and modes

The settings of MCTRL[TYPE] and MCTRL[REQUEST] determine bus types and modes.

Table 308. Determining bus types and modes

Value of MCTRL[TYPE]	Meaning when MCTRL[REQUEST] = 1 (EmitStartAddr)	Meaning when MCTRL[REQUEST] = 2 (EmitStop)	Meaning when MCTRL[REQUEST] = 6 (ForceExit)
0	SDR mode of I3C	SDR mode of I3C	Exit pattern
1	Standard I2C protocol	Standard I2C protocol	Reserved
2	The bus enters DDR mode (7E and then ENTHDR0), if the module is not in DDR mode already. The first byte written to the transmit FIFO must be a command and already in the FIFO. To end DDR mode, use ForceExit.	Reserved	Target reset
3	If not already in HDR-BT, the bus automatically enters BT mode (7E	Reserved	Reserved

Table continues on the next page...

Table 308. Determining bus types and modes (continued)

Value of MCTRL[TYPE]	Meaning when MCTRL[REQUEST] = 1 (EmitStartAddr)	Meaning when MCTRL[REQUEST] = 2 (EmitStop)	Meaning when MCTRL[REQUEST] = 6 (ForceExit)
	and then ENTHDR3). You must also configure the MHDRBTCFG register for multilane and CMD rules.		

Table 309. Deriving the dynamic address

Value of MCTRL[TYPE]	Meaning when MCTRL[REQUEST] = 4 (ProcessDAA)
0	The dynamic address is derived from MWDATAB[VALUE]; in case of a NACK, ProcessDAA continues to the next 7E/R request.
1	Reserved
2	The dynamic address is derived from MWDATAB[VALUE]; in case of a NACK, a STOP condition is sent after the assignment.
3	Reserved

41.3.2.4 Sending a CCC to I3C targets

The normal common command code (CCC) method uses an I3C write operation with an address of 7Eh (the CCC is the first byte):

- For broadcast type, any remaining bytes are sent with the CCC. This operation ends with a STOP or a Repeated START and 7Eh.
- For direct type, only the CCC byte is sent (or by a CCC and a defining byte, if required by the CCC).

  These bytes are followed by a Repeated START and the address of the I3C target (for SETDASA, this address is its I2C static address). This sequence may be repeated with more Repeated START conditions and addresses until done. It may end in STOP or a Repeated START and 7Eh. After the Repeated START and target address, the values are read or written depending on the CCC.
- I3C provides interrupts (by hardware) for unhandled and handled CCC when received by the target. Its state is reflected in Target Status (SSTATUS).

I3C controllers require to emit a single START, 7E/W sequence with both SCL high and SCL low half periods at full open-drain timing (for example, 200 ns). This sequence allows I3C targets acting as I2C legacy devices to turn off their I2C 50-ns spike filters, if they have them.

NOTE

START, 7E/W is a notation indicating a START command, followed by the 7Eh broadcast address and then by a write command.

After that sequence, addresses following START may be sent with Open-Drain Low (for example, 200 ns) but high of Push-Pull timing (for example, 40 ns). The I3C controller must emit this START, 7E/W sequence with MCONFIG[ODHPP] = 0. ODHPP is Open-Drain High period at Push-Pull speeds, and MCONFIG[ODHPP] is usually 1.

NOTE

- MCONFIG[ODHPP] is ignored when sending a message to an I2C legacy device on an I3C bus.
- Each Repeated START is a new EmitStart request. This request can be chained by an interrupt or pushed by a message model using DMA.

41.3.2.5 IBI handling

An IBI occurs when a target sends its address after a START, and that address is numerically the lowest. That is, it is lower than the address sent by the controller and addresses sent by any other targets. When the controller sends 7Eh, the controller always loses the arbitration.

The IBI can occur unexpectedly when any new START (not a Repeated START) is sent. The IBI can also occur in response to a target pulling SDA low. This condition can occur in one of these ways:

- The controller has set the request to AutoIBI mode, so the IBI occurs automatically. The controller sends 7Eh to allow the target to win the arbitration.
- The application receives the SLVSTART (target START request) interrupt, so it sends 7Eh.

The IBI response is configured by MCTRL[IBIRESP], which can be set to:

- NACK (always reject the IBI).
- ACK (always accept the IBI):
  - You must configure Controller In-band Interrupt Registry and Rules (MIBIRULES) so that the engine knows whether IBI bytes follow.
  - If IBI bytes follow (also known as IBI mandatory byte), then the COMPLETE field does not become 1 when IBIWON becomes 1. RXPEND becomes 1 for one or more bytes in the receive FIFO. When the last byte is received, then the COMPLETE field becomes 1. The controller automatically stops IBI data after 9 bytes (including the mandatory data byte). MCTRL[RDTERM] can be used with MCTRL[REQUEST] = 0 to end the process of receiving data sooner. I3C supports address ACK to mandatory byte transition during IBI from Open_Drain (controller acknowledges the target address) to Push Pull (target sends SDR data).
  - I3C target supports up to 7 bytes (maximum limit is defined by IBIEXT1[MAX]) of extended IBI data following a mandatory data byte. Extended data to send, if any, is present in Extended IBI Data 1 (IBIEXT1) and Extended IBI Data 2 (IBIEXT2).
- Manual (allow the decision to be made by the application on a case-by-case basis):
  - The application rewrites it when stopped, pending an IBI.
  - The application can ACK or NACK the IBI based on the IBI address defined in MSTATUS.
  - This mode selects whether there is an IBI byte when accepting the IBI request from targets.

Accurate timestamping of I3C target data is supported in I3C though Async mode 0. In I3C Asynchronous Timing Control mode, a target device timestamp event occurs within that target. The target notifies the controller about the event by generating an IBI.

41.3.2.6 Assigning dynamic addresses to I3C devices

If dynamic addresses (DAs) are all assigned below 40h (7-bit values from 3Fh down to 03h, except where not allowed), then MIBIRULES[MSB0] can be 1. This setting optimizes the START timing. When dynamic addresses are assigned, you must program Controller In-band Interrupt Registry and Rules (MIBIRULES) based on which I3C target uses IBI bytes (present in SIDEXT[BCR]).

Any SETDASA assignments can be performed via the normal CCC model. These assignments cannot be performed when using the static address for the directed part.

There is a built-in mechanism to process DAA mode:

1. Set up interrupts using the following fields of Controller Interrupt Set (MINTSET):
   - MCTRLDONE: Indicates when a target has sent its ID and BCR or DCR, and a new DA is needed.
   - COMPLETE: Indicates when DAA is done (NACKed by all targets).
   - RXPEND: Indicates the reading of the IDs of the targets. You can also use this field to allow DMA to read out bytes.
   - IBIWON: Indicates when an IBI has occurred (normally there are no IBI conditions when assigning dynamic addresses to the I3C device).

- Additionally, Controller Errors and Warnings (MERRWARN) indicates the causes of errors and warnings.

2. Configure Controller Control (MCTRL):
   a. Write 4 to MCTRL[REQUEST] (ProcessDAA).
   b. Set MCTRL[IBIRESP] to IBI response, if possible. If not possible, set to the first target assignment.
3. Wait for the MCTRLDONE interrupt when reading ID using the RXPEND interrupt. If MSTATUS[STATE] = 5 (DAA mode) and MSTATUS[BETWEEN] = 1, it is waiting to write a dynamic address:
   a. Write the dynamic address into Controller Write Data Byte (MWDATAB) using bits 6:0, such as 14h for DA = 7'h14.
   b. Write 4 to MSTATUS[STATE] (ProcessDAA again). This option writes the DA, then moves to the next DA.
   c. If MSTATUS[COMPLETE] = 1 and MSTATUS[STATE] = 0, then all targets are assigned.
   d. If MSTATUS indicates NACK after writing a new DA, the DA is not accepted by the target. The next step is to write 2 to MCTRL[REQUEST] (EmitStop) and start over. It is also acceptable to write 4 to MCTRL[REQUEST] (ProcessDAA again).

41.3.2.7 Using Controller Message mode

Controller Message mode is intended for use with DMA although Message mode can also be used by the processor. In Message mode, all writes are to the same location, including control and data. Reads occur from an associated location.

NOTE

The data writes for Message mode work in the same way as the halfword access registers, Controller Write Data Halfword (MWDATAH) and Controller Read Data Halfword (MRDATAH).

To send a message via Single Data Rate (SDR), follow these steps (from DMA or processor):

1. Write the control request to Controller Write Message Control in SDR mode (MWMSG_SDR_CONTROL). This request includes the following items:
   - The address.
   - I2C or I3C.
   - Read or write.
   - The count of bytes to process.
   - How to end (on STOP or ready for Repeated START).
2. Process the data:
   - For write operations, write the rest of the data to Controller Write Message Data in SDR mode (MWMSG_SDR_DATA). Use the DMA trigger (or interrupt) to keep the transmit FIFO full, and the controller stops using the data when it has already consumed the number of bytes configured in the LEN field.
   - For read operations, a DMA trigger (or interrupt) indicates when the RX FIFO is ready to be read out via Controller Read Message in SDR mode (MRMSG_SDR).
3. Select the in-band interrupt (IBI) behavior in MCTRL[IBIRESP].
4. Use END type STOP (MCTRL[REQUEST] = 2) with a zero-length message or by using Controller Control (MCTRL). To exit Message mode with the original message, use END type STOP (MCTRL[REQUEST] = 2) with a zero-length message or by using the MCTRL register.

To send a message via Double Data Rate (DDR), follow these steps (from DMA or processor):

1. Write the control request to Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL). This request includes:
   - The count of byte pairs.

- How to end the message (through HDR exit or to wait for HDR restart).

2. Write the second control data to MWMSG_DDR_CONTROL. This second write includes:
   - The address.
   - Read or write.
   - The 7-bit command value.
3. Process the data:
   - For write operations, write the rest of the data to MWMSG_DDR_DATA. Use the DMA trigger (or interrupt) to keep the transmit FIFO full, and the controller stops using the data when the program count MWMSG_DDR_CONTROL register reaches 0.
   - For read operations, a DMA trigger (or interrupt) indicates when the receive FIFO is ready to be read out.
4. Select in-band interrupt (IBI) behavior in MCTRL[IBIRESP].
5. Use END type exit (MCTRL[REQUEST] = 6) with a zero-length message or by using the MCTRL register to exit DDR Message mode with the original message.

41.3.2.8 Handing off controllership to another target and getting it back

To hand off controllership, the controller can perform either of the following actions:

- Wait for a controller request (CR) (that is, MSTATUS[IBIWON] = 1 interrupt), indicating that the CR is using MSTATUS[IBITYPE].
- Push the request manually.

In either case, the controller sends a GETACCMST request, which is a directed GET informing the target that it is being assigned controllership. If the target accepts this request, it returns its dynamic address in bits 7:1 and the negative parity of its dynamic address in bit 0. A STOP must be issued, and MCONFIG[MSTENA] must be set to 2 (switching to Target mode).

To gain controllership, the target sends a CR using Target Control (SCTRL):

- If MCONFIG[MSTENA] is 2, the GETACCCR CCC is ACKed.
- If MCONFIG[MSTENA] is not 2, the GETACCCR CCC is NACKed.

After controllership is granted, MSTATUS[NOWMASTER] becomes 1. The application must enable MINTSET[NOWMASTER], so the application is interrupted when a controllership transfer occurs.

41.3.2.9 Controller engine flow diagram

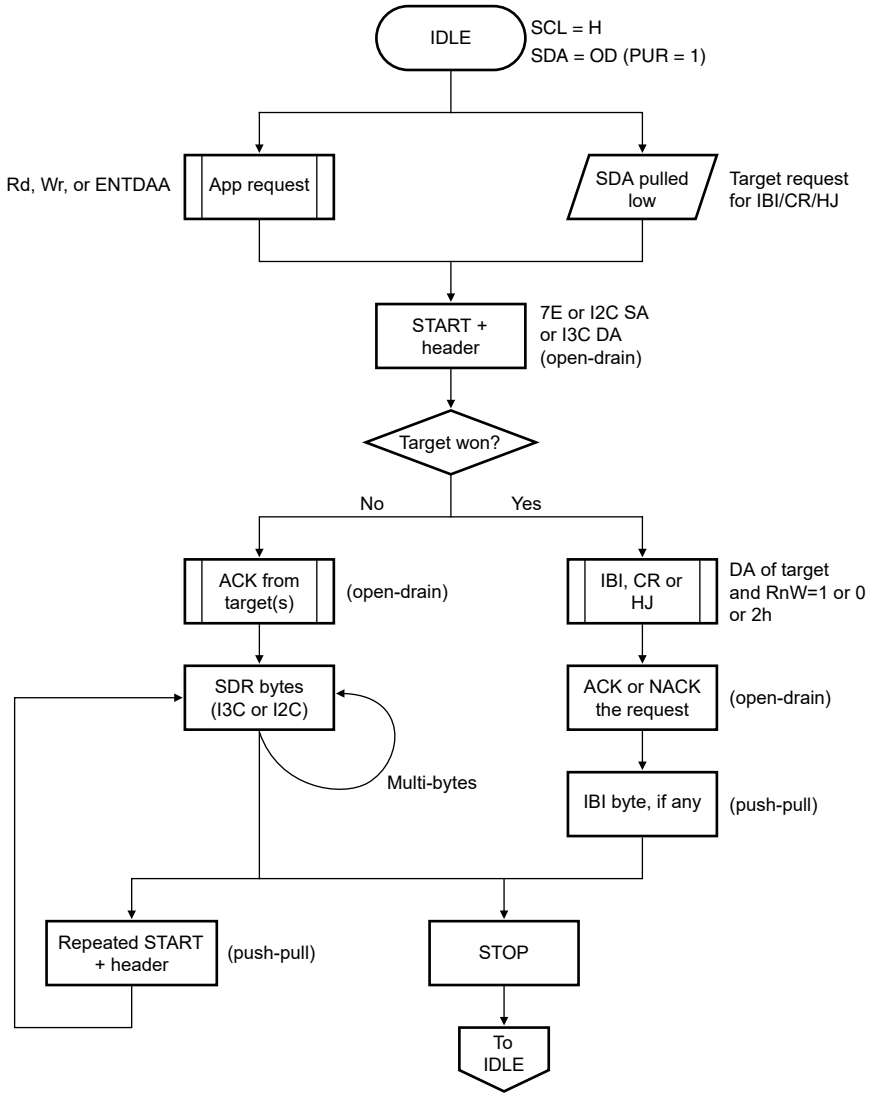


Figure 176. Controller engine flow diagram for SDR (HDR flow progresses from SDR bytes)

41.3.2.10 Target receives data from controller

When the address match is valid for the target dynamic address and the type is W (write from controller), the target ACKs the address. It does so unless some condition prevents it, such as a full buffer. On ACK, the Write state is entered. Once ACKed, it waits for each complete data byte. This process occurs in two steps:

1. Eight bits are clocked in and stored in the next buffer location when in the Write state.
2. On the ninth bit:
   - In I3C mode, the W9TH state (in I3C, the ninth data bit written by the controller is the parity of the preceding eight data bits) is used and the parity is checked. The buffer is marked as complete with or without a parity error.
   - In I2C mode, the W9TH state (in I2C, the ninth data bit written by the controller is an ACK by the target) is used and the data is ACKed and stored.

When the address match is valid for the 7Eh broadcast address, the target acknowledges it. After first data completion, it sets the in_ccc broadcast or direct flag (based on bit 7 of the command). If recognized, it parses the command, setting that bit as well. The

recognized commands are for dynamic address work and modes. The CCC and DAA blocks handle the workload afterward. If not supported, then the commands are passed up to the system.

41.3.2.11 Target sends data to controller

When the address match is valid for the target dynamic address, and the type is R (read by controller), the target ACKs the address. The target does so unless there is no data waiting for the read. On ACK, the Read state is entered.

After being ACKed, it emits the data byte at a time, with the ninth bit using the R9TH state. In I2C, the ninth data bit from target to controller is an ACK by the controller. In I3C, this bit allows the target to end a read, and allows the controller to abort a read:

- In I3C mode, the T bit is emitted to allow the device to indicate whether this byte is the last. If it is not the last byte, the controller may terminate the read via the T bit.
- In I2C mode, the ninth bit allows the controller to terminate via NACK, or else it allows the read to continue via ACK. On completion of each byte read, the "done" signal is pulsed to get the next byte.

If the read is terminated by the controller unexpectedly (abort in I3C, NACK before END marked), the application is notified.

41.3.3 Clocking

The controller works on the following primary clocks:

- The system clock, which controls access to the memory-mapped registers.
- A functional clock (FCLK), which is used to generate the SCL clock rate on the I2C or I3C bus.

I3C supports adjusting clock frequency and accuracy via Target Time Control Clock (STCCLOCK).

FCLK is used to check conditions such as bus availability, bus free for IBI, or HJ:

- To determine the Bus Available condition for IBI, the target needs a clock to generate the ~1 µs timing. To support this requirement, a slow clock (CLK_SLOW) is provided to save on power.
- I3C supports a counter field, SCONFIG[BAMATCH], to calculate the Bus Available condition. BAMATCH provides the count of the slow clock . This field counts 1 µs or more to allow an IBI to drive SDA low when the controller is free. The I3C module controls the maximum width and maximum values.

I3C supports a timeout when stalled for too long in a frame. This timeout occurs when:

- The transmit FIFO or receive FIFO is not handled and the bus is stuck in the middle of a message.
- No STOP is issued and the bus is between messages.
- IBI manual is used and no decision is made.

41.3.4 Reset

- Global or system reset fed into the module: Everything is reset by this global or system reset. Release is assumed to be synchronized with the system clock. It is not synchronized with SCL or SDA. These signals must not be active when the reset is released; otherwise, I3C enters Hot-Join mode and remains inactive.
- Software reset

41.3.5 Interrupts

This section describes all the interrupts (IRQs) that this module generates. All status interrupts are updated in Controller Status (MSTATUS) and Target Status (SSTATUS).

Supported interrupts occur:

- For pending IBI, CR, or Hot-Join that have been sent by a target.
- When a CCC is received and is handled by the module (SSTATUS[CHANDLED]).

- When an error or warning has occurred, such as data underrun, data overrun, parity error, HDR-DDR, or CRC error.

41.4 External signals

Table 310. External signals

Signal	Description	Direction
SCL	Serial clock	Input or output
SDA	Serial data	Input or output
PUR	Pull up resistance. There is an internal pullup resistance on SDA, which the I3C controls. If the internal pullup is not enough, PUR can control an external pullup resistance on SDA actively. See the chip-specific I3C information on whether this external pullup is available on your chip.	Output

41.5 Initialization

41.5.1 Controller configuration

Certain configuration is performed when initializing this module. Controller Configuration (MCONFIG) controls the frequencies, duty cycle, optimizations for performance, and other parameters.

Table 311. Configuration for module initialization

Configuration parameter		Description
MSTENA	Controller enable	Determines whether the peripheral starts in Controller mode (main controller in I3C terms) or starts in Target mode (and switches to Controller mode later).
HKEEP	High-Keeper	Determines how the high-keeper (weak pullup) is 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 frequency sets the SCL half-clock period baseline (used at least for the high time of SCL). The formula for SCL in Push-Pull mode using PPBAUD and PPLOW is: SCL frequency (in MHz) = FCLK ÷ ((2 + 2 × PPBAUD) + PPLOW) If PPLOW is 0, then SCL high = SCL low (in ns) = (1 + PPBAUD) × 1000 ÷ FCLK (in MHz) Examples: • If FCLK = 24 MHz, then PPBAUD = 0 yields 12 MHz (42.67 ns per half period) • If FCLK = 50 MHz, then 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 FCLK cycles to use for low. The formula for SCL in Push-Pull mode using PPBAUD and PPLOW is: SCL frequency (in MHz) = FCLK ÷ ((2 + 2 × PPBAUD) + PPLOW) If PPLOW is a nonzero value, then: • SCL high (in ns) = (1 + PPBAUD) × 1000 ÷ FCLK (in MHz)

Table continues on the next page...

Table 311. Configuration for module initialization (continued)

Configuration parameter		Description
		• SCL low (in ns) = (1 + PPBAUD + PPLOW) × 1000 ÷ FCLK (in MHz) For example, when FCLK is 50 MHz, PPBAUD is 1, and PPLOW is 1, then the periods are: • (1 + 1) × 1000 ÷ 50 = 20 + 20 = 40 ns high • (1 + 1 + 1) × 1000 ÷ 50 = 20 + 20 + 20 = 60 ns low This timing is equivalent to 10 MHz SCL, but this timing maintains the 40 ns high needed; therefore, I2C devices do not see the high periods. **NOTE** The PPLOW value does not have any impact on Open-Drain mode and I2C mode SCL rate calculations.
ODBAUD	Open-Drain baud	Determines the number of PPBAUD periods to make up one I3C Open-Drain half-clock baseline. If ODHPP is 0, the ODBAUD half-clock time period = (ODBAUD + 1) × PPBAUD high period. If ODHPP is 0, the formula for SCL in Open-Drain mode is: SCL (in MHz) = FCLK ÷ (2 + 2 × PPBAUD) × (ODBAUD + 1) Target to get open-drain half-clock period is 200 ns. For example: • If PPBAUD yields 12.5 MHz (40 ns per PPBAUD period), then ODBAUD = 4 can be used to get 200 ns. See ODHPP for details on short high and long low. • If PPBAUD = 1, FCLK = 50 MHz, and ODBAUD = 4, then open-drain SCL = 50 ÷ (2 + 2 × 1) × 5 = 2.5 MHz.
ODHPP	Open-Drain High Push-Pull	Optional field that allows the I3C open drain to be long low and short high. The high period of SCL is the PPBAUD period. This period leaves enough time for the pullup resistor to pull the SDA high when SCL is low. It is also quick when SCL is high and there are no changes happening. ODBAUD low half-clock time period = (ODBAUD + 1) × PPBAUD high period. ODBAUD high half-clock time period = PPBAUD high period If ODHPP is 1, the formula for SCL in Open-Drain mode is: SCL (in MHz) = FCLK ÷ (1 + PPBAUD) × (ODBAUD + 1) + (1 + PBAUD) For example: • If PPBAUD produces 12.5 MHz (40 ns per PPBAUD period), then ODBAUD = 4 and ODHPP = 1. These settings provide a high period of 40 ns and a low period of 200 ns. • If PPBAUD = 1, FCLK = 50 MHz, and ODBAUD = 4, then open-drain SCL = 50 ÷ (1 + 1) × 5 + (1 + 1) = 4.16 MHz.
I2CBAUD	I2C Baud	Indicates the number of ODBAUD periods that are required to communicate with I2C devices (MCTRL[TYPE] = 2 or MWMSG_SDR_CONTROL[I2C] = 1). For example, if ODBAUD gives 200 ns, and the goal is Fm+ (Fast Mode, 1 MHz), then the sum must be 1 µs.

Table continues on the next page...

Table 311. Configuration for module initialization (continued)

Configuration parameter		Description
		I2CBAUD acts differently for odd and even values. For example: • If I2CBAUD = 3, it gives three ODBAUD periods low and two ODBAUD periods high. • If I2CBAUD = 4, it gives three ODBAUD periods for low and three ODBAUD periods for high. • Also, if I2CBAUD = 3, this yields 200 × 3 = 600 ns and 200 × 2 = 400 ns, with the sum 600 + 400 = 1000 ns = 1 µs.
SKEW	Skew	The normal SDA skew from SCL is handled by using the time for the SCL to reach its pad and return. This time is normally 2 ns to 5 ns (or sometimes more). If the skew is too fast, then add more delay. The skew allows specifying the number of FCLK to insert.

Additional optimizations:

- Use MIBIRULES[MSB0] to obtain faster start header times. If the controller application assigns all I3C dynamic addresses to be less than 40h (it does not have MSB set), you can write 1 to MIBIRULES[MSB0]. When the controller emits 7Eh (broadcast) and a target does not drive the first bit low, the rest of the header can be at push-pull speeds. This speed is two times faster or more, depending on optimizations.
- Auto-emit 7Eh speeds up the frame when used with MIBIRULES[MSB0]. It allows the processor to sleep when the frame starts automatically (in response to a target).

41.5.2 Interrupt service flow

Set up interrupts for Controller Status (MSTATUS) in Controller Interrupt Set (MINTSET).

Set up interrupts for Target Status (SSTATUS) in Target Interrupt Set (SINTSET).

41.6 Application information

41.6.1 I2C configuration to meet timing requirements for Fm and Fm+ modes

I2C Fm and I2C Fm+ modes are supported according to the I3C standard specifications.

Table 312 shows the configuration for Fm mode, where frequency is close to 400 kHz but timing requirement is not met (T_{SU_STA} is not met) because the actual value is 600 ns. Rest of the requirements are met per the I2C specifications in the chip data sheet.

Table 313 shows the configuration for Fm mode, where all timing requirements are met.

Table 312. Configuration for Fm mode with not all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	0	4	11	370 kHz	T_{SU_STA} - 353 ns

Table 313. Configuration for Fm mode with all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	1	2	9	363 kHz	All requirements met per the I2C specifications in the chip data sheet

Table 314 shows the configuration for Fm+ mode, where the frequency is close to 1 MHz. In this case, the T_{SU_STA} timing requirement is not met because the actual value is 260 ns. Rest of the requirements are met per the I2C specifications in the chip data sheet.

Table 315 shows the configuration for Fm+ mode, where all timing requirements are met. To meet all the timing specifications, frequency may differ from 1 MHz. Use the configurations listed in Table 315.

Table 314. Configuration for Fm+ mode with not all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	0	4	3	960 kHz	T_{SU_STA} - 228 ns

Table 315. Configuration for Fm+ mode with all timing requirements met

FCLK	PPBAUD	ODBAUD	I2CBAUD	SCL frequency	Timing requirement status
24 MHz	0	3	6	749 kHz	All requirements met per the I2C specifications in the chip data sheet

NOTE

The controller module provides an option to slow down the clock for standard speed mode to support legacy I2C standard targets as well (although this option may not necessarily meet the exact timing requirement of the standard mode according to the standard I2C timing specification).

For the timing requirements under consideration, see the "I3C timing requirements when communicating with I2C legacy devices" table in the MIPI I3C v1.1.1 specification available on www.mipi.org.

41.7 I3C register descriptions

Writing to a read-only (RO) register, except to Controller Interrupt Mask (MINTMASKED), causes bus errors. This module does not verify whether programmed values in the registers are correct. You must ensure that valid programmed values are written.

However, the peripheral ignores writes to RO registers and returns 0 for reads from WO or nonexistent registers. It requires all 32-bit registers to be read and written as 32-bit and aligned to 32 bits. The peripheral uses the advanced peripheral bus (APB: AMBA 3 APB Protocol Specification v1.0); therefore, the peripheral does not know whether partial read or write operations (less than 32 bits) are used.

41.7.1 I3C memory map

I3C0 base address: 4000_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Controller Configuration (MCONFIG)	32	RW	0000_0000h
4h	Target Configuration (SCONFIG)	32	RW	0017_0000h
8h	Target Status (SSTATUS)	32	RW	0000_1400h
Ch	Target Control (SCTRL)	32	RW	0000_0000h
10h	Target Interrupt Set (SINTSET)	32	RW	0000_0000h
14h	Target Interrupt Clear (SINTCLR)	32	RW	See section

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Target Interrupt Mask (SINTMASKED)	32	R	0000_0000h
1Ch	Target Errors and Warnings (SERRWARN)	32	RW	0000_0000h
20h	Target DMA Control (SDMACTRL)	32	RW	0000_0010h
2Ch	Target Data Control (SDATACTRL)	32	RW	8000_0030h
30h	Target Write Data Byte (SWDATAB)	32	RW	See section
34h	Target Write Data Byte End (SWDATABE)	32	RW	See section
38h	Target Write Data Halfword (SWDATAH)	32	RW	See section
3Ch	Target Write Data Halfword End (SWDATAHE)	32	RW	See section
40h	Target Read Data Byte (SRDATAB)	32	R	0000_0000h
48h	Target Read Data Halfword (SRDATAH)	32	R	See section
54h	Target Write Data Byte (SWDATAB1)	32	RW	0000_0000h
54h	Target Write Data Halfword (SWDATAH1)	32	RW	0000_0000h
5Ch	Target Capabilities 2 (SCAPABILITIES2)	32	R	0000_0300h
60h	Target Capabilities (SCAPABILITIES)	32	R	E83F_FE70h
64h	Target Dynamic Address (SDYNADDR)	32	RW	0000_0000h
68h	Target Maximum Limits (SMAXLIMITS)	32	RW	0000_0000h
6Ch	Target ID Part Number (SIDPARTNO)	32	RW	3000_0000h
70h	Target ID Extension (SIDEXT)	32	RW	0066_EF00h
74h	Target Vendor ID (SVENDORID)	32	RW	0000_011Bh
78h	Target Time Control Clock (STCCLOCK)	32	RW	0000_3014h
7Ch	Target Message Map Address (SMSGMAPADDR)	32	R	0000_0000h
80h	Controller Extended Configuration (MCONFIG_EXT)	32	RW	0000_0000h
84h	Controller Control (MCTRL)	32	RW	0000_0000h
88h	Controller Status (MSTATUS)	32	RW	0000_1000h
8Ch	Controller In-band Interrupt Registry and Rules (MIBIRULES)	32	RW	0000_0000h
90h	Controller Interrupt Set (MINTSET)	32	RW	0000_0000h
94h	Controller Interrupt Clear (MINTCLR)	32	RW	See section
98h	Controller Interrupt Mask (MINTMASKED)	32	R	0000_0000h
9Ch	Controller Errors and Warnings (MERRWARN)	32	RW	0000_0000h
A0h	Controller DMA Control (MDMACTRL)	32	RW	0000_0010h
ACh	Controller Data Control (MDATACTRL)	32	RW	8000_0030h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B0h	Controller Write Data Byte (MWDATAB)	32	RW	See section
B4h	Controller Write Data Byte End (MWDATABE)	32	RW	See section
B8h	Controller Write Data Halfword (MWDATAH)	32	RW	See section
BCh	Controller Write Data Halfword End (MWDATAHE)	32	RW	See section
C0h	Controller Read Data Byte (MRDATAB)	32	R	0000_0000h
C8h	Controller Read Data Halfword (MRDATAH)	32	R	0000_0000h
CCh	Controller Write Byte Data 1 (to Bus) (MWDATAB1)	32	RW	0000_0000h
CCh	Controller Write Halfword Data (to Bus) (MWDATAH1)	32	RW	0000_0000h
D0h	Controller Write Message Control in SDR mode (MWMSG_SDR_CONTROL)	32	RW	See section
D0h	Controller Write Message Data in SDR mode (MWMSG_SDR_DATA)	32	RW	0000_0000h
D4h	Controller Read Message in SDR mode (MRMSG_SDR)	32	R	0000_0000h
D8h	Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL)	32	RW	See section
D8h	Controller Write Message in DDR Mode Control 2 (MWMSG_DDR_CONTROL2)	32	RW	See section
D8h	Controller Write Message Data in DDR mode (MWMSG_DDR_DATA)	32	RW	0000_0000h
DCh	Controller Read Message in DDR mode (MRMSG_DDR)	32	R	0000_0000h
E4h	Controller Dynamic Address (MDYNADDR)	32	RW	0000_0000h
11Ch	Map Feature Control 0 (SMAPCTRL0)	32	R	0000_0000h
140h	Extended IBI Data 1 (IBIEXT1)	32	RW	0000_0070h
144h	Extended IBI Data 2 (IBIEXT2)	32	RW	0000_0000h
FFCh	Target Module ID (SID)	32	R	EDCB_0100h

41.7.2 Controller Configuration (MCONFIG)

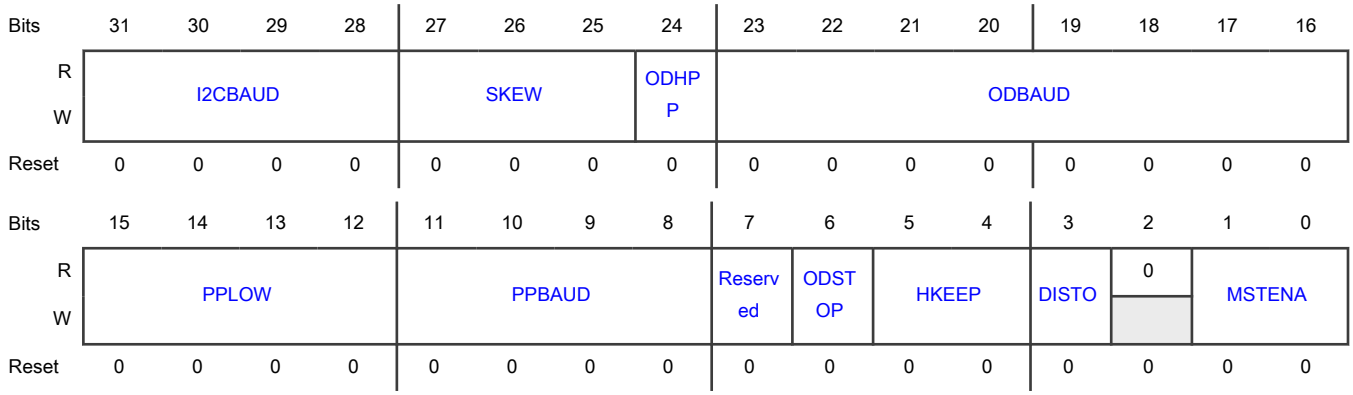
Offset

Register	Offset
MCONFIG	0h

Function

Controls all controller states when the controller operation is enabled. You must not change this register during an active transaction.

Diagram



Fields

Field	Function
31-28 I2CBAUD	I2C Baud Rate Specifies the I2C low and high times in ODBAUD counts. See I2CBAUD in Controller configuration.
27-25 SKEW	Skew Specifies the number of FCLK counts for an SDA change after SCL for I3C push-pull. This skew is in addition to the roundtrip of the SCL line from the pad back to the module (6 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. Use the following values: • In I2C Controller mode (MCONFIG[MSTENA] = 11b) — SKEW > 0 when clock stretching feature is not required. — SKEW = 0 when clock stretching feature is required. • In I3C Controller mode, use SKEW = 0.
24 ODHPP	Open-drain High Push-Pull Enables ODHPP. If you write 0 to this field, open-drain SCL high half-clock period is the same as the open-drain low SCL half period. If you write 1 to this field, open-drain high SCL half-lock period is one PPBAUD count for I3C messages. This setting is faster and works for I3C devices. Any legacy I2C devices on the bus do not see the SCL high level at all (less than the spike filter period). For open-drain timing, check upon the first 7Eh broadcast allowing I3C peripherals acting as I2C legacy devices to turn off their I2C 50-ns spike filters. See Sending a CCC to I3C targets for more information. 0b - Disable 1b - Enable
23-16	Open-drain Baud Rate

Table continues on the next page...

Table continued from the previous page...

Field	Function
ODBAUD	Specifies the open-drain baud rate. ODBAUD is configured in terms of number of PPBAUD counts - 1. This field must not be 0; this setting is not the same as push-pull. See Controller configuration for more information. The open-drain baud rate is usually 200 ns (see I2CBAUD for I2C counts). When used with MCONFIG[ODHPP], this setting produces 250 ns per clock in I3C. In this case, if MCONFIG[PPBAUD] provides 12 MHz, then one PPBAUD count is half of 12 MHz, or 41.67 ns. To obtain a rate around 200 ns, use a value of 5 - 1 = 4 for ODBAUD.
15-12 PPLOW	Push-Pull Low Acts as an adder for push-pull low to create a duty cycle with a longer low period, with up to 15 more FCLK cycles low than high. PPLOW = 0 produces a 50/50 duty cycle.
11-8 PPBAUD	Push-Pull Baud Rate Specifies the push-pull baud rate. The number of FCLK counts makes each push-pull low and normally high period. PPBAUD = 0 when run at 1/2 input FCLK speed. For example, a 24 MHz FCLK produces a 12 MHz SCL because each FCLK is SCL low or SCL high. MCONFIG[PPLOW] has an effect on the duty cycle. For example, 24 MHz with 50/50 duty cycle is 12 MHz. However, when PPLOW adds three more low beats, the push-pull baud rate becomes 4.8 MHz (from 24 MHz or 5 beats).
7 —	Reserved
6 ODSTOP	Open-drain Stop Enables open-drain stop. If you write 0 to this field, open-drain stop is disabled. ODSTOP must be disabled when sending an HDR exit pattern. If you write 1 to this field, open-drain stop is enabled. STOP condition is emitted at open-drain speeds even for I3C messages. In legacy devices, this feature can ensure that the legacy devices see the STOP condition. 0b - Disable 1b - Enable
5-4 HKEEP	High-Keeper Indicates how High-Keeper is supported. If this field is 00b, use pull-up resistor (PUR). No separate pin_HK_SDA or pin_HK_SCL is used. Only pin_PUR_oena is used for SDA. The pin_SCL_oena pin is held high in this mode, SCL clock is push-pull, and pin_SCL_out toggles, which is the actual clock. If this field is 01b, High-Keeper controls are in place; pin_HK_SDA or pin_HK_SCL (High-Keeper) controls are used. SCL may use an HK or that signal (pin_HK_SCL) may only OR into pin_SCL_oena. The pin_HK_SDA pin is used as well for SDA high keeper, along with pin_PUR_oena. Use HKEEP = 10b or 11b for I2C modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Use HKEEP = 00b or 01b for I3C modes. **NOTE** Your chip may not support any or all High-Keeper methods. You must check the availability of High-Keeper methods. 00b - None 01b - WIRED_IN 10b - PASSIVE_SDA (I2C mode, no clock stretches mode). SDA is open drain, and will be Hi-Z (high-impedance) for Bus Free (IDLE) and hold. The pin pin_HK_SDA is not used in this option, only a combination of SDA_oena and pin_PUR_oena are used. SCL is push-pull. The pin pin_HK_SCL is not used in this option, only pin_SCL_oena is used. Since the controller does not support multi-controller mode, there is no need to have open-drain SCL for I2C. For a single controller application, the controller's SCL output can be a push-pull driver design if there are no devices on the bus which would stretch the clock. 11b - PASSIVE_ON_SDA_SCL. For I2C clock stretching mode, where both SCL and SDA are open-drain. Open-Drain on both SDA and SCL, can Hi-Z (high-impedance) both for Bus Free (IDLE), and can Hi-Z SDA for hold. Does not support multi-controller mode, SCL open-drain is only to support clock stretching.
3 DISTO	Disable Timeout Disables the timeout that produces application errors. If the controller is left in a state other than Stopped for more than 100 µs (because 10 kHz is the slowest allowed I3C speed), the timeout sends a MERRWARN interrupt. To prevent the MERRWARN interrupt during development or testing, write 1 to DISTO to disable the timeout. In systems that support timeouts, timeout is disabled automatically during debug. 0b - Enabled 1b - Disabled, if configured
2 —	Reserved
1-0 MSTENA	Controller Enable Indicates whether the controller is enabled and the states it can use. If this field is 0b, the controller is disabled. The module can only use Target mode. If this field is 1b, the controller is enabled. When used upon start-up, this module is the main controller by default. I3C controls the bus unless the controller is handed off. When this happens, MSTENA must become 2 so that it has the capability to become the controller again. Performing the handoff means emitting the GETACCMST CCC command. If the command is accepted, I3C emits a STOP condition and sets this field to 2 (or 0). If this field is 10b, I3C is controller-capable, but is operating as a target now. When used from the start, I3C starts as a target, but is prepared to switch to Controller mode. To switch to this mode, the target

Table continues on the next page...

Table continued from the previous page...

Field	Function
	emits a controller request (CR) or receives a GETACCMST CCC command and accepts it (to switch on the STOP condition). Legacy I2C—if this field is 11b, I2C Controller mode uses an open-drain clock (to allow clock stretching) and relies on passive pull-up resistors (PURs) on both SCL and SDA. This mode uses only I2CBAUD. START requests can only be made for I2C type, and I3C requests are not accepted. If there are only I2C targets on the bus and no clock stretching, use type 1 instead. Type 1 has a push-pull SCL clock, which is faster and uses less power. 00b - CONTROLLER_OFF 01b - CONTROLLER_ON 10b - CONTROLLER_CAPABLE 11b - I2C_CONTROLLER_MODE

41.7.3 Target Configuration (SCONFIG)

Offset

Register	Offset
SCONFIG	4h

Function

Contains fields that must be configured before the module is activated.

Diagram

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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0		OFFLINE	0	0	0		HDROK	S0IGNORE	MATCHSS	NACK	SLVENA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25	Static Address

Table continues on the next page...

Table continued from the previous page...

Field	Function
SADDR	Sets the I2C 7-bit static address, which otherwise must be 0.
24 —	Reserved
23-16 BAMATCH	Bus Available Match Specifies the bus available condition match value for the current slow clock. BAMATCH provides the count of the slow clock to count out 1 µs (or more) to allow an IBI to drive SDA low when the controller is not doing so. I3C controls the maximum width and maximum values.
15-12 —	Reserved
11-10 —	Reserved
9 OFFLINE	Offline Enables wait to ensure that the bus is not in HDR mode. If this field is 1 when SCONFIG[SLVENA] is 1, then I3C waits for either 60 µs of bus quiet or an HDR exit pattern. This waiting ensures that the bus is not in HDR mode, and so can safely monitor the next activity in Single Data Rate (SDR) mode. 0b - Disable 1b - Enable
8 —	Reserved
7 —	Reserved
6-5 —	Reserved
4 HDROK	HDR OK Enables HDR OK. If you write 1 to this field, it allows HDR-DDR and/or HDR-BT messaging, if available, by writing 1 to the corresponding SIDEXT[BCR] field to indicate HDR is available, and the corresponding GETCAPS field for DDR and/or BT bit permitting use. This is a deprecated field; use the SIDEXT[BCR] field instead. 0b - Disable HDR OK 1b - Enable HDR OK
3	Ignore TE0 or TE1 Errors

Table continues on the next page...

Table continued from the previous page...

Field	Function
S0IGNORE	Ignores TE0 or TE1 errors. If you write 1 to this field, the target does not detect TE0 or TE1 errors, so it does not lock up waiting on an exit pattern. You must not use this setting when the bus does not use HDR mode. 0b - Do not ignore TE0 or TE1 errors 1b - Ignore TE0 or TE1 errors
2 MATCHSS	Match Start or Stop Enables matched START, that is Repeated START, or STOP condition. If you write 1 to this field, SINTSET[START] and SINTSET[STOP] become 1 only when SSTATUS[MATCHED] is 1. This setting allows the START and STOP fields to be used to detect the end of a message to or from this target. **NOTE** SSTATUS[START] does not assert for START of transaction if SCONFIG[MATCHSS] = 1. SSTATUS[START] only asserts for Repeated Start within that transaction, because SSTATUS[START] only asserts after SSTATUS[MATCHED] occurs, which is after START of transaction. 0b - Disable 1b - Enable
1 NACK	Not Acknowledge Controls the ACK or NACK capability. If you write 1 to this field, the target rejects all requests, except for a CCC broadcast. NACK = 1 must be used with caution because the controller may decide that the target is missing, if NACK is overused. 0b - Always disable NACK mode 1b - Always enable NACK mode (works normally)
0 SLVENA	Target Enable Enables the target. If you write 0 to this field, the target ignores the I2C or I3C bus. If you write 1 to this field, the target can operate on the I2C or I3C bus. You must not write 1 to this field before registers such as Target Configuration (SCONFIG), Target ID Part Number (SIDPARTNO), Target ID Extension (SIDEXT), and others are configured because these registers affect the data to and from the controller. Target enable is configured just once before the bus comes up. If target enable is used at other times, SCAPABILITIES[IBI_MR_HJ] must be 1 before writing 1 to SLVENA so that the device does not see a START or STOP condition incorrectly. 0b - Disable 1b - Enable

41.7.4 Target Status (SSTATUS)

Offset

Register	Offset
SSTATUS	8h

Function

Indicates the sticky status for interrupts, states, and modes related to the I3C bus. Not all fields of the register are used if the module only acts as a target. The fields are divided into current activity, interrupt maskable actions, and 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	0	MRDIS	IBIDIS	0		EVDET		0	EVENT	CHANDLED	HDRMATCH
W														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	ERRWARN	CCC	DACHG	TXNOTFU...	RX_PEND	STOP	MATCHED	START	0	STHDR	STDAA	STREQWR	STREQRD	STCCCH	STMSG	STNOTST...
W		W1C	W1C			W1C	W1C	W1C								
Reset	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 TIMECTRL	Time Control Indicates whether time control is enabled, and in which mode. 00b - NO_TIME_CONTROL (no time control is enabled) 01b - SYNC_MODE (Synchronous mode is enabled) 10b - ASYNC_MODE (Asynchronous standard mode (0 or 1) is enabled) 11b - BOTHSYNCASYNC (both Synchronous and Asynchronous modes are enabled)
29-28 ACTSTATE	Activity State from Common Command Codes (CCC) Indicates the activity state from 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)

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 HJDIS	Hot-Join Disabled Indicates whether hot-join is disabled. When hot-join is disabled, CTRL requests are not responded to. 0b - Enabled 1b - Disabled
26 —	Reserved
25 MRDIS	Controller Requests Disable Indicates whether controller requests are disabled. When controller requests are disabled, CTRL requests are not responded to. 0b - Enabled 1b - Disabled
24 IBIDIS	In-Band Interrupts Disable Indicates whether in-band interrupts are disabled. When in-band interrupts are disabled, CTRL requests are not responded to. 0b - Enabled 1b - Disabled
23-22 —	Reserved
21-20 EVDET	Event Details Indicates current details of the last (EVENT = 1) or pending event. 00b - NONE (no event or no pending event) 01b - NO_REQUEST (request is not sent yet; either there is no START condition yet, or is waiting for Bus-Available or Bus-Idle (HJ)) 10b - NACKed (not acknowledged, request sent and rejected); I3C tries again 11b - ACKed (acknowledged; request sent and accepted), so done (unless the time control data is still being sent)
19 —	Reserved
18 EVENT	Event Flag Indicates, for a target, whether a pending in-band interrupt (IBI), controller request (CR), or hot-join (HJ) has been sent as requested. See the upper status register fields for details. This field becomes 1 only when acknowledged by a controller.

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 - No event occurred 1b - IBI, CR, or HJ occurred When writing 0b - No effect 1b - Clear the flag
17 CHANDLED	Common Command Code Handled Flag Indicates whether an HDRMATCH CCC is being handled by I3C. This field is for notification only, but it may result in updates to this register. **NOTE** This field behaves differently for register reads and writes. When reading 0b - CCC handling not in progress 1b - CCC handling in progress When writing 0b - No effect 1b - Clear the flag
16 HDRMATCH	High Data Rate Command Match Flag Indicates whether the HDR command matched the I3C dynamic address of this device. The HDR command is available as the first byte, with RXPEND = 1. The MSB of the command byte indicates whether 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 is ACKed and the data is sent back. Otherwise, the HDR command is NACKed. **NOTE** When this field is 1, you must check SSTATUS[ERRWARN] because the HPAR error may occur after signaling this HDR command. The parity occurs after the destination address and command. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Did not match 1b - Matched the I3C dynamic address When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
15 ERRWARN	Error Warning Indicates that 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 Target Errors and Warnings (SERRWARN) for more information).
14 CCC	Common Command Code Flag Indicates whether a CCC has been received, and is not handled by I3C. There are two types of common command codes: • Broadcast CCC, which corresponds with RXPEND, and the first byte is the CCC (command). • Direct CCC, which may never be directed to this device. If the direct CCC is directed to this device, then TXSEND or RXPEND are triggered, and RXPEND contains the command. **NOTE** This field behaves differently for register reads and writes. When reading 0b - CCC not received 1b - CCC received When writing 0b - No effect 1b - Clear the flag
13 DACHG	Dynamic Address Change Flag Indicates an occurrence of dynamic address (DA) change. Actual DA can be seen in the DYNADDR register. If this field is 1, DA change is detected. The target DA has been assigned, reassigned, or reset (lost) and is now in the state of being valid or none. This field is also used when the MAP auto feature is configured, changing one or more MAP items. See DYNADDR and MAPCTRL*n* for more information. DYNAADDR for the main DA (0) indicates whether the last change was because of Auto-MAP. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No DA change detected 1b - DA change detected When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 TXNOTFULL	Transmit Buffer Not Full Indicates that the transmit buffer is not full. If DMA is enabled for transmitting, then it is also signaled to provide more data. To-bus buffer or FIFO can accept more data to be transmitted. For all but external FIFOs, this process uses SDATACTRL[TXTRIG], which defaults to "not full". 0b - Transmit buffer full 1b - Transmit buffer not full
11 RX_PEND	Received Message Pending Indicates whether a received message is pending. The field indicates when a message from the controller that is not being handled by I3C (not a CCC message) is received. I3C processes such messages internally. For all but external FIFOs, this process uses SDATACTRL[RXTRIG], which defaults to "not empty". If DMA is enabled for receiving, then DMA is signaled as well. This field automatically becomes 0 if data is read (from FIFO and non-FIFO sources). 0b - No received message pending 1b - Received message pending
10 STOP	Stop Flag Indicates whether the Stopped state is detected. The STNOTSTOP state also indicates when the module is in Stop mode. A fast STOP and START combination may not trigger the Stop status. In that case, the Start status is always set. If this field is 1, it indicates that the Stop state was present on the bus since the bus was last cleared. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No Stopped state detected 1b - Stopped state detected When writing 0b - No effect 1b - Clear the flag
9 MATCHED	Matched Flag Indicates whether an incoming header matched the I3C dynamic or I2C static address (if any) of this device since the bus was last cleared. **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
	0b - Header not matched 1b - Header matched When writing 0b - No effect 1b - Clear the flag
8 START	Start Flag Indicates whether a START or Repeated START was detected after this flag was last cleared. This flag is not usually needed, but can be used for wake events. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
7 —	Reserved
6 STHDR	Status High Data Rate Indicates whether the I3C bus is in HDR-DDR mode, regardless of whether HDR mode is supported by this module, and regardless of whether the message is intended for this module or some other module. 0b - I3C bus not in HDR-DDR mode 1b - I3C bus in HDR-DDR mode
5 STDAA	Status Dynamic Address Assignment Indicates whether the I3C bus is in Enter Dynamic Address Assignment (ENTDAA) mode, regardless of whether this bus target has a dynamic address. 0b - Not in ENTDAA mode 1b - In ENTDAA mode
4 STREQWR	Status Request Write Indicates whether the REQ in process is SDR write data from the controller to this bus target (or all targets), but not in ENTDAA mode. See status high data rate (STHDR) for double data rate (DDR) handling.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not an SDR write 1b - SDR write data from the controller, but not in ENTDAA mode
3 STREQRD	Status Request Read Indicates whether the REQ in process is an SDR read from this target. See status high data rate (STHDR) for double data rate (DDR) handling. 0b - Not an SDR read 1b - SDR read from this target or an IBI is being pushed out
2 STCCCH	Status Common Command Code Handler Indicates whether a CCC message is being handled automatically. 0b - No CCC message handled 1b - Handled automatically
1 STMSG	Status Message Indicates whether the bus target is busy (listening to the bus traffic or responding). If STNOSTOP = 1, STMSG is 0 when a nonmatching address is seen, until the next Repeated START or STOP condition occurs. 0b - Idle 1b - Busy
0 STNOTSTOP	Status not Stop Indicates the status of the bus. If this field is 0, I3C is in a STOP condition. If this field is 1, the bus is busy (has activity). Other fields of this register may also be set when busy. STNOTSTOP can also become 1 after a TE0 or TE1 error, when I3C is waiting for an exit pattern. 0b - In STOP condition 1b - Busy

41.7.5 Target Control (SCTRL)

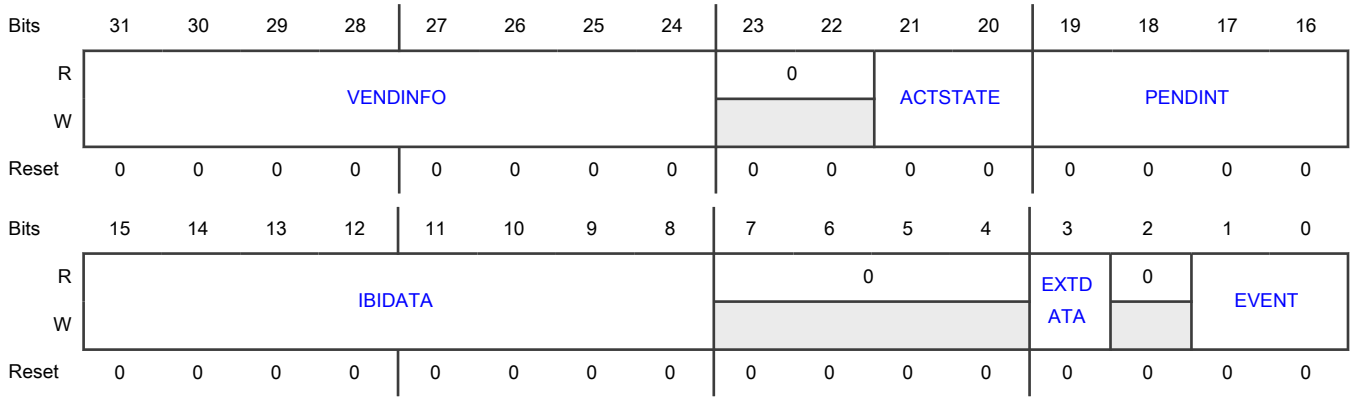
Offset

Register	Offset
SCTRL	Ch

Function

Contains controls for the active use of the I3C bus (for example, event generation such as interrupts to the controller). Only if I3C is configured to support various special operations for the target, this register is used to activate those operations. These events include IBI and GETSTATUS fields (except the protocol error, which is automatically set).

Diagram



Fields

Field	Function
31-24 VENDINFO	Vendor Information Controls vendor information that the GETSTATUS CCC returns. You must set this field to the Vendor Reserved field that the GETSTATUS CCC returns. The application must maintain the vendor information because the controller reads this field. If this field is not configured, then the GETSTATUS field always returns 0.
23-22 —	Reserved
21-20 ACTSTATE	Activity State of Target Controls the activity state of the target. You must set this field to the activity state of the target that the GETSTATUS CCC command returns as activity mode. The application must maintain the activity state because the controller reads this field. If you do not configure the activity state, then the GETSTATUS command always returns 0.
19-16 PENDINT	Pending Interrupt Specifies whether an IBI interrupt is pending. You must set this field to the pending interrupt that the GETSTATUS CCC command returns. The application must maintain the pending interrupt because the controller reads this field. If PENDINT = 0: • The GETSTATUS command returns 1 if an IBI interrupt is pending. • The GETSTATUS field returns 0 if an IBI interrupt is not pending.
15-8 IBIDATA	In-Band Interrupt Data Controls the data byte accompanying the IBI, if the module is enabled for IBI. If SCTRL[IBIDATA] is enabled, then IBI is required.
7-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 EXTDATA	Extended Data Enables extended data. After IBIDATA is emitted, extended data is acquired from IBIEXT1 and IBIEXT2, if configured. If extended data is used with time control, the data follows the time information. See IBIEXT1[MAX] for more information. 0b - Disable 1b - Enable
2 —	Reserved
1-0 EVENT	Event Requests an event. If this field is 0b, it indicates Normal mode, in which if this field becomes 0 from a nonzero value and event processing has not yet started, the event processing is canceled. However, event processing is not canceled if it has already started. If this field is 1b, an IBI is pushed onto the I3C bus. If there is data associated with the IBI, the data is read from SCTRL[IBIDATA]. If time control is enabled, this data includes any time-control-related bytes. Additionally, SCTRL[IBIDATA][7] becomes 1 automatically (as is required for time control). The IBI interrupt occurs after the first (mandatory) IBIDATA, if any. If this field is 10b, a controller request is started; the meaning depends on SIDEXT[BCR] configured in I3C. If this field is 11b, a hot-join (HJ) request is started, which is used when the device is powered on after the I3C bus is already powered up. It is also used when the device is connected using hot-insertion methods (the device is powered up when it is physically inserted in the powered-up I3C bus). The HJ waits for Bus Idle, and SCTRL[EVENT] = HOT_JOIN_REQUEST must be set before the target enable (SCONFIG[SLVENA]). If this field is a nonzero value, it requests an event. After the request, SSTATUS[EVENT] and SSTATUS[EVDET] show the status as it progresses. After completion, this field automatically returns to 0. After this field becomes a nonzero value, you can write only 0 it (to cancel) until the event processing is finished. 00b - NORMAL_MODE 01b - IBI 10b - CONTROLLER_REQUEST 11b - HOT_JOIN_REQUEST

41.7.6 Target Interrupt Set (SINTSET)

Offset

Register	Offset
SINTSET	10h

Function

Sets interrupt enables for select Target Status (SSTATUS) fields. Reading this register (SINTSET) returns the status of the interrupt enable:

- To activate an interrupt enable, write 1 to its corresponding field in this register (SINTSET).
- To disable an interrupt, write 1 to its corresponding field in Target Interrupt Clear (SINTCLR). Writing 0 to the interrupt enable in this register (SINTSET) does not disable the interrupt.

The Interrupt registers allow the masking of interrupt sources. They also allow the checking of which interrupts are activated. The normal method is to enable an interrupt, and then after the interrupt occurs, clear the interrupt either by writing to Target Status (SSTATUS) or by performing an 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; if a new event occurs, that new event is not lost:

- SINTSET sets interrupt enables for Target Status (SSTATUS) fields. Reading the SINTSET register returns the status of the interrupt enables.
- SINTCLR clears interrupt enables for the SSTATUS fields.
- SINTMASKED returns the value of the SSTATUS fields ANDed with their interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												0	EVENT	CHANDLED	DDRMATC...
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	ERRWARN	CCC	DACHG	TXSEND	RXPEND	STOP	MATCHED	START	0							
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 —	Reserved
18 EVENT	Event Interrupt Enable Enables event interrupts. This field indicates, for a target, whether a pending in-band interrupt (IBI), controller request (CR), or hot-join (HJ) has been sent as requested. The field is configured to support events. See SSTATUS[EVDET] for more information.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
17 CHANDLED	Common Command Code (CCC) Interrupt Enable Enables CCC interrupts. This field specifies whether CCC is being handled by I3C. Target Status (SSTATUS) shows new results. You can use this field to track when activity states and masks on events (for example, IBIs) occur. The field is used for CCCs that are enabled. 0b - Disable 1b - Enable
16 DDRMATCHED	Double Data Rate Interrupt Enable Enables DDR match for read or write command. This field indicates when DDR matches for read or write command, and the field is used only if HDR is enabled. 0b - Disable 1b - Enable
15 ERRWARN	Error or Warning Interrupt Enable Enables error or warning interrupts. This field indicates whether an error or warning has occurred, such as data underrun, data overrun, parity error, or HDR-DDR CRC error, or any other error or warning condition. See Target Errors and Warnings (SERRWARN) for cause of error. Only available for errors related to configured features. 0b - Disable 1b - Enable
14 CCC	Common Command Code (CCC) Interrupt Enable Enables CCC interrupts. This field indicates whether a CCC has been received, and is not handled by I3C. For CCCs that the block does not handle, RXPEND also interrupts, and SSTATUS[STREQRD] indicates that it is a CCC sending a read request. 0b - Disable 1b - Enable
13 DACHG	Dynamic Address Change Interrupt Enable Enables dynamic address change interrupts: interrupt on dynamic address defined (SETDASA or ENTDAA) or lost (RSTDAA). See Controller Dynamic Address (MDYNADDR) for more information.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
12 TXSEND	Transmit Interrupt Enable Enables transmit interrupts. This field indicates whether to-bus buffer or FIFO can accept more data to be transmitted. The corresponding interrupt occurs when the controller requests data to read from this target. This interrupt occurs on the first request (header) as well as when it is ready for more data. The corresponding interrupt occurs when the controller requests data to be read from this target. If this interrupt is for a FIFO, it triggers on the transmit emptiness trigger. If this interrupt is for DMA, then it indicates message end (DMA end or termination). 0b - Disable 1b - Enable
11 RXPEND	Receive Interrupt Enable Enables receive interrupts. This field specifies when a message from the controller that is not being handled by the module is received. For example, where data is consumed by hardware directly when it goes into the FIFO (excludes CCCs being handled automatically). If this interrupt is for a FIFO, it indicates a receive fullness trigger. If this interrupt is for DMA, then it indicates message end. 0b - Disable 1b - Enable
10 STOP	Stop Interrupt Enable Enables stop interrupts. This field detects the Stopped state present on the bus since the bus was last cleared. Use SINTSET[START] as the preferred interrupt when needed. This interrupt may not trigger for a quick STOP and START combination because it relates to the state of being stopped. 0b - Disable 1b - Enable
9 MATCHED	Match Interrupt Enable Enables match interrupts. Incoming header matched the I3C dynamic or I2C static address of this device since the bus was last cleared. If configured and if no dynamic address is set, this interrupt is also for matching a header on an I2C static address. See Controller Dynamic Address (MDYNADDR) for related information. 0b - Disable 1b - Enable
8	Start Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
START	Enables start interrupts. A START or Repeated START (such as wakeup) is seen after this field last became 0. See SINTSET[STOP] for related information. 0b - Disable 1b - Enable
7-0 —	Reserved

41.7.7 Target Interrupt Clear (SINTCLR)

Offset

Register	Offset
SINTCLR	14h

Function

Clears interrupt enables for select Target Status (SSTATUS) fields. To clear an interrupt enable, write 1 to the corresponding field 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	0												0	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	0							
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
31-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 —	Reserved
18 EVENT	EVENT Interrupt Enable Clear Flag
17 CHANDLED	CHANDLED Interrupt Enable Clear Flag
16 DDRMATCHED	DDRMATCHED Interrupt Enable Clear Flag
15 ERRWARN	ERRWARN Interrupt Enable Clear Flag
14 CCC	CCC Interrupt Enable Clear Flag
13 DACHG	DACHG Interrupt Enable Clear Flag
12 TXSEND	TXSEND Interrupt Enable Clear Flag
11 RXPEND	RXPEND Interrupt Enable Clear Flag
10 STOP	STOP Interrupt Enable Clear Flag
9 MATCHED	Matched Interrupt Enable Clear Flag
8 START	START Interrupt Enable Clear Flag
7-0 —	Reserved

41.7.8 Target Interrupt Mask (SINTMASKED)

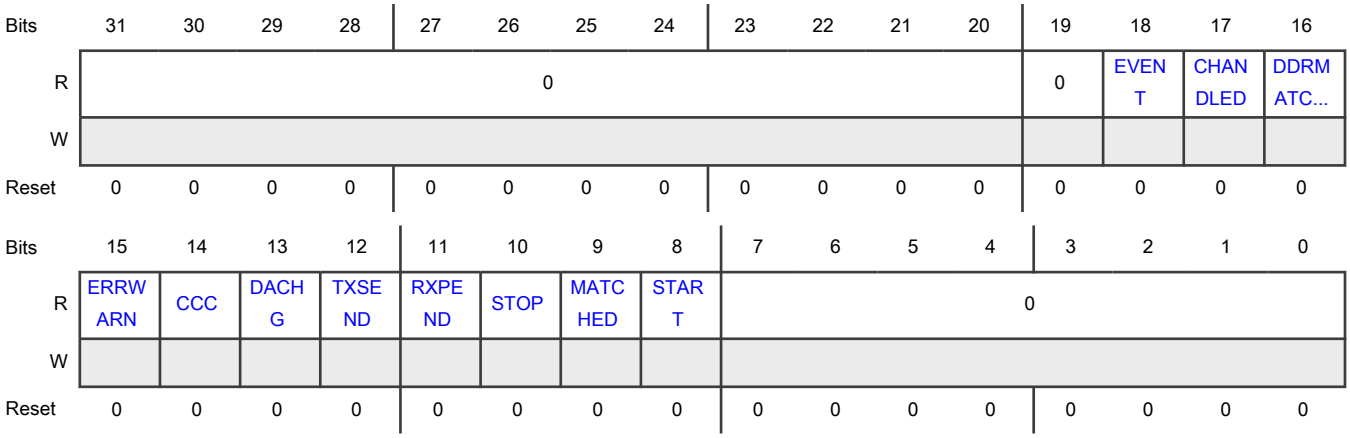
Offset

Register	Offset
SINTMASKED	18h

Function

Returns the status of enabled interrupts (the value of Target Status (SSTATUS) ANDed with the value of Target Interrupt Set (SINTSET)).

Diagram



Fields

Field	Function
31-20 —	Reserved
19 —	Reserved
18 EVENT	EVENT Interrupt Mask
17 CHANDLED	CHANDLED Interrupt Mask
16 DDRMATCHED	DDRMATCHED Interrupt Mask
15	ERRWARN Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERRWARN	
14 CCC	CCC Interrupt Mask
13 DACHG	DACHG Interrupt Mask
12 TXSEND	TXSEND Interrupt Mask
11 RXPEND	RXPEND Interrupt Mask
10 STOP	STOP Interrupt Mask
9 MATCHED	MATCHED Interrupt Mask
8 START	START Interrupt Mask
7-0 —	Reserved

41.7.9 Target Errors and Warnings (SERRWARN)

Offset

Register	Offset
SERRWARN	1Ch

Function

Contains errors and warnings from I3C and I2C protocols. This includes internal issues such as overrun and underrun, detected errors and conditions like parity errors, CRC errors, and read terminations by the controller. See SSTATUS[ERRWARN] and SINTSET[ERRWARN] for related information.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										0		0		OWRITE	OREAD
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	0		0	0	S0S1	HCRC	HPAR	SPAR	0			INVSTART	TERM	URUNNACK	URUN	ORUN
W					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
31-22 —	Reserved
21 —	Reserved
20-18 —	Reserved
17 OWRITE	Over-Write Error Flag Indicates that Target Write Data Byte (SWDATAB) or Target Write Data Byte End (SWDATABE) was written to when full. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overwrite error 1b - Overwrite error When writing 0b - No effect 1b - Clear the flag
16 OREAD	Over-Read Error Flag Indicates that Target Read Data Byte (SRDATAB) was read for more bytes than were available, by the application. This field also indicates over-read errors for Target Read Data Halfword (SRDATAH), if it is enabled.

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 - No over-read error 1b - Over-read error When writing 0b - No effect 1b - Clear the flag
15-14 —	Reserved
13 —	Reserved
12 —	Reserved
11 S0S1	TE0 or TE1 Error Flag Indicates whether a TE0 or TE1 error occurred and the target is locked and waiting for an HDR exit pattern. Writing 1 to S0S1 causes I3C to release the lock, but this method must be used with great care. S0S1 becomes 0 automatically when an exit pattern is detected, so writing 1 to S0S1 must be used under controlled circumstances to avoid problems. Before starting to operate normally after this error, I3C waits for a START (or Repeated START) or STOP state. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No TE0 or TE1 error occurred 1b - TE0 or TE1 error occurred When writing 0b - No effect 1b - Clear the flag
10 HCRC	HDR-DDR CRC Error Flag Indicates whether an HDR-DDR CRC error occurred on a message from the controller. Error reasons include an HDR restart and an exit being issued before an HDR-DDR message from the controller has finished. This error calls into question the data from the entire DDR command frame.

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 - No HDR-DDR CRC error occurred 1b - HDR-DDR CRC error occurred When writing 0b - No effect 1b - Clear the flag
9 HPAR	HDR Parity Error Flag Indicates when an HDR parity error or framing error on a message from the controller occurs. The corresponding command or data that has the error is usually in the RX buffer, which can be read using Target Read Data Byte (SRDATAB). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No HDR parity error 1b - HDR parity error When writing 0b - No effect 1b - Clear the flag
8 SPAR	SDR Parity Error Flag Indicates when an SDR parity error on a message from the controller occurs. This error also sets the GETSTATUS protocol error field (which becomes 0 after a GETSTATUS read). For read operations, this field becomes 1 when a read abort (timeout) occurs because of the controller not driving clock for more than 100 µs during an SDR read. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No SDR parity error 1b - SDR parity error When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-5 —	Reserved
4 INVSTART	Invalid Start Error Flag Indicates an invalid condition with SCL falling before SDA falls, so there is no start. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No invalid start error 1b - Invalid start error When writing 0b - No effect 1b - Clear the flag
3 TERM	Terminated Error Flag Indicates when the controller terminates a read from a target, if an END is not set by the target (on the same read or the previous read). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No terminated error 1b - Terminated error When writing 0b - No effect 1b - Clear the flag
2 URUNNACK	Underrun and Not Acknowledged (NACKed) Error Flag Indicates when the internal to-bus buffer or FIFO is underrun in the read header and so the module NACKed the header. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underrun; not acknowledged error 1b - Underrun; not acknowledged error When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
1 URUN	Underrun Error Flag Indicates when the internal to-bus buffer or FIFO is underrun during data read (the application is not providing the data fast enough). The END field or register must be used if the read was the last one. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No underrun error 1b - Underrun error When writing 0b - No effect 1b - Clear the flag
0 ORUN	Overrun Error Flag Indicates when the internal from-bus buffer or FIFO has overrun (arrival of too many characters that you cannot process fast enough). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No overrun error 1b - Overrun error When writing 0b - No effect 1b - Clear the flag

41.7.10 Target DMA Control (SDMACTRL)

Offset

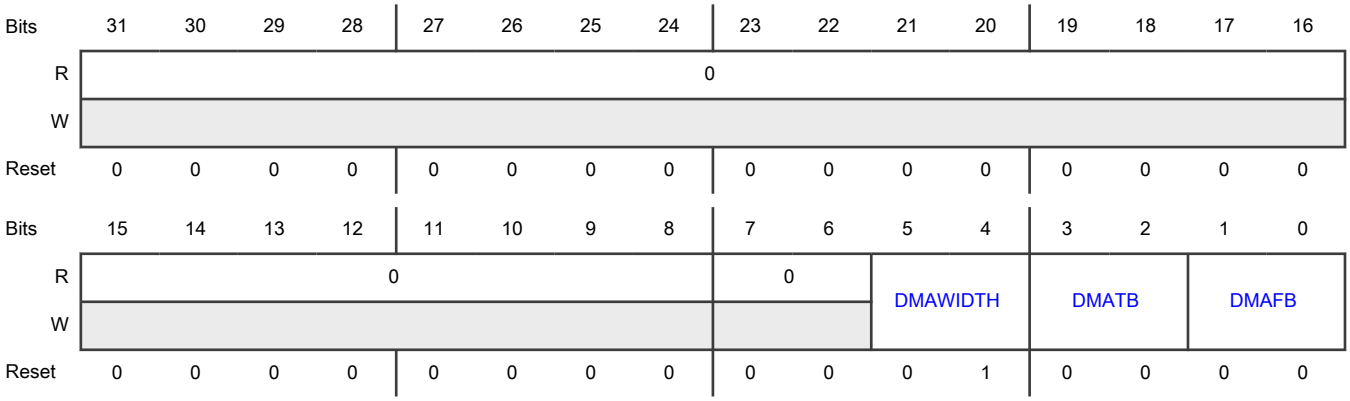
Register	Offset
SDMACTRL	20h

Function

Allows DMA to be used for inbound and outbound messages. This register is limited in value for target use because the target must be reactive. These are the two common use case models:

- To avoid an overrun in from-bus collection: if SCONFIG[MATCHSS] becomes 1, then the processor enables the interrupts for START and STOP and enables the DMA to collect the data. The START or STOP interrupt only occurs after a message is directed to the target (MATCHED is 1). The DMA copied data can then be examined.
- To perform larger reads from the to-bus: I3C and I2C reads are preceded by a write that indicates what will be read (or in response to an IBI from the target). Because of this process, the DMA can be used to push through the data:
  - For I3C, the processor must manage the last value, unless the DMA moves wider words and is able to write 1 to the END field. These values are 16-bit values when in Byte mode or 32-bit values when in Halfword mode.
  - For I2C, the controller determines the last value, so the DMA may end early or run out when the controller expects more values.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-6 —	Reserved
5-4 DMAWIDTH	Width of DMA Operations Indicates the width of DMA operations, if configured to allow halfword data accesses. 00b,01b - Byte 10b - Halfword (16 bits) (this value ensures that two bytes are available in the FIFO) 11b - Reserved
3-2 DMATB	DMA Write (To-Bus) Trigger Enables DMA writes. If this field is enabled, it starts a request DMA on a transmit trigger (see Target Data Control (SDATACTRL)). DMATB requests until full, unless the DMA is set up as a trigger. DMATB becomes 0 when MSTATUS[ERRWARN] becomes 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If you write 1b to this field, DMA is enabled for one frame (ended by DMA or terminated). DMATB automatically becomes 0 after a STOP or repeated START. See SCONFIG[MATCHSS] for more information. If you write 10b to this field, DMA is enabled until turned off. The value must only be used with Controller Message mode. 00b - DMA not used 01b - DMA enabled for one frame 10b - DMA enabled until turned off. Recommended mode, instead of 01b, to enable DMA. This setting provides better software control. 11b - Reserved
1-0 DMAFB	DMA Read (From-Bus) Trigger Enables DMA reads. If this field is enabled, DMAFB requests DMA on receive trigger (see SDATACTRL). It requests until empty, unless the DMA is set up as a trigger. This field becomes 0 when SSTATUS[ERRWARN] becomes 1. If this field is 1b (DMA is enabled for one frame), it automatically becomes 0 on STOP or repeated START. See SCONFIG[MATCHSS] for more information. **NOTE** Do not enable DMA after a transaction starts. It must be enabled only when enabling the target and interrupts to avoid any kind of RX FIFO overrun. 00b - DMA not used 01b - DMA enabled for one frame 10b - DMA enabled until turned off. Recommended mode, instead of 01b, to enable DMA. This setting provides better software control. 11b - Reserved

41.7.11 Target Data Control (SDATACTRL)

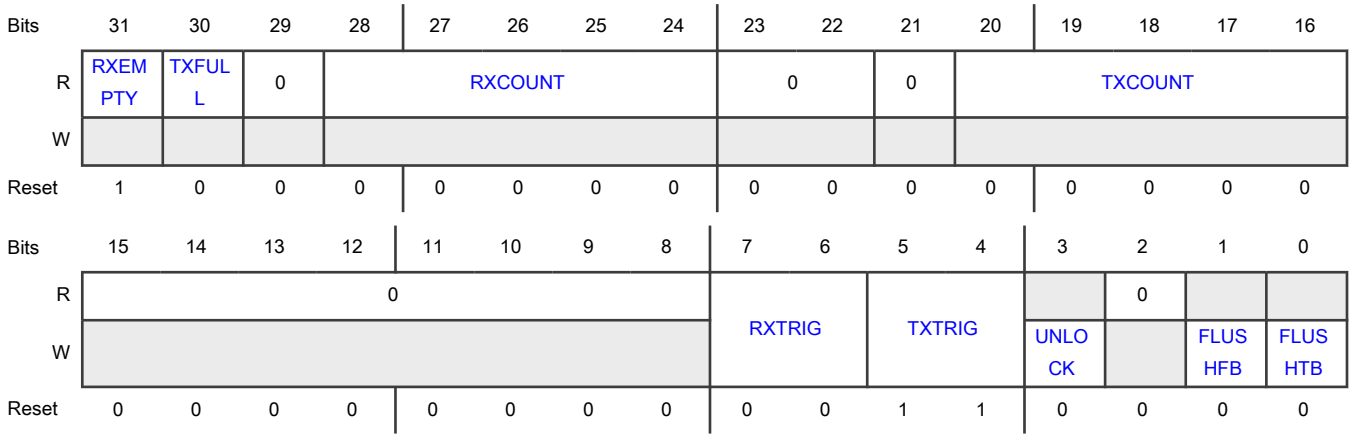
Offset

Register	Offset
SDATACTRL	2Ch

Function

Assists in data control when no FIFO is used. Also assists in data control of FIFO when the FIFO is available (regardless of size) allowing some control over the FIFO behavior. This register controls when to interrupt based on fullness or emptiness of a buffer or FIFO. It also controls behavior related to width, when the buffer or FIFO is not 1 byte wide.

Diagram



Fields

Field	Function
31 RXEMPTY	Receive is Empty Indicates whether the receive FIFO or buffer is empty. 0b - Not empty 1b - Empty
30 TXFULL	Transmit is Full Indicates whether the transmit FIFO or buffer is full. 0b - Not full 1b - Full
29 —	Reserved
28-24 RXCOUNT	Count of Entries in Receive Contains the count of entries in the receive FIFO or buffer. Entry size for SDR and HDR-DDR is in bytes.
23-22 —	Reserved
21 —	Reserved
20-16 TXCOUNT	Count of Entries in Transmit Contains the count of entries waiting in the TXFIFO. This count is the number of entries that the application has written to the transmit FIFO that have not yet gone onto the bus. Entry size for SDR and HDR-DDR is in bytes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7-6 RXTRIG	Receive Trigger Level Indicates the trigger level for receive fullness when using a FIFO. The field affects the RXPEND interrupt. 00b - Trigger when not empty (default) 01b - Trigger when 1/4 or more full 10b - Trigger when 1/2 or more full 11b - Trigger when 3/4 or more full
5-4 TXTRIG	Transmit Trigger Level Indicates the trigger level for transmit emptiness when using a FIFO. The field affects the TXNOTFULL interrupt. 00b - Trigger when empty 01b - Trigger when 1/4 full or less 10b - Trigger when 1/2 full or less 11b - Default (trigger when 1 less than full or less)
3 UNLOCK	Unlock Indicates whether the RXTRIG and TXTRIG fields can be changed on a write. This field must be 1 in the same cycle while writing to TXTRIG or RXTRIG. 0b - Cannot be changed 1b - Can be changed
2 —	Reserved
1 FLUSHFB	Flush From-Bus Buffer or FIFO Not normally used. **NOTE** Before using FLUSHFB, disable the DMA (SDMACTRL[DMAFB] = 0). Then use FLUSHFB to flush the FIFO and then re-enable the DMA. 0b - No action 1b - Flush the buffer
0 FLUSHTB	Flush To-Bus Buffer or FIFO Indicates when the controller terminates a to-bus (read) message prematurely.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Before using FLUSHTB, disable the DMA (SDMACTRL[DMATB] = 0). Then use FLUSHTB to flush the FIFO and then re-enable the DMA. 0b - No action 1b - Flush the buffer

41.7.12 Target Write Data Byte (SWDATAB)

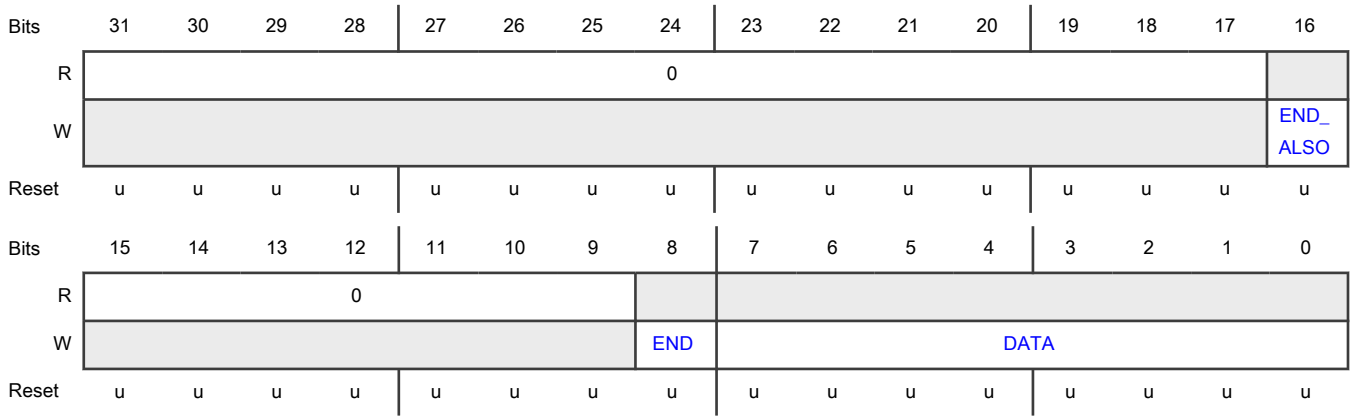
Offset

Register	Offset
SWDATAB	30h

Function

Allows writing a byte to the bus (to the controller) unless an external FIFO is used. Writing a byte requires a byte plus an end-of-data (last) marker bit. A byte must not be written unless there is room, indicated by SSTATUS[TXNOTFULL] = 1.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END_ALSO	End Also Marks the last byte of the message. This field is required for I3C, but is optional for I2C. If this field is 0, it indicates that there are more bytes in the message.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For HDR-DDR, the byte with END_ALSO must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. 0b - Not the end 1b - End
15-9 —	Reserved
8 END	End Marks the last byte of the message. If this field is 0, it indicates that there are more bytes in the message. This field is required for I3C, but is optional for I2C. For HDR-DDR, the byte with END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. 0b - Not the end 1b - End
7-0 DATA	Data Indicates the data byte to be sent to the controller.

41.7.13 Target Write Data Byte End (SWDATABE)

Offset

Register	Offset
SWDATABE	34h

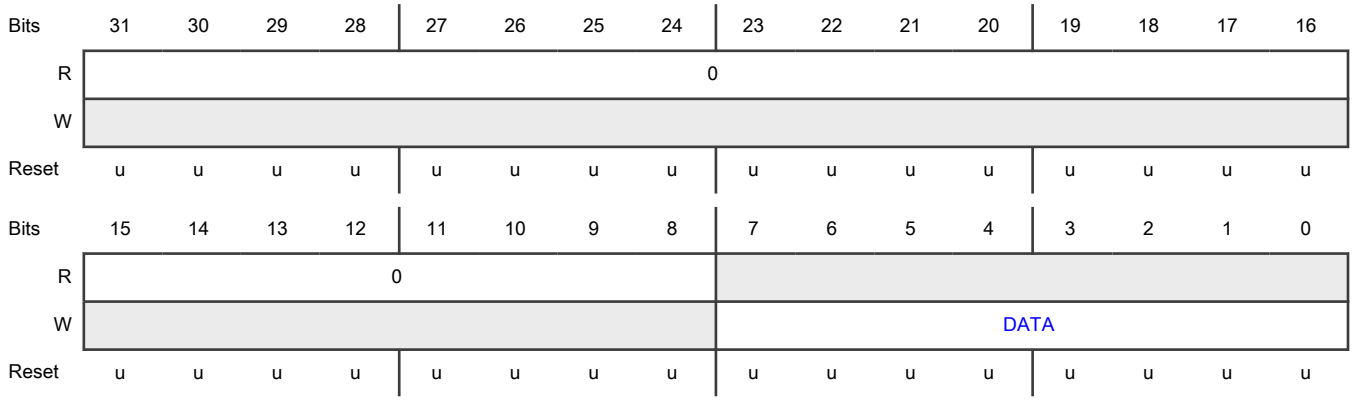
Function

Allows writing a byte to the bus (to the controller) unless an external FIFO is used.

Unlike Target Write Data Byte (SWDATAB), writing a byte only requires the byte itself, and is marked as end-of-data (last byte). A byte must not be written unless there is room, indicated by SSTATUS[TXNOTFULL] = 1.

For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Data Indicates the data byte to be sent to the controller.

41.7.14 Target Write Data Halfword (SWDATAH)

Offset

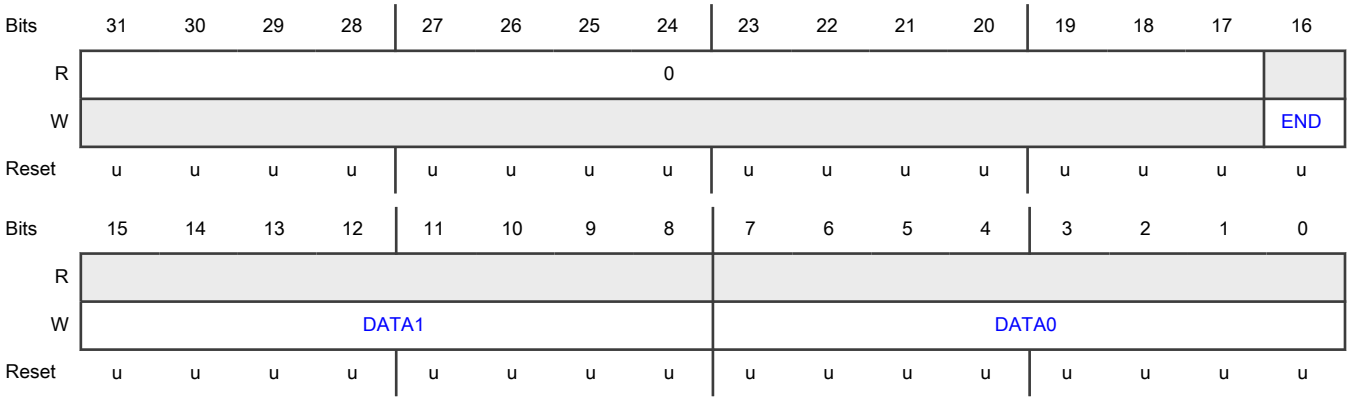
Register	Offset
SWDATAH	38h

Function

Allows writing a halfword (pair of bytes) to the bus unless an external FIFO is used. The low byte is followed by the high byte. The 16th bit marks the end; that is, the last byte of the halfword is the end.

An end-of-data (last) marker bit is allowed (or must be 0). A halfword must not be written unless there is room for both, as indicated by the use of transmit FIFO level trigger or SDATACTRL[TXCOUNT].

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of Message Marks the last byte of the message. This field always marks DATA1 as the end. The field is required for I3C, but is optional for I2C. If this field is 0, it indicates that there are more bytes in the message. For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. 0b - Not the end 1b - End
15-8 DATA1	Data 1 Indicates the second byte to be sent to the controller.
7-0 DATA0	Data 0 Indicates the first byte to be sent to the controller.

41.7.15 Target Write Data Halfword End (SWDATAHE)

Offset

Register	Offset
SWDATAHE	3Ch

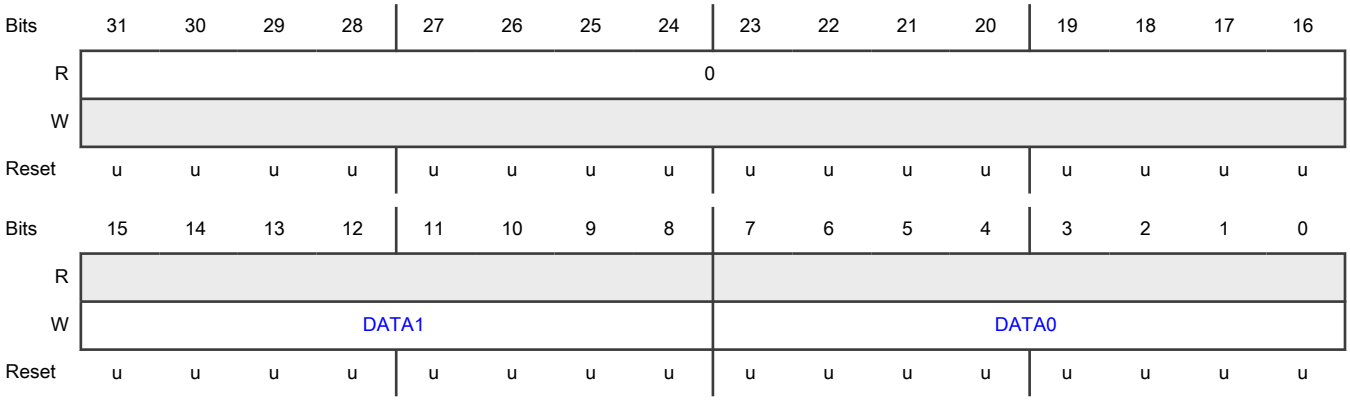
Function

Allows writing a halfword of data, which is the end (the last byte of the halfword is the end). The target writes the halfword (byte pair) in the same way it writes to Target Write Data Halfword (SWDATAH), but marks the second byte as end-of-data (last byte).

For HDR-DDR, the byte with the END must be even (second, fourth, sixth, and so on) because DDR uses byte pairs.

A halfword must not be written unless there is room for both, as indicated by the use of transmit FIFO level trigger or SDATACTRL[TXCOUNT].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DATA1	Data 1 Indicates the second byte to be sent to the controller.
7-0 DATA0	Data 0 Indicates the first byte to be sent to the controller.

41.7.16 Target Read Data Byte (SRDATAB)

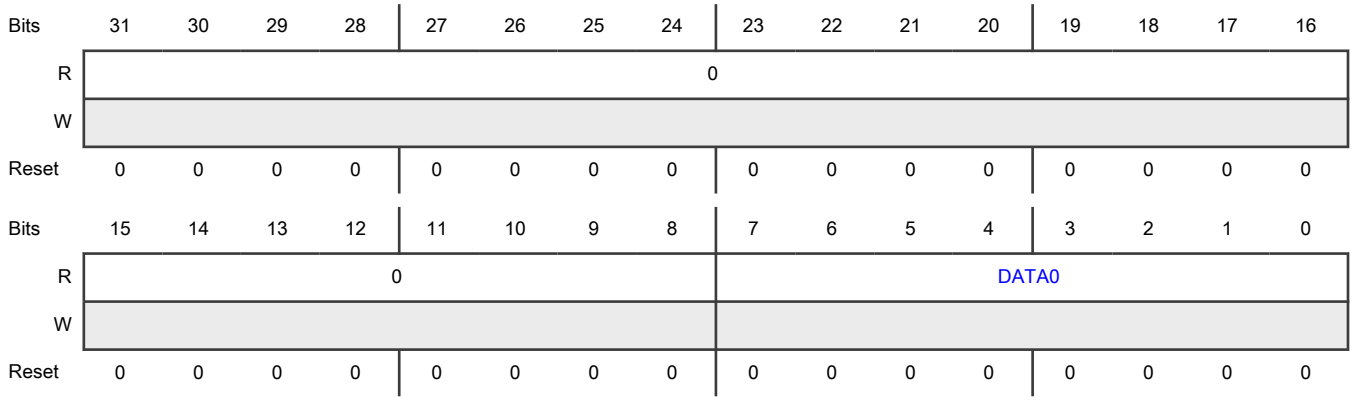
Offset

Register	Offset
SRDATAB	40h

Function

Allows reading a byte from the bus (controller). A byte must not be read unless there is data waiting, as indicated by SSTATUS[RX_PEND] = 1.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA0	Data 0 Indicates the byte read from the controller.

41.7.17 Target Read Data Halfword (SRDATAH)

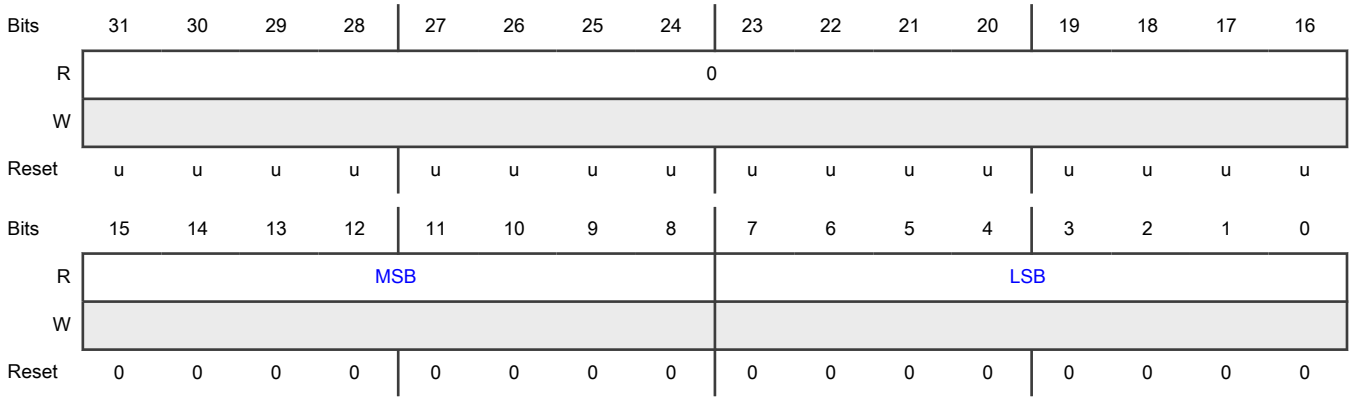
Offset

Register	Offset
SRDATAH	48h

Function

Allows reading a halfword (byte pair) written by the target after an SDR read or DAA or DDR. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively. A halfword must not be read unless there are at least two bytes of data waiting, as indicated by the use of receive FIFO level trigger or SDATACTRL[RXCOUNT].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 MSB	High Byte Indicates the second byte read from the target.
7-0 LSB	Low Byte Indicates the first byte read from the target.

41.7.18 Target Write Data Byte (SWDATAB1)

Offset

Register	Offset
SWDATAB1	54h

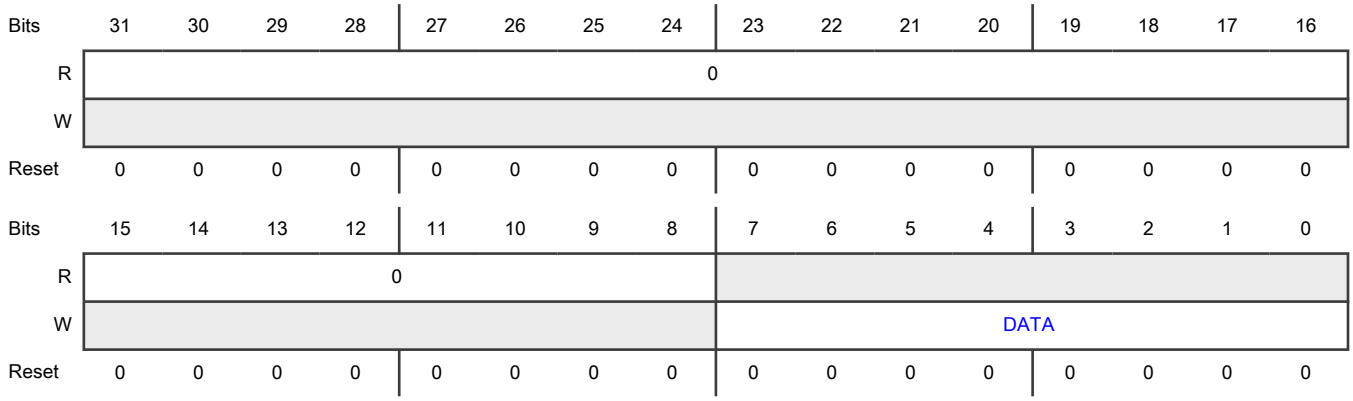
Function

Allows writing a single byte to the bus (to the controller) in a way that only bits 7:0 are used.

This register is intended to be used by DMAs (the upper bytes of the APB word are ignored).

A byte must not be written unless there is room, as indicated by SSTATUS[TXNOTFULL] = 1.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 DATA	Data Indicates the byte to be sent to the controller.

41.7.19 Target Write Data Halfword (SWDATAH1)

Offset

Register	Offset
SWDATAH1	54h

Function

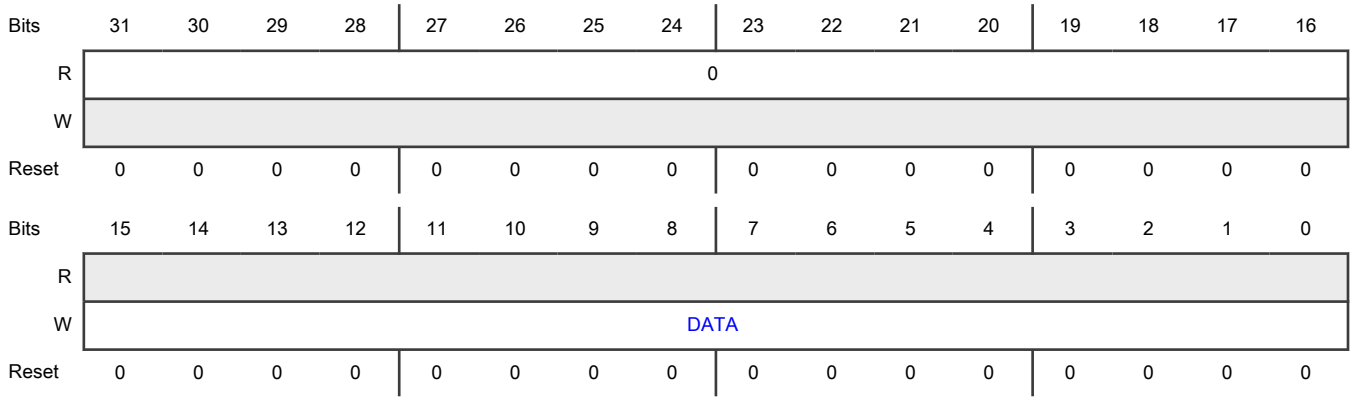
Shares the same address as Target Write Data Byte (SWDATAB1), and SDMACTRL[DMAWIDTH] determines which one is set.

This register allows writing a single-byte pair (halfword) to the bus (to the controller) in a way that only bits 15:0 are used.

This register is intended for use by DMAs, which do not format the upper part of the APB word.

A halfword must not be written unless there is room, as indicated by SSTATUS[TXNOTFULL] = 1.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data Indicates the byte pair (halfword) to be sent to the controller.

41.7.20 Target Capabilities 2 (SCAPABILITIES2)

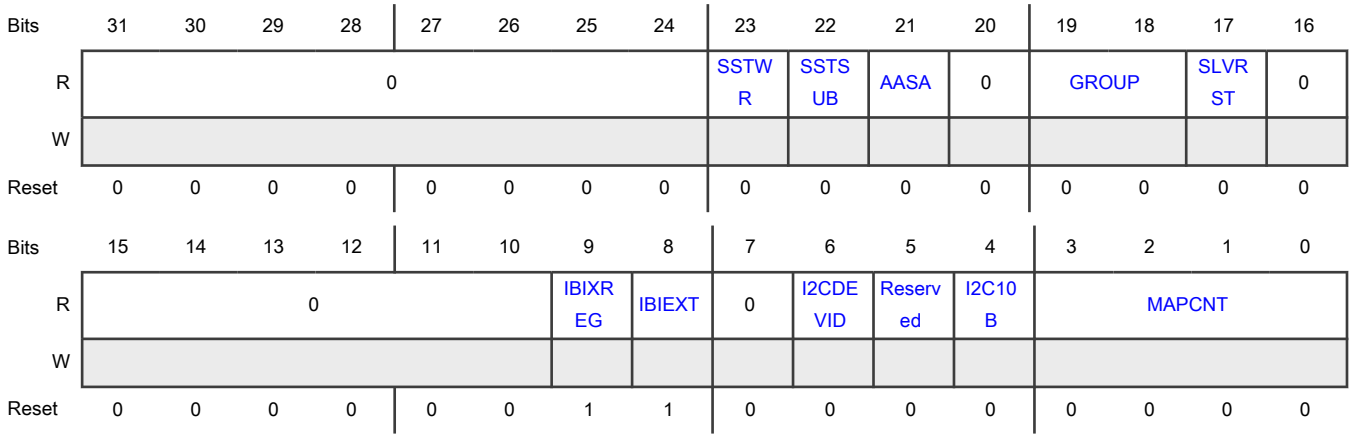
Offset

Register	Offset
SCAPABILITIES2	5Ch

Function

Indicates which features are available and supported in this module, including controller and target capabilities, HDR modes, and others.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 SSTWR	Target-Target(s)-Tunnel Write Capable Indicates whether target-target(s)-tunnel is write capable. 0b - Not write capable 1b - Write capable
22 SSTSUB	Target-Target(s)-Tunnel Subscriber Capable Indicates whether target-target(s)-tunnel is subscriber capable. 0b - Not subscriber capable 1b - Subscriber capable
21 AASA	SETAASA Supports the set static address as dynamic address CCC (SETAASA) feature. 0b - SETAASA not supported 1b - SETAASA supported
20 —	Reserved
19-18 GROUP	Group Indicates whether v1.1 group addressing is supported. Groups use mapping, so MAPCNT must be the same or greater than GROUP. 00b - v1.1 group addressing not supported 01b - One group supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Two groups supported 11b - Three groups supported
17 SLVRST	Target Reset Indicates whether v1.1 target reset is supported. 0b - Not supported 1b - Supported
16 —	Reserved
15-10 —	Reserved
9 IBIXREG	In-Band Interrupt Extended Register Supports extended registers for IBIs. This field indicates whether Extended IBI Data 1 (IBIEXT1) is available. Extended IBI Data 2 (IBIEXT2) is available if IBIEXT1[MAX] > 3. 0b - Not supported 1b - Supported
8 IBIEXT	In-Band Interrupt EXTDATA Supports SCTRL[EXTDATA] to allow data beyond the mandatory data byte (MDB) for IBIs. 0b - Not supported 1b - Supported
7 —	Reserved
6 I2CDEVID	I2C Device ID Indicates whether I2C device ID is supported. 0b - Not supported 1b - Supported
5 —	Reserved
4 I2C10B	I2C 10-bit Address Supports 10-bit I2C address in MAP 1. If this field is 1, MAPCNT must be 1 or greater.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not supported 1b - Supported
3-0 MAPCNT	Map Count Indicates the number of maps that are allowed. If there is no mapping, this field is 0. (DYNADDR is used to add more static and/or dynamic addresses to act as virtual targets.)

41.7.21 Target Capabilities (SCAPABILITIES)

Offset

Register	Offset
SCAPABILITIES	60h

Function

Indicates which features are available and supported in this module, including controller and/or target capabilities, HDR modes, and others.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA	INT	FIFORX		FIFOTX		EXTFIFO			0	TIMECTRL		IBI_MR_HJ			
W																
Reset	1	1	1	0	1	0	0	0	0	0	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CCCHANDLE				SADDR		MASTER	0	HDRSUPP		IDREG				IDENA	
W																
Reset	1	1	1	1	1	1	1	0	0	1	1	1	0	0	0	0

Fields

Field	Function
31 DMA	Direct Memory Access Indicates whether DMA is supported. 0b - Not supported 1b - Supported

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 INT	Interrupts Indicates whether interrupts are supported. 0b - Not supported 1b - Supported
29-28 FIFORX	FIFO Receive Indicates the size of receive (from-bus) FIFO. **NOTE** Controller and target use the same receive FIFO, so size is the same for the controller receive FIFO as well. FIFO size of SDR and HDR-DDR is in bytes. 00b - Two or three 01b - Four 10b - Eight 11b - 16 or larger
27-26 FIFOTX	FIFO Transmit Indicates the size of transmit (to-bus) FIFO. **NOTE** Controller and target use the same transmit FIFO, so size is the same for the controller transmit FIFO as well. FIFO size of SDR and HDR-DDR is in bytes. 00b - Two 01b - Four 10b - Eight 11b - 16 or larger
25-23 EXTFIFO	External FIFO Indicates whether external FIFOs are enabled. If they are not enabled, then check FIFOTX and FIFORX for the internal FIFO. 000b - No external FIFO available 001b - Standard available or free external FIFO 010b - Request track external FIFO All other values are reserved.
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 TIMECTRL	Time Control Specifies whether any time-control type is supported. 0b - No time control supported 1b - At least one time-control type supported
20-16 IBI_MR_HJ	In-Band Interrupts, Controller Requests, Hot-Join Events Indicates which events (IBI, CR, and HJ) are allowed. For example, if this field is 00011b, IBI (bit 0) and IBI_HAS_DATA (bit 1) functionality are both enabled. 0_0000b - Application cannot generate IBI, CR, or HJ 1_xxxxb - Application can use SCONFIG[BAMATCH] for bus-available timing x_1xxxb - Application can generate a Hot-Join event x_x1xxb - Application can generate a controller request for a secondary controller x_xx1xb - When bit 0 = 1, the IBI has data from the SCTRL register x_xxx1b - Application can generate an IBI
15-12 CCCHANDLE	Common Command Codes Handling Indicates what manages CCC between I3C and the application. 0000b - All handling features disabled 1xxxb - GETSTATUS CCC returns the value of SCTRL[VENDINFO] x1xxb - GETSTATUS CCC returns the values of SCTRL[PENDINT] and SCTRL[ACTSTATE] xx1xb - The I3C module manages maximum read and write lengths, and max data speed xxx1b - The I3C module manages events, activities, status, HDR, and if enabled for it, ID and static-address-related items
11-10 SADDR	Static Address Indicates how the static address is managed. 00b - No static address 01b - Static address is fixed in hardware 10b - Hardware controls the static address dynamically (for example, from the pin strap) 11b - SCONFIG register supplies the static address
9 MASTER	Controller Specifies whether controller capability is supported. 0b - Not supported 1b - Supported
8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-6 HDRSUPP	High Data Rate Support Indicates which HDR modes are supported. 00b - No HDR modes supported 01b - DDR mode supported All other values are reserved.
5-2 IDREG	ID Register Indicates which ID features are available in the configurable registers. 0000b - All ID register features disabled 1xxxb - A Bus Characteristics Register (BCR) is available x1xxb - A Device Characteristic Register (DCR) is available xx1xb - An ID Random field is available xxx1b - ID Instance is a register; used if there is no PARTNO register
1-0 IDENA	ID 48b Handler Indicates what handles the 48-bit ID value. 00b - Application 01b - Hardware 10b - Hardware, but the I3C module instance handles ID 48b 11b - A part number register (PARTNO)

41.7.22 Target Dynamic Address (SDYNADDR)

Offset

Register	Offset
SDYNADDR	64h

Function

Fills with the assigned address after the controller has assigned it via SETDASA or ENTDAA CCC commands. It clears if the RESETDAA CCC is used. The current validity state is also indicated through SSTATUS[DAVALID], which can be used to interrupt the processor. The DCAUSE field can be used to determine the changes, if enabled.

NOTE

If mapped addresses are enabled, you can access them through the SMAPCTRL*n* registers.

The first dynamic address (DA[0]) may be written irrespective of whether the mapping is enabled, if configured to allow writes 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 required if state-retention flops are used for the DA. This mechanism only allows writes when target is disabled

(and is ignored otherwise). If the controller uses RSTDAA or SETNEWDA, then it overrides this mechanism and yields to the controller-assigned DA. When enabling the target, SCONFIG[OFFLINE] must be 1. This waits for evidence that the bus is not in I3C HDR mode and exits when an HDR exit pattern is seen or when 60 µs has expired. This makes it safe to monitor START and STOP. If the application needs to do an IBI, then the application must either wait for a STOP (see STATUS) or make sure that 200 µs have gone by with no activity (no START or STOP) before the application emits the IBI.

The MAPIDX or MAPSA model allows writing additional DAs (and SAs) into a list (the number in the list is preconfigured); 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 the upper map, and then invalidate the 0 map) when matching ENTDAA over and over. Any mapped location can be invalidated by writing MAPIDX and DAVALID = 0. The mapped ones are not reset by the RSTDA CCC (see Target Message Map Address (SMSGMAPADDR)).

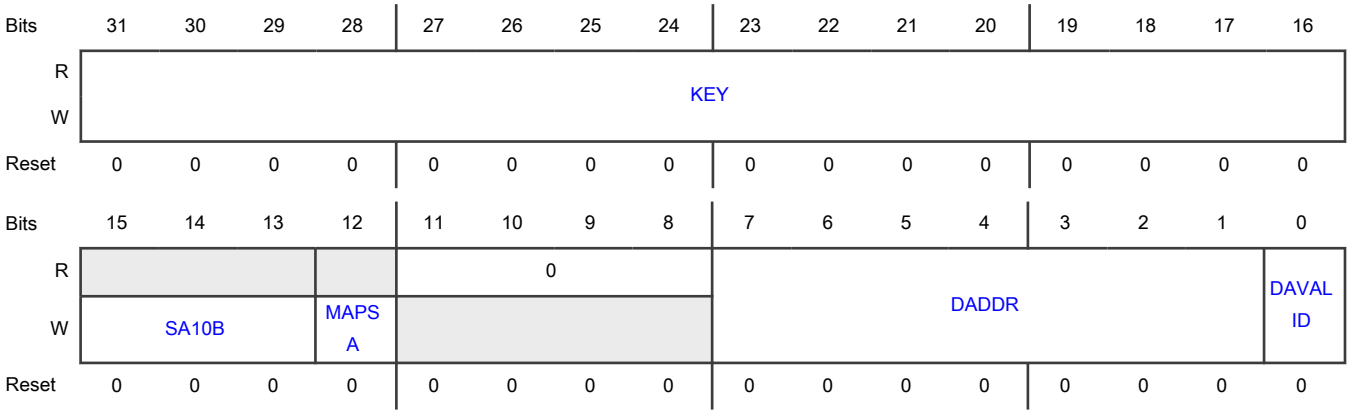
To copy the base DA to the mapped set, you must perform the configuration to allow it; otherwise it leads to distinct mechanisms:

- If used in I3C mode, a base DA is required, so the last one must not be cleared (or writing a dynamic address with DAVALID = 1 is necessary).

NACK/ACK of mapped DAs/SAs: The register also permits specific DAs (above the base) to NACK or ACK writes or reads, such as for bridges when the endpoint is busy. The NACK must not be set for too long, or the controller may consider the target in an error state. The model is to write with KEY = A731h and the MAPIDX and DAVALID to select ACK or NACK.

Mapping: If MAPIDX ≠ 0 and KEY = 0, then the next read returns details on the static or dynamic address at that location. If the read does not have MAPIDX ≠ 0, then it is not an acceptable location.

Diagram



Fields

Field	Function
31-16 KEY	Key Specifies the value of KEY. Must be set to A4D9h to write to the DADDR field (and 1 to the DAVALID field). Write only when a target is not enabled (for restoring after power-down sleep with auto-restore). The mapped locations and base may be written when a target is enabled, but do not write when there are transactions on the I3C bus. KEY reads back 1 if overwritten, else KEY is 0 if assigned by the controller including when the controller changes it. If address mapping is allowed, then writing with KEY = CB19h is used to clear the base DA.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If mapping is allowed, then writing with the key of A731h means that MAPIDX indicates which mapped address to set to ACK or NACK, such that DAVALID = 1 if ACK, DAVALID = 0 if NACK. This allows having the address refuse write and read requests when NACK. When using read back from map index (write with MAPIDX ≠ 0, KEY = 0, read) bit [16] = 1 if NACK, else 0.
15-13 SA10B	10-Bit Static Address Stores a 10-bit static address composed of {SA10B,DADDR}, if not 0 when MAPSA is set. Only one static address may be stored this way, and it must be MAPIDX == 1.
12 MAPSA	Map a Static Address Sets a static address into the list if MAPSA = 1 on a write with MAPIDX ≠ 0; otherwise, a dynamic address is used.
11-8 —	Reserved
7-1 DADDR	Dynamic Address Specifies the dynamic address (DA). This is the assigned dynamic address when DAVALID is 1 and is a static address if the MAPSA field is 1.
0 DAVALID	Dynamic Address Valid Determines whether a dynamic address is assigned. This field is 1 for ACK and 0 for NACK when using the A731h KEY. 0b - DANOTASSIGNED: a dynamic address is not assigned 1b - DAASSIGNED: a dynamic address is assigned

41.7.23 Target Maximum Limits (SMAXLIMITS)

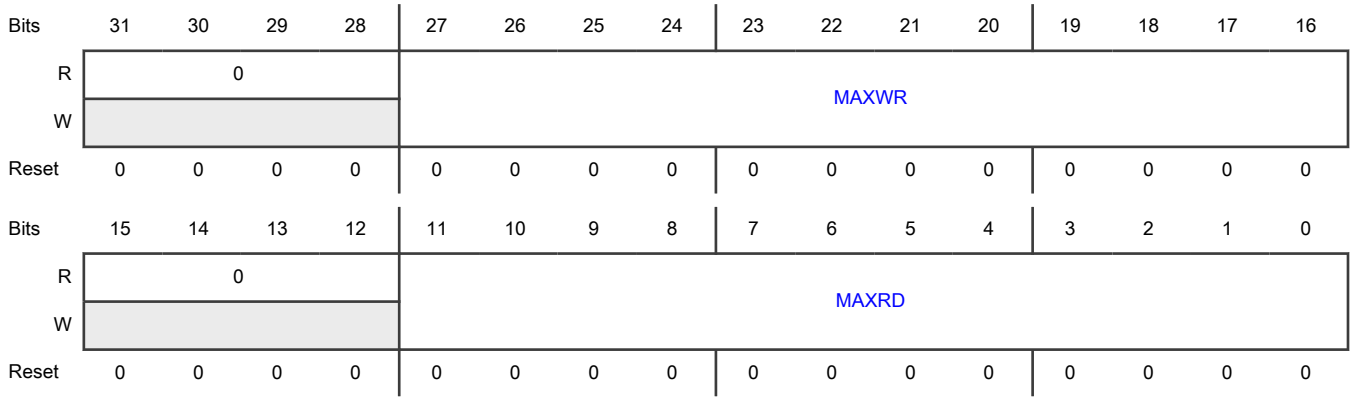
Offset

Register	Offset
SMAXLIMITS	68h

Function

Indicates the limits set by the controller (or the originally requested limits). The maximum limits are not enabled in the hardware design, including maximum read and write lengths. If the maximum read and write lengths are enabled, then the current setting (including the default request) shows up in this register (SMAXLIMITS).

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 MAXWR	Maximum Write Length Indicates the maximum write length, which must be between 8 to 4095 (saturation). The application must not set the maximum write length to a higher value than the maximum write length set by the controller.
15-12 —	Reserved
11-0 MAXRD	Maximum Read Length Indicates the maximum read length, which must be between 16 to 4095 (saturation). The application must not set the maximum read length to a higher value than the maximum read length set by the controller.

41.7.24 Target ID Part Number (SIDPARTNO)

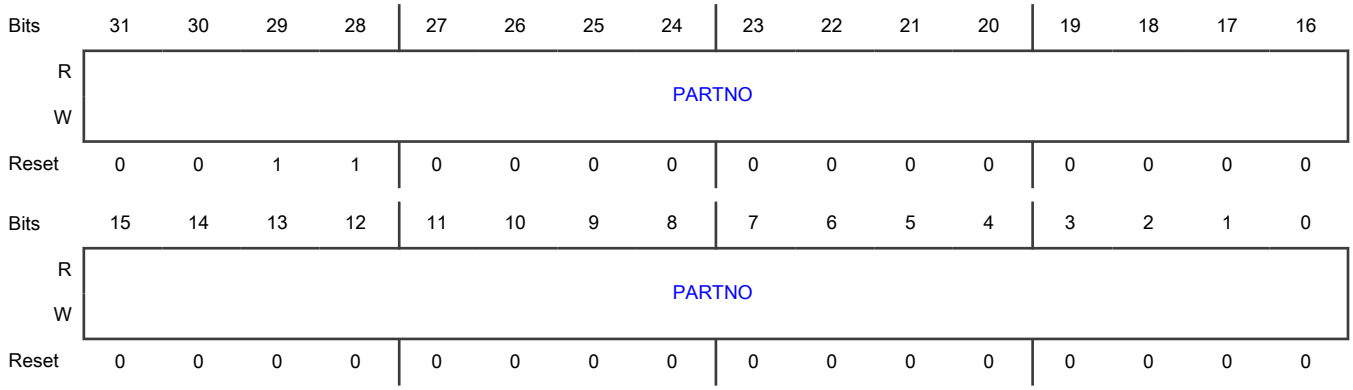
Offset

Register	Offset
SIDPARTNO	6Ch

Function

Allows you to write the ID part number. You must write a nonzero value into the PARTNO field because 0 is not valid.

Diagram



Fields

Field	Function
31-0 PARTNO	Part Number

41.7.25 Target ID Extension (SIDEXT)

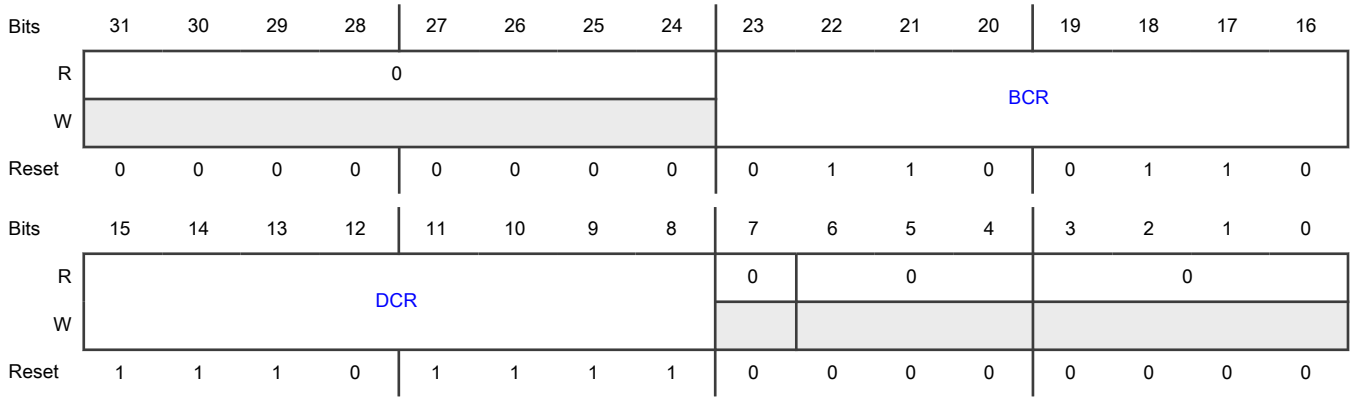
Offset

Register	Offset
SIDEXT	70h

Function

Allows you to write the ID extension of DCR and/or BCR.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 BCR	Bus Characteristics Register Sets the value for BCR, if this field is configured. Otherwise, the default value is considered. This field controls features such as secondary controller and slow-speed requirements.
15-8 DCR	Device Characteristic Register Sets the value for DCR, if this field is configured. Otherwise, the default value is considered.
7 —	Reserved
6-4 —	Reserved
3-0 —	Reserved

41.7.26 Target Vendor ID (SVENDORID)

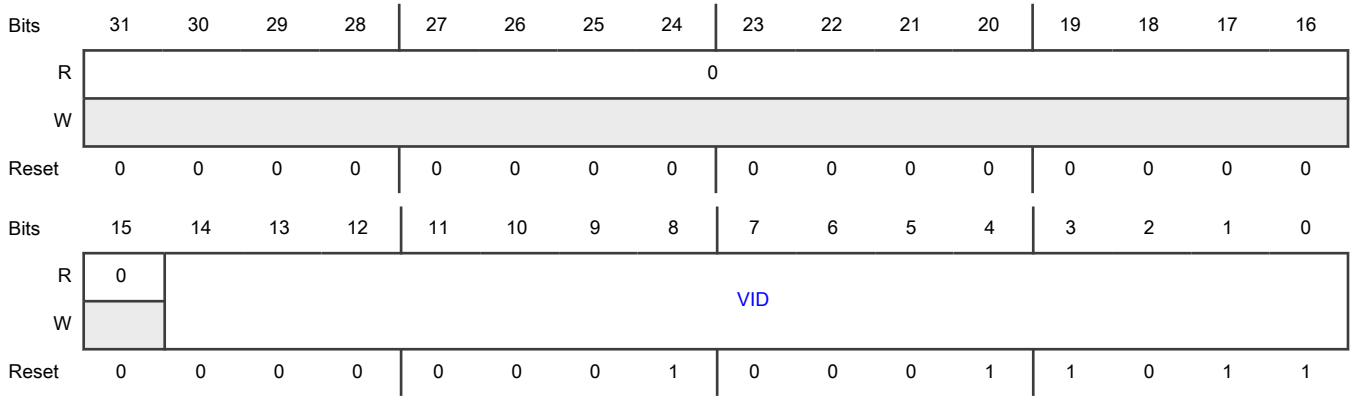
Offset

Register	Offset
SVENDORID	74h

Function

Allows you to write the vendor ID. The default value is the chip vendor ID, and is set from the constant field. When using the chip vendor ID, the part number (PARTNO) does 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 Configures the 15-bit MIPI vendor ID. If not configured, the default value is considered.

41.7.27 Target Time Control Clock (STCCLOCK)

Offset

Register	Offset
STCCLOCK	78h

Function

Allows you to dynamically set the time control clock and accuracy information. The clock frequency and accuracy are constants set by the hardware. If the clock can be adjusted (that is, divided) or if the accuracy could vary with knowable information, then the clock may be set via this register, which must 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	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	FREQ								ACCURACY							
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-16 —	Reserved
15-8 FREQ	Clock Frequency Indicates the clock frequency in 0.5 MHz steps. For example, a value of 20 in this field indicates a frequency of 10 MHz. Default set by parameters if configured.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 ACCURACY	Clock Accuracy Indicates the clock accuracy in 1/10ths of %. For example, a value of 15 indicates an accuracy of 1.5%. Default set by parameters if configured.

41.7.28 Target Message Map Address (SMSGMAPADDR)

Offset

Register	Offset
SMSGMAPADDR	7Ch

Function

Allows you to determine which address is matched when STATUS[MATCHED] = 1. This register is used when the DYNADDR register builds a list of extra DAs or SAs to match. It holds the last three matches.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												MAPLASTM2			
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				MAPLASTM1				0			LASTSTA...	MAPLAST			
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 MAPLASTM2	Matched Previous Index 2 Indicates index 2 of the previous matched address (0 for the base address).
15-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-8 MAPLASTM1	Matched Previous Address Index 1 Indicates index 1 of the previous matched address (0 for the base address).
7-5 —	Reserved
4 LASTSTATIC	Last Static Address Matched Indicates whether the last matched address was an I2C static address or an I3C dynamic address. 0b - I3C dynamic address 1b - I2C static address
3-0 MAPLAST	Matched Address Index Indicates the matched address index for current or last matched message (0 for the base address).

41.7.29 Controller Extended Configuration (MCONFIG_EXT)

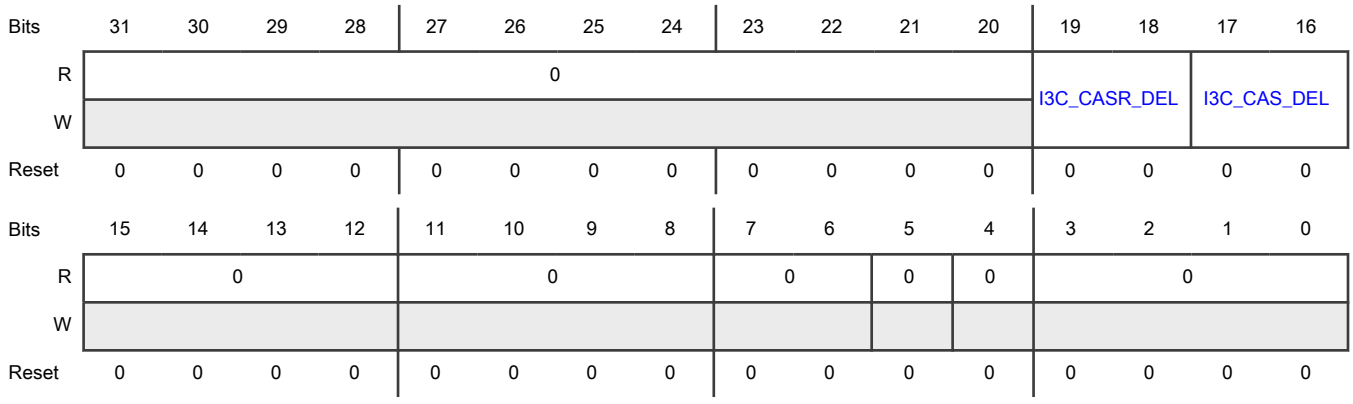
Offset

Register	Offset
MCONFIG_EXT	80h

Function

Allows special extended configurations, including for I2C use.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-18 I3C_CASR_DEL	I3C CAS Delay After Repeated START Increases SCL clock period by a certain value. You can configure this field to delay the clock after a Repeated START to provide more time for slow I3C devices. The maximum value may be less if the internal clock is very fast. 00b - No delay 01b - Increases SCL clock period by 1/2 10b - Increases SCL clock period by 1 11b - Increases SCL clock period by 1 1/2
17-16 I3C_CAS_DEL	I3C CAS Delay After START Increases SCL clock period by a certain value. By increasing the SCL clock period, you can delay the clock to be longer after the START. The maximum value may be less if the internal clock is very fast. 00b - No delay 01b - Increases SCL clock period by 1/2 10b - Increases SCL clock period by 1 11b - Increases SCL clock period by 3/2
15-12 —	Reserved
11-8 —	Reserved
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3-0 —	Reserved

41.7.30 Controller Control (MCTRL)

Offset

Register	Offset
MCTRL	84h

Function

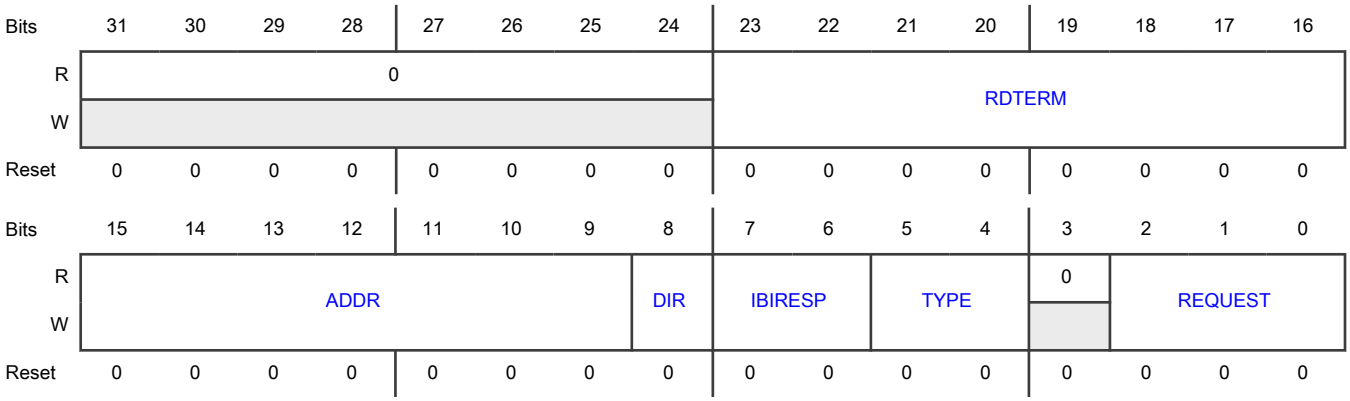
Starts activities on the I3C or I2C bus (see the MWMSG registers). A request cannot be changed when a message is in progress; the REQUEST field becomes 0 automatically.

You must write to MCTRL fields as per use case or as mentioned in Operating modes. Bit read and write checking may not work independently.

NOTE

If MCONFIG[MSTENA] is configured for I2C Controller mode (legacy I2C), only REQUEST = 1 and REQUEST = 2 are accepted. Also, fields are constrained to I2C-supported fields.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 RDTERM	Read Terminate Counter Determines when to terminate a read operation: • For I2C, this field controls when to NACK a read. • For I3C, this field can be used to terminate (end) a read: — If RDTERM is 0, it has no effect. — If RDTERM is 1, the read terminates after the next character.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— If RDTERM is 2, the read terminates after the next two characters. The field supports up to 255 characters. In DDR mode, RDTERM terminates the read based on word counts (for DDR) instead of byte counts (for SDR). The field self-clears when MSTATUS[COMPLETE] becomes 1. **NOTE** You may write 1 to this field anytime, but a value greater than 1 must be written when starting EmitStartAddress.
15-9 ADDR	Address Indicates the type of address: I3C dynamic address or I2C static address. Some values are not allowed based on the bus that is used (I3C or I2C).
8 DIR	Direction Indicates the direction, write or read. 0b - Write 1b - Read
7-6 IBIRESP	In-Band Interrupt Response Indicates the response to use when you get an IBI from START, and when to force using an IBI ACK NACK request when completing a manual IBI. Completion of a manual IBI means that the target DA is known, and so the mandatory byte (or not) is specified by the application when acknowledging. This field is also used when a message is emitted in Message mode using the MWMSG_SDR or MWMSG_DDR registers. If an IBI with MDB (mandatory byte) is ACKed, the controller limits it to a maximum of eight more bytes after the MDB. RDTERM = 1 can be used with REQUEST = NONE to terminate data from a target sooner. If this field is 0b, it indicates ACK (acknowledge). When REQUEST = 1 (EmitStartAddr) or REQUEST = 7 (AutoIBI), Controller In-band Interrupt Registry and Rules (MIBIRULES) decides whether to ACK with mandatory byte. When REQUEST = 3 (IBIAckNack), ACK with no mandatory byte. If this field is 1b, it indicates NACK (reject) or no acknowledgement. If this field is 10b, it indicates acknowledge with mandatory byte. When REQUEST = 1 or REQUEST = 7, ignore Controller In-band Interrupt Registry and Rules (MIBIRULES). Do not use this setting unless only targets with a mandatory byte can cause an IBI. When REQUEST = 3, ACK with mandatory byte. If this field is 11b, it indicates manual. When REQUEST = 1 or REQUEST = 7, stop and wait for a decision using the IBI ACK NACK request. When REQUEST = 3, this field is reserved. **NOTE** A CR and HJ always cause IBIRESP to become 3 (manual) so the application must decide. 00b - ACK (acknowledge) 01b - NACK (reject) 10b - Acknowledge with mandatory byte

Table continues on the next page...

Table continued from the previous page...

Field	Function
	11b - Manual
5-4 TYPE	Bus Type with EmitStartAddr Works with REQUEST to determine the bus type. See Determining bus types and modes for more information. The following values are valid only if REQUEST is 1. 00b - I3C 01b - I2C 10b - DDR 11b - Reserved
3 —	Reserved
2-0 REQUEST	Request Emits the requested operation when performing in pieces, instead of performing by message. You must check Controller Status (MSTATUS) because some requests can only be made in some states. For example, the system cannot enter SDR mode from DDR mode, and it cannot use an incorrect request in DAA mode. If this field is 0b, it indicates that no request is present (NONE). The REQUEST field returns to NONE after finishing a request. Controller Status (MSTATUS) indicates the state of the controller. See AutoIBI mode. NONE is written as 0 only in these cases: when writing 1 to MCTRL[RDTERM] (to stop a read in progress) or when configuring MCTRL[IBIRESP] for MSG use. If this field is 1b, it emits START with address and direction (EMITSTARTADDR), either from Stopped state or in the middle of a single data rate (SDR) message. If from Stopped state (Idle), then EmitStart may be prevented by an event (such as IBI, CR, HJ). In this case, an appropriate interrupt is signaled. EmitStart can be resubmitted. If this field is 10b, it emits a STOP on bus (EMITSTOP). Must be in SDR mode. In DAA mode, emitting STOP exits DAA mode. If this field is 11b, it indicates manual IBI ACK or NACK (IBIACKNACK). When MCTRL[IBIRESP] indicates a hold on an IBI to allow a manual decision, this request completes it. The field uses MCTRL[IBIRESP] to provide the information. If this field is 100b, and if currently not in DAA mode, it emits START, 7E, ENTDAA sequence, and then emits 7E/R to process the first target (PROCESSDAA). Stops just before the new dynamic address (DA) is to be emitted. The DA is written using Controller Write Data Byte (MWDATAB), then process DAA is requested again to write the new address, and then it starts the next unless marked to STOP. An MSTATUS indicating NACK means DA was not accepted (for example, parity error). If PROCESSDAA is NACKed on the 7E/R request, meaning no more targets need a DA, then a COMPLETE is signaled (along with DONE) and a STOP issued. If TYPE = 2 or 3, the DA is assigned and then it emits a STOP (instead of starting a new 7E/R request). If TYPE = 1 or 3, then the DA is taken from the ADDR field (bits 6:0). If this field is 110b, it emits an exit pattern from any state (force exit and target reset). End DDR mode (including MSGDDR), including a STOP afterward. If MCTRL[TYPE] is 2, perform a target reset action

Table continues on the next page...

Table continued from the previous page...

Field	Function
	(RSTACT can prevent the reset in the target). Target reset may follow immediately after RSTACT CCC or STOP. If this field is 111b, the target is held in a Stopped state, but autoemits a START, 7E sequence when the target holds SDA low for an IBI (AUTOIBI). Actual IBI handling is defined by MCTRL[IBIRESP]. 000b - NONE 001b - EMITSTARTADDR 010b - EMITSTOP 011b - IBIACKNACK 100b - PROCESSDAA 101b - Reserved 110b - Force Exit and Target Reset 111b - AUTOIBI

41.7.31 Controller Status (MSTATUS)

Offset

Register	Offset
MSTATUS	88h

Function

Indicates the controller status, including which events cause interrupts. The peripherals share the IRQ (called parallel-to-target status).

Because a peripheral can either be in Controller or Target mode, but not both at the same time, only one (target or controller peripheral) can cause the IRQ. If there is an IRQ and the peripheral is a controller, then this register (MSTATUS) indicates the status.

If there is an IRQ and the peripheral is a target, then Target Status (SSTATUS) indicates the status.

This register self-clears if the COMPLETE field becomes 1.

If the MCONFIG register is configured only for I2C, some states and fields never remain active (for example, DAA or IBI).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	IBIADDR							0				NOWMAST...	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	ERRWARN	0	IBIWON	TXNOTFU...	RXPEND	COMPLETE	MCTRLDO...	SLVSTART	IBITYPE		NACKED	BETWEEN	0	STATE		
W			W1C			W1C	W1C	W1C								
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-24 IBIADDR	IBI Address Indicates the address of: • The IBI when MSTATUS[IBITYPE] = 1. • The CR, when MSTATUS[IBITYPE] = 2. • 7'h2 when HJ, when MSTATUS[IBITYPE] = 3.
23-20 —	Reserved
19 NOWMASTER	Module is now Controller Flag Indicates whether the module is now a controller. That is, it was previously a target, and controllership acceptance was requested from the previous controller and controllership was accepted. The reverse operation (controller becomes a target) does not need an interrupt because the application grants it through the GETACCMST CCC. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not a controller 1b - Controller When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 —	Reserved
15 ERRWARN	Error or Warning Indicates whether an error occurred, such as improper register use, overrun, or underrun of FIFO or buffer, or invalid parity or CRC in a DDR read. See Controller Errors and Warnings (MERRWARN) for more information. 0b - No error or warning 1b - Error or warning
14 —	Reserved
13 IBIWON	In-Band Interrupt (IBI) Won Flag Indicates whether an IBI, CR, or HJ has won the arbitration on a header address, regardless of whether it was NACKed or ACKed. Arbitration requires manual intervention for CR and HJ, and optionally requires it for IBI if MCTRL[IBIRESP] = 3 (manual). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No IBI arbitration won 1b - IBI arbitration won When writing 0b - No effect 1b - Clear the flag
12 TXNOTFULL	TX Buffer or FIFO Not Full Indicates whether the buffer, FIFOm, or message register can accept another byte or halfword. FIFO uses trigger level. If DMA is enabled for transmitting, it transfers data as long as it is not full. 0b - Receive buffer or FIFO full 1b - Receive buffer or FIFO not full
11 RXPEND	RXPEND Indicates whether a message is being received from a target and bytes are in the input buffer or FIFO: • When using a FIFO, this message is at least one FIFO trigger's worth (a minimum of one byte in the FIFO). • When DMA is enabled for receiving, the DMA is signaled. RXPEND becomes 0 when the data is read.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No receive message pending 1b - Receive message pending
10 COMPLETE	Complete Flag Indicates whether a message is complete: • With MWMSG_SDR or MWMSG_DDR, the COMPLETE condition occurs when MWMSG_SDR_CONTROL[LEN] or MWMSG_DDR_CONTROL[LEN] reaches 0. • When MCTRL[REQUEST] = 1 (EmitStartAddr), this COMPLETE condition occurs after the end of a write operation, or after a read operation has terminated or ended. • With an IBI (AutoIBI or EmitStartAddr), the COMPLETE condition occurs at the end of IBI data (if any). **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
9 MCTRLDONE	Controller Control Done Flag Indicates whether the module has completed an MCTRL request. MCTRLDONE automatically becomes 0 when writing a new control. If MCTRL[REQUEST] = 1 (EmitStartAddr), MCTRLDONE becomes 1 when the address goes out (and is ACKed, NACKed, or ended in an IBI). If ACKed, MSTATUS[COMPLETE] becomes 1 after the write or read data is complete. If MCTRL[REQUEST] = 4 (ProcessDAA), MCTRLDONE becomes 1 when the module is ready to emit the dynamic address (DA) for the target, or when no more targets are ACKing. This condition can be determined by using the MSTATUS[BETWEEN] and MSTATUS[STATE] fields. **NOTE** This field behaves differently for register reads and writes. When reading 0b - Not done 1b - Done When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 SLVSTART	Target Start Flag Indicates whether a target is or was requesting a START by holding SDA low. Handling starts automatically when MCTRL[REQUEST] = 7 (AutoIBI). **NOTE** This field behaves differently for register reads and writes. When reading 0b - Target not requesting START 1b - Target requesting START When writing 0b - No effect 1b - Clear the flag
7-6 IBITYPE	In-Band Interrupt (IBI) Type Indicates the type of IBI of the last event that won the arbitration, whether the interrupt was ACKed, NACKed, or pending. 00b - NONE (no IBI: this status occurs when MSTATUS[IBIWON] becomes 0) 01b - IBI 10b - CR 11b - HJ
5 NACKED	Not Acknowledged Indicates whether the last start and address sequence was NACKed (was not ACKed by the addressed target). 0b - Not NACKed 1b - NACKed (not acknowledged)
4 BETWEEN	Between Indicates whether the controller is active between messages or dynamic address assignments (DAA). It is active when: • MSTATUS[STATE] is MSGSDR, DDR, or DAA, or NORMACT, and the state is between messages or DAAs. It is expecting a new message or DAA to start (or STOP or exit). • MSTATUS[STATE] is NORMACT. The module is waiting for the transmit FIFO to be not empty or the receive FIFO to be not full. 0b - Inactive (for other cases) 1b - Active
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 STATE	State of the Controller Indicates the current controller state. If this field is 0b, it indicates that the controller is idle (the bus has stopped.) If this field is 1b, it indicates target request. The bus has stopped but a target is holding SDA low. When using auto-emit IBI (MCTRL[REQUEST] = 7), the controller does not remain in this state. If this field is 10b, it indicates entry into Single Data Rate Message mode, using MWMSG_SDR. If this field is 11b, it indicates entry into normal active SDR mode, using MCTRL, MWDATA*n*, and MRDATA*n* registers. The controller remains in this state until a STOP is issued. If this field is 100b, it indicates entry into Double Data Rate Message mode, using MWMSG_DDR or the normal method with DDR. The controller remains in the DDR state until the controller exits using EXIT (emitting the Exit pattern). If this field is 101b, it indicates entry into Dynamic Address Assignment (ENTDAA) mode. If this field is 110b, it indicates wait for an IBI ACK/NACK decision. If this field is 111b, it indicates receiving an IBI. This state is used after IBI, CR, or HJ has won an arbitration. The IBIRCV state is also used for IBI mandatory byte (if any) and any bytes that follow. 000b - IDLE (bus has stopped) 001b - SLVREQ (target request) 010b - MSGSDR 011b - NORMACT 100b - MSGDDR 101b - DAA 110b - IBIACK 111b - IBIRCV

41.7.32 Controller In-band Interrupt Registry and Rules (MIBIRULES)

Offset

Register	Offset
MIBIRULES	8Ch

Function

Contains the rules for using IBI, and keeps a registry of the targets that use the IBI byte.

This register defines the IBI mandatory byte rules for the targets and determines which targets do (or do not) have a mandatory byte.

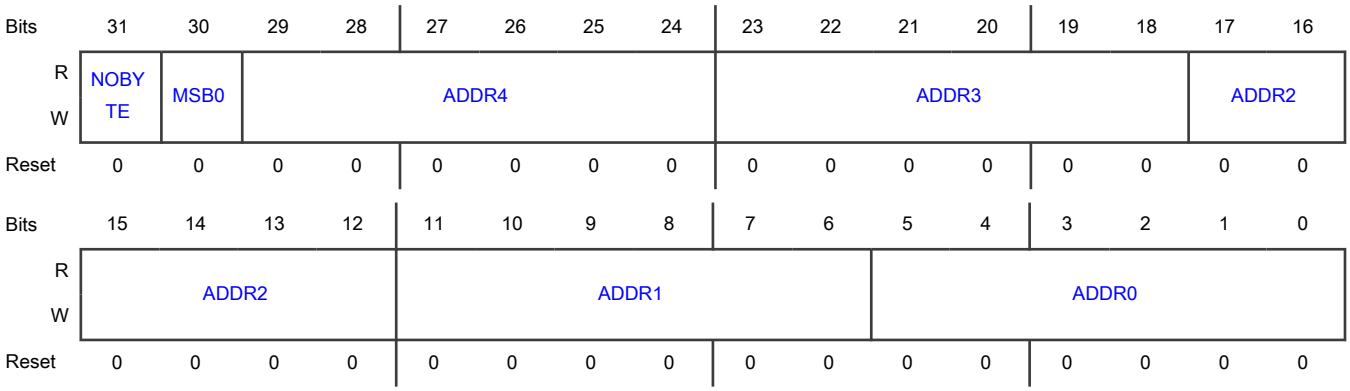
Concerning ADDR*n* fields: The address has 6 bits. Assuming that MIBIRULES[MSB0] = 1, the most significant bit of each address is 0. In this case, each address is 7 bits (usually written as A7 to A1) and A7 must be 0. If the application does not use that optimal convention, the "manual" method of IBI ACK handling must be used (see MCTRL[IBIRESP] for more information).

By default, the ADDR*n* values indicate the targets with a mandatory byte. If MIBIRULES[NOBYTE] = 1, the ADDR*n* values indicate targets that do not have a mandatory IBI byte.

NOTE

A7 = 0 is only needed for targets that use IBI. For legacy I2C devices, A7 can use any valid value because I2C devices cannot cause an IBI.

Diagram



Fields

Field	Function
31 NOBYTE	No IBI byte Specifies whether the ADDR*n* fields refer to targets with or without a mandatory IBI byte. 0b - With mandatory IBI byte 1b - Without mandatory IBI byte
30 MSB0	Most Significant Address Bit is 0 Specifies whether MSB is 0 for all I3C dynamic addresses. Assigning 1 to this field allows the START header to be optimized. 0b - MSB is not 0 1b - MSB is 0
29-24 ADDR4	ADDR4 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
23-18 ADDR3	ADDR3 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
17-12 ADDR2	ADDR2 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-6 ADDR1	ADDR1 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.
5-0 ADDR0	ADDR0 Specifies the address of the target with or without mandatory IBI byte. If 0, then the address does not apply.

41.7.33 Controller Interrupt Set (MINTSET)

Offset

Register	Offset
MINTSET	90h

Function

Returns the status of the interrupt enables when read. This register sets interrupt enables for select fields in Controller Status (MSTATUS):

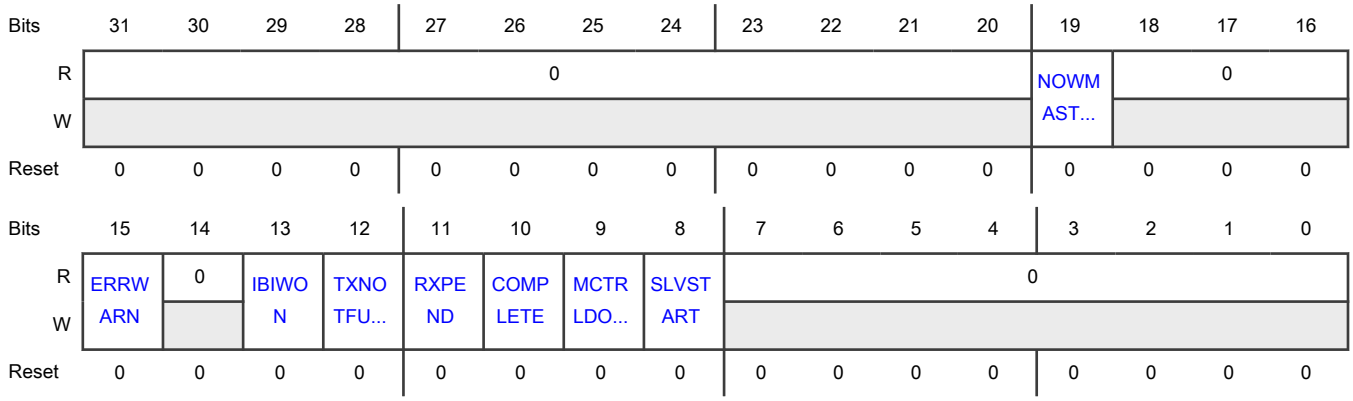
- To activate an interrupt enable, write 1 to its corresponding field.
- To disable an interrupt enable, write 1 to the appropriate field in Controller Interrupt Clear (MINTCLR). Writing 0 to the interrupt enable in MINTSET does not disable the interrupt.

The interrupt registers allow masking of interrupt sources as well as checking which interrupts are activated in Controller Status (MSTATUS). The normal method is to enable an interrupt and then after that interrupt fires, clear the interrupt either by writing to Controller Status (MSTATUS) or via an action on the corresponding data register. The interrupt is level-held, meaning that the interrupt remains 1 until the cause is cleared by some method. The module prevents races; if a new event comes in, that new event is not lost.

These interrupts are parallel to Target Interrupt Set (SINTSET) and Target Interrupt Clear (SINTCLR), and only one interrupt set is active depending on the state (controller or target operation):

- MINTSET: Sets interrupt enables for MSTATUS fields. Reading the MINTSET register returns the status of the interrupt enables.
- MINTCLR: Clears interrupt enables for MSTATUS fields.
- MINTMASKED: Returns the value of the MSTATUS fields ANDed with their interrupt enables.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	Now Controller Interrupt Enable Enables the corresponding interrupt to indicate whether the module is now a controller (now this I3C module is a controller). That is, it was previously a target; controllership acceptance was requested from the previous controller and it was accepted. 0b - Disable 1b - Enable
18-16 —	Reserved
15 ERRWARN	Error or Warning (ERRWARN) Interrupt Enable Enables the corresponding interrupt to indicate whether an error occurred, such as improper register use, overrun or underrun of FIFO or buffer, or invalid parity or CRC in a DDR read. 0b - Disable 1b - Enable
14 —	Reserved
13 IBIWON	IBI Won Interrupt Enable Enables the corresponding interrupt to indicate whether an IBI, CR, or HJ has won the arbitration on a header address, regardless of whether it was NACKed or ACKed. 0b - Disable 1b - Enable
12	Transmit Buffer/FIFO Not Full Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXNOTFULL	Enables the corresponding interrupt to indicate whether the buffer, FIFOm, or message register can accept another byte or halfword. 0b - Disable 1b - Enable
11 RXPEND	Receive Pending Interrupt Enable Enables the corresponding interrupt to indicate whether a message is being received from a target and bytes are in the input buffer or FIFO.
10 COMPLETE	Completed Message Interrupt Enable Enables the corresponding interrupt to indicate whether a message is complete. 0b - Disable 1b - Enable
9 MCTRLDONE	Controller Control Done Interrupt Enable Enables the corresponding interrupt to indicate whether the module has completed an MCTRL request. 0b - Disable 1b - Enable
8 SLVSTART	Target Start Interrupt Enable Enables the corresponding interrupt to indicate whether a target is or was requesting a START by holding SDA low. 0b - Disable 1b - Enable
7-0 —	Reserved

41.7.34 Controller Interrupt Clear (MINTCLR)

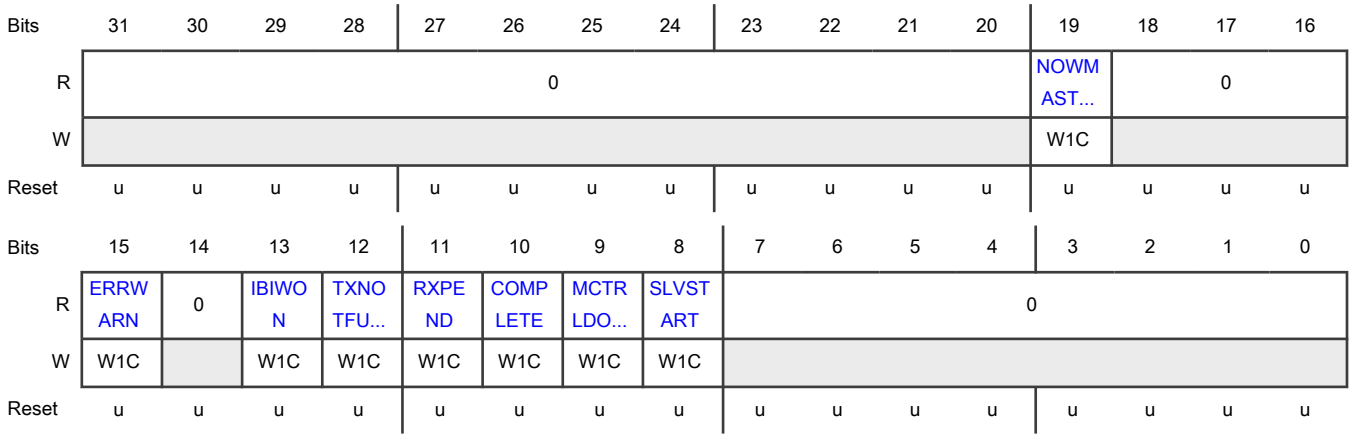
Offset

Register	Offset
MINTCLR	94h

Function

Clears interrupt enables for select fields in Controller Status (MSTATUS). Writing 1 clears the corresponding interrupt enable. Writing 0 has no effect.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	NOWCONTROLLER Interrupt Enable Clear Flag Clears the corresponding NOWCONTROLLER interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
18-16 —	Reserved
15 ERRWARN	ERRWARN Interrupt Enable Clear Flag Clears the corresponding ERRWARN interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When writing 0b - No effect 1b - Clear the flag
14 —	Reserved
13 IBIWON	IBIWON Interrupt Enable Clear Flag Clears the corresponding IBIWON interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
12 TXNOTFULL	TXNOTFULL Interrupt Enable Clear Flag Clears the corresponding TXNOTFULL interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
11 RXPEND	RXPEND Interrupt Enable Clear Flag Clears the corresponding RXPEND interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
10 COMPLETE	COMPLETE Interrupt Enable Clear Flag Clears the corresponding COMPLETE interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
9 MCTRLDONE	MCTRLDONE Interrupt Enable Clear Flag Clears the corresponding MCTRLDONE interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing 0b - No effect 1b - Clear the flag
8 SLVSTART	SLVSTART Interrupt Enable Clear Flag Clears the corresponding SLVSTART interrupt enable. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No effect 1b - Interrupt enable cleared When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
7-0 —	Reserved

41.7.35 Controller Interrupt Mask (MINTMASKED)

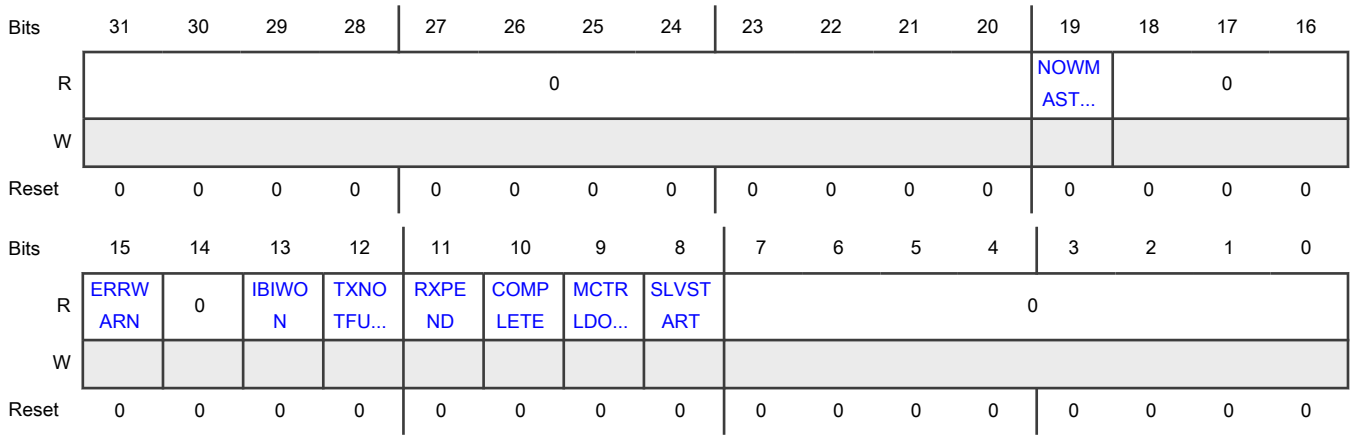
Offset

Register	Offset
MINTMASKED	98h

Function

Returns the status of enabled interrupts (the value of Controller Status (MSTATUS) ANDed with the value of Controller Interrupt Set (MINTSET)).

Diagram



Fields

Field	Function
31-20 —	Reserved
19 NOWMASTER	NOWCONTROLLER Interrupt Mask Indicates whether the NOWCONTROLLER interrupt is enabled and active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
18-16 —	Reserved
15 ERRWARN	ERRWARN Interrupt Mask Indicates whether the ERRWARN interrupt is enabled and active. 0b - Disabled 1b - Enabled
14 —	Reserved
13 IBIWON	IBIWON Interrupt Mask Indicates whether the IBIWON interrupt is enabled and active. 0b - Disabled 1b - Enabled
12 TXNOTFULL	TXNOTFULL Interrupt Mask Indicates whether the TXNOTFULL interrupt is enabled and active. 0b - Disabled 1b - Enabled
11 RXPEND	RXPEND Interrupt Mask Indicates whether the RXPEND interrupt is enabled and active.
10 COMPLETE	COMPLETE Interrupt Mask Indicates whether the COMPLETE interrupt is enabled and active. 0b - Disabled 1b - Enabled
9 MCTRLDONE	MCTRLDONE Interrupt Mask Indicates whether the MCTRLDONE interrupt is enabled and active. 0b - Disabled 1b - Enabled
8 SLVSTART	SLVSTART Interrupt Mask Indicates whether the SLVSTART interrupt is enabled and active.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
7-0 —	Reserved

41.7.36 Controller Errors and Warnings (MERRWARN)

Offset

Register	Offset
MERRWARN	9Ch

Function

Contains errors and warnings. When any errors or warnings are not 0, then MSTATUS[ERRWARN] is 1.

Parallel-to-target ERRWARN:

- In Controller mode, use this register (MERRWARN).
- In Target mode, use Target Errors and Warnings (SERRWARN).

The error fields in both registers (MERRWARN and SERRWARN) are similar in meaning.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										0	TIMEOUT	INVREQ	MSGERR	OWRITE	OREAD
W												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	0			0	0	HCRC	HPAR	0				TERM	WRABT	NACK	URUN	0
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
31-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 TIMEOUT	Timeout Error Flag Indicates an error caused by the module stalling for too long in a frame. This stalling occurs when: • The transmit FIFO or receive FIFO is not handled, and the bus is stuck in the middle of a message. • No STOP is issued after data transfer (there is a gap between messages). • IBI manual is used and no decision has been made. The maximum stall period is 10 kHz or 100 µs. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
19 INVREQ	Invalid Request Error Flag Indicates an error caused by an invalid use of a request: • Not using IBI ACK NACK when stopped in manual hold for IBI acknowledgment. • Using a request other than ForceStop or ForceExit when in a message. Other requests are valid when the message is done. • Other mismatched uses (for example, IBI ACK NACK in normal states). **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
18 MSGERR	Message Error Flag Indicates an error caused by:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Trying to write to or read from the MWMSG_SDR register when in a DDR message. • Trying to write to or read from the MWMSG_DDR register when in an SDR message. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
17 OWRITE	Overwrite Error Flag Indicates an error caused by trying to write to Controller Write Data Byte (MWDATAB) when the FIFO is full. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
16 OREAD	Overread Error Flag Indicates an error caused by: • Trying to read from Controller Read Data Byte (MRDATAB) when the FIFO is empty. • Trying to read from the MRMSG_SDR register or the MRMSG_DDR register when no message has yet started. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No effect 1b - Clear the flag
15-13 —	Reserved
12 —	Reserved
11 —	Reserved
10 HCRC	High Data Rate CRC Error Flag Indicates that a cyclic redundancy check (CRC) error occurred from a DDR read. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
9 HPAR	High Data Rate Parity Flag Indicates a parity error from a DDR read, including a bad preamble on a read. This does not stop the read because it is not safe to terminate; the read data may become misframed. Ends on a run of 1 second. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
8-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 TERM	Terminate Error Flag Indicates an error when this controller terminates a target read because the read exceeded the count for the message. This error is valid only when using the MWMSG_SDR or MWMSG_DDR register. If you write to the MWMSG_SDR or MWMSG_DDR register, this field automatically becomes 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
3 WRABT	Write Abort Error Flag Indicates an error caused by the I2C target NACKing the write data, terminating the message. For example, the controller is writing in I2C and the target NACKed the write. If you write to the MCTRL register, this field automatically becomes 0. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
2 NACK	Not Acknowledge Error Flag Indicates an error caused by the target or targets NACKing (not acknowledging) the last address. If 7Eh is the address, then it indicates all targets NACKed the last address. If you write to the MCTRL register, this field automatically becomes 0. **NOTE** In HDR mode, this error occurs when an address is not accepted (as opposed to NACKed). **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
	0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
1 URUN	Underrun Error Flag Indicates an underrun for HDR-BT. This error occurs when attempting to write from an empty transmit FIFO. **NOTE** This field behaves differently for register reads and writes. When reading 0b - No error 1b - Error When writing 0b - No effect 1b - Clear the flag
0 —	Reserved

41.7.37 Controller DMA Control (MDMACTRL)

Offset

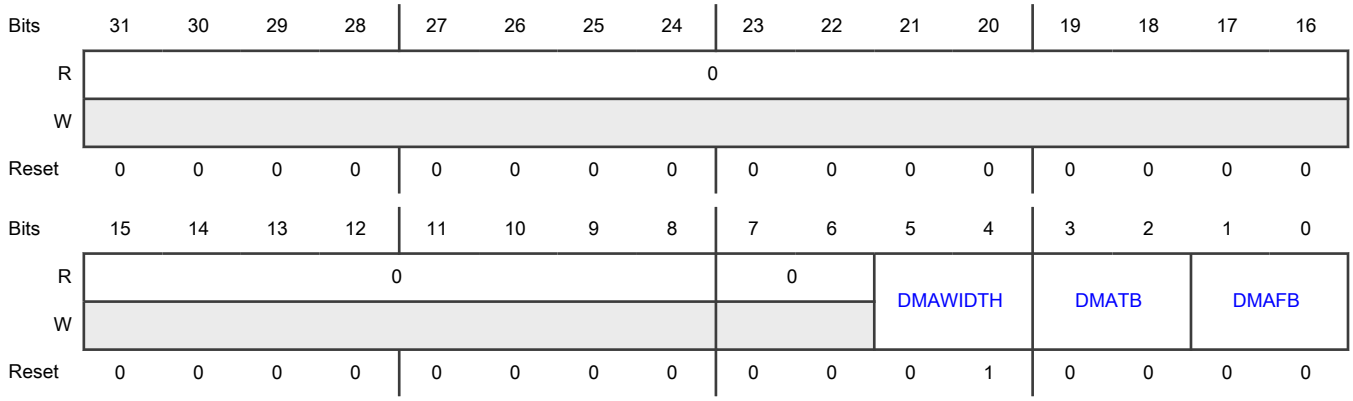
Register	Offset
MDMACTRL	A0h

Function

Allows DMA to be used for inbound and outbound messages. DMA is much more useful for a controller than for a target because the controller directs the bus traffic and actions. DMA can be used with a controller in these ways:

- Push or pull data for MCTRL[REQUEST] = 1 (EmitStartAddr) request written by the processor.
- Implementing message mode, completely controlled by DMA.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-6 —	Reserved
5-4 DMAWIDTH	DMA Width Specifies the data width of DMA operations. The halfword (16 bits) setting ensures that two bytes are free or available in FIFO. 00b,01b - Byte 10b - Halfword (16 bits) 11b - Reserved
3-2 DMATB	DMA to Bus Represents the DMA write (to-bus) trigger. When DMAFB = 1 or DMAFB = 2, I3C requests DMA when a transmit trigger occurs (see Controller Data Control (MDATACTRL)). I3C requests until full, unless the DMA is set up as a trigger. DMAFB becomes 0 when MSTATUS[ERRWARN] = 1. If this field is 1b, STOP or repeated START causes DMATB to become 0 (see SCONFIG[MATCHSS]). Normally, DMA enable must be used only in Controller Message mode. 00b - DMA not used 01b - Enable DMA for one frame (ended by DMA or terminated) 10b - Enable DMA until DMA is turned off. Recommended mode, instead of 01b, to enable DMA. This setting provides better software control. 11b - Reserved
1-0	DMA from Bus

Table continues on the next page...

Table continued from the previous page...

Field	Function
DMAFB	Represents the DMA read (from-bus) trigger. When DMAFB = 1 or DMAFB = 2, I3C requests DMA when a receive trigger occurs (see Controller Data Control (MDATACTRL)). I3C requests until empty unless the DMA is set up as a trigger. DMAFB becomes 0 when MSTATUS[ERRWARN] = 1. If this field is 1b, STOP or Repeated START causes DMAFB to automatically become 0 (see SCONFIG[MATCHSS]). **NOTE** Do not enable DMA after a transaction starts. It must be enabled only when enabling the controller and interrupts to avoid any kind of RX FIFO overrun. 00b - DMA not used 01b - Enable DMA for one frame 10b - Enable DMA until DMA is turned off. Recommended mode, instead of 01b, to enable DMA. This setting provides better software control. 11b - Reserved

41.7.38 Controller Data Control (MDATACTRL)

Offset

Register	Offset
MDATACTRL	ACh

Function

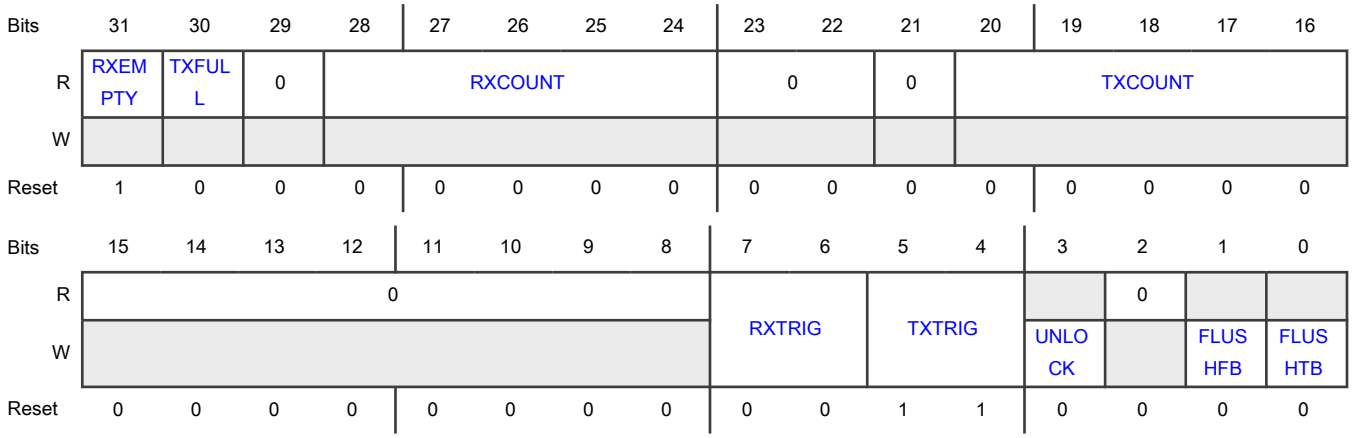
Assists in data control when there is no FIFO. This register also assists the FIFO when FIFO is available (regardless of size), and this assistance allows some control over the FIFO behavior. In particular, the register provides control over when to interrupt when a particular state of fullness or emptiness is reached. It also controls behavior related to width, when the width is not 1 byte wide.

This register acts as an alias of Target Data Control (SDATACTRL).

NOTE

When flushing a FIFO if DMA is in use, disable the DMA channel first. You must not use FIFO flush when a message (in that direction) is in flight.

Diagram



Fields

Field	Function
31 RXEMPTY	Receive is Empty Indicates whether the receive FIFO or buffer is empty. 0b - Not empty 1b - Empty
30 TXFULL	Transmit is Full Indicates whether the transmit FIFO or buffer is full. 0b - Not full 1b - Full
29 —	Reserved
28-24 RXCOUNT	Receive Entry Count Contains the count of entries in the receive FIFO or buffer. Entry size for SDR and HDR-DDR is in bytes.
23-22 —	Reserved
21 —	Reserved
20-16 TXCOUNT	Transmit Entry Count Contains the count of entries waiting in the TXFIFO. This count is the number of entries that the application has written to the transmit FIFO that have not yet gone onto the bus. Entry size for SDR and HDR-DDR is in bytes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7-6 RXTRIG	Receive Trigger Level Indicates the trigger level for receive fullness when using a FIFO. This field affects the RXPEND interrupt. 00b - Trigger when not empty (default) 01b - Trigger when 1/4 full or more 10b - Trigger when 1/2 full or more 11b - Trigger when 3/4 full or more
5-4 TXTRIG	Transmit Trigger Level Indicates the trigger level for transmit emptiness when using a FIFO. This field affects the TXNOTFULL interrupt. 00b - Trigger when empty 01b - Trigger when 1/4 full or less 10b - Trigger when 1/2 full or less 11b - Trigger when 1 less than full or less (default)
3 UNLOCK	Unlock Unlocks the functionality so that the RXTRIG and TXTRIG fields can be changed on a write. This field must be 1 (unlocked) in the same cycle when writing to TXTRIG or RXTRIG. If this field is 0 (locked), RXTRIG and TXTRIG fields cannot be changed on a write. 0b - Locked 1b - Unlocked
2 —	Reserved
1 FLUSHFB	Flush From-Bus Buffer or FIFO Flushes from-bus buffer or FIFO. You normally do not use this field. **NOTE** Before using FLUSHFB, disable the DMA (MDMACTRL[DMAFB] = 0). Then use FLUSHFB to flush the FIFO and then re-enable the DMA. 0b - No action 1b - Flush the buffer
0	Flush To-Bus Buffer or FIFO

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLUSHTB	Flushes to-bus buffer or FIFO. This field is used when the controller terminates a to-bus message (read) prematurely. **NOTE** Before using FLUSHTB, disable the DMA (MDMACTRL[DMATB] = 0). Then use FLUSHTB to flush the FIFO and then re-enable the DMA. 0b - No action 1b - Flush the buffer

41.7.39 Controller Write Data Byte (MWDATAB)

Offset

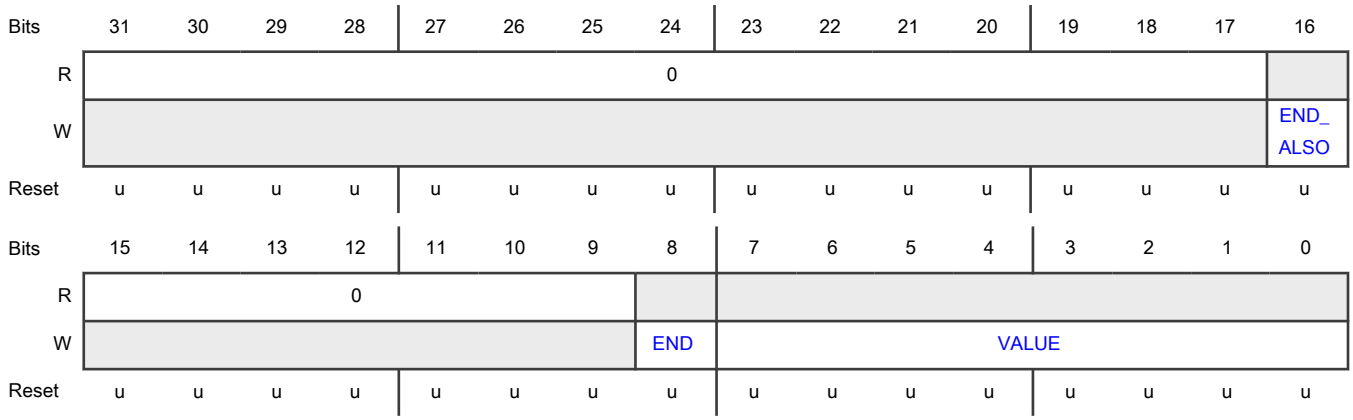
Register	Offset
MWDATAB	B0h

Function

Allows writing bytes to send onto the bus. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

This register acts as an alias of Target Write Data Byte (SWDATAB).

Diagram



Fields

Field	Function
31-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 END_ALSO	End of Message ALSO Indicates end of message, used to end an outbound message normally. Every message must indicate when it is the last message to be sent. This method can be used with the MDATABE register. This field is required for I3C and is optional for I2C. For HDR-DDR, the byte with END_ALSO must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. If this field is 0, more bytes are assumed to be in the message. If this field is 1, the END bit marks the last byte of the message. 0b - Not the end 1b - End
15-9 —	Reserved
8 END	End of Message Indicates the end of a message. This field is required for I3C and is optional for I2C. For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. If this field is 0, more bytes are assumed to be in the message. If this field is 1, the END bit marks the last byte of the message. 0b - Not the end 1b - End
7-0 VALUE	Data Byte Represents the byte written to the target (stored in transmit FIFO): • I3C computes the parity. • I2C manages the ACK or NACK.

41.7.40 Controller Write Data Byte End (MWDATABE)

Offset

Register	Offset
MWDATABE	B4h

Function

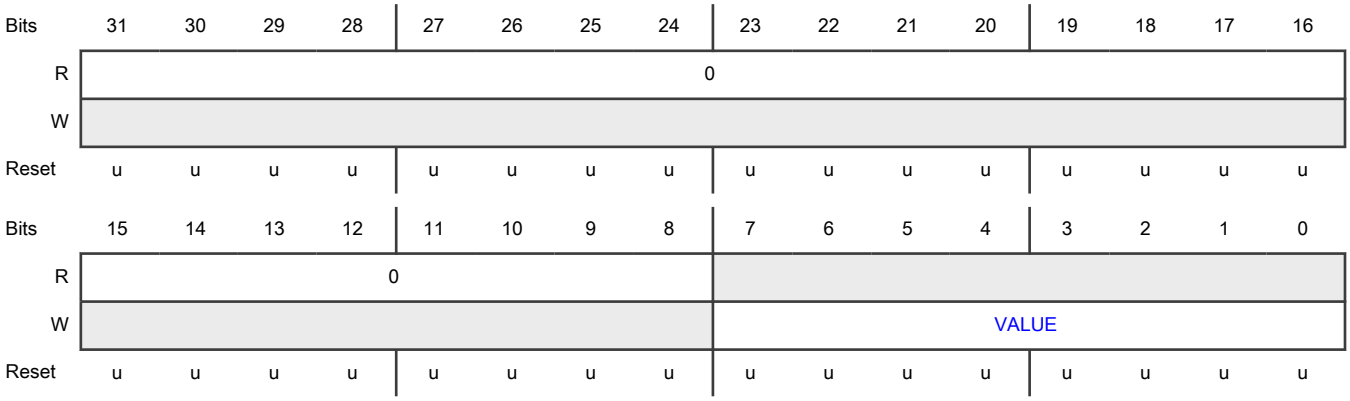
Allows writing the last byte to send onto the bus. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

This register acts as an alias of Target Write Data Byte End (SWDATABE).

NOTE

MWDATABE can also indicate END using bit 8.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	Data Represents the last byte written to the target (stored in transmit FIFO): • I3C computes the parity. • I2C manages the ACK or NACK.

41.7.41 Controller Write Data Halfword (MWDATAH)

Offset

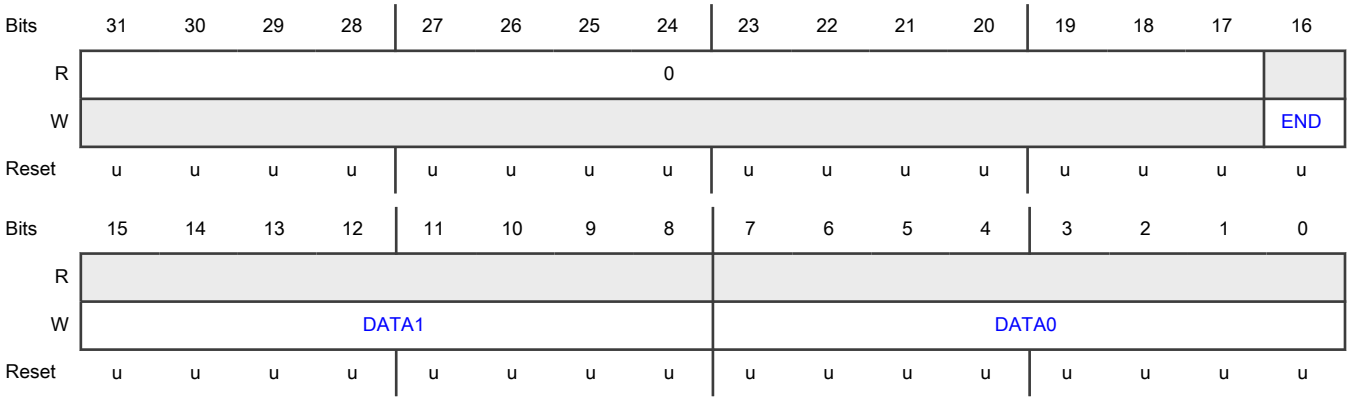
Register	Offset
MWDATAH	B8h

Function

Allows writing a halfword (pair of bytes) to the bus unless an external FIFO is used, sending the low byte followed by the high byte. An end-of-data (last) marker bit is allowed (or must be 0). A halfword must not be written unless there is room for both, as indicated by the use of transmit FIFO level trigger or MDATACTRL[TXCOUNT].

This register acts as an alias of Target Write Data Halfword (SWDATAH).

Diagram



Fields

Field	Function
31-17 —	Reserved
16 END	End of Message Indicates the end of a message. For this register, this field always marks DATA1 as the end. This field is required for I3C and is optional for I2C. For HDR-DDR, the byte with the END must be an even byte (second, fourth, sixth, and so on) because DDR uses byte pairs. If this field is 0, more bytes are assumed to be in the message. If this field is 1, the END bit marks the last byte of the message. 0b - Not the end 1b - End
15-8 DATA1	Data Byte 1 Represents the second byte that is sent to the target (written to transmit FIFO).
7-0 DATA0	Data Byte 0 Represents the first byte that is sent to the target (written to transmit FIFO).

41.7.42 Controller Write Data Halfword End (MWDATAHE)

Offset

Register	Offset
MWDATAHE	BCh

Function

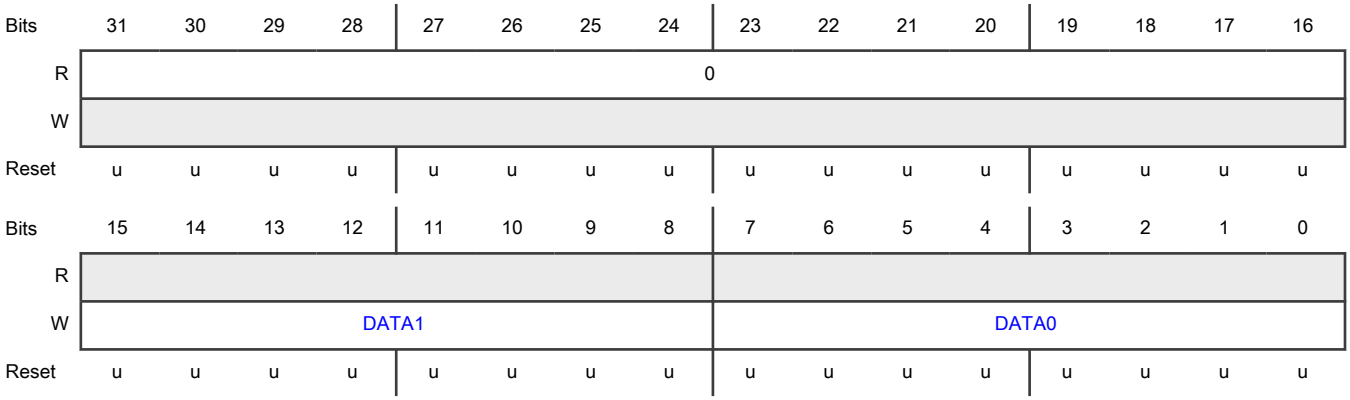
Allows writing a halfword of data, which is the end (the last byte of the halfword is the end). The target writes the halfword (byte pair) in the same way as it does to Controller Write Data Halfword (MWDATAH), but marks the second byte as end-of-data (last byte).

For HDR-DDR, the byte with the END must be even (second, fourth, sixth, and so on) because DDR uses byte pairs.

You must not write a halfword unless there is room for both, as indicated by the use of transmit FIFO level trigger or MDATACTRL[TXCOUNT].

This register acts as an alias of Target Write Data Halfword End (SWDATAHE).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 DATA1	Data Byte 1 Represents the second byte that is sent to the target (written to transmit FIFO).
7-0 DATA0	Data Byte 0 Represents the first byte that is sent to the target (written to transmit FIFO).

41.7.43 Controller Read Data Byte (MRDATAB)

Offset

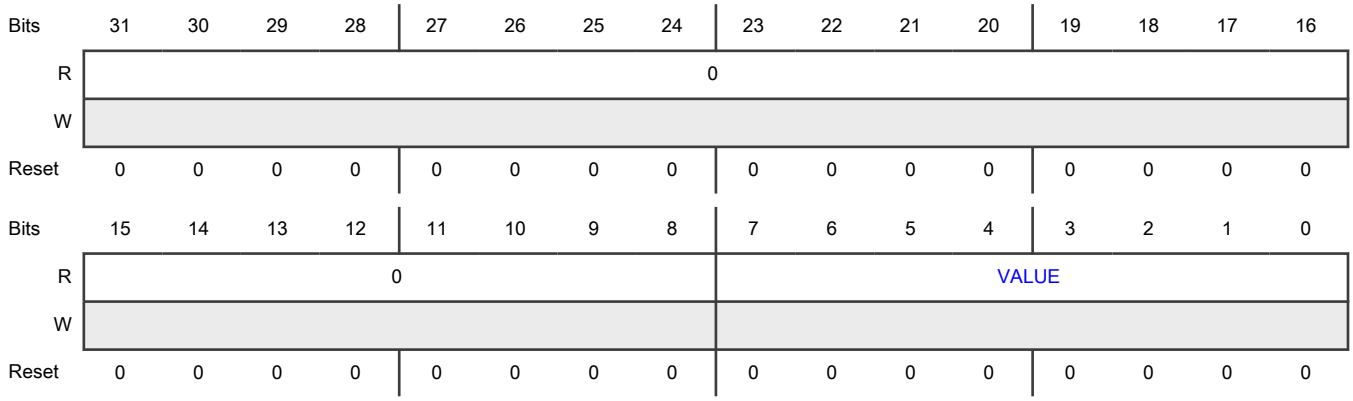
Register	Offset
MRDATAB	C0h

Function

Allows reading bytes written by the target after an SDR read, or DAA or DDR. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

This register acts as an alias of Target Read Data Byte (SRDATAB).

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 VALUE	Value Represents the byte read from the controller (and written by the target).

41.7.44 Controller Read Data Halfword (MRDATAH)

Offset

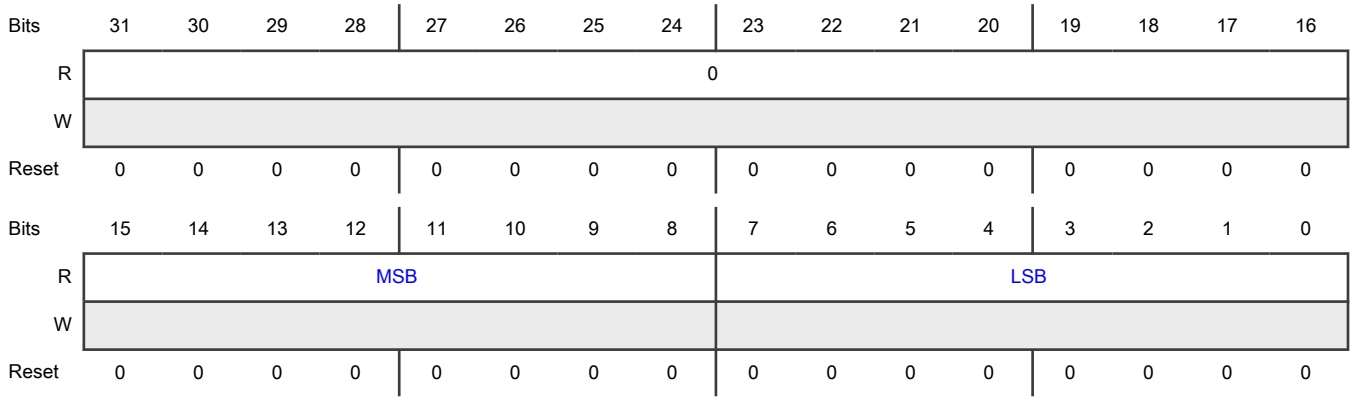
Register	Offset
MRDATAH	C8h

Function

Allows reading a halfword (byte pair) written by the target after an SDR read or DAA or DDR. This register is used only when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

A halfword must not be read unless there are at least two bytes of data waiting, as indicated by the receive FIFO level trigger or MDATACTRL[RXCOUNT].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 MSB	High Byte Represents the second byte read from the controller (and written by the target).
7-0 LSB	Low Byte Represents the first byte read from the controller (and written by the target).

41.7.45 Controller Write Byte Data 1 (to Bus) (MWDATAB1)

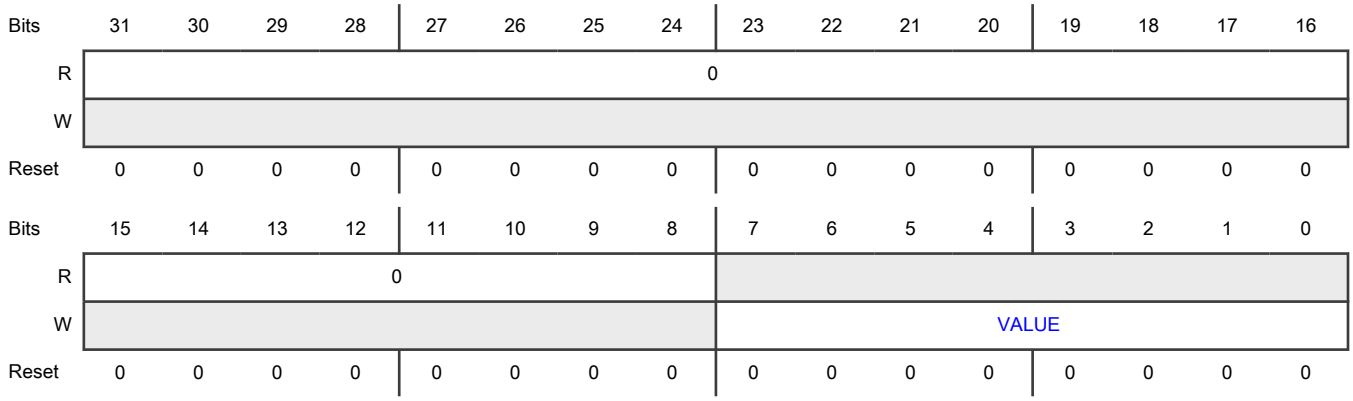
Offset

Register	Offset
MWDATAB1	CCh

Function

Allows writing bytes to send onto the bus, and is intended for DMAs that do not clear the upper bits of the word. This register does not have the END bits and is only used when using Controller Control (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
31-8 —	Reserved
7-0 VALUE	Value Represents the byte to write out. The I3C module computes the parity for I3C, or manages the ACK or NACK for I2C.

41.7.46 Controller Write Halfword Data (to Bus) (MWDATAH1)

Offset

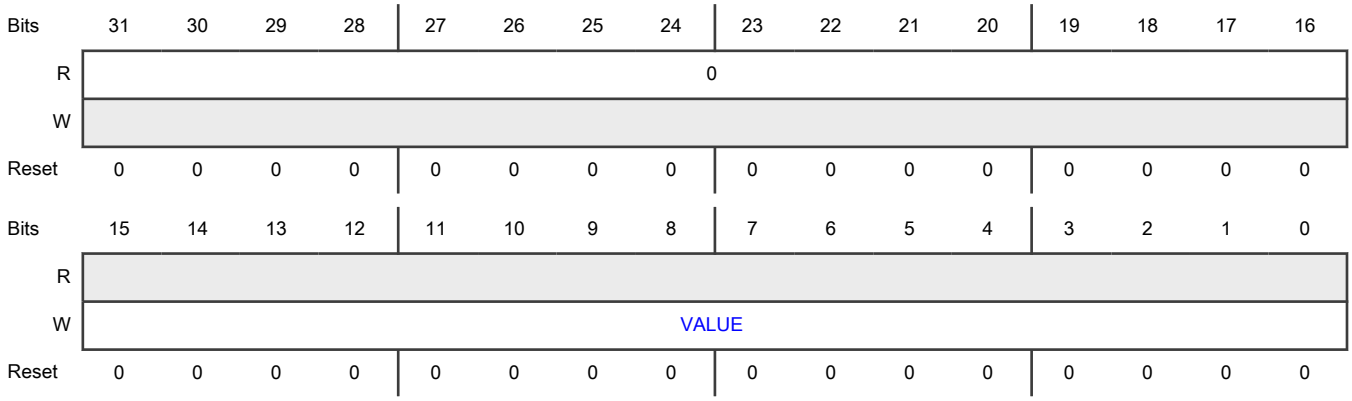
Register	Offset
MWDATAH1	CCh

Function

Allows writing a byte pair (halfword) to send onto the bus, and is intended for DMAs that do not clear the upper bits of the word. It does not have the END bits as present in Controller Write Data Halfword (MWDATAH). This register is only used when using Controller Control (MCTRL) to start the message. If MWMSG_SDR or MWMSG_DDR is used to start a message, that interface must be used exclusively.

MWDATAH1 shares the same address as MWDATAB1. MDMACTRL[DMAWIDTH] determines which one is set.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 VALUE	Value Represents the byte to write out. The I3C module computes the parity for I3C, or manages the ACK or NACK for I2C.

41.7.47 Controller Write Message Control in SDR mode (MWMSG_SDR_CONTROL)

Offset

Register	Offset
MWMSG_SDR_CONTROL	D0h

Function

Allows setting up and writing 16-bit words in Single Data Rate (SDR) mode. The MWMSG_SDR_ register is modal and has two modes—control and data:

- For the first write to set up a new message, this register functions as the MWSMSG_SDR_CONTROL register.
- For subsequent writes, this register functions as MWMSG_SDR_DATA.
- The control information is not pipelined. You must write to control information registers for a start and do not write to them until data is sent.

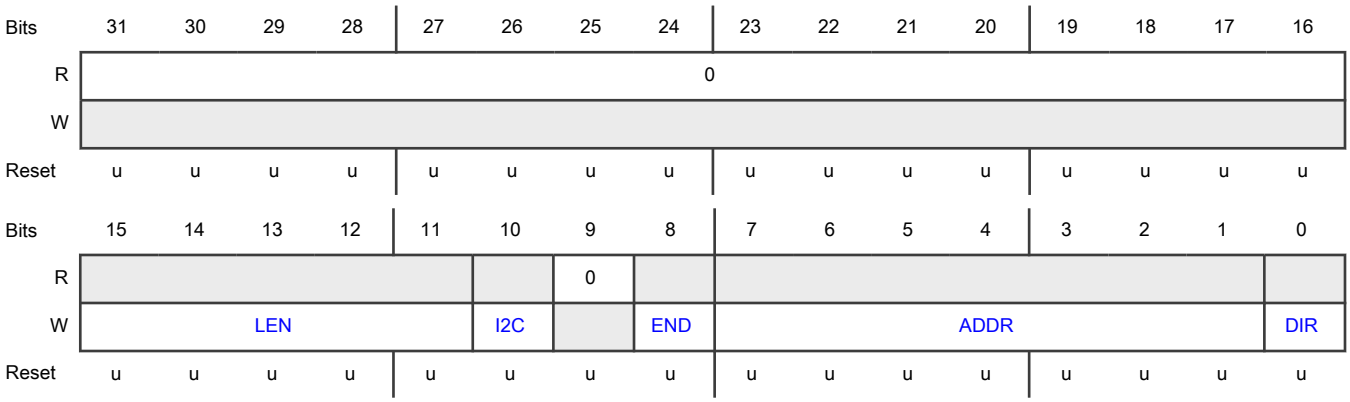
When not in the middle of a message, the MWMSG_SDR_CONTROL register is used to start a new message (or emit a STOP). After starting a message, the MWMSG_SDR_DATA register is used until the length (see the LEN field) counts down, or until data with END = 1 is used.

The MWMSG_SDR_CONTROL register is written with the 16-bit control word if currently stopped or after the end of the previous message. If MSTATUS[STATE] = 2 (MSGSDR) and MSTATUS[BETWEEN] = 0, the register (at this offset address) functions as the MWMSG_SDR_DATA register. Otherwise, this register (at this offset address) functions as the MWMSG_SDR_CONTRL register, as long as MSTATUS[STATE] is not in another mode.

The control word contains the byte length (6-bit), address, direction, and how it ends (stop, ready for next, continuation with more length). If the command is START and an event (IBI, CR, HJ) occurs, MCTRL[IBIRESP] is used to determine action, and the corresponding interrupt occurs. In that case, the message is restarted.

The MWMSG_SDR_CONTROL and MWMSG_SDR_DATA registers are only targeted for DMA operations, but the processor can also write to the MWMSG_SDR_CONTROL register (instead of using MCTRL and MxDATAB).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-11 LEN	Length Indicates the byte length of the message. If LEN = 0, then only END is used (and it does not use the address if stopped). LEN = 1 must not be used. The minimal LEN size is 2 bytes (LEN = 2).
10 I2C	I2C Specifies whether the message is I2C or I3C. 0b - I3C message 1b - I2C message
9 —	Reserved
8 END	End of SDR Message Indicates the end of SDR message. MSTATUS[COMPLETE] = 1 when done. The end can happen: • Before LEN bytes are read in I3C mode, if a target ends sooner. • Before LEN bytes are written in I2C mode, if a target NACKs. If this field is 0, the SDR message ends waiting for a new SDR message (issues a Repeated START for a new message). If this field is 1, the SDR message ends at the STOP.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not the end 1b - End
7-1 ADDR	Address Contains the address to be written.
0 DIR	Direction 0b - Write 1b - Read

41.7.48 Controller Write Message Data in SDR mode (MWMSG_SDR_DATA)

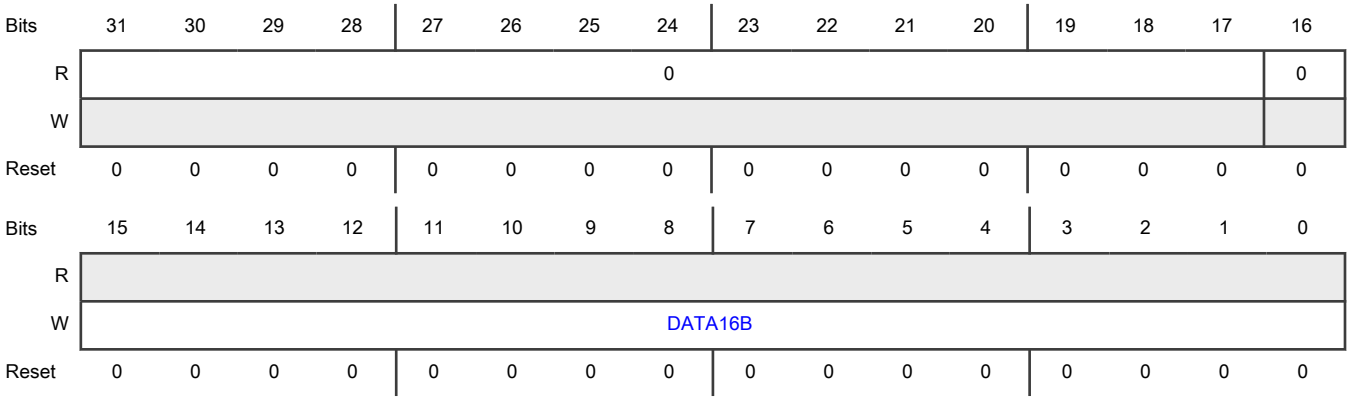
Offset

Register	Offset
MWMSG_SDR_DATA	D0h

Function

Contains the 16-bit word to be written in Single Data Rate (SDR) mode. This register functions in a way similar to MWMSG_SDR_CONTROL.

Diagram



Fields

Field	Function
31-17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 —	Reserved
15-0 DATA16B	Data

41.7.49 Controller Read Message in SDR mode (MRMSG_SDR)

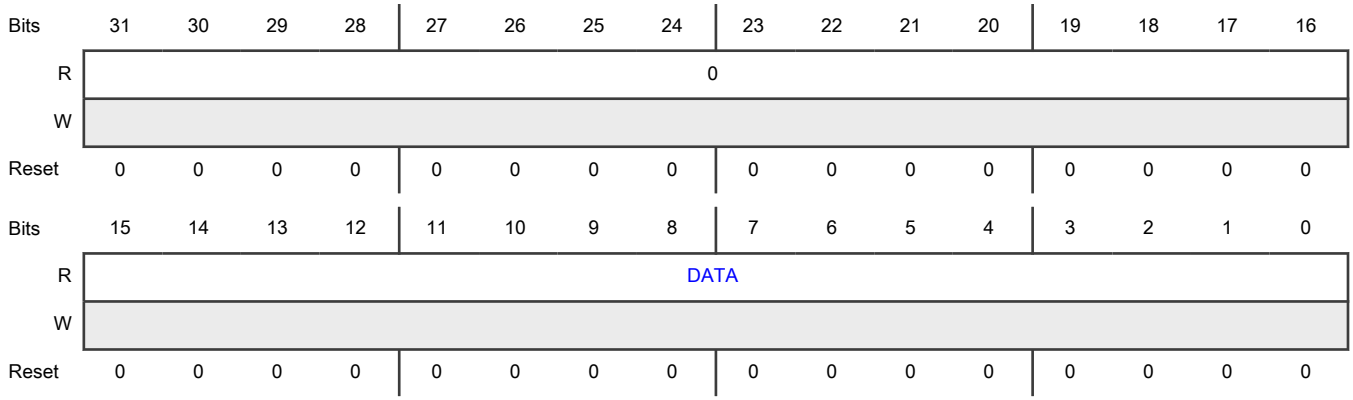
Offset

Register	Offset
MRMSG_SDR	D4h

Function

Allows reading 16-bit words from a target in 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 Contains the 16-bit word read from the target: • If the length (LEN) is an odd number, the upper byte is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• If the target ends before LEN is finished, the module treats the read as completed. • If the target is not done before LEN is finished and END is not a continuation, the read is terminated.

41.7.50 Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL)

Offset

Register	Offset
MWMSG_DDR_CONTROL	D8h

Function

Allows setting up and writing 16-bit words in DDR mode. The register has three modes:

- The first write to set up a new message functions as the MWSMSG_DDR_CONTROL register.
- The second write contains the functions as shown in the MWMSG_DDR_CONTROL2 register.
- For subsequent writes, this register functions in a way similar to Controller Write Message Data in DDR mode (MWMSG_DDR_DATA).
- The control information is not pipelined. You must write to control information registers for a start and do not write to them until data is sent.

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 after the end of the previous message. If MSTATUS[STATE] = 4 (MSGDDR) and MSTATUS[BETWEEN] = 0, then the register (at this offset address) functions as the MWMSG_DDR_DATA register. Otherwise, this register (at this offset address) functions as the MWMSG_DDR_CONTROL register, as long as MSTATUS[STATE] is not in another mode.

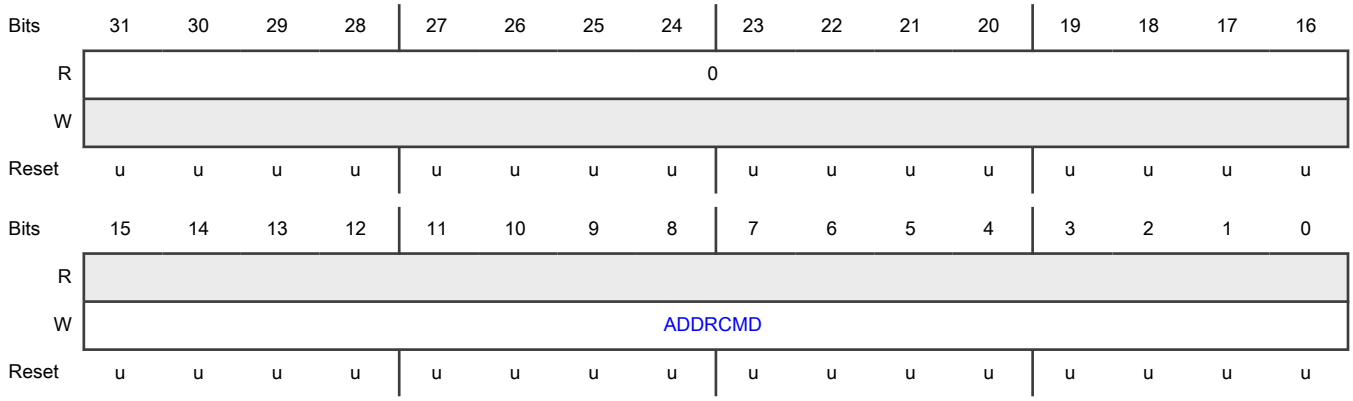
The main control word contains the 16-bit word length and how it ends (stop, ready for next, continuation with more length). Then the command word contains the command and address for read or write. If the command is START and an event (IBI, CR, HJ) occurs, MCTRL[IBIRESP] is used to determine action, and the corresponding interrupt occurs. In that case, the message is restarted.

The MWMSG_DDR_CONTROL, MWMSG_DDR_CONTROL2, and MWMSG_DDR_DATA registers are only targeted for DMA operations, but the processor can also write to the MWMSG_DDR_CONTROL register (instead of using MCTRL and MxDATAB).

NOTE

The module handles preamble, parity, and CRC.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ADDRCMD	Address Command Indicates the first data write after control write, with LEN ≠ 0. This is formatted as: • Bits 15:9: target address to read or write • Bit 8: reserved, must be 0 • Bit 7: 1 if read, 0 if write • Bits 6:0: CMD as 7-bit value, always for controller

41.7.51 Controller Write Message in DDR Mode Control 2 (MWMSG_DDR_CONTROL2)

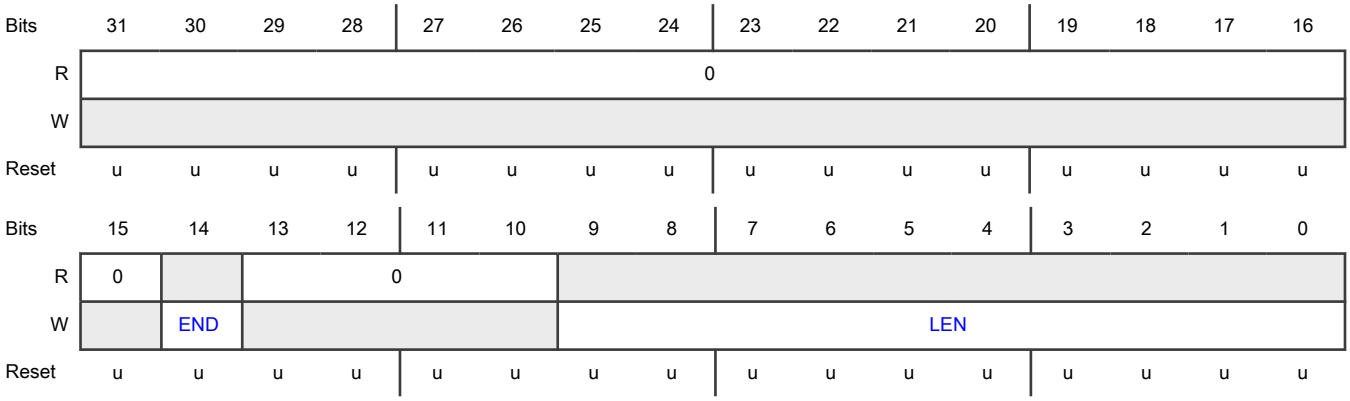
Offset

Register	Offset
MWMSG_DDR_CONTROL2	D8h

Function

Contains the second control word instructions with the length of message and end.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 —	Reserved
14 END	End of Message Indicates the end of DDR message. MSTATUS[COMPLETE] becomes 1 when done. The end can happen before LEN bytes are read if the target ends sooner. If this field is 0, DDR message ends waiting for a new DDR message (issues an HDR restart for the new message). If this field is 1, DDR message ends on HDR exit. 0b - Not the end 1b - End
13-10 —	Reserved
9-0 LEN	Length of Message Contains the length of the message (including the command) in halfwords, up to 2046 bytes. If LEN = 0, then only END is applied: • For reads, + 1 for the CRC. For example, to read 4 bytes (2 halfwords), use 1 + 2 + 1 for CMD + 2 halfwords + CRC. • For writes, LEN is the number of halves of data bytes + 1 (for command).

41.7.52 Controller Write Message Data in DDR mode (MWMSG_DDR_DATA)

Offset

Register	Offset
MWMSG_DDR_DATA	D8h

Function

Contains the 16-bit word to be written in DDR mode. This register functions in a way similar to Controller Write Message in DDR mode: First Control Word (MWMSG_DDR_CONTROL).

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																
W	DATA16B															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-0 DATA16B	Data

41.7.53 Controller Read Message in DDR mode (MRMSG_DDR)

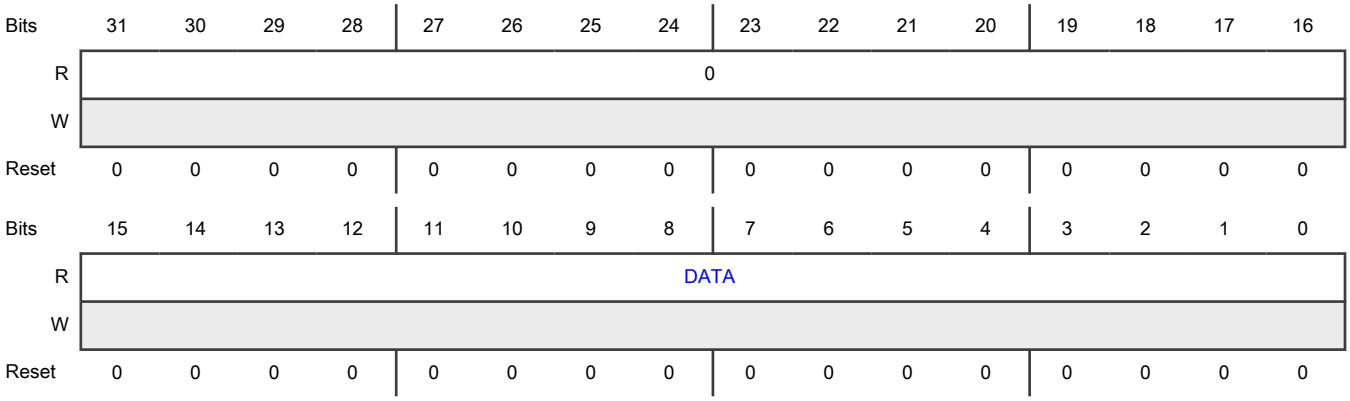
Offset

Register	Offset
MRMSG_DDR	DCh

Function

Allows reading 16-bit words from a target in DDR Message mode from an active message started with MWMSG_DDR. These words are intended to be read by DMA.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 DATA	Data Contains the 16-bit word read from a target. The first byte is the LSB, and is in DATA[7:0]. The second byte is the MSB, and is in DATA[15:8]: • If the target ends before the entire length of the message (MWMSG_DDR[LEN]) is read, the module considers the DATA read as completed. In I3C mode, the target can indicate the end of message (the last byte). Otherwise, the controller terminates the message if the message is more than the controller can accept. • If the target has not yet finished sending DATA before the entire length of the message (MWMSG_DDR[LEN]) is read and END is not a continuation, the DATA read is terminated.

41.7.54 Controller Dynamic Address (MDYNADDR)

Offset

Register	Offset
MDYNADDR	E4h

Function

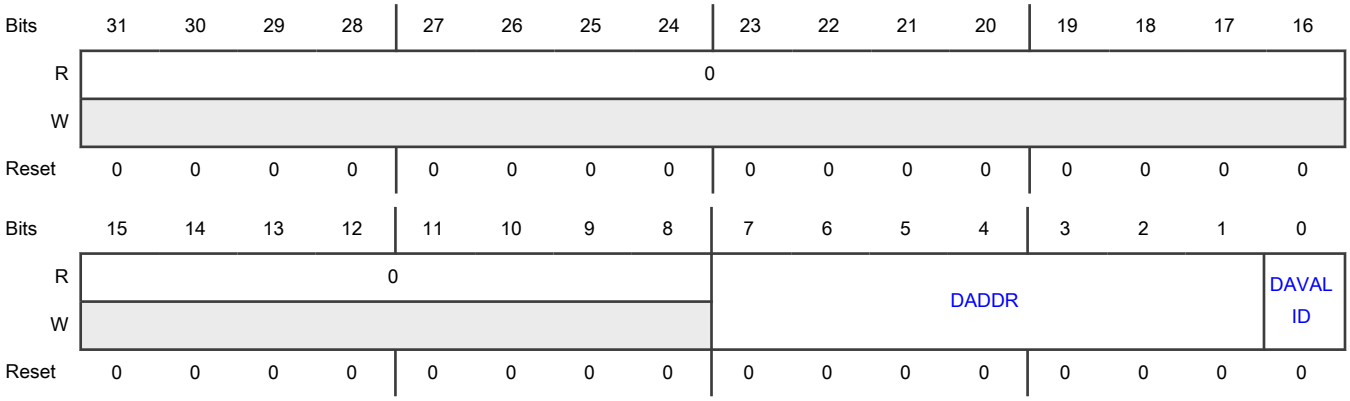
Allows the I3C module to write its own dynamic address (DA) when the module changes from Controller mode to Target mode.

If this device is the main controller (the controller during bus initialization), then the device may use this mechanism to assign itself its DA. When the device hands off control to a secondary controller, it becomes a target itself. This DA must be written before switching to Target mode and must not be changed once in Target mode (it is not clock-safe to do so). It must be written with a valid address value in DADDR if DAVALID = 1.

NOTE

The main controller also uses DEFSLVS CCC to define the target addresses, including itself. This mechanism is how secondary controllers know this address. If the controller is not the main controller, then this mechanism must not be used.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-1 DADDR	Dynamic Address Contains the assigned dynamic address when DAVALID = 1.
0 DAVALID	Dynamic Address Valid 0b - No valid DA assigned 1b - Valid DA assigned

41.7.55 Map Feature Control 0 (SMAPCTRL0)

Offset

Register	Offset
SMAPCTRL0	11Ch

Function

Provides map feature control.

The SMAPCTRL*n* registers are named SMAPCTRL0, SMAPCTRL1, and so on based on the number of mapped addresses. MAPCTRL0 represents the primary DA or SA with SMAPCTRL1 onwards being the mapped addresses.

The features of the SMAPCTRL*n* registers depend on configuration. SMAPCTRL0 acts differently, as described in this chapter.

In general, this mechanism is intended to replace the DYNADDR register for all MAP-related uses.

When using the Auto-MAP and DASA or AASA, the slot is changed from SA to DA. If the controller then issues RSTDAA, the application must rewrite the static addresses and enable them because they are marked disabled.

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				CAUSE				DA							ENA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-17 —	Reserved
16 —	Reserved
15-11 —	Reserved
10-8 CAUSE	Cause Indicates the cause of the most recent DA assignment, which lead to SSTATUS[DACHG] interrupt. This field has MAP enabled. If this field is 100b (auto MAP change happened last), the change may have changed this DA as well (for example, ENTDAA and SETAASA), but at least one MAP entry automatically changed after that. 000b - No information (this value occurs when not configured to write DA) 001b - Set using ENTDAA 010b - Set using SETDASA, SETAASA, or SETNEWDA 011b - Cleared using RSTDAA 100b - Auto MAP change happened last All other values are reserved.
7-1 DA	Dynamic Address Contains primary DA when ENA = 1. When ENA = 0, static address is used (but not described here). This field has MAP enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 ENA	Enable Primary Dynamic Address Indicates whether the MAP is enabled. 0b - Disabled 1b - Enabled

41.7.56 Extended IBI Data 1 (IBIEXT1)

Offset

Register	Offset
IBIEXT1	140h

Function

Contains extended IBI data.

The CTRL register is used to submit IBI, CR, and Hot-Join when enabled to do so. If allowed, an IBI may have additional bytes following the mandatory data byte (MDB). Extended IBI data is allowed when SCTRL[EXTDATA] = 1. If allowed, the extra bytes are indicated using these registers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EXT3								EXT2							
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	EXT1								0	MAX			0	CNT		
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0

Fields

Field	Function
31-24 EXT3	Extra Byte 3 Contains the third extra byte.
23-16 EXT2	Extra Byte 2 Contains the second extra byte.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 EXT1	Extra Byte 1 Contains the first extra byte.
7 —	Reserved
6-4 MAX	Maximum Indicates the maximum number of extra bytes allowed by configuration. This field is 0 if there are no extra bytes.
3 —	Reserved
2-0 CNT	Count Contains the number of extra bytes beyond the MDB to be used. This field is 0 if there are no extra bytes.

41.7.57 Extended IBI Data 2 (IBIEXT2)

Offset

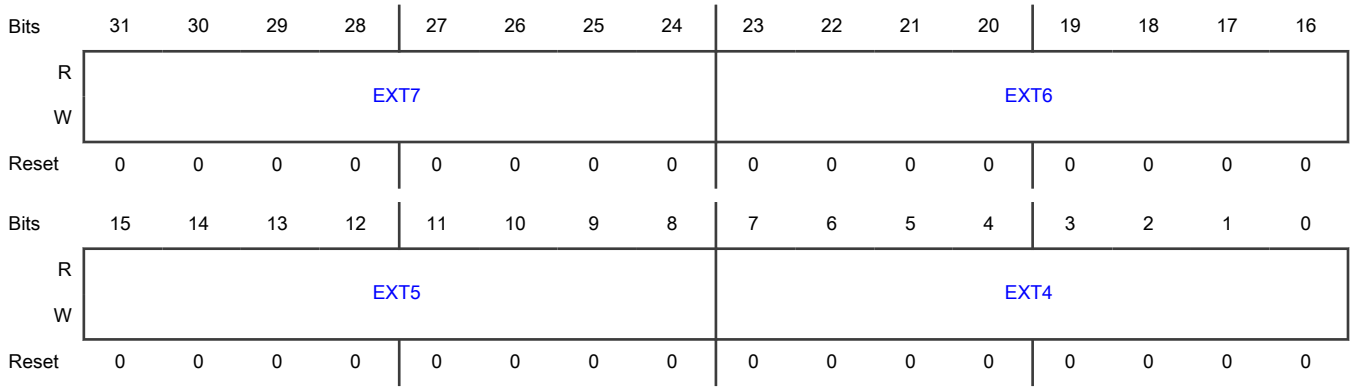
Register	Offset
IBIEXT2	144h

Function

Contains extended IBI data.

The CTRL register is used to submit IBI, CR, and Hot-Join when enabled to do so. If allowed, an IBI may have additional bytes following the mandatory data byte (MDB). Extended IBI data is allowed when SCTRL[EXTDATA] = 1. If allowed, the extra bytes are indicated using these registers.

Diagram



Fields

Field	Function
31-24 EXT7	Extra Byte 7 Contains the seventh extra byte.
23-16 EXT6	Extra Byte 6 Contains the sixth extra byte.
15-8 EXT5	Extra Byte 5 Contains the fifth extra byte.
7-0 EXT4	Extra Byte 4 Contains the forth extra byte.

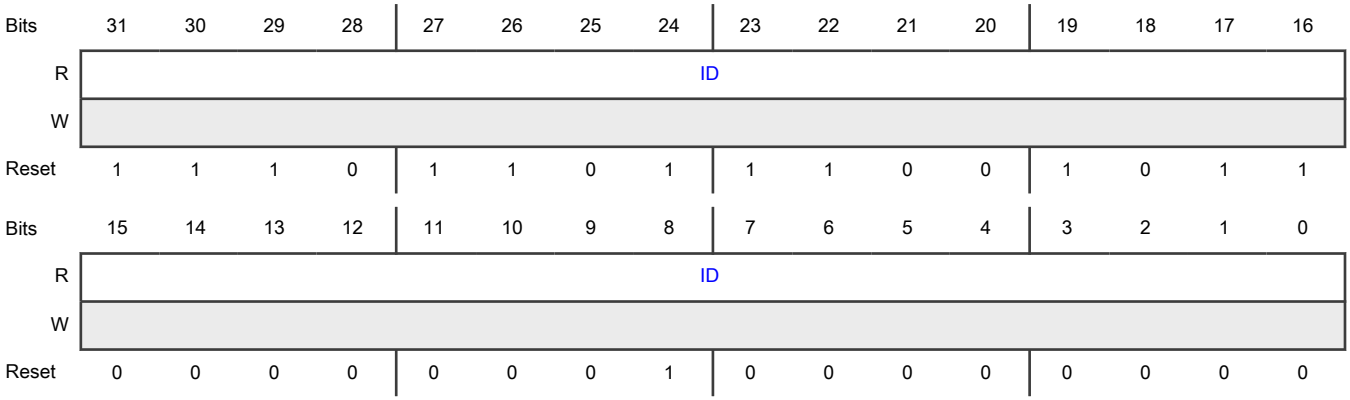
41.7.58 Target Module ID (SID)

Offset

Register	Offset
SID	FFCh

Function
Allows software to detect the module and its version information, if BlockID is enabled.

Diagram



Fields

Field	Function
31-0 ID	ID Indicates the ID. The ID meaning is specific to each use of the I3C module. ID = EDCB0100h.

Chapter 42
Analog-to-Digital Converter (ADC)

42.1 Chip-specific ADC information

Table 316. Reference links to related information

Topic	Related module	Reference
Full description	ADC	ADC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

42.1.1 Module instances

This device has two instances of 16-bit ADC: ADC0.ADC1

42.1.2 ADC input connections

Table 317. ADC analog channel input connections

ADC Channel (CMDLn[ADCH])	ADC0 A Connection	ADC1 A Connection	Description
0	ADC0_A0	ADC1_A0	VDD domain, 40 Ohm
1	ADC0_A1	ADC1_A1	VDD domain, 40 Ohm
2	ADC0_A2	ADC1_A2	VDD domain, 40 Ohm
3	OpAMP0_INT	ADC1_A3	VDD domain, 40 Ohm
4	ADC0_A4	ADC1_A4	VDD domain, 40 Ohm
5	ADC0_A5	ADC1_A5	VDD domain, 40 Ohm
6	ADC0_A6	ADC1_A6	VDD domain, 40 Ohm
7	ADC0_A7/VREFI	ADC1_A7	VDD domain, 300 Ohm
8	ADC0_A8	ADC1_A8	VDD domain, analog mux
9	ADC0_A9	ADC1_A9	VDD domain, analog mux
10	ADC0_A10	ADC1_A10	VDD domain, analog mux
11	ADC0_A11	ADC1_A11	VDD domain, analog mux
12	ADC0_A12	ADC1_A12	VDD domain, analog mux
13	ADC0_A13	ADC1_A13	VDD domain, analog mux
14	ADC0_A14	Reserved	VDD domain, analog mux

Table continues on the next page...

Table 317. ADC analog channel input connections (continued)

15	ADC0_A15	Reserved	VDD domain, analog mux
16	ADC0_A16	Reserved	VDD domain, analog mux
17	ADC0_A17	Reserved	VDD domain, analog mux
18	ADC0_A18	Reserved	VDD domain, analog mux
19	ADC0_A19	Reserved	VDD domain, analog mux
20	ADC0_A20	ADC1_A20	VDD domain, analog mux
21	ADC0_A21	ADC1_A21	VDD domain, analog mux
22	ADC0_A22	ADC1_A22	VDD domain, analog mux
23	ADC0_A23	Reserved	VDD domain, analog mux
24	VSSA	VSSA	-
25	Reserved	Reserved	-
26	Temperature+	Temperature+	-
27	PMC BG+	PMC BG+	ADC Internal PMC Bandgap, pmc_1vbuf_ana_1p8v
28	OpAMP0_OBS	ADC1_A20~A22	ADC1_A20~ADC1_A22 are on P3. P3 is supplied by VDD_P3, while other pins are supplied by VDD. Should use different physical channel.
29	VDDA/4	VDDA/4	PMC resistive dividers
30	ATX0	ATX0	Allocate ATX0 to ADC1, so ATX0/1 can be tested same time.
31	ATX1	ATX2	Changed to ATX2

42.1.3 ADC voltage reference options

The ADC voltage references can be selected by CFG[REFSEL] as follows:

- CFG[REFSEL]=00, VREFH reference pin
- CFG[REFSEL]=01, VREFI
- CFG[REFSEL]=10, VDDA_ANA supply pin

42.1.4 ADC trigger inputs

ADC trigger sources get routed through the Input Multiplexing (INPUTMUX). See the INPUTMUX chapter for available trigger sources.

42.2 Overview

16-bit ADC is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

For the chip specific modes of operation, see the power management information of the device.

42.2.1 Block diagram

The following figure is the ADC module block diagram.

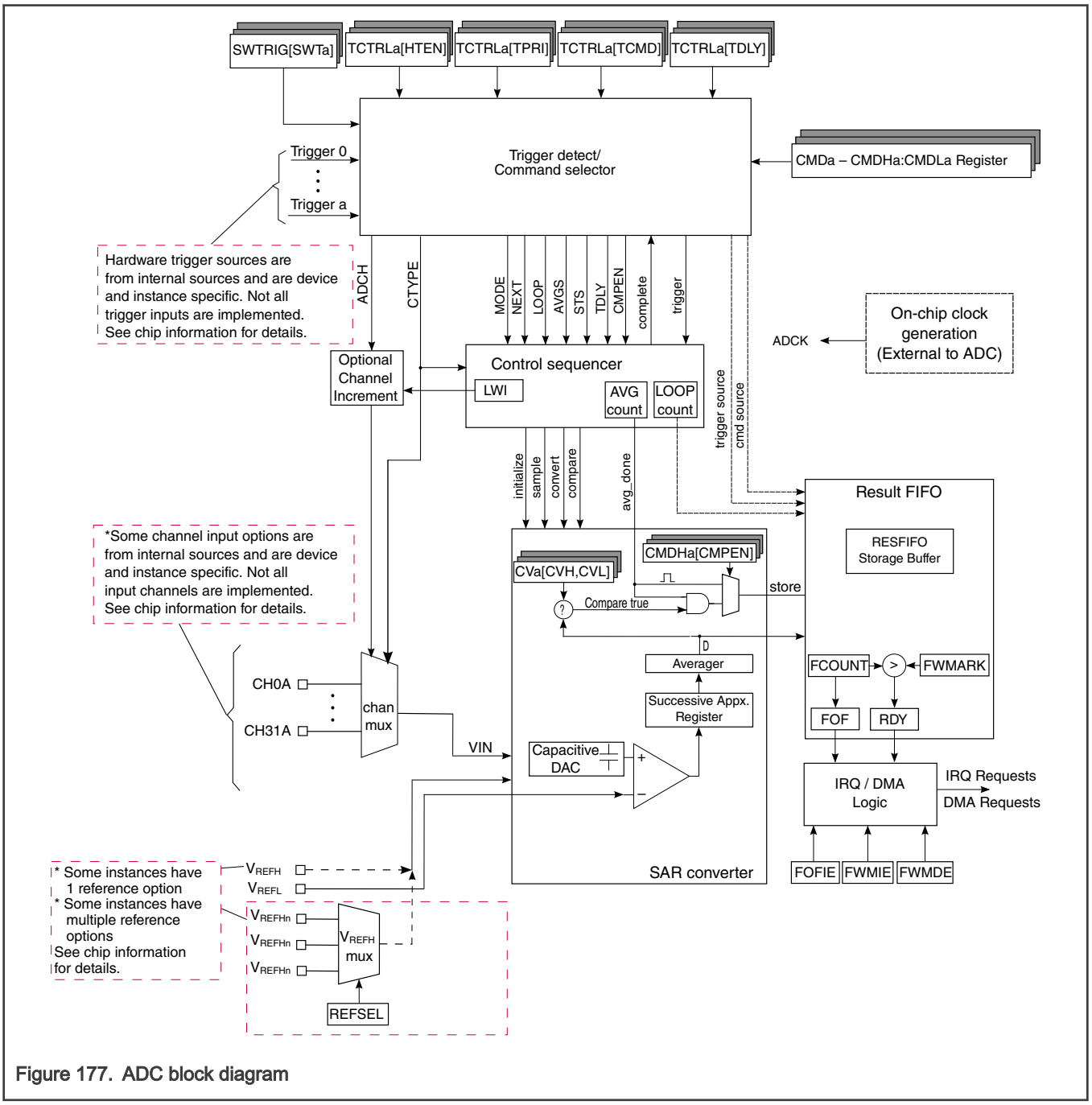


Figure 177. ADC block diagram

42.2.2 Features

Following are the features of the ADC module.

- Linear successive approximation algorithm
  - single-ended operation with 12-bit resolution or reduced bandwidth 16-bit resolution.
- Configurable analog input sample time
- Configurable speed options to accommodate operation in low power modes of SoC

- Trigger detect with up to 4 trigger sources with priority level configuration. Software or hardware trigger option for each.
- 7 command buffers allow independent options selection and channel sequence scanning.
- Automatic compare for less-than, greater-than, within range, or out-of-range with "store on true" and "repeat until true" options
- 8-entry conversion result data FIFO with configurable watermark and overflow detection
- Interrupt, DMA or polled operation
- Linearity and gain adjustment calibration logic

42.3 Functional description

The ADC module performs analog-to-digital conversions on any of the software selectable analog input channels by a successive approximation algorithm. The module initializes to its lowest power state during reset. The ADC analog circuits can optionally be pre-enabled for faster starts to conversions at the expense of higher idle currents. Conversions are initiated by selectable trigger events from software or hardware sources. The trigger detect logic includes a configurable enable and priority scheme for the available trigger sources. The module includes multiple command buffers that provide configurable flexibility for channel scanning and independent channel selections for different trigger sources. Multiple command buffers also allow variable option selection such as sample time and averaging on a per-channel basis.

The ADC module optionally averages the result of multiple conversions on a channel before storing the calculated result. The hardware average function is enabled by setting CMDH*a*[AVGS] bitfield to a non-zero value and operates in any of the conversion modes and configurations.

When the conversion and averaging loops are completed, the resulting data is placed in a FIFO data buffer along with other tag information associated with the result. A configurable watermark level supports interrupt or DMA requests when the number of stored datawords exceeds the setting. Interrupts can also be enabled to indicate when FIFO overflow errors occur.

The ADC module optionally compares 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 of the conversion modes and configurations.

The ADC module includes offset and linearity calibration logic. A request for calibration should be made any time upon reset or power up. Each SAR conversion utilizes calibration data calculated during the calibration routine.

The sequencing of a ADC command is summarized in the flow diagram shown below.

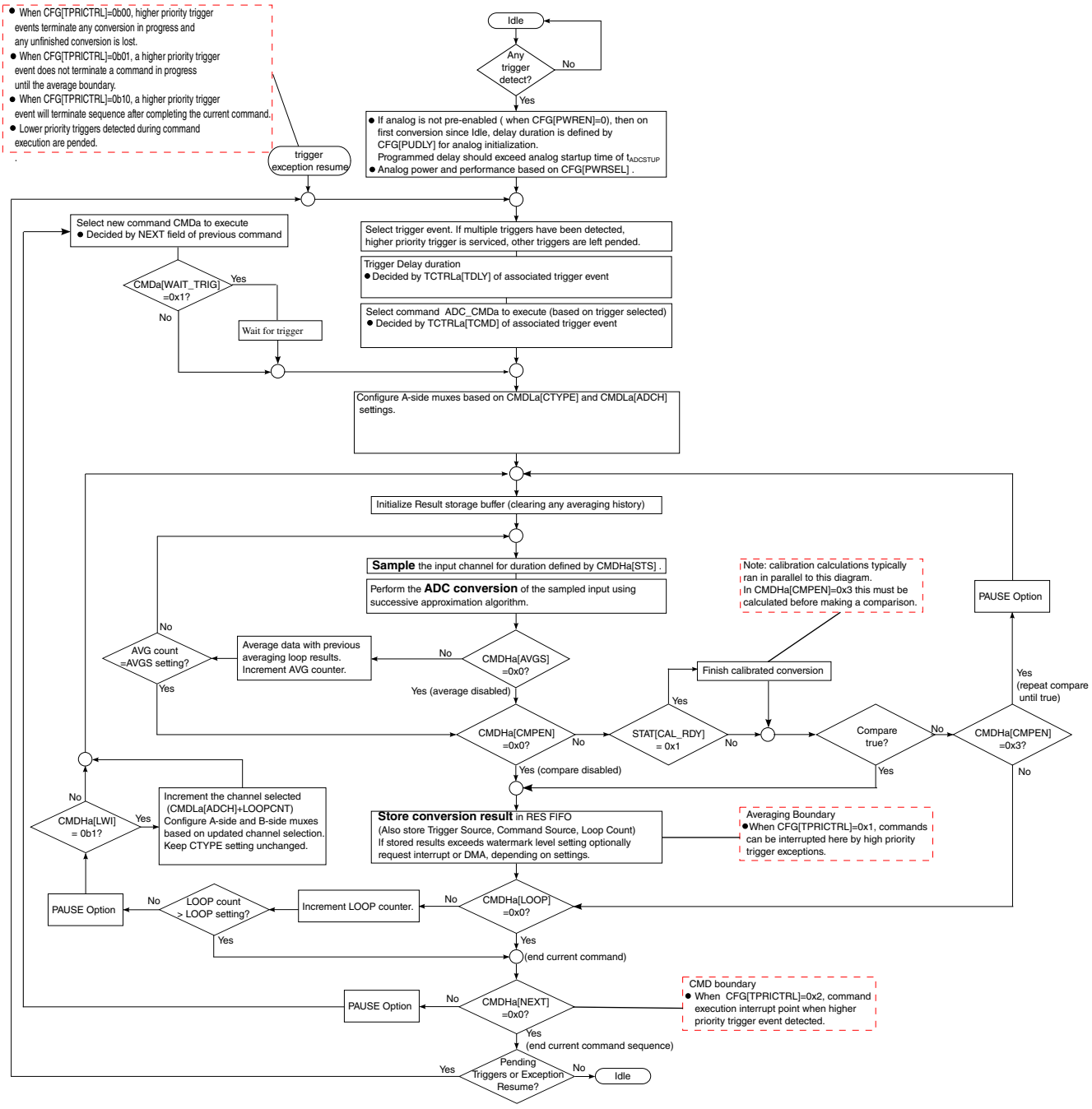


Figure 178. ADC command execution flow diagram

42.3.1 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 318. Power option settings

CFG[PWRSEL]	Description
0b (Default)	Slow speed and low power
1b	Fast speed and high power

42.3.2 ADC behavior in low power mode

The ADC module supports system level low power modes. In system low power modes where the ADC can remain functional, the CTRL[DOZEN] bit controls ADC behavior.

With the CTRL[DOZEN] bit clear, if the ADC was configured and enabled before low power mode entry, 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.

With the CTRL[DOZEN] bit set the ADC waits for the current averaging iteration/FIFO storage to complete before acknowledging low power entry request. After system entry to the low power state, the ADC is kept in inactive state until the system exits the low power state.

When entering a low power mode where the ADC cannot remain functional, the CTRL[DOZEN] bit is ignored and the ADC always waits for the current averaging iteration/FIFO storage to complete before acknowledging the low power entry request. After entry to the low power state, the ADC is kept in an inactive state until the system exits the low power state. The CFG[PWREN] bit setting is ignored and the ADC analog is forced to its lowest power state. Upon wakeup from the low power state, if the analog was pre-enabled before stop mode entry (i.e., the CFG[PWREN] bit is set) and the control registers have retained state in the low power mode, the analog is automatically re-enabled and the power up delay timer begins. Any triggered conversion is stalled until the power up delay associated with the CFG[PUDLY] control field has finished.

42.3.3 Voltage reference

The voltage reference high (V_{REFH}) used by the ADC is supplied from either an on-chip voltage reference source or from an off-chip source supplied through external pins. V_{REFL} is always from an external pin and must be at the same voltage potential as V_{SSA}. See the chip configuration information on the voltage reference options specific to this packaged device.

This block supports a programmable selection of the Voltage Reference High used for ADC conversions (via the CFG[REFSEL] field). See the chip configuration information on the voltage reference options specific to this packaged device.

42.3.4 Trigger detect and command execution

See Figure 178 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 0b1 to the corresponding SWTRIG[SWT*n*] bitfield. Alternatively, hardware triggers can be generated from asynchronous input sources at the periphery of the module. The number and sources of hardware triggers implemented is device specific. Refer to chip configuration information for description of available hardware trigger sources for this device.

Each hardware trigger source is enabled by setting the associated enable bit (TCTRL*a*[HTEN]). Each trigger source is assigned a priority via the associated priority control field (TCTRL*a*[TPRI]). Each of the trigger sources is associated with a command buffer via the associated command select field (TCTRL*a*[TCMD]).

When a hardware trigger input is enabled, hardware trigger events are detected on the rising-edge of the associated hardware trigger source.

When the ADC is inactive, achieving the minimum startup time to begin an initial conversion after a software or hardware generated trigger requires that the ADC be pre-enabled (by setting CFG[PWREN] during ADC initialization and allowing enough time for ADC power-up delay to timeout before allowing any triggers to be generated). In the case of a pre-enabled ADC, it takes 9 ADCK cycles for the trigger to be recognized and begin sampling on the associated analog channel.

In the case that the ADC is not pre-enabled, an initial conversion is delayed further as determined by the power-up delay (CFG[PUDLY]). Conversion startup can also be delayed on a trigger source basis (TCTRLn[TDLY]).

Each trigger source has an associated priority field TCTRLa[TPRI] which allows for arbitration between trigger sources. Arbitration is in control of two things: selecting which trigger sequence to execute next, and selecting 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. If CFG[HPT_EXDI] is set to 1'b1 then trigger exceptions are disabled and any higher priority triggers are left pending until the current sequence completes. Note that new triggers are accepted based on priority.
2. If CFG[HPT_EXDI] is set to 1'b0 (default), then exceptions are enabled and the higher priority sequence begins executing at a user specified breakpoint.

Breakpoint locations are determined by the register CFG[TPRICTRL]. CFG[TPRICTRL] has an affect on latency for accepting a trigger exception.

1. When CFG[TPRICTRL]=0x0, a higher priority trigger causes an immediate command abort and the new command specified by the trigger is immediately started. It takes 9 ADCK cycles for the higher priority trigger to be recognized, terminate the conversion in progress, and complete a switch to the new associated ADC command. 9 ADCK cycles after the trigger is driven to the ADC, the ADC begins sampling on the associated new analog channel.
2. When CFG[TPRICTRL]=0x1, the current conversion is allowed to complete (including averaging) before the higher priority exception is initiated. 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 4 loop iterations, there is no requirement to complete all 4 iterations before the interrupt occurs.
3. When CFG[TPRICTRL]=0x2, a higher priority trigger begins once the current command is completed. If a command consists of 5 loop iterations each containing 8 averages, then all 5x8 conversions must be completed before accepting the trigger exception.

CFG[TCMDRES] and CFG[TRES] determine what the ADC does after accepting a trigger exception. The module can be programmed to resume commands after returning from a trigger exception.

1. If CFG[TRES] = 0x0 then commands are not automatically resumed after being stopped by an exception. However, an interrupt is set to indicate this case has occurred. The flag TSTAT[TEXC_NUM] can be used to resolve which trigger was stopped by the exception.
2. If CFG[TRES] = 0x1 the ADC automatically resumes commands after they were stopped by an exception.

By utilizing CFG[TRES] in conjunction with CFG[TCMDRES], the module can be programmed to resume commands at one of two possible locations.

1. If CFG[TCMDRES] = 0x0 then the trigger which was stopped by an exception is resumed from the beginning of its associated command sequence. Note, triggers which are waiting to be resumed take the same priority programmed to TCTRLa[TPRI].
2. If CFG[TCMDRES] = 0x1 then the trigger is resumed from the command that it was executing before being interrupted by an exception.

If a lower priority trigger occurs (i.e., a trigger event occurs that is configured for a lower priority than the trigger source associated with the currently executing command), the trigger detect is left pending until completion of the current command sequence. 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 CMDHa[AVGS] is non-zero), the result is placed in a RESFIFO buffer. When an ADC command selects looping (when CMDHa[LOOP] is non-zero) a command stores multiple conversion results to the FIFO during execution of that command.

At the end of command execution, the CMDHa[NEXT] field of the command selects the next command to be executed. Multiple commands can be executed sequentially by configuration of each commands CMDHa[NEXT] field. Setting the next command to 0x0 causes conversions to terminate at the completion of the command. Unending circular command execution is allowed by setting the CMDHa[NEXT] field in the last command in a sequence to the first command in the sequence.

By default, command sequences executes automatically in the order that CMDHa[NEXT] fields are programmed. However, by utilizing the CMDHa[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 receiving TRIGGER 2 one time unconditionally runs this sequence to completion. If CMDH2[WAIT_TRIG] is set to 0x1, however, then the sequence pauses after CMD1 until TRIGGER2 is received again. Therefore, sequences can be stalled until receiving a trigger assertion.

Disabling the ADC by writing 0b0 to the CTRL[ADCEN] bitfield terminates any active ADC command processing. Writing 0b0 to the CTRL[ADCEN] bitfield causes the current command (or command sequence) to terminate, clears any pending triggers and sends the ADC module to an IDLE state.

42.3.5 Pause option

When the maximum conversion rate is not required by an application 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 ADACK source. Both of these 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 results in a limit on the maximum conversion rate when using periodic triggering.

Another means of reducing conversion rates is by inserting a pause of a programmable duration between LOOP iterations, between commands in a sequence, and between conversions when command is executing in the "Compare Until True" configuration. When PAUSE[PAUSEEN] is set, the PAUSE[PAUSEDLY] field controls the duration of pausing during command execution sequencing. The pause delay is a count of (PAUSE[PAUSEDLY]*4) ADCK cycles. Note, the PAUSE register should not be changed while the CTRL[ADCEN] bit is set. Writes to the PAUSE register while CTRL[ADCEN] is set can lead to metastable operation.

See Figure 178 for the places during command execution sequencing where the pause is optionally inserted.

42.3.6 Resync functionality

Any trigger source (SW or HW) can be configured to act as a resync trigger. Trigger based resync functionality is used to interrupt a running trigger (resync target) and clear the FIFO it's writing to. This can either be used to abort a running sequence, or restart a running sequence depending on the configuration of CFG[TRES]. If CFG[TRES] = 0b1 then the target sequence is aborted, the FIFO cleared, and the sequence restarted after the resync occurs. If CFG[TRES] = 0b0 then the target sequence is aborted and the FIFO is cleared after the RESYNC occurs.

A resync trigger needs to have a specific target. The resync only occurs if the resync target is running at the time of the trigger. For the following description, let n be the resync trigger number, let m be the resync target number. According to these variables, trigger n should resync trigger m. To enable a trigger source to act as a resync trigger, the following conditions must be satisfied:

1. The resync trigger TCTRLn[RSYNC] must be set to 0b1.
2. The resync trigger must have higher priority than the resync target (TCTRLn[TPRI] must be less than TCTRLm[TPRI]).
3. The resync target is specified using TCTRLn[TCMD]. In this case the resync target, m, must be equal to TCTRLn[TCMD].
4. The resync target, m, must be executing commands when the resync trigger, n, is asserted.
5. Trigger m must have at least one conversion left to begin when trigger n is received.

If a trigger source n has TCTRLn[RSYNC] set to 0b1, but some of the above conditions are not met then the trigger source n is ignored.

The following figure illustrates a resync trigger sequence executing. Note, in this example, trigger source 1 is configured to resync trigger source 0.

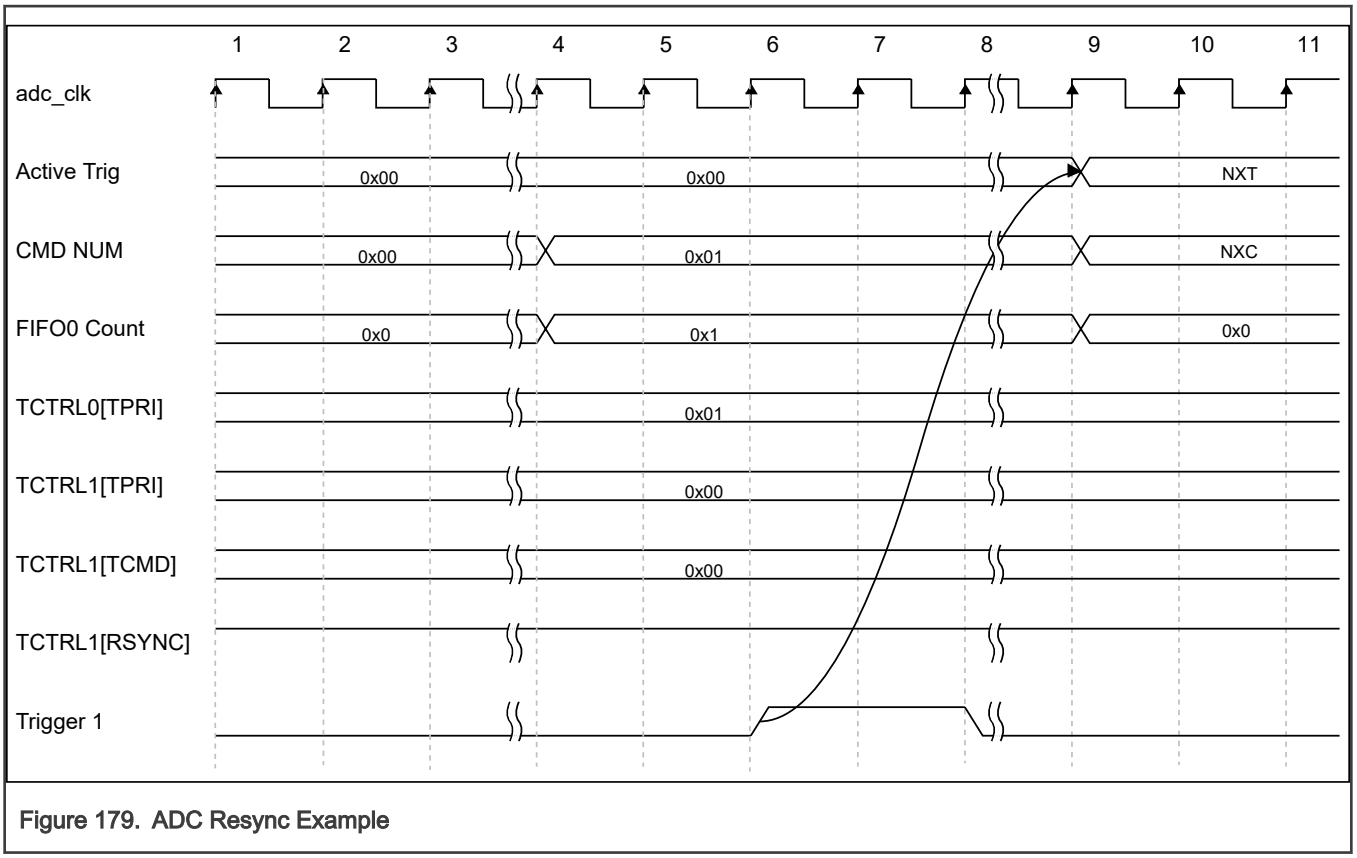


Figure 179. ADC Resync Example

In this figure, trigger source 0 was executing a sequence of commands when trigger source 1 was asserted. Notice that trigger source 0 is stopped when trigger source 1 is asserted (after some synchronization delay). In addition, the FIFO being written to by trigger source 0 is cleared (FIFO0 in this example). After the trigger 0 sequence is stopped, the ADC runs the next trigger pending with the highest priority. This is marked as NXT, for next trigger. If resume functionality is enabled, and trigger source 0 had the highest priority pending, then NXT = 0x00.

42.3.7 Temperature sensor

The ADC module has a dedicated input channel for an on-chip temperature sensor. Refer to 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 some configuration requirements. This sequence for converting the temperature sensor channel is outlined below:

1. Refer to chip specific channel definition to determine which channel corresponds to the temperature sensor.
2. Configure a command buffer register to convert the temperature sensor channel. CMDLa[ADCH] = Temperature Sensor Channel.
3. The command buffer must also be programmed with the following parameters:
   - 16b mode (CMDLa[MODE] = 0x1)
   - Max averaging (CMDHa[AVGS] = 0xA)
   - Max sample time (CMDHa[STS] = 0x7)
   - LOOP set to 1 (CMDHa[LOOP] = 0x1)
   - Loop with increment disabled (CMDHa[LWI] = 0x0)
   - Compare function disabled (CMDHa[CMPEN] = 0x0)
4. Configure a trigger control register TCTRLa with the following parameters:

- TCTRLa[TCMD] = command buffer used in steps 2 and 3.

5. Trigger a conversion of the temperature sensor channel. Software triggers the conversion by writing "1" to the associated bit in SWTRIG register (i.e., the trigger configured in step 4).

After completion of the conversion of the temperature sensor channel, two results are stored in the FIFO. 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.

$$\text{Temp} = A\left[\frac{\alpha(\text{Vbe8} - \text{Vbe1})}{\text{Vbe8} + (\alpha(\text{Vbe8} - \text{Vbe1}))}\right] - B$$

Equation 22. 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.

42.3.8 Result FIFO operation

The ADC includes 1 8-entry FIFO 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 along with the data. FCTRLa[FCOUNT] indicates how many valid datawords are stored in each RESFIFO.

A programmable watermark threshold supports configurable notification of data availability. When FCTRLa[FCOUNT] is greater than FCTRLa[FWMARK], the associated RDY flag is asserted. When IE[FWMIEx] is set, a watermark interrupt request is issued. When DE[FWMDEx] is set, a DMA request is issued. Reading RESFIFO provides the oldest unread dataword entry in the FIFO and decrements FCTRLa[FCOUNT]. When FCTRLa[FCOUNT] falls equal to or below FCTRLa[FWMARK], the RDY flag is cleared.

Each FIFO can be emptied by successive reads of RESFIFOa. When the RESFIFOa[VALID] bit is 1 the associated FIFO entry is valid. Reading RESFIFOa when the FIFO is empty (when RESFIFOa[VALID] is clear and FCTRLa[FCOUNT]=0x0) provides an undefined dataword. All FIFOs are reset by writing 0b1 to the CTRL[RSTFIFOx] bit.

If the ADC attempts to store a dataword to the FIFO when the FIFO is full the FIFO overflow flag (FCTRLa[FOF]) is set. When IE[FOFIEx] is set, a overflow interrupt request is issued. The FOF flag is cleared by writing 1 to STAT[FOFx]. On overflow events no new data is stored and the data associated with the store that triggered the overflow is lost.

42.3.9 Compare function

After the input is sampled and converted and any averaging iterations are performed, the CMDH*a*[CMPEN] field guides operation of the automatic compare function to optionally only store when the compare operation is true. There are multiple options on command sequencing related to the compare function as 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. This means that 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 5 ADC clock cycles.

NOTE

Not all Command Buffers have an associated Compare Value register. The compare function is only available on Command Buffers that have a corresponding Compare Value register.

Table 319. Compare modes

CMDH*a*[CMPEN]	Compare Function	Description
0b00	Compare disabled	Do not perform compare operation. Always store the conversion result to the FIFO.
0b01	Reserved	
0b10	Store on true	Perform compare operation. Store conversion result to FIFO at end of averaging only if compare is true. If compare is false do not store the result to the FIFO. In either the true or false condition, the LOOP setting is considered and increments the LOOP counter before deciding whether the current command has completed or additional LOOP iterations are required.
0b11	Repeat compare until true	Perform compare operation. Store conversion result to FIFO at end of averaging only if compare is true. Once the true condition is found the LOOP setting is considered and increments the LOOP counter before deciding whether the current command has completed or additional LOOP iterations are required. If the compare is false do not store the result to the FIFO. The conversion is repeated without consideration of LOOP setting and does not increment the LOOP counter.

Depending on CV*a*[CVH] and CV*a*[CVL] values programmed, the compare operation checks whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. The compare values are used as described in the following table.

Table 320. Compare operations

CV*a*[CVL] vs. CV*a*[CVH]	Operation	Description
set CVa[CVL] < CVa[CVH]	Outside range (General form)	Compare true if the result is less than CVa[CVL] value **OR** greater than CVa[CVH] value.
set CVa[CVH] to max value set CVa[CVL] to compare point	Less than	Compare true if the result is less than CVa[CVL] value.
set CVa[CVL] to min value set CVa[CVH] to compare point	Greater than	Compare true if the result is greater than CVa[CVH] value.
set CVa[CVL] > CVa[CVH]	Inside range	Compare true if the result is less than CVa[CVL] value **AND** greater than CVa[CVH] value.

NOTE

In low power modes where the ADC continues to operate, the compare function can monitor the voltage and only wake the device when the compare condition is met.

42.3.10 Cycles per conversion

The number of ADCK cycles needed to complete a conversion varies based on the selected resolution (CMDLa[MODE]), the sample time configured, and several other configuration options discussed below.

To calculate the cycle count, first, determine the Base Cycles Count from Table 321. Next, add the Sample Time Adder based on CMDHa[STS] setting as summarized in Table 322. If averaging is enabled (CMDHa[AVGS] does not equal to 0), then the total cycles (Base Cycles Count + Sample Time Adder) should be multiplied by the number of averages configured as summarized in Table 323.

$$CycleCount\ /\ Conversion = [(BaseCycleCount + SampleTimeAdder) * AverageMultiplier]$$

Equation 23. Cycle Count/Conversion

Note that there is latency associated with trigger detection and starting an initial conversion. There is additional latency associated with offset and gain error adjustment of a raw conversion result and storage of a final result to the FIFO. Due to pipe-lining of conversions when looping or command chaining is configured, the next conversion is immediately started while a raw conversion result is being adjusted and stored and thus the conversion rate is maximized.

The base cycles per conversion is variable depending on CMDLa[MODE], CFG2[HS], CFG2[HSEXTRA], and CFG2[TUNE] settings as summarized in the following table.

Table 321. Base cycles per conversion

CFG2[HS] (default 0)	CFG2[HSEXTRA] (default 0)	CFG2[TUNE] (default 01)	Base ADCK cycles/conversion[1]		Description
			16-bit CMDLa[MODE] = 1	12-bit CMDLa[MODE] = 0	
0	X	00	24	20	
0	X	01	23	19	Reset default
0	X	10	22	18[2]	
1	0	00	21	17	
1	0	01	20	16	
1	0	10	19	15[2]	
1	1	00	22	18	
1	1	01	21	17	
1	1	10	20	16[2]	

1. This cycle count includes min sample time of 3.5 ADCK cycles.
2. Setting CFG2[TUNE] to 10 in 12-bit mode can result in missing codes (i.e., some result codes may not be produced).

In addition to the base cycles per conversion, the configured sample time needs to be considered. The sample time adder is variable depending on CMDHa[STS] setting and is summarized in the following table:

Table 322. Sample time cycle adder

CMDHa[STS] (default 000)	Add ADCK cycles/conversion
000	0
001	2

Table continues on the next page...

Table 322. Sample time cycle adder (continued)

CMDHa[STS] (default 000)	Add ADCK cycles/conversion
010	4
011	8
100	16
101	32
110	64
111	128

Table 323. Averaging multiplier

CMDHa[AVGS] (default 0000)	Averaging multiplier
0000	1
0001	2
0010	4
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256
1001	512
1010	1024

42.3.11 Clocking

ADC uses the ADCK clock input provided by an on-chip clock select block. It 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 CFG[PWRSEL]. See the device data sheet for supported frequency ranges.

ADC continues operating in low-power mode as long as CTRL[DOZEN] = 0 and the on-chip clock select block supplies an ADCK clock source.

NOTE

When in low-power mode with CTRL[DOZEN] = 0, the bus clock can be shut off, and asynchronous interrupts and DMA requests can be configured. ADC continues processing commands and writing data to the internal FIFO.

ADC has four sources for asynchronous interrupts during low-power mode: watermark, FIFO overflow, TCOMP, and TEXC. To enable them, configure IE[FWMIEx], IE[FOFIEx], IE[TCOMP_IE], and IE[TEXC_IE] before entering low-power mode.

When CTRL[DOZEN] = 1 in low-power mode, ADC waits for the current averaging iteration or FIFO storage to complete before acknowledging the low-power mode entry. Any pending triggers are dropped when a low-power mode request is made in this mode. ADC is forced into its lowest-power setting after acknowledging the low-power mode request.

42.3.12 Resets

Table 324. 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]). After a FIFO is reset, that FIFO is empty.

42.3.13 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 325. 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

42.4 External signals

The ADC module supports analog channel inputs. ADC requires supply and ground connections.

Table 326. ADC signal descriptions

Signal	Description	I/O
V_{DDA}	Analog Power Supply	I
V_{SSA}	Analog Ground	I
CH0A - CHnA[1]	A-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.

42.4.1 Analog channel inputs (CHnA)

CMDLn[ADCH] and CMDLn[CTYPE] control selection of individual input channels. Each ADC command independently makes a channel and conversion type selection.

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.

42.5 Initialization

42.5.1 Calibration

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. Prior to 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 the ADC is actively converting, that sequence completes before starting calibration.

Averaging multiple conversions can achieve improved accuracy during the calibration routines. It is recommended to set CTRL[CAL_AVGS] for minimum of 256 averaging during calibration steps. 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.

42.5.1.1 Offset calibration

The Offset Trim Register (OFSTRIM) register is used to trim for ADC comparator offset voltage. The ADC supports an offset calibration function in which the OFSTRIM register is automatically updated. Below are the steps for executing the function.

1. Configure for the desired averaging via CTRL[CAL_AVGS]. The minimum recommended setting is for 256 averaging (CTRL[CAL_AVGS] set to 0x8 or greater).
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.

42.5.1.2 ADC Calibration

ADC includes hardware calibration logic in which the converter is calibrated for gain error and linearity error correction of the raw conversion result. The GCR0[GCALR] and CAL_GAR registers control calibration for the A-side converter. The ADC supports a calibration function in which the CAL_GAR registers are automatically updated. The calibration function also updates GCC0[GAIN_CAL] (associated with A-side calibration) . For A-side calibration, a software calculation is required to derive GCR0[GCALR] from GCC0[GAIN_CAL]. Below are the steps for completing calibration setup.

1. Execute Offset calibration steps. The Offset Trim Register (OFSTRIM) is used during calibration to trim for comparator offset voltage.
2. Configure for the desired averaging via CTRL[CAL_AVGS]. The minimum recommended setting is for 256 averaging (CTRL[CAL_AVGS] set to 0x8 or greater).
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. 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 needed 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 signed value. This results in a floating point value between 0 and 2.
7. Convert the floating point value to its integer component (0 or 1) and its fractional component rounded to 16-bits. The integer value is stored to GCR0[GCALR[16]] and the fractional component is stored to GCR0[GCALR[15:0]]. Write this value to the GCR0[GCALR] register.
8. Once GCR0[GCALR] contains the result from the gain_adjustment calculation, set the GCR0[RDY] flag to indicate it is valid.

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, the 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.

42.5.1.3 Calibration General A-Side Register valid ranges

The general calibration value registers CAL_GARa have 34 registers in each array. The data in each register is a signed 16-bit value but has a variable range of possible values. The following table defines the valid range each register supports.

Table 327. Calibration General Registers Valid Values

Element CAL_GxR[N]	Valid Range
N = 0x00, 0x20, 0x21	–1024 to +1023
N = 0x01	–2048 to +2047
N = 0x02–0x03	–4096 to +4095
N = 0x04–0x07	–8192 to +8191
N = 0x08–0x0F	–16384 to +16383
N = 0x10–0x1F	–32768 to +32767

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 and 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.

42.5.1.4 High speed calibration

The High Speed Trim Register (HSTRIM) register is used to trim for high speed mode conversions (when CFG2[HS] is set). The ADC supports a high speed calibration function in which the HSTRIM register is automatically updated. Below are the steps for executing the function.

1. Configure for the desired averaging via CTRL[CAL_AVGS]. The minimum recommended setting is for 256 averaging (CTRL[CAL_AVGS] set to 0x8 or greater).
2. Initiate High speed calibration by setting CTRL[CALHS].
3. Poll the STAT[CAL_RDY] flag. When STAT[CAL_RDY] is asserted, the high speed calibration function has completed and the HSTRIM register has updated.

42.6 ADC register descriptions

This section describes the ADC registers.

NOTE

Writing to a read-only (RO) register can cause bus errors.

42.6.1 ADC memory map

HSADC0 base address: 400A_F000h

HSADC1 base address: 400B_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0200_1409h
4h	Parameter Register (PARAM)	32	R	0707_0804h
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
48h	High Speed Trim Register (HSTRIM)	32	RW	0000_0000h
A0h - ACh	Trigger Control Register (TCTRL0 - TCTRL3)	32	RW	0000_0000h
E0h	FIFO Control Register (FCTRL0)	32	RW	0000_0000h
F0h	Gain Calibration Control (GCC0)	32	R	0000_0000h
F8h	Gain Calculation Result (GCR0)	32	RW	0001_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

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
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
200h - 218h	Compare Value Register (CV1 - CV7)	32	RW	0000_0000h
300h	Data Result FIFO Register (RESFIFO0)	32	R	0000_0000h
400h - 484h	Calibration General A-Side Registers (CAL_GAR0 - CAL_GAR33)	32	RW	0000_0000h
FF8h	Configuration 2 Register (CFG2)	32	RW	0000_1000h

42.6.2 Version ID Register (VERID)

Offset

Register	Offset
VERID	0h

Function

The Version ID register indicates the version integrated for this instance on the device and also indicates inclusion/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	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	0	1	0	1	0	0	0	0	0	0	1	0	0	1

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 —	Reserved This read-only field is reserved and always has the value 0.
14-12 NUM_FIFO	Number of FIFOs This read-only field indicates the number of result FIFOs implemented in the design. 000b - N/A 001b - This design supports one result FIFO. 010b - This design supports two result FIFOs. 011b - This design supports three result FIFOs. 100b - This design supports four result FIFOs.
11 NUM_SEC	Number of Single Ended Outputs Supported This read-only field indicates the number of single ended channels which can be processed simultaneously. 0b - This design supports one single ended conversion at a time. 1b - This design supports two simultaneous single ended conversions.
10 CALOFSI	Calibration Function Implemented This read-only field indicates if the ADC contains hardware calibration functions. When supported, CTRL[CALOFS] and CTRL[CAL_REQ] can be used to request the calibration routines to run. 0b - Calibration Not Implemented. 1b - Calibration Implemented.
9 IADCKI	Internal ADC Clock Implemented This read-only field indicates if this implementation of the ADC block includes an internal clock source. When supported, the CFG[ADCKEN] field is available and documented for enabling the clock source. When available, this clock source is used in clock selection logic external to the module for use within the system. 0b - Internal clock source not implemented. 1b - Internal clock source (and CFG[ADCKEN]) implemented.
8 VR1RNGI	Voltage Reference 1 Range Control Bit Implemented

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This read-only field indicates if a control bit is implemented for selecting the input voltage range on Voltage Reference Option 1.When range control is required, the CFG[VREF1RNG] field is available and documented for controlling the Voltage Reference Option 1. 0b - Range control not required. CFG[VREF1RNG] is not implemented. 1b - Range control required. CFG[VREF1RNG] is implemented.
7 —	Reserved This read-only field is reserved and always has the value 0.
6-4 CSW	Channel Scale Width This read-only field indicates if channel scaling is supported by this implementation. When supported, each command buffer has a control field (CMDLa[CSCALE]) for setting input scaling. 000b - Channel scaling not supported. 001b - Channel scaling supported. 1-bit CSCALE control field. 110b - Channel scaling supported. 6-bit CSCALE control field.
3 MVI	Multi Vref Implemented This read-only field indicates if multiple Voltage Reference High inputs are supported by this implementation. When multiple voltage references are supported, the CFG[REFSEL] field is available and documented for selecting the Voltage Reference High options. 0b - Single voltage reference high (VREFH) input supported. 1b - Multiple voltage reference high (VREFH) inputs supported.
2 —	Reserved This read-only field is reserved and always has the value 0.
1 DIFFEN	Differential Supported This read-only field indicates if differential operation is supported by this implementation. When supported, CMDLa[CTYPE] in each command buffer is used for configuring for differential operation. 0b - Differential operation not supported. 1b - Differential operation supported.
0 RES	Resolution This read-only field indicates the maximum accuracy supported by this implementation. When RES=1, each command buffer has a control field (CMDLa[MODE]) for selecting resolution of conversions for the associated command. 0b - Up to 12-bit single ended resolution supported (and 13-bit differential resolution if VERID[DIFFEN] = 1b). 1b - Up to 16-bit single ended resolution supported (and 16-bit differential resolution if VERID[DIFFEN] = 1b).

42.6.3 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	0	1	1	1	0	0	0	0	0	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	0	1	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-24 CMD_NUM	Command Buffer Number Number of command buffers implemented.
23-16 CV_NUM	Compare Value Number Number of compare value registers implemented.
15-8 FIFOSIZE	Result FIFO Depth The maximum number of conversion datawords that can be stored in the result FIFO before an overflow occurs. This field is read-only. 0000_0001b - Result FIFO depth = 2 dataword. 0000_0100b - Result FIFO depth = 4 datawords. 0000_1000b - Result FIFO depth = 8 datawords. 0001_0000b - Result FIFO depth = 16 datawords. 0010_0000b - Result FIFO depth = 32 datawords. 0100_0000b - Result FIFO depth = 64 datawords.
7-0 TRIG_NUM	Trigger Number Number of Triggers implemented.

42.6.4 Control Register (CTRL)

Offset

Register	Offset
CTRL	10h

Function

Control Register includes primary control bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0				CAL_AVGS			
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	CALHS	0	CALOFS	CAL_REQ	DOZEN	RST	ADCEN
W								RSTFIFO0								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved This read-only field is reserved and always has the value 0.
24-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 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 recommended minimum setting is for 256 averaging (CAL_AVGS set to 0x8 or greater). The CAL_AVGS bit field applies to CALOFS, CAL_REQ and CALHS calibration functions. CAL_AVGS should be fixed when requesting and running calibration with CTRL[CAL_REQ], CTRL[CALOFS], or CTRL[CALHS]. . 0000b - Single conversion. 0001b - 2 conversions averaged. 0010b - 4 conversions averaged. 0011b - 8 conversions averaged.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - 16 conversions averaged. 0101b - 32 conversions averaged. 0110b - 64 conversions averaged. 0111b - 128 conversions averaged. 1000b - 256 conversions averaged. 1001b - 512 conversions averaged. 1010b - 1024 conversions averaged.
15-9 —	Reserved This read-only field is reserved and always has the value 0.
8 RSTFIFO0	Reset FIFO 0 0b - No effect. 1b - FIFO 0 is reset.
7 —	Reserved This read-only field is reserved and always has the value 0.
6 CALHS	High Speed Mode Trim Request Setting CALHS to 1 initiates a high speed mode trim calculation. Any conversions in progress are completed before launching the high speed mode calibration function. After being accepted, the ADC calculates the trim value and updates the HSTRIM register automatically. The CALHS bit is cleared by hardware upon completion of the high speed mode trim calculation. The STAT[CAL_RDY] bit indicates when the high speed mode calibration routine has completed. 0b - No request for high speed mode trim has been made 1b - Request for high speed mode trim has been made
5 —	Reserved This read-only field is reserved and always has the value 0.
4 CALOFS	Offset Calibration Request Setting CALOFS to 1 initiates a hardware offset calibration calculation. Any conversions in progress are completed before launching the offset calibration function. After being accepted, the ADC calculates the offset value and updates either the OFSTRIM register automatically. The CALOFS bit is cleared by hardware upon completion of the offset calibration calculation. The STAT[CAL_RDY] bit indicates when the offset calibration routine has completed. 0b - No request for offset calibration has been made 1b - Request for offset calibration function
3 CAL_REQ	Auto-Calibration Request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Request the hardware calibration routine to run. This bit is automatically cleared when the system accepts the hardware calibration request. Note, the status flag STAT[CAL_RDY] will indicate when this calibration routine has been completed. 0b - No request for hardware calibration has been made 1b - A request for hardware calibration has been made
2 DOZEN	Doze Enable Controls system transition to low power modes while ADC is converting. When DOZEN is clear, immediate entries to low power modes are allowed. Note that the selected clock source provided from the on-chip clock source must be able to continue operating. When the DOZEN bit is set, the ADC waits for the current averaging iteration/FIFO storage to complete. When the system is entering a low power mode in which ADC operation is not supported, the DOZEN bit is ignored and the 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 Control Register. Remains set until cleared by software. 0b - ADC logic is not reset. 1b - ADC logic is reset.
0 ADCEN	ADC Enable 0b - ADC is disabled. 1b - ADC is enabled.

42.6.5 Status Register (STAT)

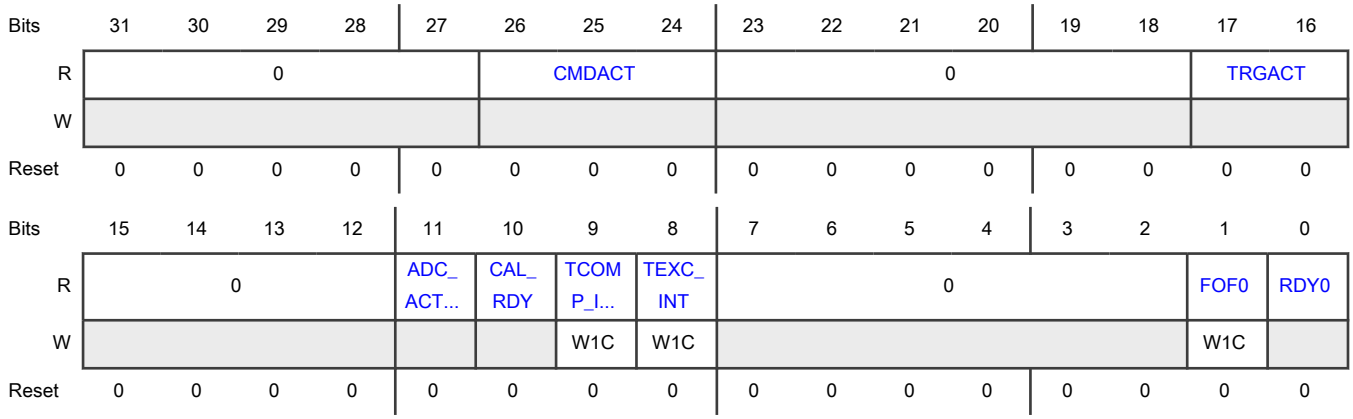
Offset

Register	Offset
STAT	14h

Function

The Status Register provides the current status of the ADC module.

Diagram



Fields

Field	Function
31-27 —	Reserved This read-only field is reserved and always has the value 0.
26-24 CMDACT	Command Active CMDACT is a read-only status field indicating the command that is actively being processed. Note, commands are only shown here when they are actively being processed by the SAR routine. Use CMDACT in conjunction with ADC_ACTIVE to determine which conversion is running. 000b - No command is currently in progress. 001b - Command 1 currently being executed. 010b - Command 2 currently being executed. 011b-111b - Associated command number is currently being executed.
23-18 —	Reserved This read-only field is reserved and always has the value 0.
17-16 TRGACT	Trigger Active TRGACT is a read-only status field indicating the trigger actively being processed. This can be used to determine which trigger is running through a trigger delay or being converted by the SAR routine. The command associated with TRGACT will be running through a conversion when CMDACT is non-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 This read-only field is reserved and always has the value 0.
11	ADC Active

Table continues on the next page...

Table continued from the previous page...

Field	Function
ADC_ACTIVE	This flag shows if the module is currently processing a conversion, or has pending triggers to service. When the ADC is IDLE, this bit is set to 0b0. 0b - The ADC is IDLE. There are no pending triggers to service and no active commands are being processed. 1b - The ADC is processing a conversion, running through the power up delay, or servicing a trigger.
10 CAL_RDY	Calibration Ready This flag shows if the ADC CTRL[CAL_REQ] or CTRL[CALOFS] request for calibration 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 hasn't been ran. 1b - The ADC is calibrated.
9 TCOMP_INT	Interrupt Flag For Trigger Completion When a trigger sequence has been completed (all associated commands have been run) the TCOMP_INT flag is asserted. Note that IE[TCOMP_IE] must be set for the specific trigger source to flag an interrupt upon completion. 0b - Either IE[TCOMP_IE] is set to 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 When a high priority trigger exception has occurred and CFG[TRES] = 0, this flag is asserted. This flag only asserts if trigger exception interrupts are enabled (IE[TEXC_IE] = 0x1). This flag can be used in conjunction 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 acknowledgement.
7-2 —	Reserved This read-only field is reserved and always has the value 0.
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 remains holding the original contents. This flag asserts regardless of the value of IE[FOFIE0]. However, an interrupt request is issued only if IE[FOFIE0] is set. This flag is cleared by writing a 1. 0b - No result FIFO 0 overflow has occurred since the last time the flag was cleared. 1b - At least one result FIFO 0 overflow has occurred since the last time the flag was cleared.
0 RDY0	Result FIFO 0 Ready Flag Indicates when the number of valid datawords in the result FIFO 0 is greater than the watermark level set in the FCTRL[FWMARK] bitfield. This flag asserts regardless of the value of IE[FWMIE0]. However,

Table continues on the next page...

Table continued from the previous page...

Field	Function
	an interrupt request or DMA request occurs only when the associated control bit (IE[FWMIE0] and DE[FWMDE0]) is set. This flag is cleared when the FCTRLa[FCOUNT] (which decrements on each read of the RESFIFO register) is less than or equal to the watermark level set in the FCTRL[FWMARK] bitfield. 0b - Result FIFO 0 data level not above watermark level. 1b - Result FIFO 0 holding data above watermark level.

42.6.6 Interrupt Enable Register (IE)

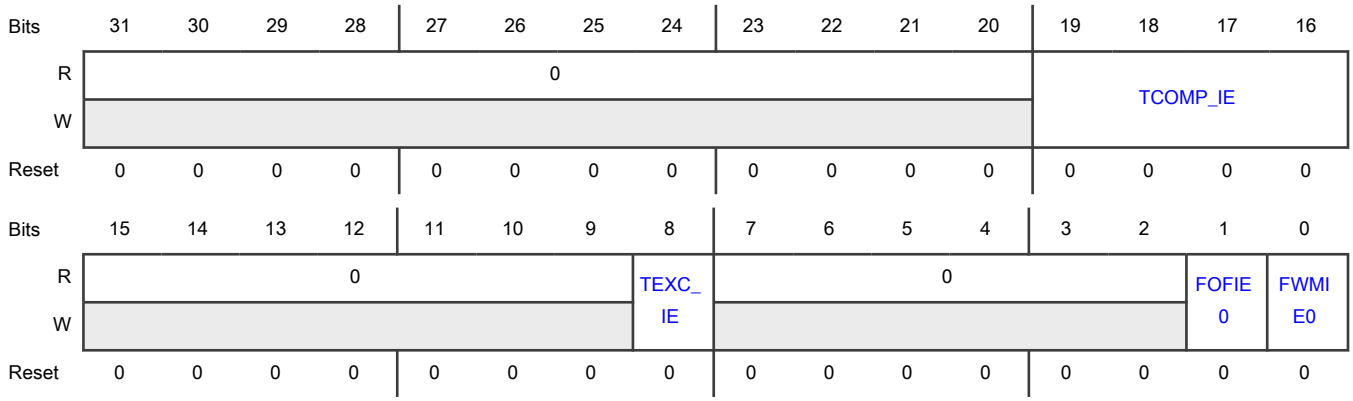
Offset

Register	Offset
IE	18h

Function

Interrupt Enable Register includes system interrupt masking control bits.

Diagram



Fields

Field	Function
31-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 TCOMP_IE	Trigger Completion Interrupt Enable Each bit in TCOMP_IE corresponds to a trigger source. (TCOMP_IE[0] corresponds to trigger 0, TCOMP_IE[1] corresponds to trigger 1, ...) A trigger completion interrupt is used to indicate when a complete trigger command sequence has been executed. All results will be stored in the FIFO when a sequence is considered complete.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0000b - Trigger completion interrupts are 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 - Trigger completion interrupts are enabled for every trigger source.
15-9 —	Reserved This read-only field is reserved and always has the value 0.
8 TEXC_IE	Trigger Exception Interrupt Enable This register field enables the ADC to assert an interrupt request when a high priority trigger exception occurs. TSTAT[TEXC_NUM] will contain the value of the corresponding trigger which was affected by the exception. 0b - Trigger exception interrupts are disabled. 1b - Trigger exception interrupts are enabled.
7-2 —	Reserved This read-only field is reserved and always has the value 0.
1 FOFIE0	Result FIFO 0 Overflow Interrupt Enable Configures the module to generate overflow interrupt requests when FOF flag is asserted. 0b - FIFO 0 overflow interrupts are not enabled. 1b - FIFO 0 overflow interrupts are enabled.
0 FWMIE0	FIFO 0 Watermark Interrupt Enable Configures the module to generate watermark interrupt requests when STAT[RDY0] flag is asserted. 0b - FIFO 0 watermark interrupts are not enabled. 1b - FIFO 0 watermark interrupts are enabled.

42.6.7 DMA Enable Register (DE)

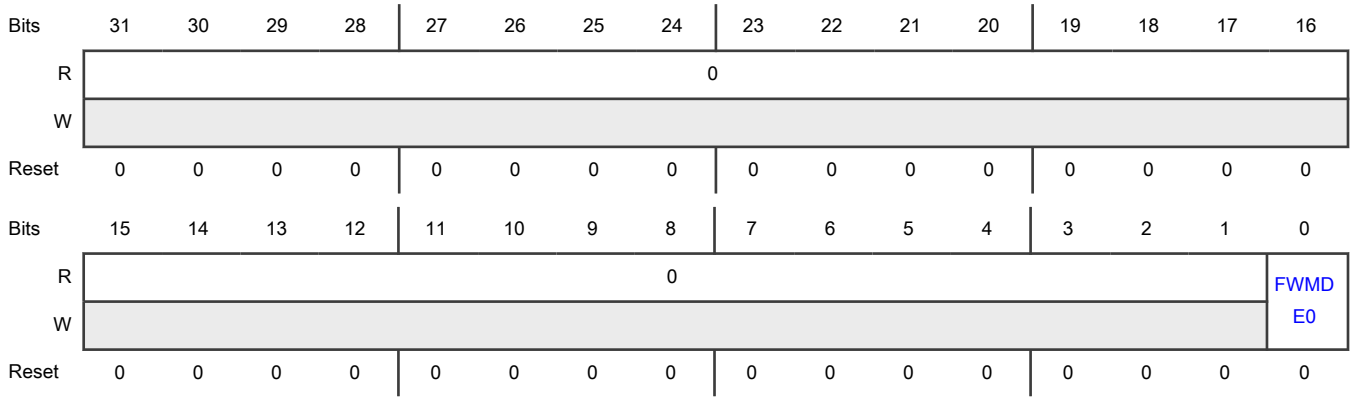
Offset

Register	Offset
DE	1Ch

Function

DMA Enable Register includes DMA request masking control bits.

Diagram



Fields

Field	Function
31-1 —	Reserved This read-only field is reserved and always has the value 0.
0 FWMDE0	FIFO 0 Watermark DMA Enable Configures the module to generate DMA requests when STAT[RDY0] flag is asserted. 0b - DMA request disabled. 1b - DMA request enabled.

42.6.8 Configuration Register (CFG)

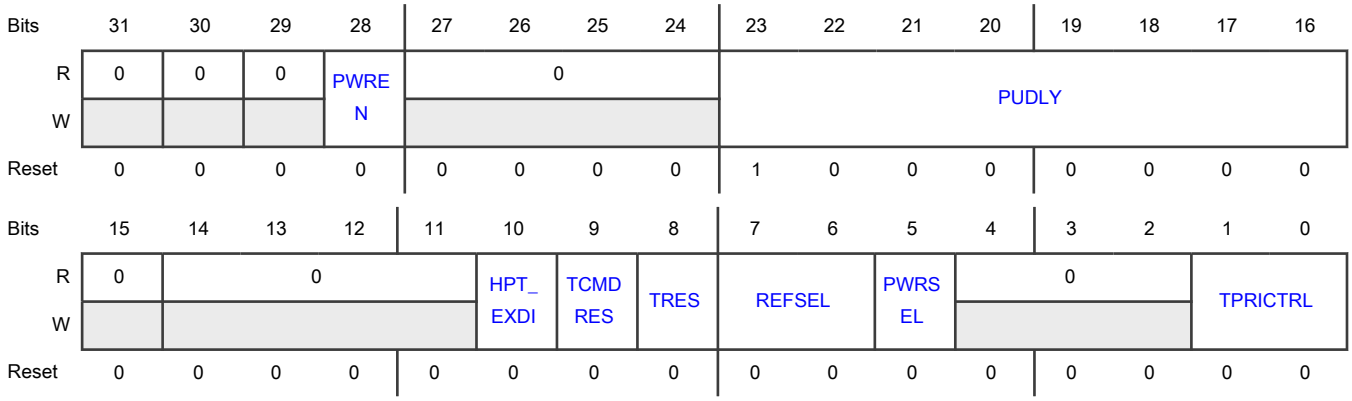
Offset

Register	Offset
CFG	20h

Function

The Configuration Register controls ADC functions that are common to all commands. The CFG cannot be changed while the CTRL[ADCEN] bit is set. Writes to CFG while CTRL[ADCEN] is set are ignored.

Diagram



Fields

Field	Function
31 —	Reserved This read-only field is reserved and always has the value 0.
30 —	Reserved This read-only field is reserved and always has the value 0.
29 —	Reserved This read-only field is reserved and always has the value 0.
28 PWREN	ADC Analog Pre-Enable Enables the ADC analog circuits. When setting CFG[PWREN], user code should delay for a period exceeding the analog startup time of $t_{ADCSTUP}$ before enabling the ADC for operation. The module is still operational even when CFG[PWREN] is left clear but command execution start is delayed for a period defined by CFG[PUDLY]. Refer to the device datasheet for ADC idle and ADC active power consumption parameters. 0b - ADC analog circuits are only enabled while conversions are active. Performance is affected due to analog startup delays. 1b - ADC analog circuits are pre-enabled and ready to execute conversions without startup delays (at the cost of higher DC current consumption). Note that a single power up delay (CFG[PUDLY]) is executed immediately once PWREN is set, and any detected trigger does not begin ADC operation until the power up delay time has passed. After this initial delay expires the analog remains pre-enabled and no additional delays are executed.
27-24 —	Reserved This read-only field is reserved and always has the value 0.
23-16 PUDLY	Power Up Delay The PUDLY must be programmed to a non-zero value to enable the ADC. Note, the PUDLY is executed in one of two modes depending on the value configured in PWREN:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When CFG[PWREN]=0b0, the ADC analog circuits are only powered on while the module is active. The delay defined by CFG[PUDLY] is executed after an initial trigger transitions the ADC from its Idle state. This 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 conversion(s) after activation is degraded if CFG[PUDLY] is set to too small a value. At the end of active conversions, if there are no subsequent pending conversions, the ADC analog is automatically reverted back to its low power idle state and the PUDLY runs again after the ADC is awakened with a later trigger. When CFG[PWREN]=0b1 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 remains enabled regardless of ADC activity. This configuration has the benefit of beginning conversions immediately after trigger event detection at the cost of increased DC power consumption of the analog circuits. Note, the ADC does not begin an initial conversion until the PUDLY has been executed regardless of the PWREN setting.
15 —	Reserved This read-only field is reserved and always has the value 0.
14-11 —	Reserved This read-only field is reserved and always has the value 0.
10 HPT_EXDI	High Priority Trigger Exception Disable Determines if high priority trigger exceptions are disabled. See section Trigger detect and command execution for a detailed description on trigger exception handling. If HPT_EXDI = 1, then exceptions are disabled and fields CFG[TCMDRES], CFG[TRES], and CFG[TPRICTRL] are ignored. 0b - High priority trigger exceptions are enabled. 1b - High priority trigger exceptions are disabled.
9 TCMDRES	Trigger Command Resume Determines where to resume from a high priority trigger exception. CFG[TRES] must be asserted for TCMDRES to be used. If asserted, the interrupted trigger sequence is resumed from the command where it was interrupted. If de-asserted the trigger sequence is restarted from the beginning. 0b - Trigger sequences interrupted by a high priority trigger exception is automatically restarted. 1b - Trigger sequences interrupted by a high priority trigger exception is resumed from the command executing before the exception.
8 TRES	Trigger Resume Enable Determines what happens after completing a high priority trigger exception. If CFG[TRES] is asserted, the interrupted trigger sequence is resumed or restarted automatically. The sequence can be resumed along command boundaries or restarted from the beginning of a sequence. This is controlled by CFG[TCMDRES]. If CFG[TRES] is de-asserted an interrupted trigger sequence does not resume automatically. In this case, interrupted triggers can be resumed via software by monitoring STAT[TEXC_INT] bit (or ISR handling for a trigger exception interrupt when IE[TEXC_IE] is set). Software determines which trigger was

Table continues on the next page...

Table continued from the previous page...

Field	Function
	interrupted by reading TSTAT[TEXC_NUM] and re-starts the trigger with a write of 1 to the corresponding SWTRIG[SWTx] bit. 0b - Trigger sequences interrupted by a high priority trigger exception are not automatically resumed or restarted. 1b - Trigger sequences interrupted by a high priority trigger exception are 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 on the voltage reference options specific to this packaged device. 00b - (Default) Option 1 setting. 01b - Option 2 setting. 10b - Option 3 setting. 11b - Reserved
5 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. 0b - Low power 1b - High power
4-2 —	Reserved
1-0 TPRICTRL	ADC Trigger Priority Control Controls how higher priority trigger exceptions are handled. See Trigger detect and command execution for a detailed explanation trigger event handling. 00b - If a higher priority trigger is detected during command processing, the current conversion is aborted and the new command specified by the trigger is started. 01b - If a higher priority trigger is received during command processing, the current command is stopped after completing the current conversion. If averaging is enabled, the averaging loop will be completed. However, CMDHa[LOOP] will be ignored and the higher priority trigger will be serviced. 10b - If a higher priority trigger is received during command processing, the current command will be completed (averaging, looping, compare) before servicing the higher priority trigger. 11b - Reserved

42.6.9 Pause Register (PAUSE)

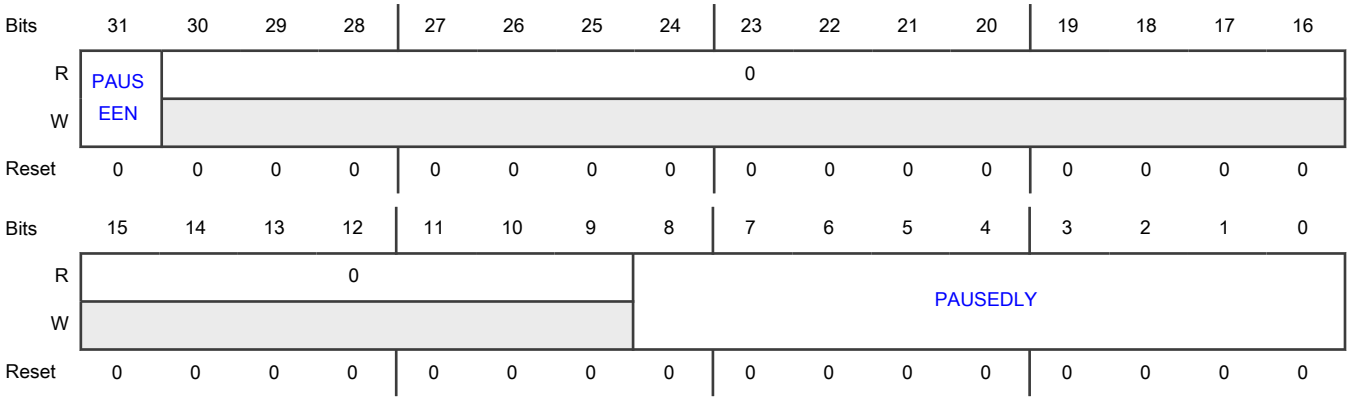
Offset

Register	Offset
PAUSE	24h

Function

The Pause Register controls an optional inserted delay between conversions. Note, the PAUSE register should not be modified while the CTRL[ADCEN] bit is set.

Diagram



Fields

Field	Function
31 PAUSEEN	PAUSE Option 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, and between conversions when command is executing in for "Compare Until True" configuration. Note, CMDHa[WAIT_TRIG] and PAUSE are mutually exclusive. CMDHa[WAIT_TRIG] will take priority over PAUSE between commands when both are enabled. 0b - Pause operation disabled 1b - Pause operation enabled
30-9 —	Reserved This read-only field is reserved and always has the value 0.
8-0 PAUSEDLY	Pause Delay When PAUSEEN is set, the PAUSEDLY field controls the duration of pausing during command execution sequencing. The pause delay is a count of (PAUSEDLY*4) ADCK cycles.

42.6.10 Software Trigger Register (SWTRIG)

Offset

Register	Offset
SWTRIG	34h

Function

The Software Trigger Register (SWTRIG) is written to initiate software triggered conversions. Writes to SWTRIG register are ignored while CTRL[ADCEN] is clear. Note that there is an approximately 3 ADC Clock cycle synchronization delay between asserting CTRL[ADCEN] until SWTRIG can be accepted.

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												0	0	0	0
W													SWT3	SWT2	SWT1	SWT0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved This read-only field is reserved and always has the value 0.
3 SWT3	Software Trigger 3 Event 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 Event 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	Software Trigger 1 Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWT1	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 Event Write 1 to SWT0 generates a trigger 0 event. Writing 1 to SWT0 is ignored while the trigger 0 event is being serviced or is pending. 0b - No trigger 0 event generated. 1b - Trigger 0 event generated.

42.6.11 Trigger Status Register (TSTAT)

Offset

Register	Offset
TSTAT	38h

Function

This register contains status flags to indicate when trigger sequences have been completed or interrupted by a high priority trigger exception. Each bit in this register is set by hardware and cleared by software. To clear a bit in this register, write a 0b1 to the corresponding bit position.

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

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This read-only field is reserved and always has the value 0.
19-16 TCOMP_FLAG	Trigger Completion Flag Each bit corresponds to a trigger. When the corresponding trigger sequence has been completed a bit in TCOMP_FLAG will be set. For example, if trigger 3 has been completed, then bit 19 will be set in TSTAT. This register will only be active if the corresponding bit in IE[TCOMP_IE] = 1. Note, 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 2-4 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 This read-only field is reserved and always has the value 0.
3-0 TEXC_NUM	Trigger Exception Number Each bit corresponds to a trigger. When the corresponding trigger sequence has been interrupted by a high priority trigger exception a 1 will be set in its bit position. For example, if trigger 3 has been interrupted, then bit 3 will be set in TSTAT. This register will be 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.

42.6.12 Offset Trim Register (OFSTRIM)

Offset

Register	Offset
OFSTRIM	40h

Function

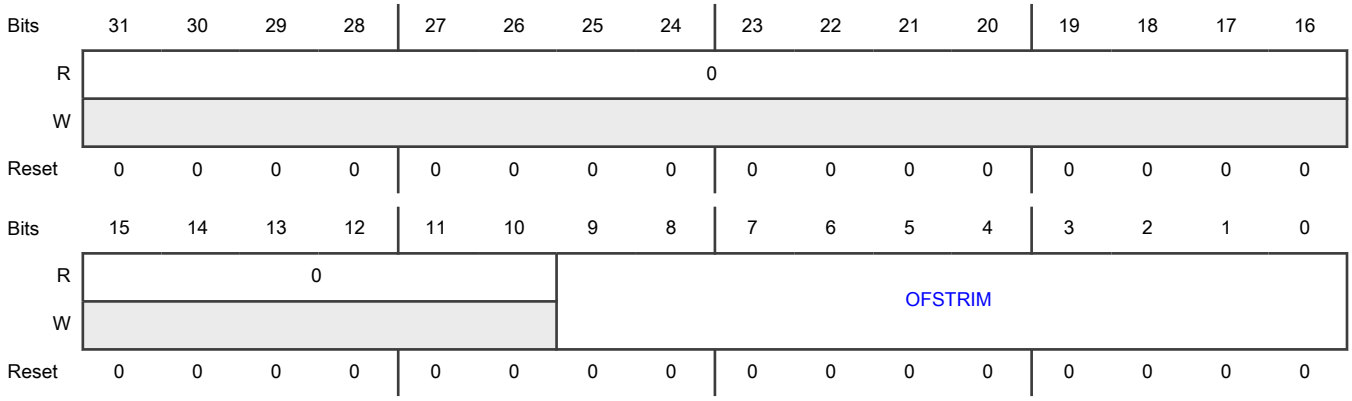
The Offset Trim register is used to trim for comparator offset. The ADC supports an offset calibration function in which the OFSTRIM register is automatically updated.

To trigger the automatic update of OFSTRIM, set the CTRL[CALOFS] bit. Upon completion of the offset calibration sequence, the STAT[CAL_RDY] bit is set to 0b1 and the OFSTRIM[OFSTRIM] bitfield is updated with a 10-bit signed value.

The OFSTRIM[OFSTRIM] value is used to minimize offset for normal conversions with a range of -512 to 511. For 16-bit conversions, each increment is 1/2 LSB resulting in a programmable offset range of -256 to +255.5 LSB. For 12-bit conversions, each increment is 1/32 LSB resulting in a programmable offset range of -16 to +15.96875 LSB.

The OFSTRIM[OFSTRIM] trim value is also used in the calibration function.

Diagram



Fields

Field	Function
31-10 —	Reserved This read-only field is reserved and always has the value 0.
9-0 OFSTRIM	Trim for Offset OFSTRIM is a 10-bit signed value between -512 and 511. This value is subtracted from the raw ADC conversion result.

42.6.13 High Speed Trim Register (HSTRIM)

Offset

Register	Offset
HSTRIM	48h

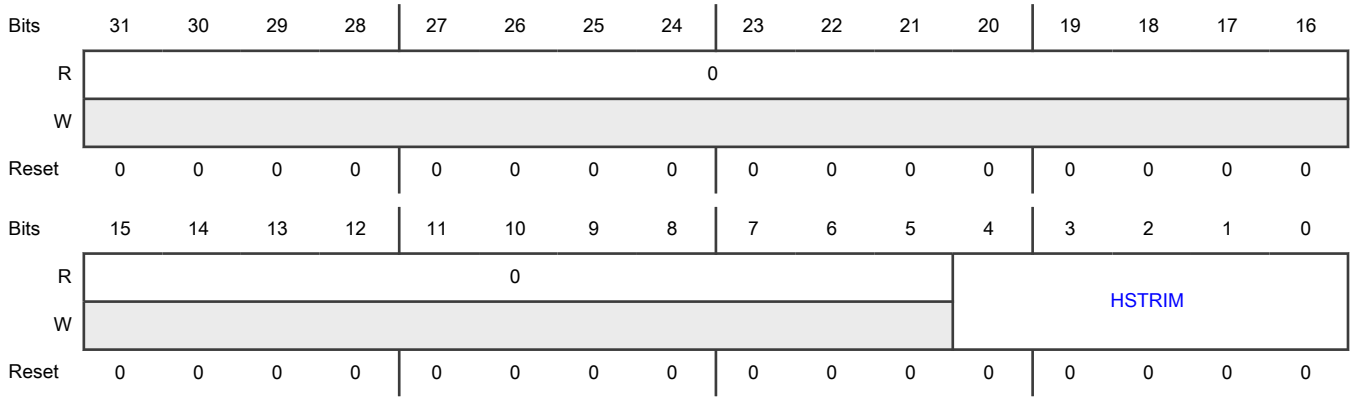
Function

The High Speed Trim register is used to trim for high speed function. The ADC supports an additional offset calibration function for high speed mode in which the HSTRIM register is automatically updated.

To trigger the automatic update of HSTRIM, set the CTRL[CALHS] bit. Upon completion of the high speed calibration sequence, the STAT[CAL_RDY] bit is set to 0b1 and the HSTRIM[HSTRIM] bitfield is updated with a 5-bit signed value.

The HSTRIM[HSTRIM] value is used to minimize offset for high speed conversions (when CFG2[HS] is set). The HSTRIM register supports a range of -16 to 15.

Diagram



Fields

Field	Function
31-5 —	Reserved This read-only field is reserved and always has the value 0.
4-0 HSTRIM	Trim for High Speed Conversions HSTRIM is a 5-bit signed value between -16 and 15 that trims circuitry for high speed mode.

42.6.14 Trigger Control Register (TCTRL0 - TCTRL3)

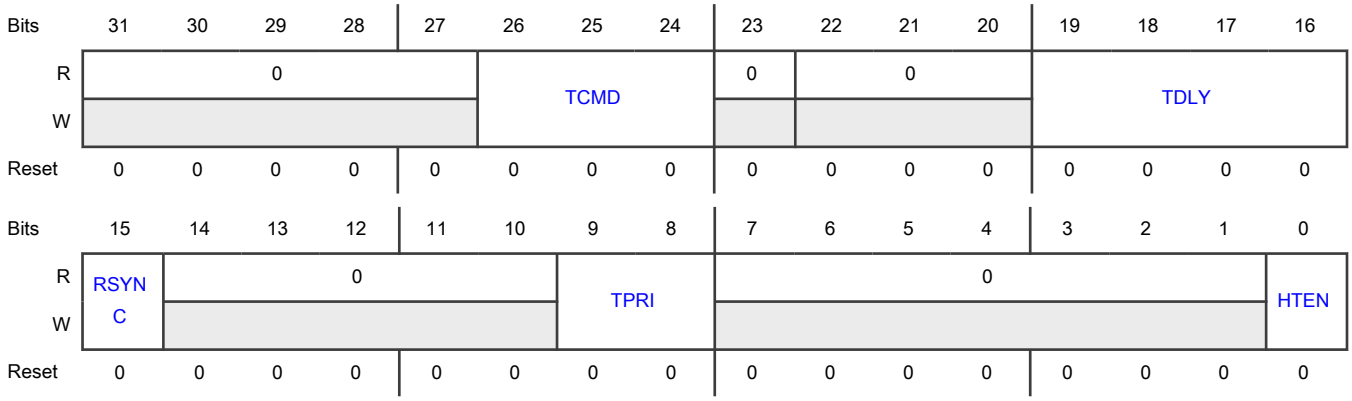
Offset

Register	Offset
TCTRL0	A0h
TCTRL1	A4h
TCTRL2	A8h
TCTRL3	ACh

Function

The Trigger Control (TCTRL*a*) register implements control fields associated with each implemented trigger source. When the ADC is actively executing commands, only one of the TCTRL*a* registers is actively controlling ADC conversions. The actively controlling TCTRL*a* register should not be updated while the ADC is active. A write to a TCTRL*a* registers while that trigger control register is controlling the ADC operation may cause unpredictable behavior.

Diagram



Fields

Field	Function
31-27 —	Reserved This read-only field is reserved and always has the value 0.
26-24 TCMD	Trigger Command Select Selects the command from command buffer to execute upon detection of the associated trigger event. When a trigger is received while the ADC is busy converting and the new trigger has higher priority than the trigger associated with the current command being executed, the command in this register, associated with the new trigger, is executed under control of CFG[TPRICTRL]. See section Trigger detect and command execution for a more detailed description on the relationship between CFG[TPRICTRL] and TCMD. 000b - Not a valid selection from the command buffer. Trigger event is ignored. 001b - CMD1 is executed 010b-110b - Corresponding CMD is executed 111b - CMD7 is executed
23 —	Reserved This read-only field is reserved and always has the value 0.
22-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 TDLY	Trigger Delay Select Selects the trigger delay duration to wait at the start of servicing a trigger event. Each trigger source has an associated programmable delay prior to beginning an initial conversion. When TDLY field is clear, then no delay is incurred. When TDLY is set to a non-zero value, the duration for the delay is 2^{TDLY} ADCK cycles.
15 RSYNC	Trigger Resync

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If RSYNC is set to 1, this trigger source will be used as a resync trigger. A resync trigger can be configured to restart or abort a running trigger sequence (the target). Each resync trigger is configured to have a resync target. When a resync trigger is asserted, the target sequence will be aborted, the target FIFO destination(s) will be cleared, and the target sequence can optionally be restarted depending on CFG[TRES]. Multiple conditions must be met for a resync trigger to execute properly. 1. The resync trigger must have higher priority than the resync target. 2. The resync trigger TCTRLa[RSYNC] must be set to 0b1. 3. The resync target, specified by TCTRLa[TCMD], must be executing when the resync trigger is asserted. Please see Resync functionality for more information.
14-10 —	Reserved This read-only field is reserved and always has the value 0.
9-8 TPRI	Trigger Priority Setting This bitfield sets the priority of the associated trigger source. If two or more triggers have the same priority level setting, the lower order trigger event has the higher priority (e. g., if Trigger 0 and Trigger 1 are both pending triggers and TCTRL0[TPRI] is configured the same as TCTRL1[TPRI], then the command associated with Trigger 0 is serviced first). 00b - Set to highest priority, Level 1 01b-10b - Set to corresponding priority level 11b - Set to lowest priority, Level 4
7-1 —	Reserved This read-only field is reserved and always has the value 0.
0 HTEN	Trigger Enable Enables hardware trigger source to initiate conversion on the rising-edge of the input trigger source. **NOTE** Enabling hardware trigger will not disable software triggers. 0b - Hardware trigger source disabled 1b - Hardware trigger source enabled

42.6.15 FIFO Control Register (FCTRL0)

Offset

Register	Offset
FCTRL0	E0h

Function

The FIFO Control (FCTRL*a*) registers contain 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 FCTRLa[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
31-19 —	Reserved This read-only field is reserved and always has the value 0.
18-16 FWMARK	Watermark Level Selection FWMARK is a programmable threshold setting. When the number of datawords stored in the ADC Result FIFO is greater than the value in FWMARK, the STAT[RDY] flag is asserted to indicate stored data has reached the programmable threshold. When IE[FWMIEx] is set, an interrupt request is generated. When DE[FWMDEx] is set, a DMA request is generated.
15-4 —	Reserved This read-only field is reserved and always has the value 0.
3-0 FCOUNT	Result FIFO Counter This read-only field indicates the number of datawords that are stored in the result FIFO. This value may be used in conjunction with PARAM[FIFOSIZE] to calculate how much room is left in the result FIFO. FCOUNT is incremented with each store of new data to the result FIFO and decrements with each read of the result FIFO. The FIFO is reset by writing to the CTRL[RSTFIFOx] bit, resulting in FCTRLa[FCOUNT] initialized to 0x0.

42.6.16 Gain Calibration Control (GCC0)

Offset

Register	Offset
GCC0	F0h

Function

The Gain Calibration Control register (GCC0) hold intermediate values that are utilized as part of the calibration steps. The GCC*a*[GAIN_CAL] field is calculated with hardware and auto updated during the calibration steps. Once the hardware calibration sequence has updated GCC*a*[GAIN_CAL] the GCC*a*[RDY] status flag is asserted automatically. Note that requesting a calibration routine automatically clears the GCC*a*[RDY] bit until the new GCC*a*[GAIN_CAL] value is calculated. For the calibration steps to be completed, further software processing is needed where the GCC*a*[GAIN_CAL] value is used to calculate the gain adjustment and the result is stored to GCR*a*[GCALR]. For more information see Calibration in the functional description.

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 This read-only field is reserved and always has the value 0.
24 RDY	Gain Calibration Value Valid This status flag indicates when the data stored in GCCa[GAIN_CAL] is valid. 0b - The GAIN_CAL value is invalid. Run the hardware calibration routine for this value to be set. 1b - The GAIN_CAL value is valid. GAIN_CAL should be used by software to derive GCRa[GCALR].
23-16 —	Reserved This read-only field is reserved and always has the value 0.
15-0 GAIN_CAL	Gain Calibration Value GAIN_CAL is read-only. As part of the hardware calibration steps, GAIN_CAL is automatically updated with a 16-bit signed number and is used in the gain adjustment calculations to derive the value written to the GCRa[GCALR] register. For more information see Calibration in the functional description.

42.6.17 Gain Calculation Result (GCR0)

Offset

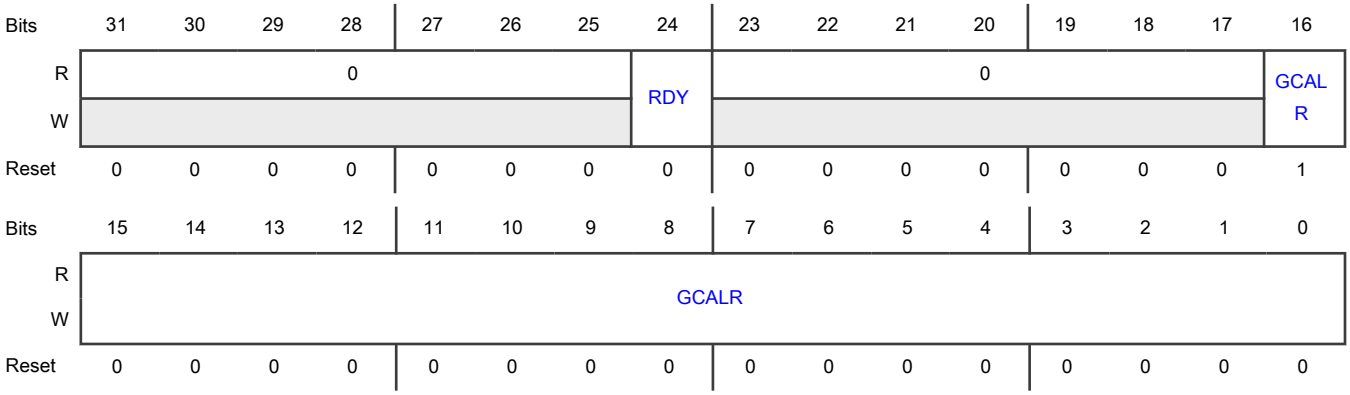
Register	Offset
GCR0	F8h

Function

The Gain Calculation Result register (GCR0) is used during ADC conversions for automated gain error adjustment. Determining the values to store to GCC0[GCALR] is the result of software calculations described in setup steps in Calibration. There is a RDY status flag in the GCR register which indicates whether the value GCRa[GCALR] is valid. Upon writing the GCRa[GCALR] value, the user should also assert GCRa[RDY] to indicate that the gain adjustment result is valid.

After beginning a calibration sequence by asserting CTRL[CAL_REQ], the calibration sequence is not complete until GCRa[GCALR] is calculated and GCRa[RDY] is asserted. The gain adjustment calculation results in a floating point value between 0 and 2. GCRa[GCALR][16] holds the integer value (0 or 1) and GCRa[GCALR][15:0] holds the 16-bit fractional component of the gain adjustment calculation. To convert the GCRa[GCALR] value back into the gain adjustment value in decimal format, this would be calculated as: 1*GCRa[16]+0.5*GCRa[15]+0.25*GCRa[14]+0.125*GCRa[13]+...

Diagram



Fields

Field	Function
31-25 —	Reserved This read-only field is reserved and always has the value 0.
24 RDY	Gain Calculation Ready The RDY flag indicates to the ADC when the data stored in GCRa[GCALR] is valid and is used for gain adjustment. The GCRa[GCALR] must be written from memory or calculated each time the calibration routine is run. The RDY flag must be asserted by the user after writing valid data into the GCALR field. The RDY flag is cleared automatically when requesting a new calibration sequence. 0b - The GCALR value is invalid. 1b - The GCALR value is valid.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-17 —	Reserved This read-only field is reserved and always has the value 0.
16-0 GCALR	Gain Calculation Result This value holds the 17-bit value generated from the gain adjustment calculation during the calibration steps.

42.6.18 Command Low Buffer Register (CMDL1 - CMDL7)

Offset

Register	Offset
CMDL1	100h
CMDL2	108h
CMDL3	110h
CMDL4	118h
CMDL5	120h
CMDL6	128h
CMDL7	130h

Function

There are 7 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDL*a*:CMDH*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRL*a*[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer should not be updated while the ADC is active. A write to a CMD buffer while that CMD buffer is controlling the ADC operation may cause unpredictable behavior.

NOTE

The 7 command buffers are numbered [CMDH1:CMDL1] through [CMDH7:CMDL7]. In NXP-supplied header files, these are likely to be defined as two 7-element arrays that are indexed from 0. I.e., the type declaration would be:

```
unsigned int CMDL[7];
unsigned int CMDH[7];
```

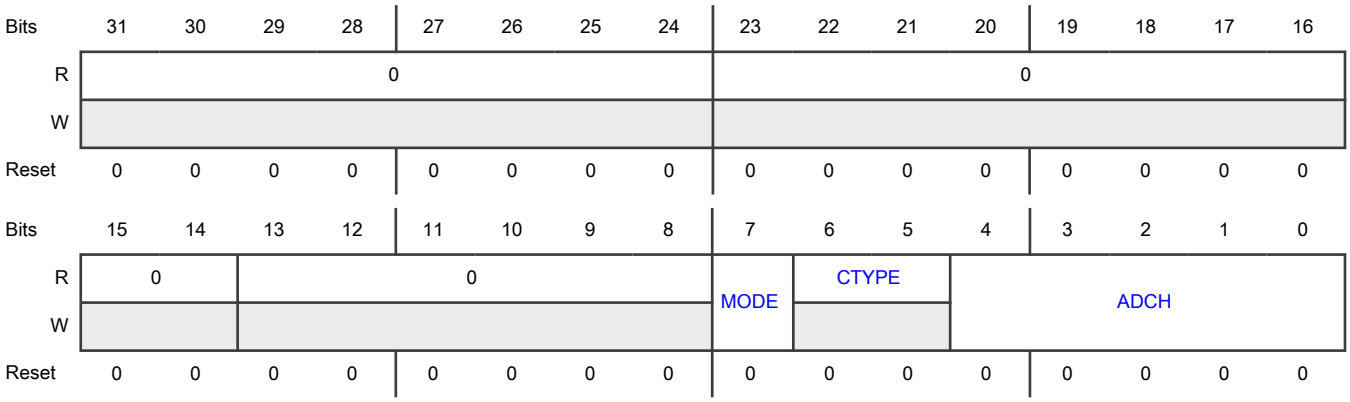
and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL7 as `ADC0->CMDL[6]`.

The CMDL*a*[ADCH] and CMDL*a*[CTYPE] bitfields control selection of input channels. Each ADC command independently makes a channel selection.

NOTE

Some of the input channels are from on-chip sources such as temperature sensors and reference voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions may not be available for your device. Refer to chip-specific information for the channels supported on this device.

Diagram



Fields

Field	Function
31-24 —	Reserved This read-only field is reserved and always has the value 0.
23-16 —	Reserved This read-only field is reserved and always has the value 0.
15-14 —	Reserved This read-only field is reserved and always has the value 0.
13-8 —	Reserved This read-only field is reserved and always has the value 0.
7 MODE	Select Resolution of Conversions The MODE field, selects the ADC resolution. 0b - Standard resolution. Single-ended 12-bit conversion. 1b - High resolution. Single-ended 16-bit conversion.
6-5 CTYPE	Conversion Type Zero is the only valid program value and selects A-side channels for conversion. 00b - Single-Ended Mode. Only A side channel is converted. 01b-11b - Reserved.
4-0	Input Channel Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
ADCH	**NOTE** Some of the input channels are from internal resources such as temperature sensors and bandgap voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the ADC channel assignments for your device, see the chip-specific information. 0_0000b - Select CH0A. 0_0001b - Select CH1A. 0_0010b - Select CH2A. 0_0011b - Select CH3A. 0_0100b-1_1101b - Select corresponding channel CHnA. 1_1110b - Select CH30A. 1_1111b - Select CH31A.

42.6.19 Command High Buffer Register (CMDH1 - CMDH7)

Offset

Register	Offset
CMDH1	104h
CMDH2	10Ch
CMDH3	114h
CMDH4	11Ch
CMDH5	124h
CMDH6	12Ch
CMDH7	134h

Function

There are 7 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDL*a*:CMDH*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRL*a*[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer should not be updated while the ADC is active. A write to a CMD buffer while that CMD buffer that is controlling ADC operation may result in unpredictable behavior.

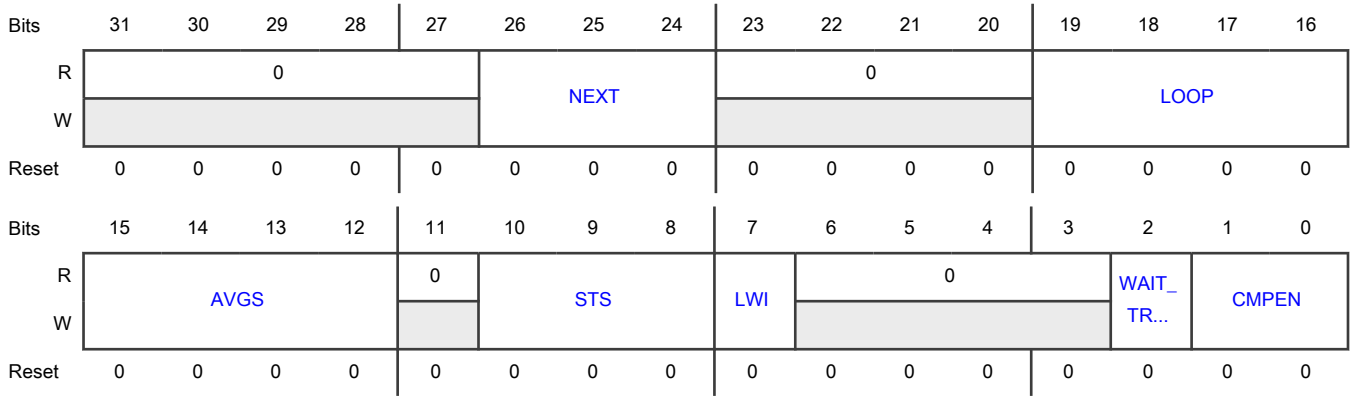
NOTE

The 7 command buffers are numbered [CMDH1:CMDL1] through [CMDH7:CMDL7]. In NXP-supplied header files, these are likely to be defined as two 7-element arrays that are indexed from 0. I.e., the type declaration would be:

```
unsigned int CMDL[7];
unsigned int CMDH[7];
```

and software would access ADC0 CMDH1 as `ADC0->CMDH[0]` and ADC0 CMDL7 as `ADC0->CMDL[6]`.

Diagram



Fields

Field	Function
31-27 —	Reserved This read-only field is reserved and always has the value 0.
26-24 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 and the order of execution is strictly controlled by the NEXT field (for example, a sequence of commands could be CMD2 to CMD1 to CMD3). Unending circular command execution is allowed 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 0x0 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. 000b - No next command defined. Terminate conversions at completion of current command. If lower priority trigger pending, begin command associated with lower priority trigger. 001b - Select CMD1 command buffer register as next command. 010b-110b - Select corresponding CMD command buffer register as next command 111b - Select CMD7 command buffer register as next command.
23-20 —	Reserved 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
19-16 LOOP	Loop Count Select Selects how many times this command executes (and stores conversion result to RESFIFO) before finish and transition to the next command or Idle state. LWI field controls whether a single channel is converted on each iteration or auto channel increment results in channel scanning functionality. 0000b - Looping not enabled. Command executes 1 time. 0001b - Loop 1 time. Command executes 2 times. 0010b - Loop 2 times. Command executes 3 times. 0011b-1110b - Loop corresponding number of times. Command executes LOOP+1 times. 1111b - Loop 15 times. Command executes 16 times.
15-12 AVGS	Hardware Average Select Selects how many ADC conversions are averaged to create the ADC result (2^{AVGS}). An internal storage buffer is used to capture 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 more detailed description on usage of AVGS, LOOP, and NEXT bitfields in command execution sequencing. 0000b - Single conversion. 0001b - 2 conversions averaged. 0010b - 4 conversions averaged. 0011b - 8 conversions averaged. 0100b - 16 conversions averaged. 0101b - 32 conversions averaged. 0110b - 64 conversions averaged. 0111b - 128 conversions averaged. 1000b - 256 conversions averaged. 1001b - 512 conversions averaged. 1010b - 1024 conversions averaged.
11 —	Reserved This read-only field is reserved and always has the value 0.
10-8 STS	Sample Time Select When programmed to 0b000 the minimum sample time of 3.5 ADCK cycles is selected. When STS is programmed to 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 accurately sampled. Longer sample times can also be used to lower overall power consumption when command looping and sequencing is configured and high conversion rates are not required.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Minimum sample time of 3.5 ADCK cycles. 001b - 3.5 + 2^1 ADCK cycles; 5.5 ADCK cycles total sample time. 010b - 3.5 + 2^2 ADCK cycles; 7.5 ADCK cycles total sample time. 011b - 3.5 + 2^3 ADCK cycles; 11.5 ADCK cycles total sample time. 100b - 3.5 + 2^4 ADCK cycles; 19.5 ADCK cycles total sample time. 101b - 3.5 + 2^5 ADCK cycles; 35.5 ADCK cycles total sample time. 110b - 3.5 + 2^6 ADCK cycles; 67.5 ADCK cycles total sample time. 111b - 3.5 + 2^7 ADCK cycles; 131.5 ADCK cycles total sample time.
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 - Auto channel increment disabled 1b - Auto channel increment enabled
6-3 —	Reserved This read-only field is reserved and always has the value 0.
2 WAIT_TRIG	Wait for Trigger Assertion before Execution. Controls if commands are automatically executed or if a trigger must be received before execution. If WAIT_TRIG is asserted, wait states will be added before the command until the active trigger is asserted again. When WAIT_TRIG is disabled, each command will be automatically executed when called. 0b - This command will be automatically executed. 1b - The active trigger must be asserted again before executing this command.
1-0 CMPEN	Compare Function Enable After an ADC channel input is sampled and converted and any averaging iterations are performed, the CMDH*a*[CMPEN] field guides operation of the automatic compare function to optionally only store when the compare operation is true. When compare is enabled, the conversion result is compared to the compare value registers (CV*a*[CVH] and CV*a*[CVL]). There are multiple options on command sequencing related to the compare function. See Compare function for a detailed explanation of the compare options.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Not all Command Buffers have implemented the CMPEN field. The CMPEN field is only available in Command Buffers that have a corresponding Compare Value register. 00b - Compare disabled. 01b - Reserved 10b - Compare enabled. Store on true. 11b - Compare enabled. Repeat channel acquisition (sample/convert/compare) until true.

42.6.20 Compare Value Register (CV1 - CV7)

Offset

Register	Offset
CV1	200h
CV2	204h
CV3	208h
CV4	20Ch
CV5	210h
CV6	214h
CV7	218h

Function

The compare value registers (CV*a*) contain values used to compare the conversion result when the compare function is enabled. This register is formatted in the same way as the D field in the RESFIFO registers in different modes of operation for both bit position definition and value format using unsigned or signed 2's complement. There is a direct association of each compare value register to a specific command buffer register (i.e., CV1 is only used during execution of CMD1 command).

NOTE

Not all Command Buffers have an associated Compare Value register. The compare function is only available on Command Buffers that have a corresponding Compare Value register.

When the ADC is actively executing commands, the CV*a* register associated with the active command (CMD*a*) should not be updated. Writes to associated CV*a* register during this time may result in unpredictable behavior.

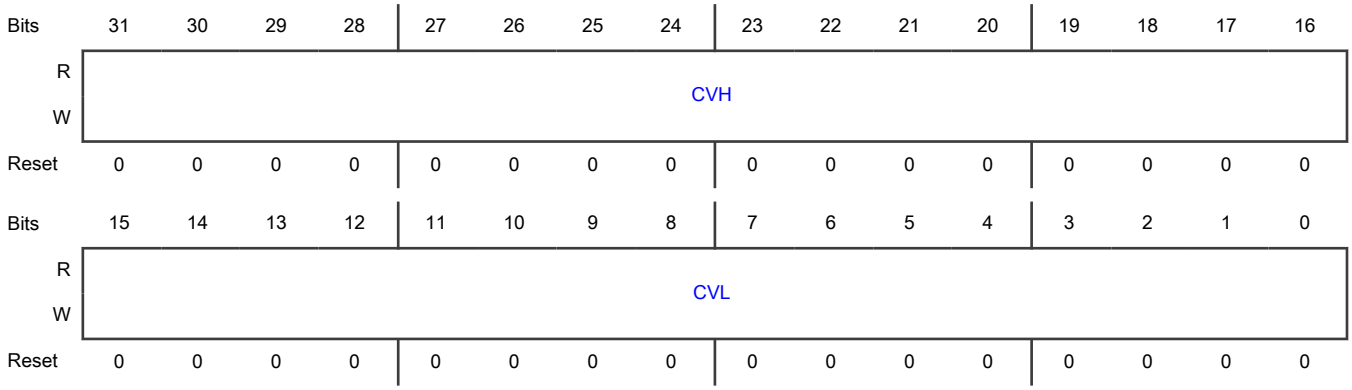
NOTE

The 7 compare value registers are numbered [CV1] through [CV7]. In NXP-supplied header files, this is likely to be defined as a 7-element array that is indexed from 0. I.e., the type declaration would be:

```
unsigned int CV[7];
```

and software would access ADC0 CV1 as `ADC0->CV[0]` and ADC0 CV7 as `ADC0->CV[6]`.

Diagram



Fields

Field	Function
31-16 CVH	Compare Value High The compare function can be configured to check whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. After the input is sampled and converted and any averaging iterations are performed, the compare values in CVL and CVH are optionally used in a compare operation on the result. See Compare function for a description on how CVH is used.
15-0 CVL	Compare Value Low The compare function can be configured to check whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. After the input is sampled and converted and any averaging iterations are performed, the compare values in CVL and CVH are optionally used in a compare operation on the result. See Compare function for a description on how CVL is used.

42.6.21 Data Result FIFO Register (RESFIFO0)

Offset

Register	Offset
RESFIFO0	300h

Function

The data result FIFO register (RESFIFO) is a 8 entry FIFO that stores the data result of ADC conversions. In addition, several tag fields of source command and trigger information are stored along with the data. FCTRLa[FCOUNT] indicates how many valid datawords are stored in the RESFIFO. Reading RESFIFO provides the oldest unread dataword entry in the FIFO and decrements FCTRLa[FCOUNT]. The FIFO can be emptied by successive reads of RESFIFO. The FIFO is reset by writing 0b1 to the CTRL[RSTFIFOx] bit.

The following table describes the format of data in the result FIFO.

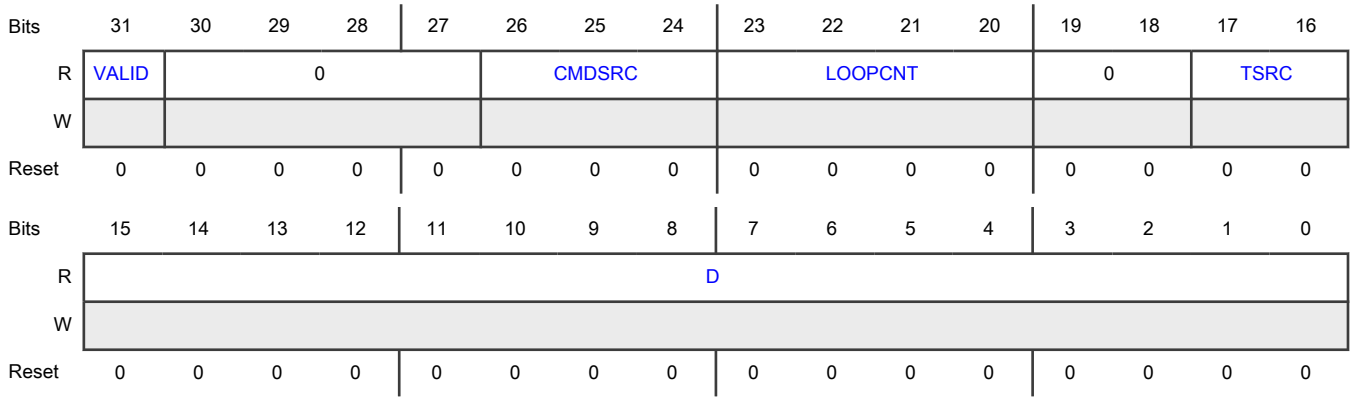
Table 328. Data result register format description

Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned, 16-bit magnitude, JLEFT=X
12-bit single-ended	0	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Unsigned, JLEFT=0, zero in D[15] and D[2:0]
12-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	0	Unsigned, JLEFT=1, zero in D[3:0]

NOTE

D: Data

Diagram



Fields

Field	Function
31 VALID	FIFO Entry is Valid Indicate the FIFO entry is valid. When the FIFO is holding data the VALID bit is set. When the FIFO is empty the VALID bit is clear. 0b - FIFO is empty. Discard any read from RESFIFO. 1b - FIFO record read from RESFIFO is valid.
30-27 —	Reserved This read-only field is reserved and always has the value 0.
26-24 CMDSRC	Command Buffer Source Indicate the command buffer being executed that generated this result. 000b - Not a valid value CMDSRC value for a dataword in RESFIFO. 0x0 is only found in initial FIFO state prior to an ADC conversion result dataword being stored to a RESFIFO buffer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001b - CMD1 buffer used as control settings for this conversion. 010b-110b - Corresponding command buffer used as control settings for this conversion. 111b - CMD7 buffer used as control settings for this conversion.
23-20 LOOPCNT	Loop Count Value Indicate the loop count value during command execution that generated this result. When CMDH*a*[LOOP] is non-zero, the command execution stores off a result 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 16^{th} conversion in command.
19-18 —	Reserved This read-only field is reserved and always has the value 0.
17-16 TSRC	Trigger Source Indicate the trigger source that initiated a conversion and generated this result. When multiple commands are chained together using the CMDHa[NEXT] field, the TSRC field for all datawords stored to the RESFIFO indicate the trigger source that started the sequence of commands. 00b - Trigger source 0 initiated this conversion. 01b - Trigger source 1 initiated this conversion. 10b - Trigger source 2 initiated this conversion. 11b - Trigger source 3 initiated this conversion.
15-0 D	Data Result The formatting for the data in D is summarized in Table 328.

42.6.22 Calibration General A-Side Registers (CAL_GAR0 - CAL_GAR33)

Offset

For a = 0 to 33:

Register	Offset
CAL_GARa	400h + (a × 4h)

Function

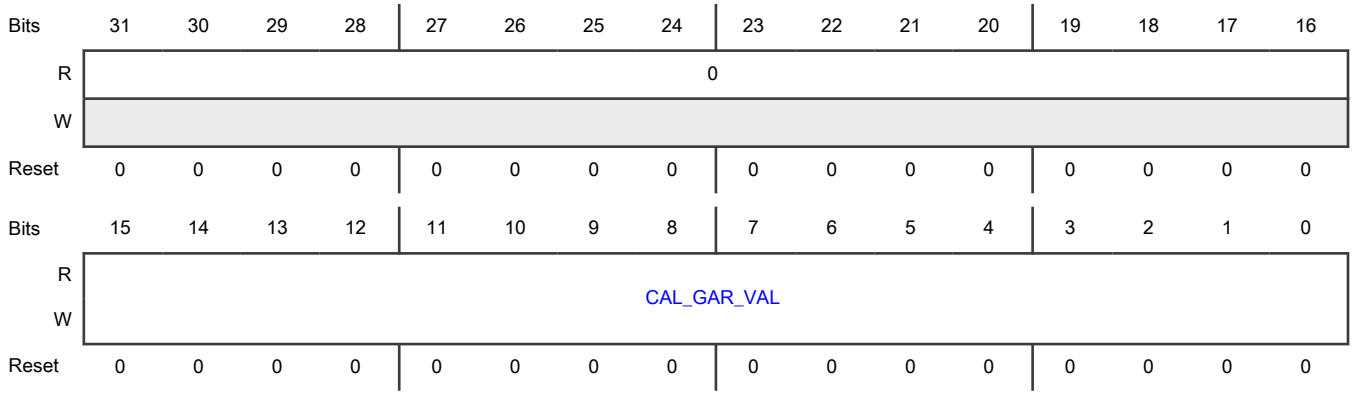
The A-side general calibration value registers (CAL_GAR0 - CAL_GAR33) are used to correct for linearity errors of the A-side converter. All 34 CAL_GAR registers contain calibration information that is automatically updated during the self calibration sequence. See Calibration for more information on completing ADC calibration steps.

The calibration values in the CAL_GAR registers affect the end conversion result by conditionally being subtracted from the conversion before the end result is transferred into the FIFOs. Calibration must be run each time the 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 and 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 write accessible when the ADC is disabled with CTRL[ADCEN] = 0b0. Note that access time when writing to these registers is larger than 3 ADC clock cycles. Wait states are inserted on the bus to meet synchronization timing to the associated CAL_GAR register.

Note that the values in each register is a 16-bit signed value but has non-uniform supported range of values. The valid range of each register is summarized in Calibration General A-Side Register valid ranges.

Diagram



Fields

Field	Function
31-16 —	Reserved This read-only field is reserved and always has the value 0.
15-0 CAL_GAR_VAL	Calibration General A Side Register Element The CAL_GAR_VAL is a 16-bit signed value. The valid range for each register in this array is non-uniform and summarized in Calibration General A-Side Register valid ranges.

42.6.23 Configuration 2 Register (CFG2)

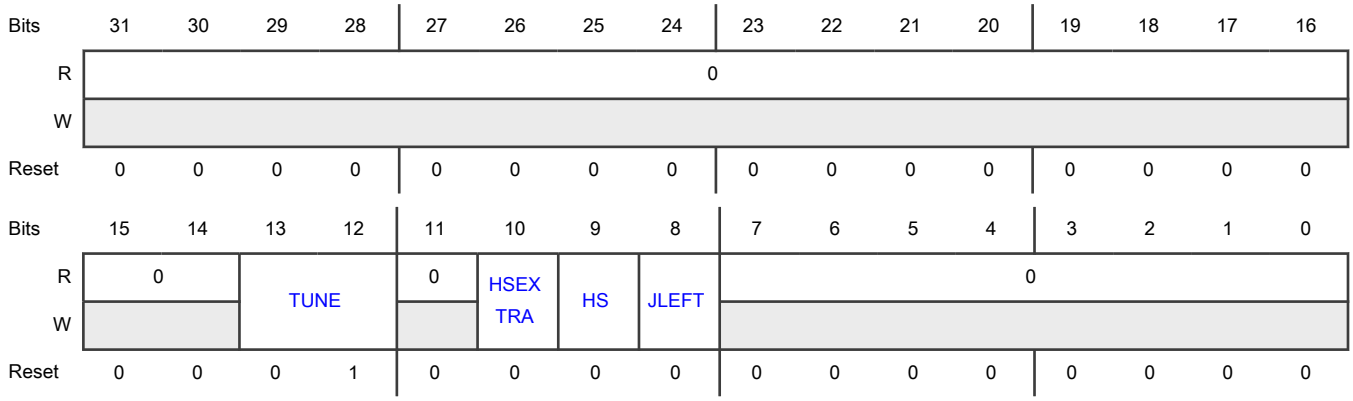
Offset

Register	Offset
CFG2	FF8h

Function

The Configuration 2 Register controls ADC functions that are common to all commands. The CFG2 cannot be changed while the CTRL[ADCEN] bit is set. Writes to CFG2 while CTRL[ADCEN] is set are ignored.

Diagram



Fields

Field	Function
31-14 —	Reserved This read-only field is reserved and always has the value 0.
13-12 TUNE	Tune Mode register TUNE provides some variability in how many cycles are needed to complete a conversion. Conversions can be terminated early at the expense of conversion resolution. See Cycles per conversion for more information on TUNE affect on conversion duration.
11 —	Reserved This read-only field is reserved and always has the value 0.
10 HSEXTRA	High Speed Extra register In high speed conversions (when CFG2[HS] bit set to 1), HSEXTRA provides a secondary control to add an additional ADCK cycle to the conversion. Setting to 1 is needed in a specific corner of operation where VDDA < TBD V and ADCK > TBD Hz. See Cycles per conversion for more information on HSEXTRA affect on conversion duration. 0b - No extra cycle added 1b - Extra cycle added
9 HS	High Speed Enable register Configures ADC for high speed conversions at the expense of higher operating current. When HS is set, conversions complete 2 or 3 ADCK cycles sooner (depending on the CFG2[HSEXTRA] setting) compared to conversion cycle counts when HS is clear. See Cycles per conversion for more information on HS affect on conversion duration. 0b - High speed conversion mode disabled 1b - High speed conversion mode enabled
8 JLEFT	Justified Left Enable register

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When JLEFT is set and the ADC is converting in 12-bit single-ended mode, conversion result stores to RESFIFOn[D] are in left-justify format. The formatting for the data in RESFIFOn[D] is summarized in Table 328. **NOTE** In JLEFT mode, CVL and CVH must be programmed in left-justify format to generate correct compare results.
7-0 —	Reserved This read-only field is reserved and always has the value 0.

Chapter 43
Low Power Comparator (LPCMP)

43.1 Chip-specific CMP information

Table 329. Reference links to related information

Topic	Related module	Reference
Full description	CMP	CMP
System memory map		Memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

43.1.1 Module instances

This device has two instances of the CMP module, CMP0, and CMP1.

43.1.2 Input connections

Table 330. Input connections

Index	CMP0_Positive	CMP0_Negative	CMP1_Positive	CMP1_Negative	Description
0	CMP0_IN0	CMP0_IN0	CMP1_IN0	CMP1_IN0	VDD
1	CMP0_IN1	CMP0_IN1	CMP1_IN1	CMP1_IN1	VDD
2	CMP0_IN2	CMP0_IN2	CMP1_IN2	CMP1_IN2	VDD
3	CMP0_IN3	CMP0_IN3	CMP1_IN3	CMP1_IN3	VDD
4	OPAMP0_OUT	Reserved	Reserved	Reserved	VDDA
5	OPAMP0_INT	Reserved	Reserved	Reserved	VDDA
6	Reserved	Reserved	Reserved	Reserved	Internal
7	CMP0 DAC	CMP0 DAC	CMP1 DAC	CMP1 DAC	VDD

43.1.3 Input references

In the DAC Control register, when:

- DCR[VRSEL]=0, VREFH0 is selected, which is VDD
- DCR[VRSEL]=1, VREFH1 is selected, which is VREFI

43.1.4 External trigger sources

RRCR0[RR_EXTTRG_SEL] controls the trigger source for the CMP. Note that only CMP1 supports multiple options. RRCR0[RR_EXTTRG_SEL] is not writeable in CMP0 and only source '0' is available.

Table 331. Input connections

CMP instance	Source number	RR_EXTTRG_SEL trigger source
CMP0	0	LPTMR0
CMP1	0	Reserved
	1	CTimer0_MAT0
	2	CTimer0_MAT2
	3	CTimer1_MAT0
	4	CTimer1_MAT2
	5	CTimer2_MAT0
	6	CTimer2_MAT2
	7	LPTMR0
	8	Reserved (LPTMR1)
	9	PWM0_SM0_MUX_TRIG0
	10	PWM0_SM0_MUX_TRIG1
	11	PWM0_SM1_MUX_TRIG0
	12	PWM0_SM1_MUX_TRIG1
	13	PWM0_SM2_MUX_TRIG0
	14	PWM0_SM2_MUX_TRIG1
	15	Reserved

43.2 Overview

The LPCMP module provides a circuit to compare two analog input voltages. It includes the following:

- A low power comparator (CMP)
- A DAC
- An analog mux (ANMUX)

See Block diagram for more information.

LPCMP 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.

43.2.1 Block diagram

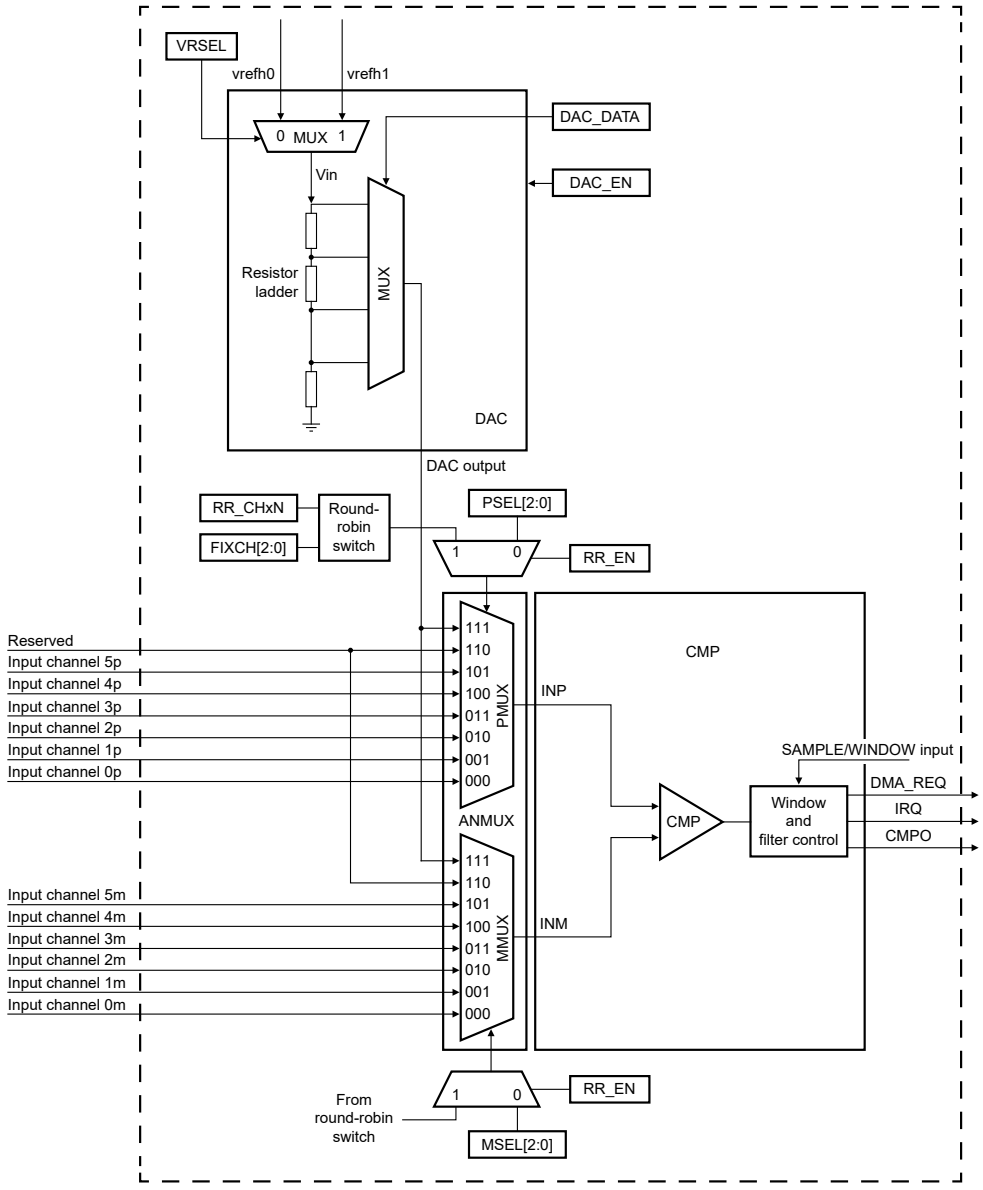


Figure 180. Block diagram

43.2.2 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.
  - COUT (comparator output) rising, falling or both edges closes the window.
  - COUT 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
- Supports Round Robin Trigger mode
- 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

43.3 Functional description

43.3.1 Functional block diagram

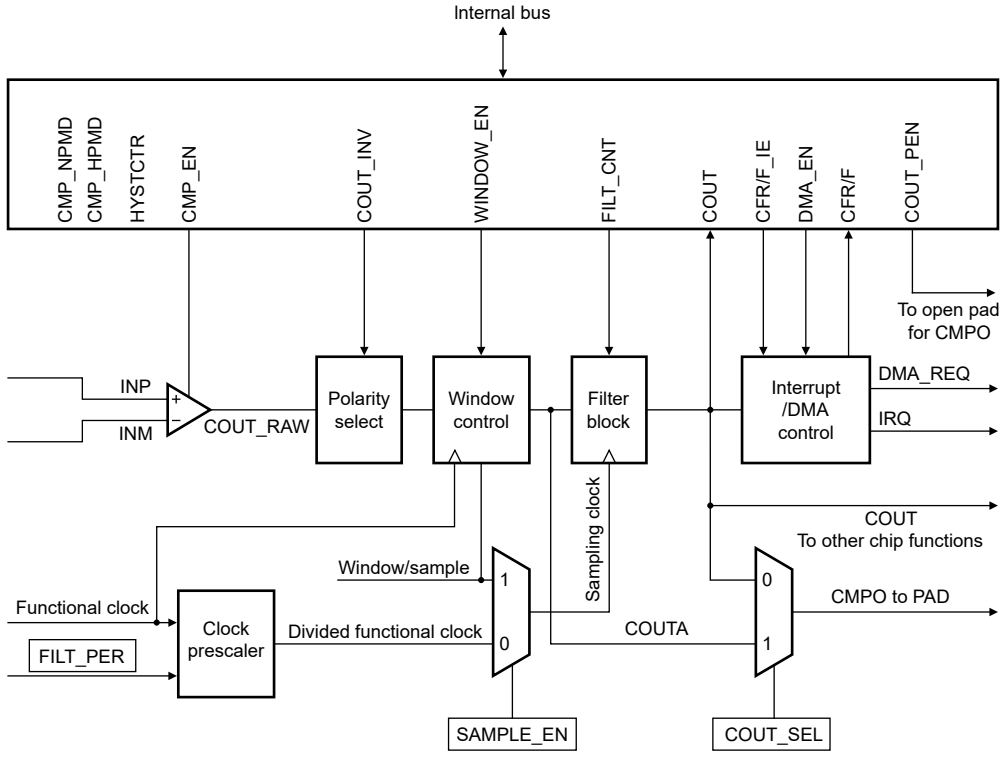


Figure 181. Functional block diagram

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 functional 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 functional 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 COUT'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.

43.3.2 Round-robin trigger mode

You can enable Round-Robin Trigger mode by setting RRCR0[RR_EN] to 1. A trigger event initiates a comparison sequence. The next trigger event should not occur before the current sequence completes.

RRCR1[FIXP] and RRCR1[FIXCH] select the reference channel for the plus side mux or the minus side mux. RRCR1[RR_CHnEN] selects active channels.

When a trigger comes, the analog comparator enables. After the comparison sequence completes, the analog comparator disables again. RRCR0[RR_INITMOD] controls the analog stabilization time.

NOTE

RR_INITMOD*round robin clock period must be longer than the initialization delay specified in the Comparator and 8-bit DAC electrical specifications section of LPCMP datasheet.

After the stabilization process completes, the round robin manner comparison sequence begins. Sample the comparison result for the selected active channel after RRCR0[RR_NSAM] defines the configurable number of operation clocks.

There are (RR_SAMPLE_CNT + 1) samples for each channel. With these samples, at least (RR_SAMPLE_THRESHOLD + 1) sampled "1" causes the channel's final sample result to be "1"; otherwise, it is "0". Compare the final result with the pre-programmed value.

After all the active channels are sampled/compared, if the comparison result changes from its pre-programmed state, the corresponding flag in RRSR[RR_CHnF] is set. Write to RRCSR[RR_CHnOUT] to configure the pre-programmed state for each channel. Update RRCSR[RR_CHnOUT] to store the last comparison result for each channel. If any flag in RRSR[RR_CHnF] sets, CSR[RRF] also sets. If IER[RRF_IE] sets, an asynchronous interrupt asserts. Note that these flags do not support generating a DMA transfer event.

The following diagram shows the basic flow of this mode. In the diagram, RRCR1[RR_CH1EN], RRCR1[RR_CH3EN], and RRCR1[RR_CH4EN] are 1, so channels #1, #3, and #4 are selected for round-robin depending on their priority setting. RRCR0[RR_NSAM] sets to 2'b01, so you can sample one clock later the comparison result of the selected channel. After you compare the channel #4, the result is sampled, and round-robin ends. If any of the comparison results from channel #1, #3, or #4 changed from their programmed value (written to RRCSR[RR_CH1OUT], RRCSR[RR_CH3OUT], and RRCSR[RR_CH4OUT]), generates an interrupt. Software can then poll RRSR[RR_CHnF] to see which channel input(s) changed value.

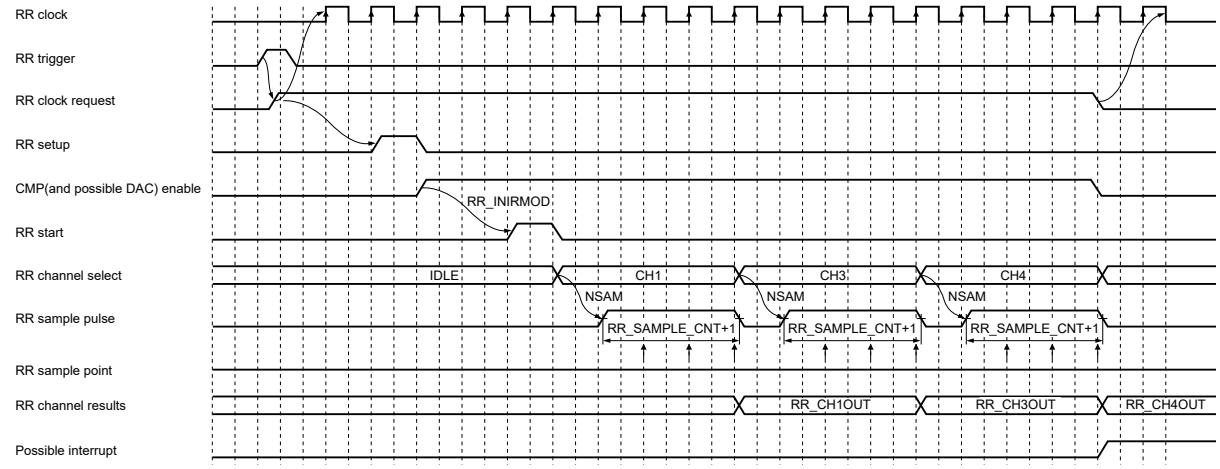


Figure 182. Trigger mode

The table below shows the channel decode in both Functional mode and Trigger mode. Other cases not in the table are illegal.

Table 332. CMP channel decode in functional mode and round-robin trigger mode

Mode	RR_EN	PSEL[2:0]	MSEL[2:0]	FIXP	FIXCH[2:0]	RR_CHxN	INP	INM	CMP Behavior
Functional mode	0	0 to 7	0 to 7	x[1]	x	x	Channel decoded from PSEL[2:0]	Channel decoded from MSEL[2:0]	Compare the channel 0 to 7 with channel 0 to 7[2]
Trigger mode	1	x	x	0	0 to 7	0 to 7	Channel fixed by FIXCH[2:0]	Channel sweep (RR_CHxN)	Sweep the channel 0 to 7 with a fixed channel (0 to 7)
		x	x	1	0 to 7	0 to 7	Channel sweep (RR_CHxN)	Channel fixed by FIXCH[2:0]	Sweep the channel 0 to 7 with a fixed channel (0 to 7)

1. "x" means "don't care"
2. PSEL should not be same as MSEL.

43.3.3 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.

43.3.3.1 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 functional clock to drive the low-pass filter, it samples COUTA every CCR1[FILT_PER] functional 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.

43.3.3.2 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 334 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.

43.3.4 Low power mode operation

Below table introduces the mode of operation of lower power.

Table 333. Low power mode operation

Mode of operation	Description
Wait	LPCMP can operate normally with window and filter function.
Deep Sleep	LPCMP can operate only in Continuous mode or Round-robin trigger mode.

43.3.5 Functional modes

You can combine the comparator window and filter features as shown in the following table.

Table 334. 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		

Table continues on the next page...

Table 334. Functional modes (continued)

Mode #	CMP_EN	WINDOW_EN	SAMPLE_EN	FILT_CNT	FILT_PER	Operation	Maximum latency[1]
3A	1	0	1	0x01	X	See the Sampled, non-filtered mode (#3A and #3B).	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		$T_{PD} + (FILT\_PER * T_{per}) + T_{per}$
4A	1	0	1	> 0x01	X	See the Sampled, filtered mode (#4A and #4B).	$T_{PD} + (FILT\_CNT * T_{SAMPLE}) + T_{SAMPLE}$
4B	1	0	0	> 0x01	> 0x00		$T_{PD} + (FILT\_CNT * FILT\_PER \times T_{per}) + T_{per}$
5A	1	1	0	0x00	X	See the Windowed mode (#5A and #5B).	$T_{PD} + 2T_{per}$
5B	1	1	0	X	0x00		
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).	$T_{PD} + (FILT\_CNT * FILT\_PER \times T_{per}) + 3T_{per}$
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 functional clock.

43.3.5.1 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].

43.3.5.2 Continuous mode (#2A and #2B)

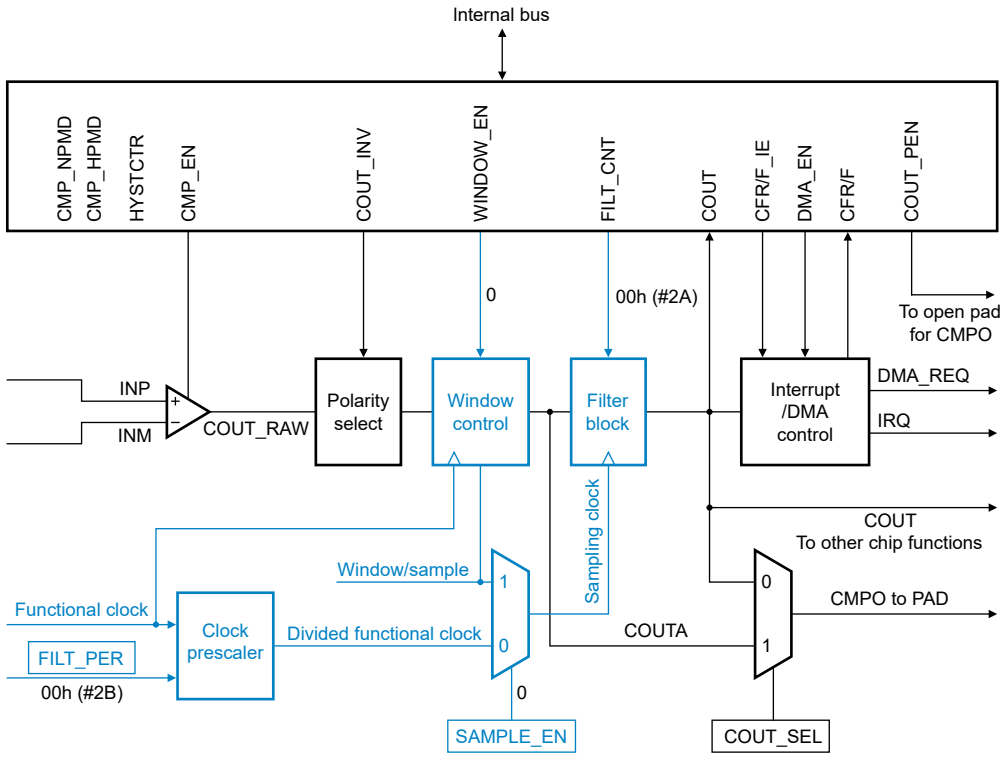


Figure 183. Comparator operation in continuous mode

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.

43.3.5.3 Sampled, non-filtered mode (#3A and #3B)

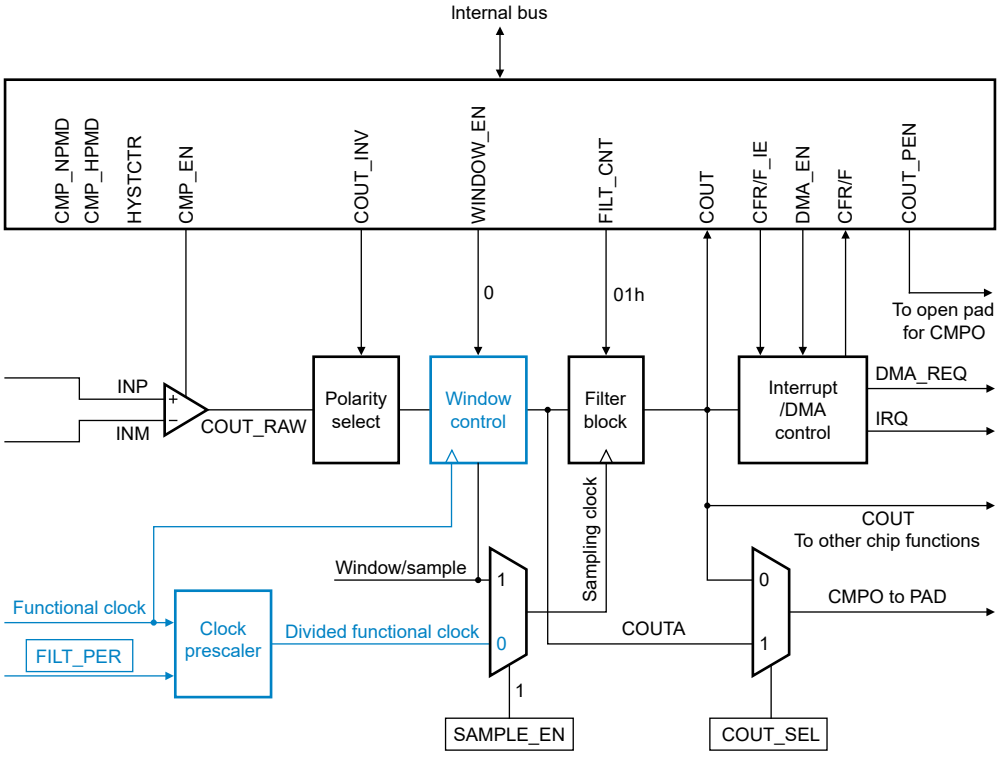


Figure 184. Sampled, non-filtered (#3A): sampling point externally driven

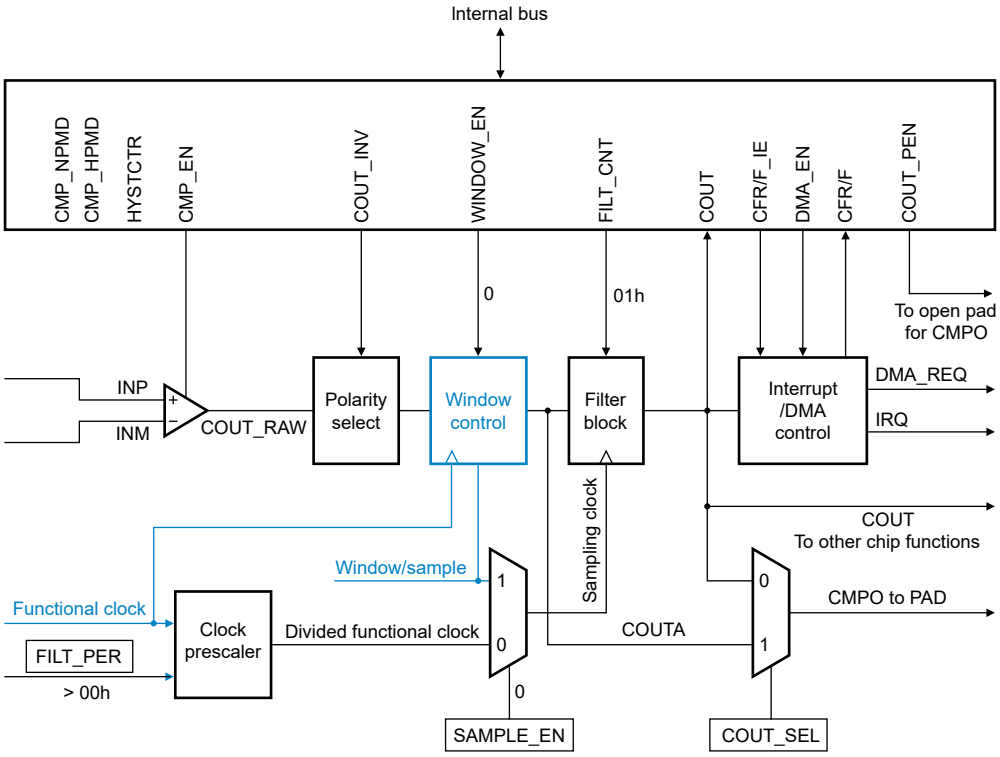


Figure 185. Sampled, Non-Filtered (#3B): sampling interval internally derived

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.

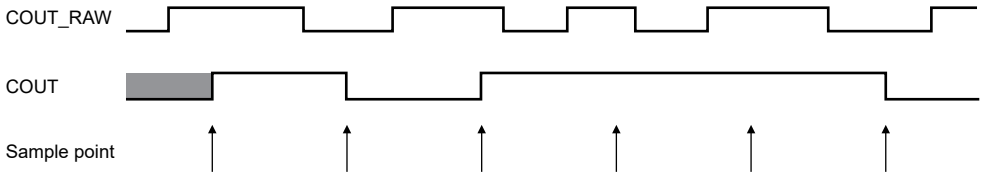


Figure 186. Sampled, Non-Filtered mode timing diagram

43.3.5.4 Sampled, filtered mode (#4A and #4B)

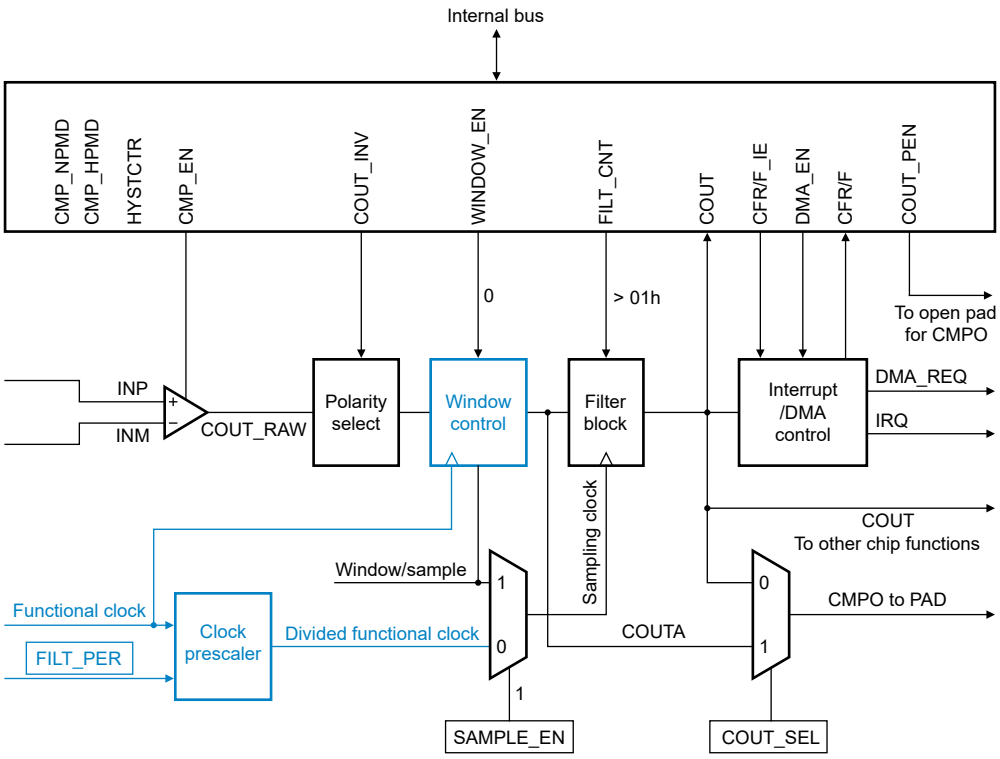


Figure 187. Sampled, filtered (#4A): sampling point externally driven

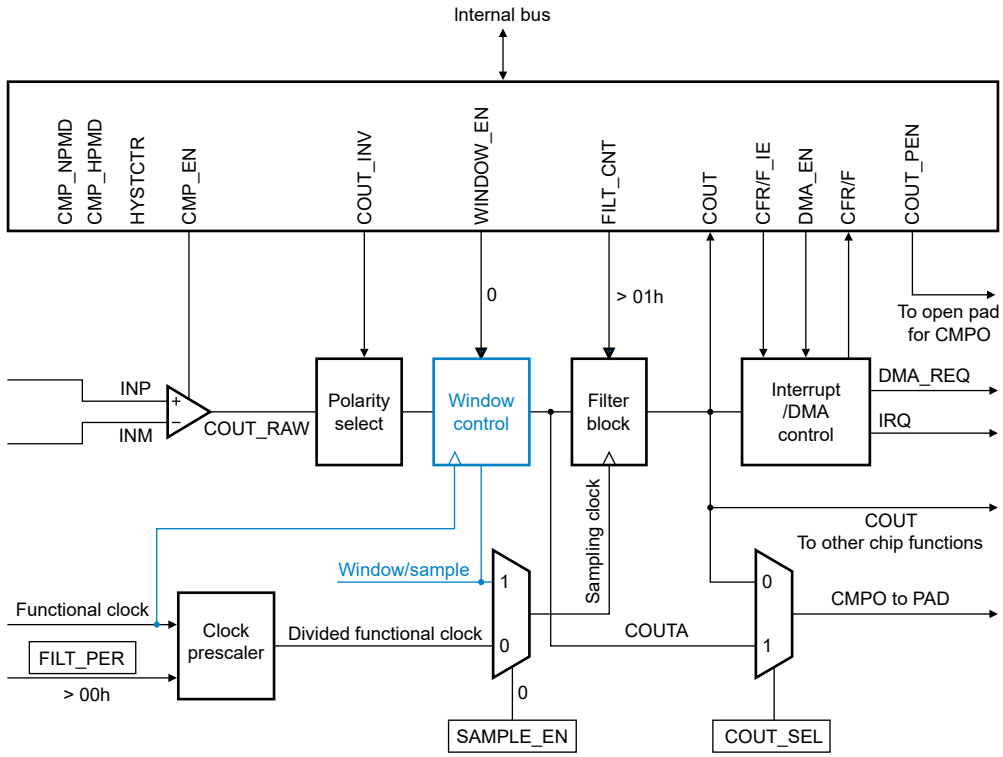


Figure 188. Sampled, filtered (#4B): sampling point internally derived

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.

43.3.5.5 Windowed mode (#5A and #5B)

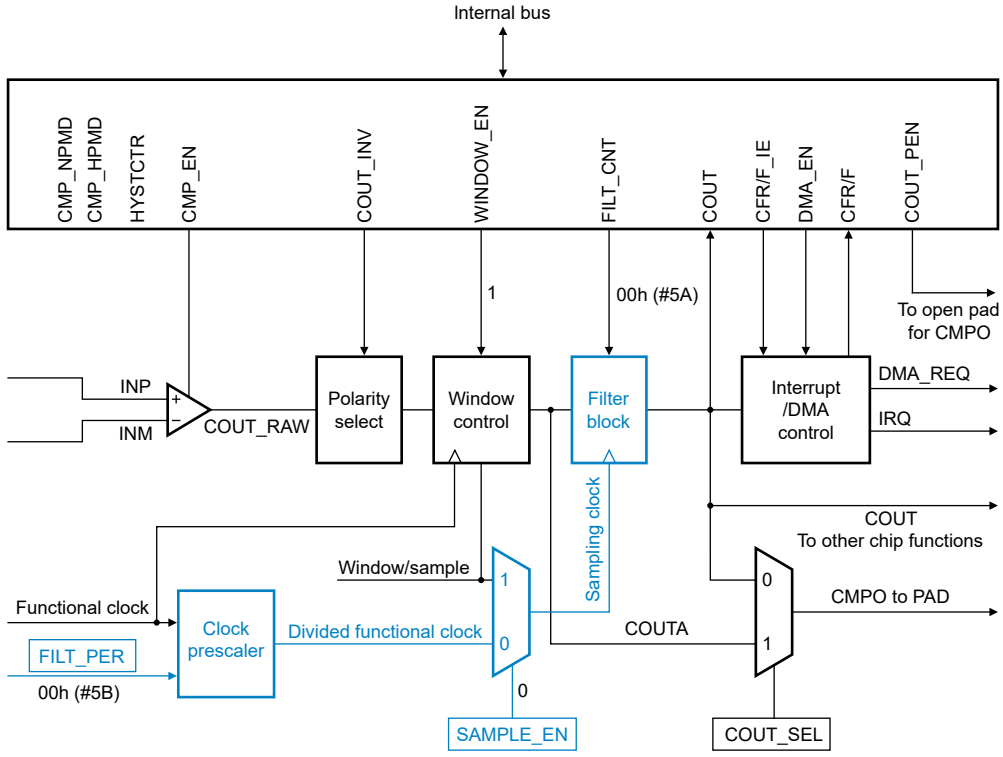


Figure 189. Windowed mode

The functional 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.

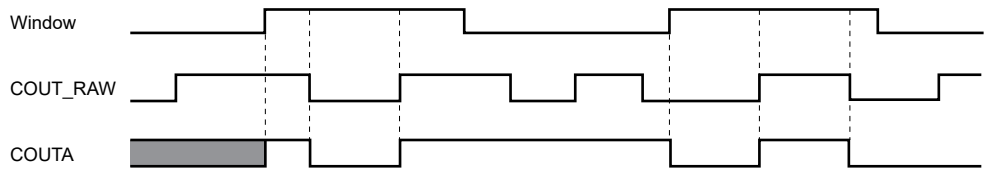


Figure 190. Windowed mode timing diagram

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.

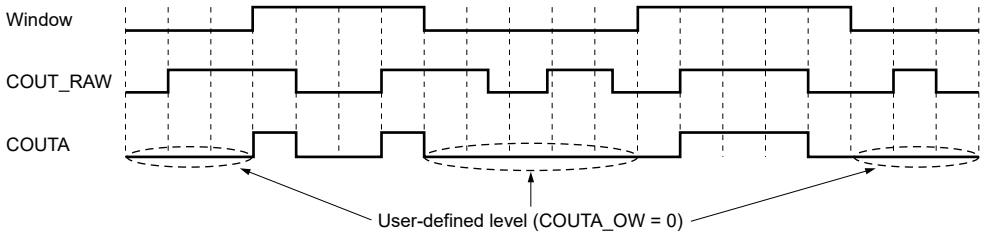


Figure 191. Windowed mode timing diagram with user defined value 0 outside window

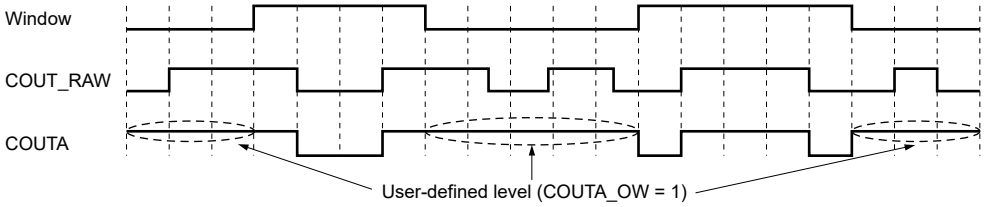


Figure 192. Windowed mode timing diagram with user defined value 1 outside window

If CCR1[WINDOW_CLS] becomes 1, you can define the COUT 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 COUT rising edge closes the internal window.

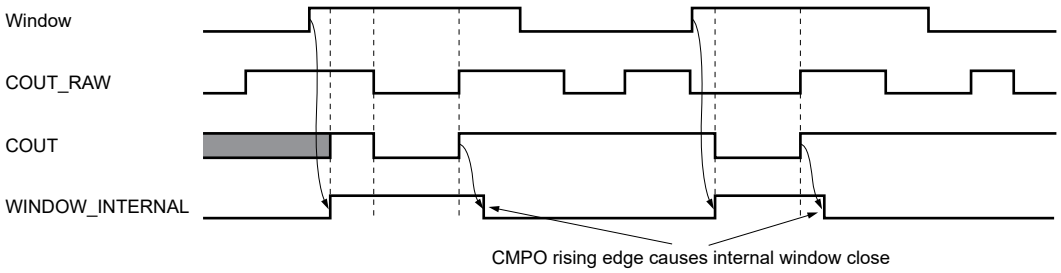


Figure 193. Windowed mode timing diagram with COUT rising edge close window

The following figure shows that if CCR1[WINDOW_INV] becomes 1, you can invert the window signal before you use it.

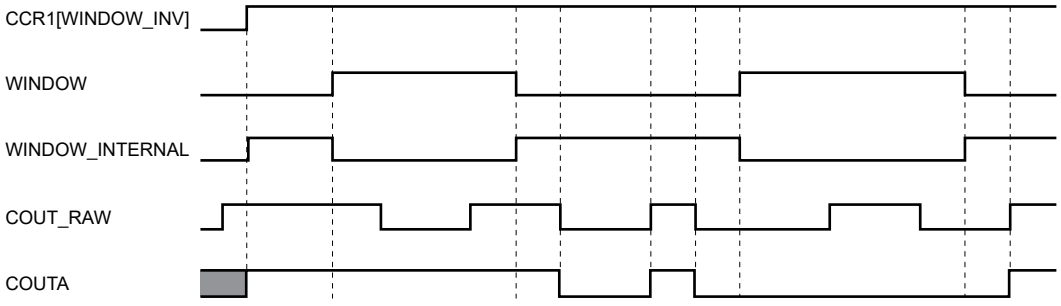


Figure 194. Windowed mode timing diagram with window signal inverted

43.3.5.6 Windowed/Resampled mode (#6)

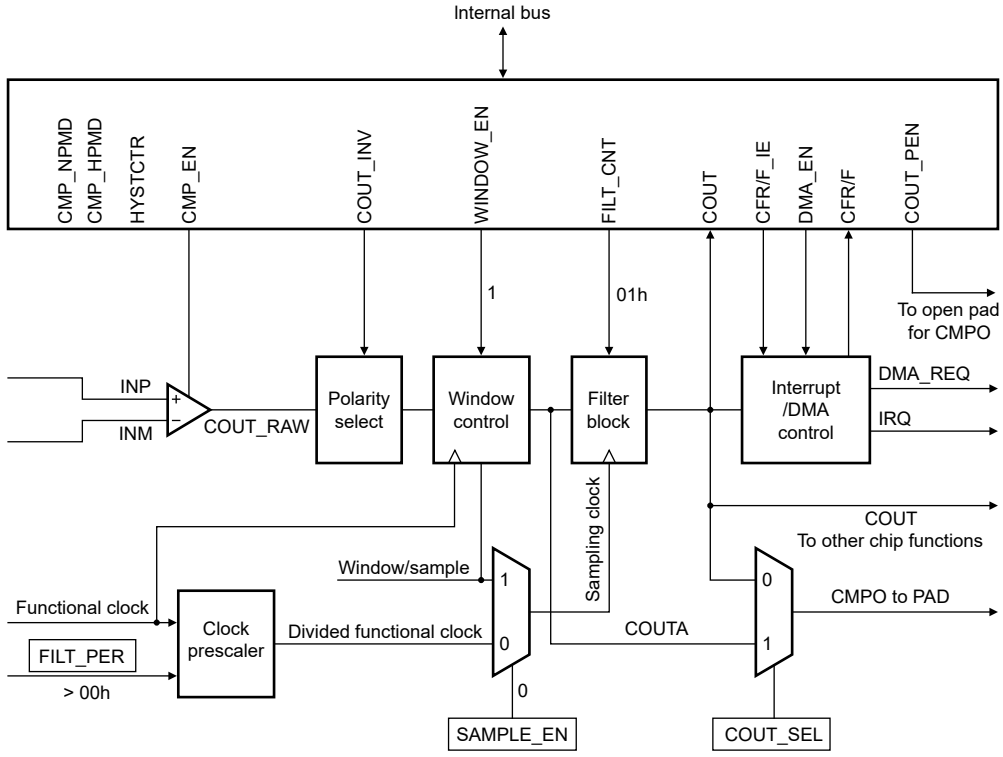


Figure 195. Windowed/Resampled mode

This mode of operation results in an unfiltered string of comparator samples where CCR1[FILT_PER] and the functional 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 190, 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.

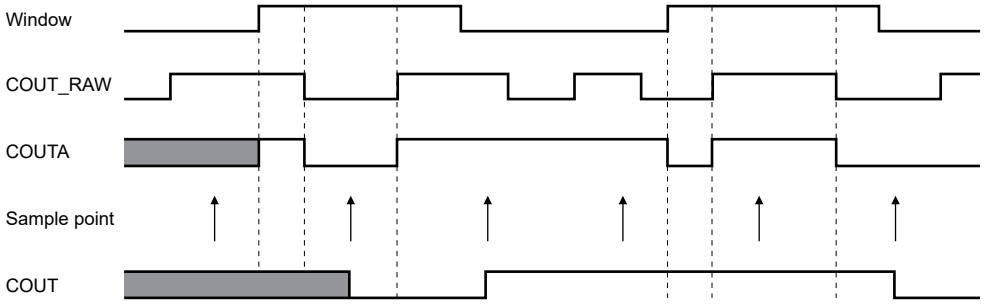


Figure 196. Windowed/Resampled mode operation

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.

43.3.5.7 Windowed/Filtered mode (#7)

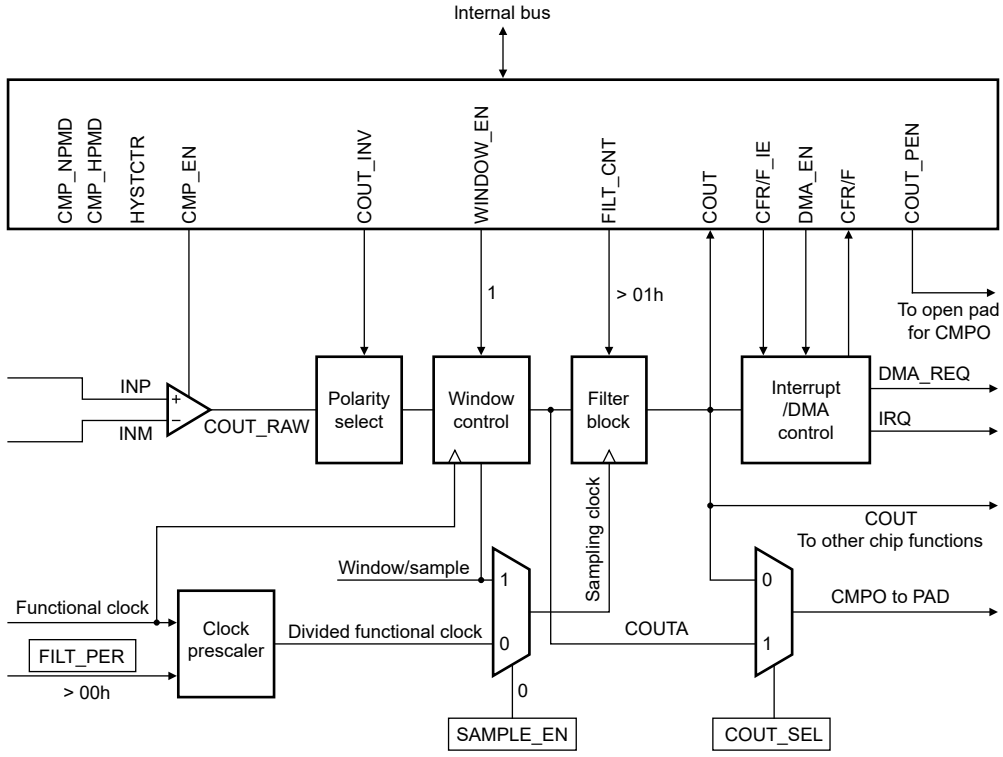


Figure 197. Windowed/Filtered mode

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.

43.3.6 DMA

After DMA is enabled by writing 1 to CCR1[DMA_EN] and interrupt is enabled 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 Deep Sleep modes if CCR0[CMP_STOP_EN] becomes 1.

A DMA transfer request wakes up the system from Deep Sleep mode. After completing the data transfer, the system go back again to Deep Sleep mode. See the DMA chapters in this document for more information on the asynchronous DMA function.

43.3.7 Clocking

LPCMP requires the following clocks to operate:

Table 335. LPCMP clocks

Type of clock	Description
Bus	Controls the access to LPCMP registers.

Table continues on the next page...

Table 335. LPCMP clocks (continued)

Type of clock	Description
Functional	Controls window/filter function. LPCMP configures CCR1[FUNC_CLK_SEL] to select one from the four functional clock sources. See the Chip-specific LPCMP information for more on the functional clock source information.
Round-robin clock (RCLK)	Controls Round-robin trigger mode. LPCMP configures CCR1[RR_CLK_SEL] to select one from the four round-robin clock sources. See the Chip-specific LPCMP information for more on the round robin clock source information.

43.3.8 Resets

The global chip reset signal resets LPCMP.

43.3.9 Interrupts

After the corresponding IER becomes 1, CSR[CFR], CSR[CFF], and CSR[RRF] can generate an interrupt, assuming that CCR1[DMA_EN] is not 1. You can clear either the flag or IER to deassert the interrupt.

43.4 External signal descriptions

Below table introduces external signals.

Table 336. 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

43.5 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 Comparator and 8-bit DAC electrical specifications section of LPCMP datasheet for more information on propagation delay and power-up delay). Table 334 specifies the delay that the windowing and filter function causes.

During operation, you must always consider the propagation delay of the selected data paths. It can take many functional 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.

43.6 Application information

43.6.1 Round-robin trigger mode programing recommendation

Configure the Round-robin trigger mode as follows:

1. Configure RRCR0[RR_CLK_SEL] to select the RR clock source.
2. 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].
3. 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 LPCMP datasheet for more information.
4. Configure RRCR0[RR_SAMPLE_CNT] and RRCR0[RR_SAMPLE_THRESHOLD].
5. Configure RRCR0[RR_TRG_SEL] to select the trigger from internal or external.
6. Enable RRCR2[RR_TIMER_EN] and configure RRCR2[RR_TIMER_RELOAD] according to the round robin clock frequency, if using an internal trigger.
7. Configure RRCR1[FIXP] to select the fixed port of CMP and RRCR1[FIXCH] to select the fixed channel.
8. Configure channels for comparison by RRCR1[RR_CHnEN].
9. Write RRCSR[RR_CHnOUT] to define the pre-set state of channel n.
10. Clear channel flags RRSR[RR_CHnF].
11. Enable round robin interrupt by IER[RRF_IE] (disable IER[CFR_IE] and IER[CFF_IE]).
12. Enable round-robin trigger mode by setting RRCR0[RR_EN] to 1.

43.6.2 Round-robin clock (RCLK) frequency requirement

(1) RCLK high frequency limit

RCLK high frequency limit depends on two facts:

1. The analog CMP and DAC initialization time (see the chip data sheet for more information on the initialization time.)
   - RRCR0[RR_INITMOD] provides a maximum 63 RCLK cycles for the analog CMP and DAC initialization.
   - RCLK must be slow to assure: $63 * (1/f_{RCLK}) > T_{initialization}$, where f_{RCLK} is in MHz, and $T_{initialization}$ is in microsecond.
   - so $f_{RCLK} < 63 / T_{initialization}$
   - Example: $T_{initialization} = 40$ µs, then f_{RCLK} should be smaller than 1.575 MHz.
2. The analog CMP propagation delay (see the Comparator and 8-bit DAC electrical specifications section of LPCMP datasheet for more information on the CMP propagation delay.)
   - RRCR0[RR_NSAM] provides a maximum 4 RCLK cycles for the analog CMP propagation delay.
   - RCLK must be slow to assure: $4 * (1/f_{RCLK}) > T_{propagation}$, where f_{RCLK} is in MHz, and $T_{propagation}$ is in microsecond.
   - $f_{RCLK} < 4 / T_{propagation}$
   - Example: $T_{propagation} = 0.1$ µs, then f_{RCLK} must be smaller than 40 MHz.

(2) RCLK low frequency limit

In theory, RCLK frequency has no low limit. But the lower the RCLK frequency, the longer the scan time. Therefore, the lower limit of the RCLK frequency depends on the system application.

43.7 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.

43.7.1 LPCMP memory map

CMP0 base address: 400B_1000h

CMP1 base address: 400B_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0001h
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
24h	Round Robin Control Register 0 (RRCR0)	32	RW	0000_0000h
28h	Round Robin Control Register 1 (RRCR1)	32	RW	0000_0000h
2Ch	Round Robin Control and Status (RRCSR)	32	RW	0000_0000h
30h	Round Robin Status (RRSR)	32	RW	0000_0000h
38h	Round Robin Control Register 2 (RRCR2)	32	RW	0000_0000h

43.7.2 Version ID (VERID)

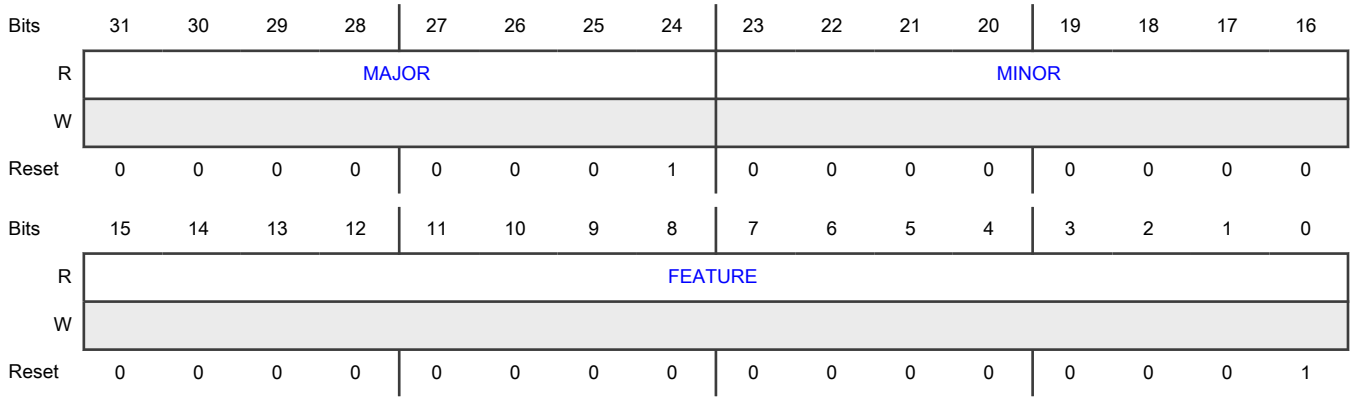
Offset

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

43.7.3 Parameter (PARAM)

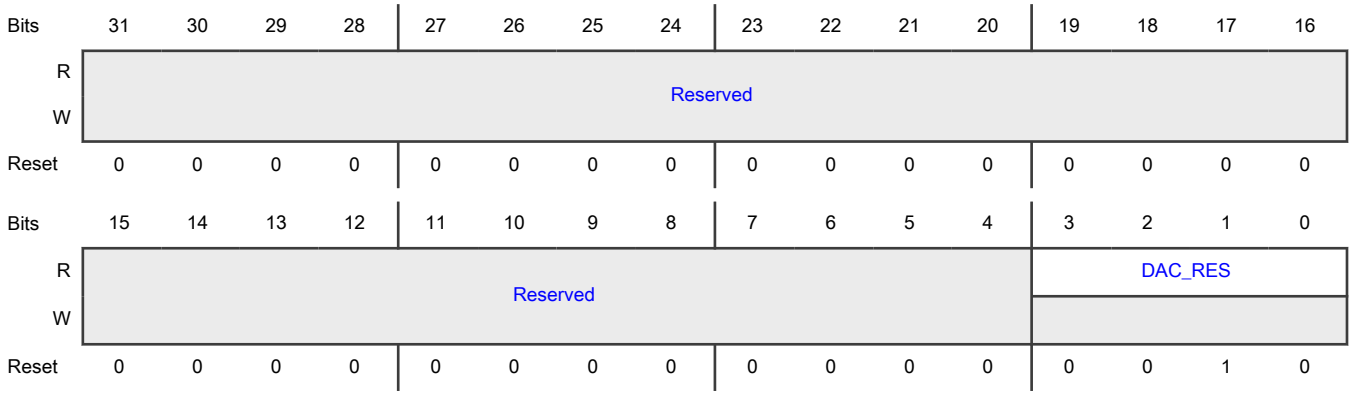
Offset

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 supported DAC resolutions. **NOTE** All other bit field values are reserved. 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

43.7.4 Comparator Control Register 0 (CCR0)

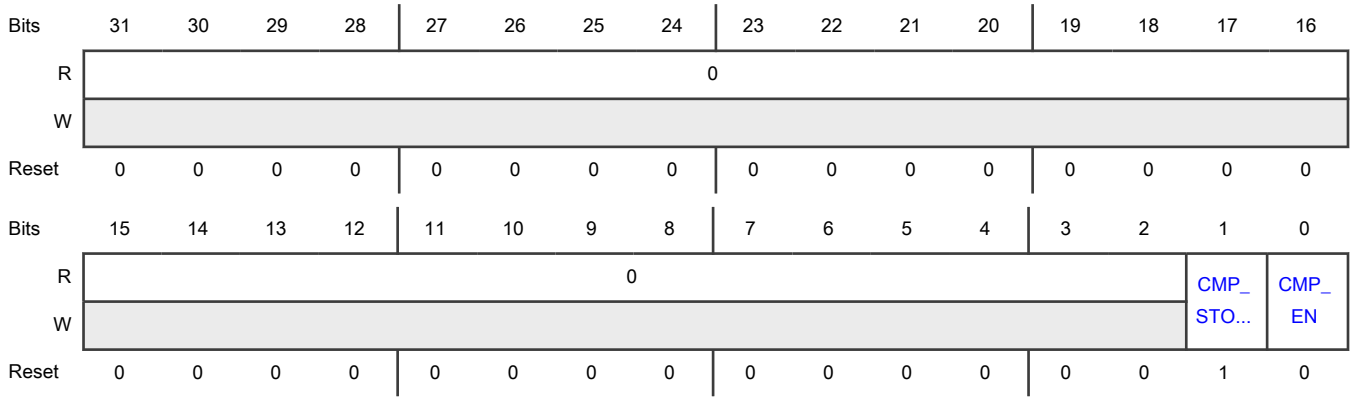
Offset

Register	Offset
CCR0	8h

Function

Contains configuration options for enabling the analog comparator.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 CMP_STOP_EN	Comparator Deep Sleep Mode Enable Enables the analog comparator or the DAC when the module is in Deep Sleep mode. **NOTE** This field has no effect in Round-robin Trigger mode. 0b - Disables 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 - Disable (The analog logic remains off and consumes no power.) 1b - Enable

43.7.5 Comparator Control Register 1 (CCR1)

Offset

Register	Offset
CCR1	Ch

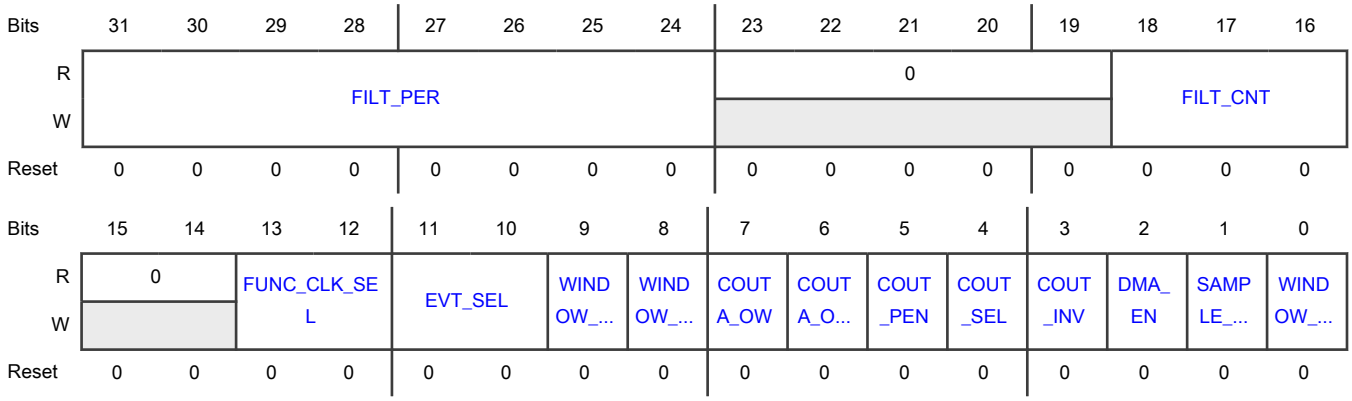
Function

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 functional 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
13-12	Functional Clock Source Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
FUNC_CLK_SEL	Selects which clock source is used for the functional clock. See the Chip-specific LPCMP information for more on the functional clock source information. 00b - Select functional clock source 0 01b - Select functional clock source 1 10b - Select functional clock source 2 11b - Select functional clock source 3
11-10 EVT_SEL	COUT Event Select Selects which COUT signal edge (rising, falling, or both) defines a COUT event. **NOTE** Valid only in Windowing mode. 00b - Rising edge 01b - Falling edge 1xb - Both edges
9 WINDOW_CLS	COUT Event Window Close Enables a COUT event (defined as a COUT rising edge, falling edge, or both) to close an active window. See EVT_SEL to configure the COUT 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 - COUT event cannot close the window 1b - COUT 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_OW Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
COUTA_OWEN	Enables the COUTA signal value to be defined by COUTA_OW when the window is closed. **NOTE** Valid only in Windowing mode. 0b - COUTA holds the last sampled value. 1b - Enables the COUTA signal value to be defined by COUTA_OW.
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. **NOTE** COUT_INV has no effect in Trigger mode. 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 - Disable 1b - Enable
1 SAMPLE_EN	Sampling Enable Enables Sampling mode. 0b - Disable 1b - Enable
0 WINDOW_EN	Windowing Enable Enables Windowing mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Valid only when SAMPLE_EN = 0. 0b - Disable 1b - Enable

43.7.6 Comparator Control Register 2 (CCR2)

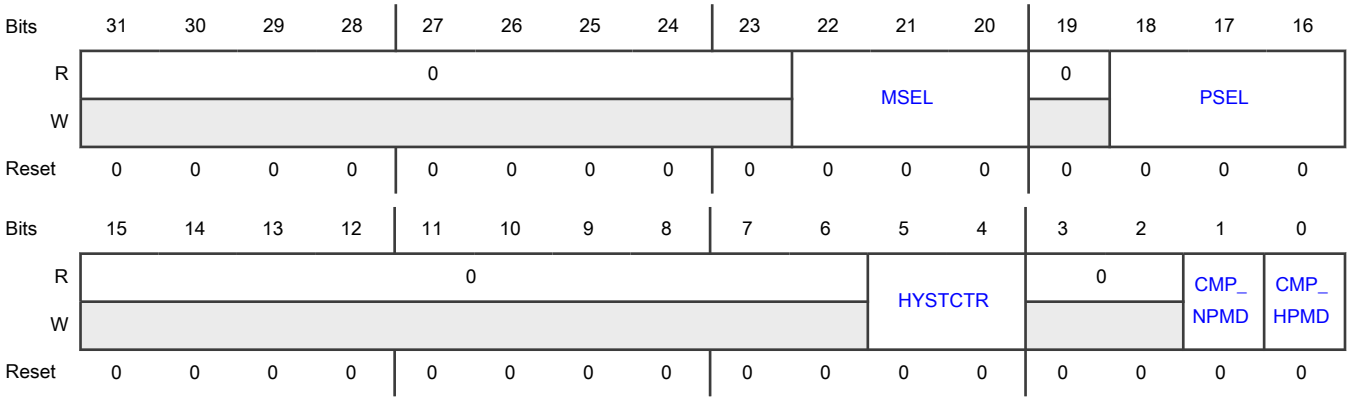
Offset

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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE MSEL has no effect in Trigger mode. 000b - Input 0m 001b - Input 1m 010b - Input 2m 011b - Input 3m 100b - Input 4m 101b - Input 5m 110b - Reserved 111b - Internal DAC output
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. **NOTE** PSEL has no effect in Trigger mode. 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. **NOTE** This applies to the comparator hard block.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - Level 0: Analog comparator hysteresis 0 mV. 01b - Level 1: Analog comparator hysteresis 10 mV. 10b - Level 2: Analog comparator hysteresis 20 mV. 11b - Level 3: Analog comparator hysteresis 30 mV.
3-2 —	Reserved
1 CMP_NPMD	CMP Nano Power Mode Select Enables Nano Power mode for the comparator. 0b - Disables CMP Nano power mode. CCR2[CMP_HPMD] determines the mode for the comparator. 1b - Enables CMP Nano power mode.
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). 0b - Low power (speed) comparison mode 1b - High power (speed) comparison mode

43.7.7 DAC Control (DCR)

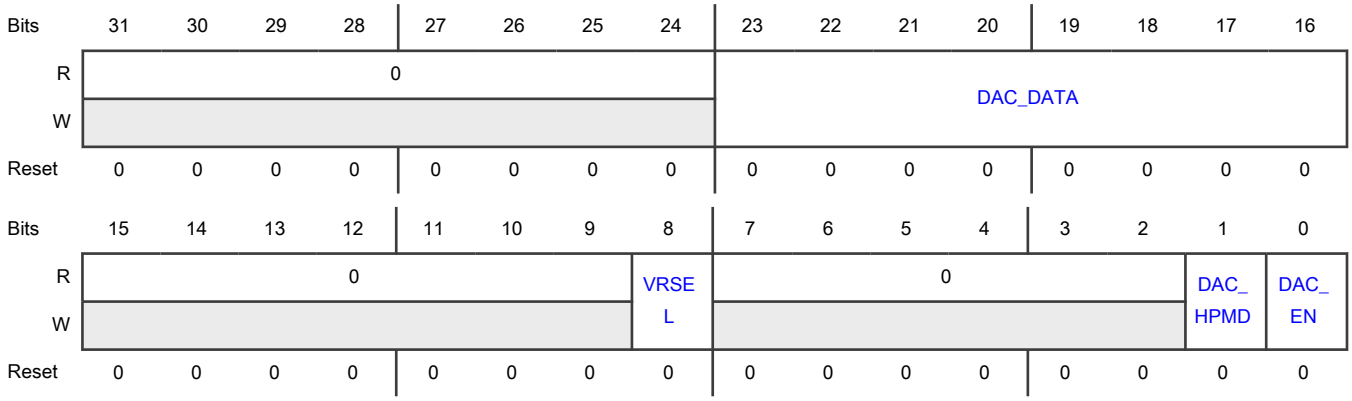
Offset

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 by configuring the value of DAC_DATA. The DACO ranges from Vin/256 to Vin. **NOTE** DACO = (Vin/256) * (DAC_DATA + 1)
15-9 —	Reserved
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 Enables the DAC high power mode. 0b - Disable 1b - Enable
0 DAC_EN	DAC Enable Enables the DAC. When disabled, power-down the DAC to conserve power.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable

43.7.8 Interrupt Enable (IER)

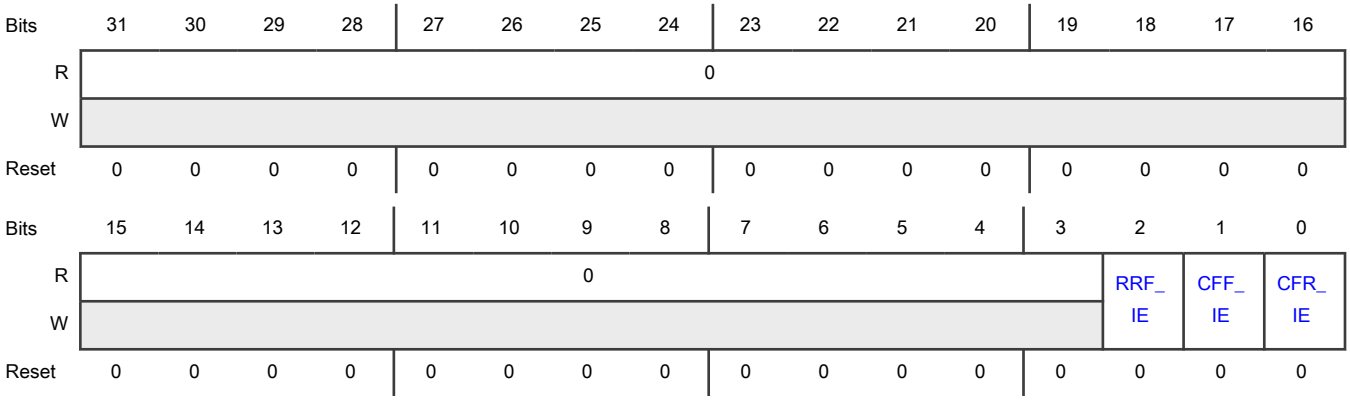
Offset

Register	Offset
IER	1Ch

Function

Provides enable fields for the comparator and round-robin flag interrupts.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 RRF_IE	Round-Robin Flag Interrupt Enable Enables or disables the round-robin flag interrupt. 0b - Disables the round-robin flag interrupt. 1b - Enables the round-robin flag interrupt when the comparison result changes for a given channel.
1	Comparator Flag Falling Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CFF_IE	Enables or disables the comparator flag falling interrupt. 0b - Disables the comparator flag falling interrupt. 1b - Enables the comparator flag falling interrupt when CFF is set.
0 CFR_IE	Comparator Flag Rising Interrupt Enable Enables or disables the comparator flag rising interrupt. 0b - Disables the comparator flag rising interrupt. 1b - Enables the comparator flag rising interrupt when CFR is set.

43.7.9 Comparator Status (CSR)

Offset

Register	Offset
CSR	20h

Function

Indicates comparator status, including COUT, CFF, CFR, and RRF.

NOTE

LPCMP may output a glitch and affect the value of CSR[CFF] and CSR[CFR] at the moment of enabling CMP. In order to ensure correctness, it is recommended to write one to clear (W1C) CSR[CFF] and CSR[CFR] before further configuring CMP.

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					RRF	CFF	CFR
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	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.
7-3 —	Reserved
2 RRF	Round-Robin Flag Detects when any channel's last comparison result is different from the pre-set value in Trigger mode. Write 1 to clear this field. This field clears when CCR0[CMP_EN] or RRCR0[RR_EN] is not 1. 0b - Not detected 1b - Detected
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

43.7.10 Round Robin Control Register 0 (RRCR0)

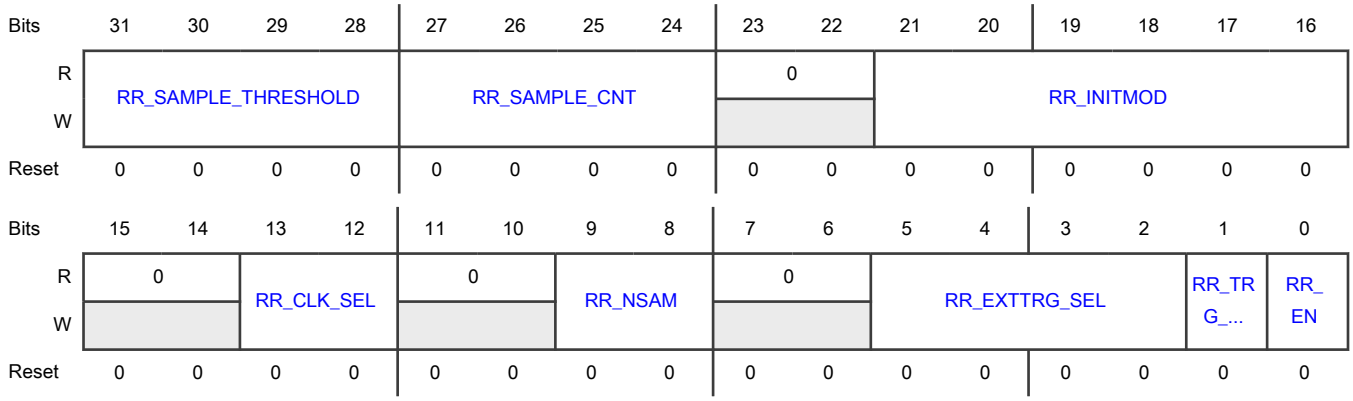
Offset

Register	Offset
RRCR0	24h

Function

Contains configuration options for the round-robin operation, such as enabling it and specifying the initialization delay.

Diagram



Fields

Field	Function
31-28 RR_SAMPLE_THRESHOLD	Sample Time Threshold Specifies that for one channel, when (RR_SAMPLE_THRESHOLD+1) sample results are "1", the final result is "1"; otherwise the final result is "0". **NOTE** This field must not be larger than RRCR0[RR_SAMPLE_CNT]. 0000b - At least 1 sampled "1", the final result is "1" 0001b - At least 2 sampled "1", the final result is "1" 0010b - At least 3 sampled "1", the final result is "1" 0011b - At least 4 sampled "1", the final result is "1" 0100b - At least 5 sampled "1", the final result is "1" 0101b - At least 6 sampled "1", the final result is "1" 0110b - At least 7 sampled "1", the final result is "1" 0111b - At least 8 sampled "1", the final result is "1" 1000b - At least 9 sampled "1", the final result is "1" 1001b - At least 10 sampled "1", the final result is "1" 1010b - At least 11 sampled "1", the final result is "1" 1011b - At least 12 sampled "1", the final result is "1" 1100b - At least 13 sampled "1", the final result is "1" 1101b - At least 14 sampled "1", the final result is "1" 1110b - At least 15 sampled "1", the final result is "1" 1111b - At least 16 sampled "1", the final result is "1"
27-24	Number of Sample for One Channel Specifies the number of samples for one channel.

Table continues on the next page...

Table continued from the previous page...

Field	Function
RR_SAMPLE_CNT	0000b - 1 samples 0001b - 2 samples 0010b - 3 samples 0011b - 4 samples 0100b - 5 samples 0101b - 6 samples 0110b - 7 samples 0111b - 8 samples 1000b - 9 samples 1001b - 10 samples 1010b - 11 samples 1011b - 12 samples 1100b - 13 samples 1101b - 14 samples 1110b - 15 samples 1111b - 16 samples
23-22 —	Reserved
21-16 RR_INITMOD	Initialization Delay Modulus Specifies the number of round-robin clock cycles that determines the comparator and DAC initialization delay specified in the chip datasheet. Calculate the initialization delay as RR_INITMOD * (round-robin clock period). For example, if the initialization delay is 80us and the round-robin clock is 100kHz, program RR_INITMOD to be 80us/10us = 8. 00_0000b - 63 cycles (same as 111111b) 00_0001b-11_1111b - 1 to 63 cycles
15-14 —	Reserved
13-12 RR_CLK_SEL	Round Robin Clock Source Select Selects which clock source is used for the Round Robin clock. See the chip-specific LPCMP information for more on the Round Robin clock source information. 00b - Select Round Robin clock Source 0 01b - Select Round Robin clock Source 1

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>10b - Select Round Robin clock Source 2
11b - Select Round Robin clock Source 3</td></tr>
<tr><td>11-10
—</td><td>Reserved</td></tr>
<tr><td>9-8
RR_NSAM</td><td>Number of Sample Clocks
Specifies the number of the round-robin clock cycles to wait after scanning the active channel before sampling the channel's comparison result. After the next cycle of the round-robin clock, the sampling takes place RR_NSAM clocks later.
00b - 0 clock
01b - 1 clock
10b - 2 clocks
11b - 3 clocks</td></tr>
<tr><td>7-6
—</td><td>Reserved</td></tr>
<tr><td>5-2
RR_EXTTRG_SEL</td><td>External Trigger Source Select
Selects the external trigger from 16 trigger sources. See the chip-specific LPCMP information for more on the trigger source 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>CMP0</td><td>—</td><td>RRCR0</td></tr>
<tr><td>CMP1</td><td>RRCR0</td><td>—</td></tr>
</table>
0000b - Select external trigger source 0
0001b - Select external trigger source 1
0010b - Select external trigger source 2
0011b - Select external trigger source 3
0100b - Select external trigger source 4
0101b - Select external trigger source 5
0110b - Select external trigger source 6
0111b - Select external trigger source 7</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1000b - Select external trigger source 8 1001b - Select external trigger source 9 1010b - Select external trigger source 10 1011b - Select external trigger source 11 1100b - Select external trigger source 12 1101b - Select external trigger source 13 1110b - Select external trigger source 14 1111b - Select external trigger source 15
1 RR_TRG_SEL	Round-Robin Trigger Select Selects the internal trigger or external trigger as the trigger source. 0b - External trigger 1b - Internal trigger
0 RR_EN	Round-Robin Enable Enables the round-robin operation. 0b - Disable 1b - Enable

43.7.11 Round Robin Control Register 1 (RRCR1)

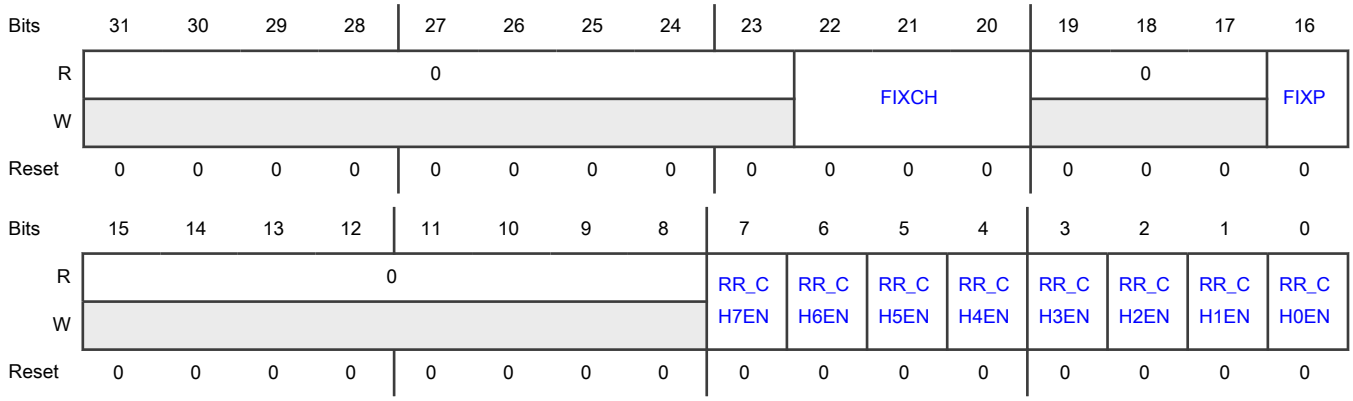
Offset

Register	Offset
RRCR1	28h

Function

Contains configuration options for the round-robin operation, such as enabling individual channels to participate.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-20 FIXCH	Fixed Channel Select Selects which channel in the mux port to fix for a given round-robin trigger mode application. 000b - Channel 0 001b - Channel 1 010b - Channel 2 011b - Channel 3 100b - Channel 4 101b - Channel 5 110b - Channel 6 111b - Channel 7
19-17 —	Reserved
16 FIXP	Fixed Port Fixes an analog mux port (plus or minus) for round-robin trigger mode. The inputs to the non-fixed port sweep during each round. 0b - Fix the plus port. Sweep only the inputs to the minus port. 1b - Fix the minus port. Sweep only the inputs to the plus port.
15-8 —	Reserved
7-0	Channel n Input Enable in Trigger Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
RR_CHnEN	Enables channel n of the non-fixed mux port to check its voltage value when in Trigger mode. 0b - Disable 1b - Enable

43.7.12 Round Robin Control and Status (RRCSR)

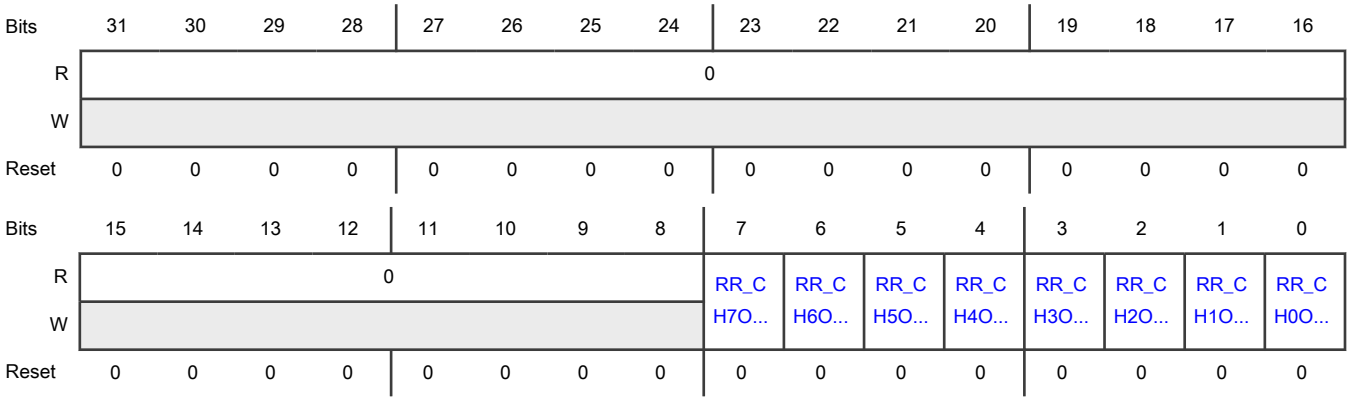
Offset

Register	Offset
RRCSR	2Ch

Function

Contains the latest comparison results of the individual channels with the fixed mux port. It also allows you to define the pre-set state for each channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 RR_CHnOUT	Comparison Result for Channel n Returns the latest comparison result for channel n when read and defines the pre-set state for channel n when written to.

43.7.13 Round Robin Status (RRSR)

Offset

Register	Offset
RRSR	30h

Function

Contains individual channel flags that indicates when a channel's last comparison result is different from its pre-set 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	0								RR_CH7F	RR_CH6F	RR_CH5F	RR_CH4F	RR_CH3F	RR_CH2F	RR_CH1F	RR_CH0F
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	0

Fields

Field	Function
31-8 —	Reserved
7-0 RR_CHnF	Channel n Input Changed Flag Indicates when the corresponding channel's last comparison result is different from its pre-set value. **NOTE** To clear a flag, write a 1 to it. 0b - No different 1b - Different

43.7.14 Round Robin Control Register 2 (RRCR2)

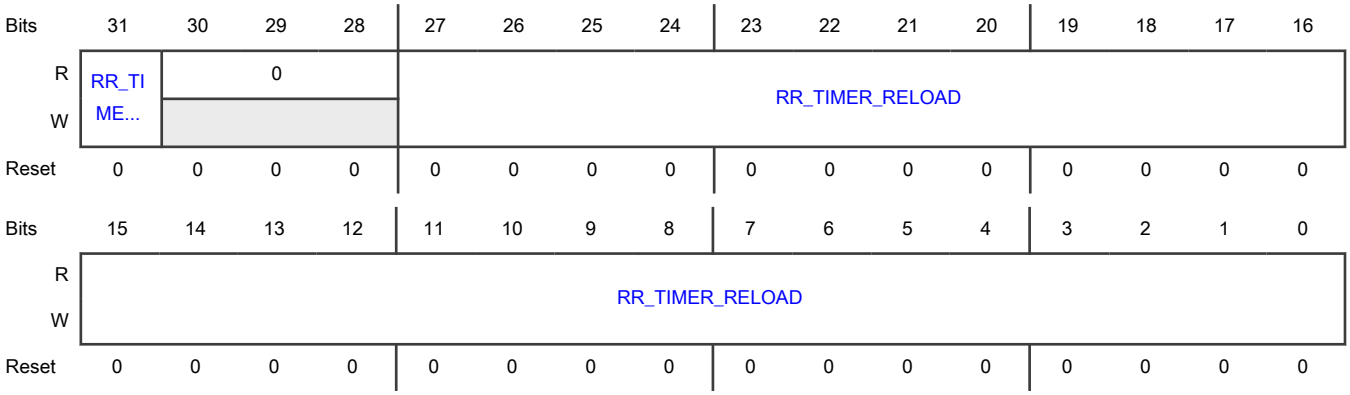
Offset

Register	Offset
RRCR2	38h

Function

Provides the controls for the round-robin internal trigger generation.

Diagram



Fields

Field	Function
31 RR_TIMER_EN	Round-Robin Internal Timer Enable Enables round-robin internal timer. 0b - Disables 1b - Enables
30-28 —	Reserved
27-0 RR_TIMER_RELOAD	Number of Sample Clocks Establishes the repetitive count rate for the round-robin internal timer. Each time the timer counts zero it is reloaded with this value. The round-robin trigger signal generates at a rate of (RR_TIMER_RELOAD + 1) times the round-robin clock period.

Chapter 44
12-bit Digital-to-Analog Converter (DAC)

44.1 Chip-specific 12-bit DAC information

Table 337. Reference links to related information

Topic	Related module	Reference
Full description	12-bit DAC	12-bit Digital-to-Analog Converter (DAC)
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

44.1.1 Module instances

This device has one instance of the 12-bit 1 Msps DAC module, DAC0, with compatible DAC Controller. Set controller FIFO size to 4 (16 words).

44.1.2 12-bit DAC references

- VREFH0=VDDA
- VREFH1=VREFO
- VREFH2=VREFH

44.1.3 12-bit DAC chip-specific initialization

To initialize the 12-bit DAC module:

1. Set the clock divider for DAC in SYSCON registers.
2. Enable the clock to DAC in SYSCON. This enables the register interface and the peripheral function clock.
3. Power up the DAC with ACTIVE_CFG1 and LP_CFG1 registers in SPC. See SOC_CNTRL fields.
4. Reset and release the DAC peripheral reset.
5. Reset the DAC logic by setting and clearing bit RCR[SWRST]. See RCR register Reset Control (RCR).
6. Reset the DAC FIFO by setting and clearing bit RCR[FIFORST].
7. If needed, initialize the DAC reference source before selecting the DAC reference with GCR[DACRFS]. See GCR register Global Control (GCR).
8. Configure DAC as described in section Initialization.

44.1.4 12-bit DAC trigger inputs

12-bit DAC trigger sources get routed through the INPUTMUX. See the DAC0_TRIG register in the INPUTMUX chapter for available trigger sources.

44.2 Overview

The 12-bit general-purpose digital-to-analog converter is a Low-power DAC. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator or ADC.

44.2.1 Block diagram

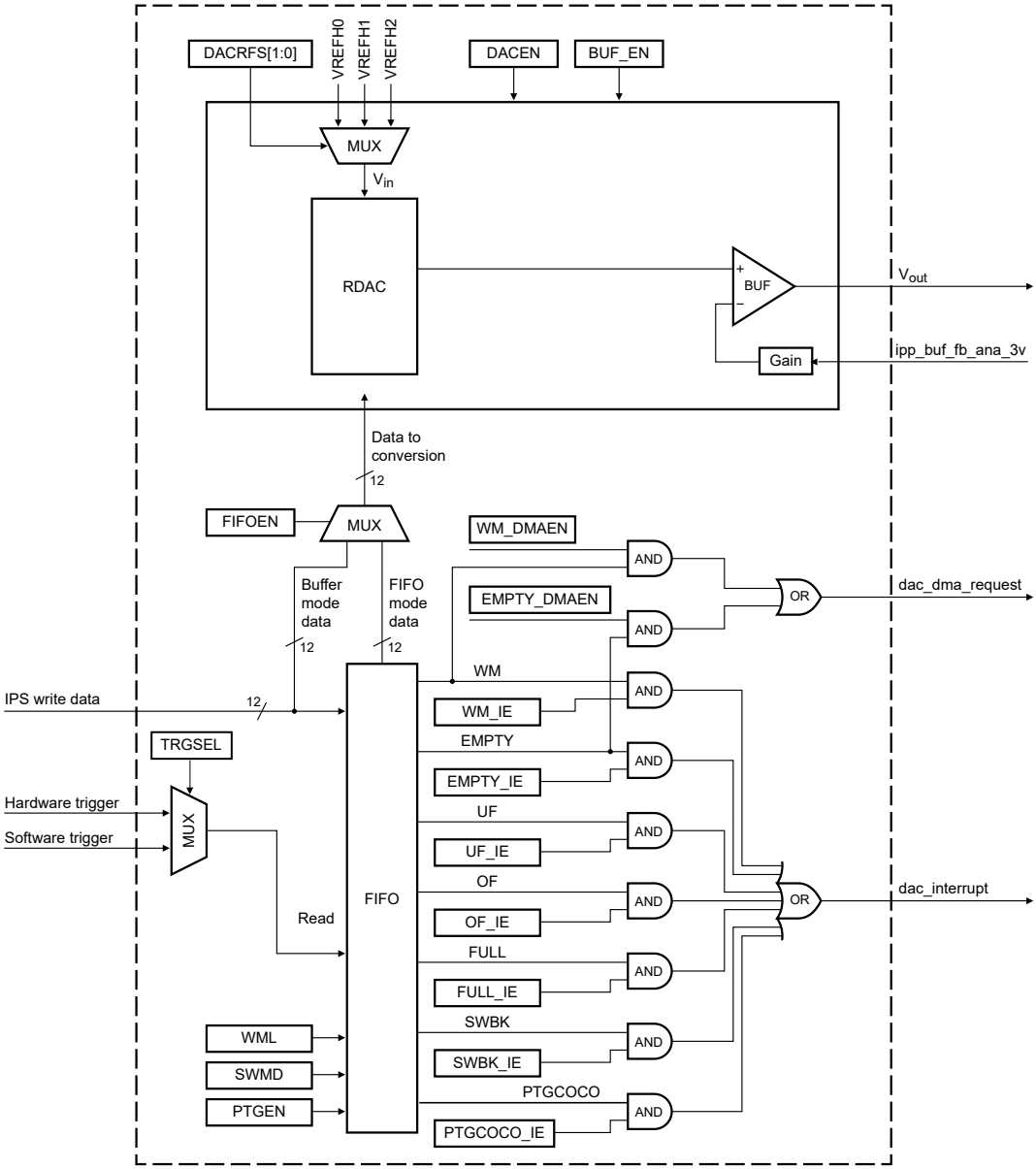


Figure 198. 12-bit DAC block diagram

44.2.2 Features

- Includes an on-chip programmable reference generator output. The voltage output ranges from 1/4096 V_{in} to V_{in}, and the step is 1/4096 V_{in}, where V_{in} is the input reference voltage.
- Selects V_{in} from three reference sources.
- Supports 16-word depth FIFO with the configurable watermark.

- Uses multiple operation modes.
  - Buffer mode
  - FIFO mode
  - Swing Back mode
  - Periodic Trigger mode
- Supports software trigger and hardware trigger.
- Provides selectable performance levels: Low-Power mode(LP) and Lower Low-Power mode(LLP).
- Provides interrupt and DMA support.

44.3 Functional description

12-bit DAC can select one of the three reference inputs: VREFH0, VREFH1, and VREFH2 as the reference voltage, by DACRFS. See the chip-specific 12-bit DAC information to determine the source options for VREFH0, VREFH1, and VREFH2.

12-bit DAC converts a 12-bit data to a stepped analog output voltage when enabled. The output voltage range is from V_{in} to $V_{in}/4096$, and the step is $V_{in}/4096$.

You can configure the 12-bit DAC to Buffer mode, FIFO mode, Swing Back mode, or Periodic Trigger mode.

NOTE

You must configure 12-bit DAC before any conversion. During conversion or FIFO write operation, you must not change the configuration or disable DACEN. If you switch to another mode, you must first write a 1 to SWRST to reset 12-bit DAC or write 0 to DACEN.

44.3.1 Buffer mode

You can enable Buffer mode with DACEN.

In Buffer mode, any write to DATA pushes the data to the analog conversion without trigger support.

NOTE

The data write speed must not be higher than the analog conversion speed.

44.3.2 FIFO mode

You can enable FIFO mode with FIFOEN and DACEN.

In FIFO mode, for a 32-/16-bit write to DATA, put the data into the FIFO and the write pointer increments automatically. For an 8-bit write, only if you write both the bytes, the write pointer increases. You must ensure that an 8-bit write is done in pair and write both the upper and lower bytes. There is no priority, which byte to write first.

When a software or hardware trigger comes, the read pointer pointed data in FIFO pushes to the analog conversion, and the read pointer increments automatically. If the trigger comes when FIFO is empty, the data to analog remains unchanged, and an underflow error flag sets.

44.3.3 Swing Back mode

You can enable Swing Back mode with FIFOEN, SWMD, and DACEN.

The read pointer swings between the writer pointer and zero in Swing Back mode. The trigger increases the read pointer till the writer pointer and decreases the read pointer till zero, and so on. EMPTY, FULL, and WM have no meaning and you should ignore them during Swing Back mode.

NOTE

In Swing Back mode, the FIFO should first write at least two data; otherwise, this mode will not work correctly. You should not write the FIFO when Swing Back mode is in progress.

44.3.4 Periodic Trigger mode

You can enable Periodic Trigger mode with FIFOEN, PTGEN, and DACEN.

Due to the trigger (hardware or software) synchronization logic, there is a delay time variation between each conversion. If there is a need to trigger DAC periodically, NXP recommends you to use Periodic Trigger mode.

In Periodic Trigger mode, you only send the first trigger. Then after every [PTG_PERIOD+1] read clock (RCLK) cycle, an internal trigger automatically triggers the DAC. There are [PTG_NUM] internal triggers, therefore, in total [PTG_NUM+1] conversions, including the first trigger that you send. Write 0 to PTGEN to terminate the current conversion queue. Then, after the current conversion completes, the conversion terminates, and the PTGCOCO sets. If PTG_NUM becomes zero, the infinite triggers follow the first hardware/software trigger until you write 0 to PTGEN. In any case, FIFORST or SWRST can terminate the conversion.

During the periodic trigger mode conversion, you can ignore the additional hardware/software triggers. You can start another queue after PTGCOCO becomes 1.

During Periodic Trigger Mode, WM feeds data to FIFO to avoid underflow.

You can combine Periodic Trigger mode with SWMD to periodically trigger DAC and swing back.

44.3.5 Low-power mode operation

If the RCLK remains enabled, 12-bit DAC can be functional with FIFO mode or Swing Back mode or Periodic Trigger mode during Stop and Wait modes, using the hardware trigger initiating the conversion.

44.3.6 DMA

After the corresponding DMA enable register field becomes 1, you can automatically generate DMA requests when WM or EMPTY becomes 1. When you enable DMA, the error flags can only generate an interrupt.

44.3.7 Clocks

12-bit DAC block requires the following clock(s):

- Bus clock for register access and writing data to FIFO or Buffer.
- Read clock to push data from FIFO or Buffer to the analog DAC.

44.3.8 Resets

The following are the 12-bit DAC resets:

1. Global reset fed into the block. This global reset resets everything.
2. SWRST: resets all 12-bit DAC registers and internal logic.
3. FIFORST: resets the FIFO pointers in DAC FIFO Pointer (FPR) and flags in FIFO Status (FSR).

44.3.9 Interrupts

After the corresponding interrupt enable register field becomes 1, warning flags and error flags can generate interrupt. The warning flags are EMPTY, FULL, WM, SWBK, and PTGCOCO. The error flags are OF and UF.

44.4 DAC external signal descriptions

Table 338. DAC external signal descriptions

Signal	Description	I/O
VREFH0	DAC reference high voltage 0	I
VREFH1	DAC reference high voltage 1	I
VREFH2	DAC reference high voltage 2	I
Vout	DAC output	O

44.5 Initialization

To initialize 12-bit DAC:

1. Configure the read clock at chip level.
2. Configure Work mode and GCR enables 12-bit DAC.
3. Configure Interrupt Enable (IER) if required.
4. Configure DMA Enable (DER) if required.
5. Configure Periodic Trigger Control (PCR) if required.
6. Write data to DATA to fill the FIFO, if FIFO mode.
7. Wait analog start-up time
8. Use hardware or software trigger to push data to the analog DAC to convert, if FIFO mode.
9. Write data directly to DATA to push data to the analog DAC to convert, if Buffer mode.

44.6 Application Information

Digital delay

Due to the resynchronization and data latch process, there is a delay between the trigger posedge and analog voltage settling. The delay is (2+ LATCH_CYC)*T to (3+ LATCH_CYC)*T, where T is the RCLK period, and LATCH_CYC*T should not be smaller than 40 ns. So the delay has one T uncertainty. The fast RCLK can achieve a small delay.

RCLK frequency

For discrete conversion, the RCLK frequency has no low limit. But slow RCLK involves large digital delay and uncertainty.

For continuous conversion, the RCLK frequency should be at least 5 times the trigger speed.

For continuous conversion, due to the one cycle uncertainty RCLK, the trigger period should not less than (1000ns + T). Where 1000 ns period means the maximum analog conversion speed 1 Msps.

To achieve the full speed of DAC analog, you must configure to the internal periodic trigger mode.

For Internal Periodic Trigger mode, RCLK frequency should be at least two times the conversion speed. For example, to achieve 1 Msps conversion speed, at least 2 MHz RCLK requires in Periodic Trigger mode.

44.7 Memory map and register definition

All registers are accessible in 8-, 16-, or 32-bit accesses. For efficiency, 32-bit accesses are recommended. Access to an address outside the valid memory map generates a bus error.

44.7.1 12-bit DAC register descriptions

Access to an address outside the valid memory map generates a bus error.

44.7.1.1 DAC memory map

DAC0 base address: 400B_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version Identifier (VERID)	32	R	0100_0000h
4h	Parameter (PARAM)	32	R	0000_0003h
8h	Data (DATA)	32	RW	0000_0000h
Ch	Global Control (GCR)	32	RW	0000_0100h
10h	DAC FIFO Control (FCR)	32	RW	0000_0000h
14h	DAC FIFO Pointer (FPR)	32	R	0000_0000h
18h	FIFO Status (FSR)	32	RW	0000_0002h
1Ch	Interrupt Enable (IER)	32	RW	0000_0000h
20h	DMA Enable (DER)	32	RW	0000_0000h
24h	Reset Control (RCR)	32	RW	0000_0000h
28h	Trigger Control (TCR)	32	RW	0000_0000h
2Ch	Periodic Trigger Control (PCR)	32	RW	0000_0000h

44.7.1.2 Version Identifier (VERID)

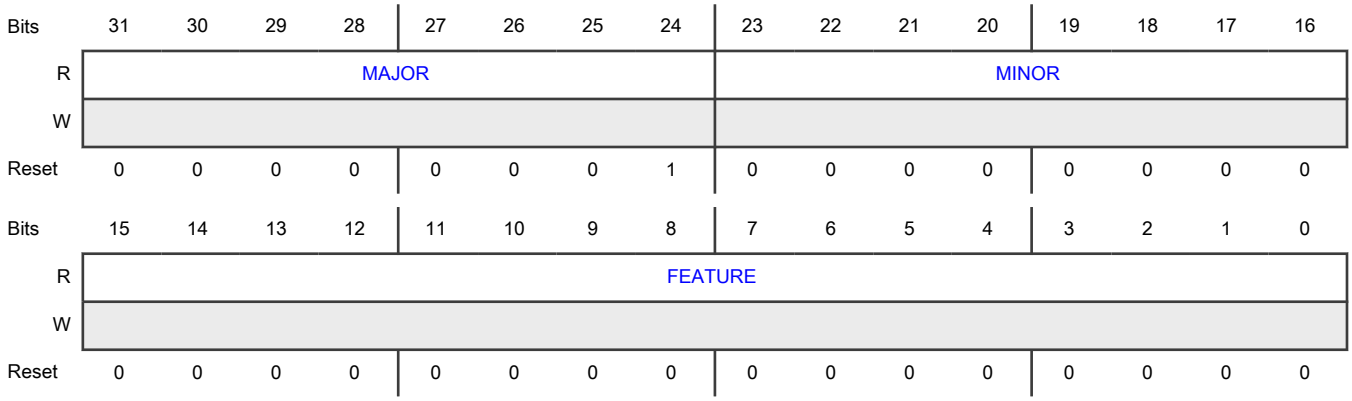
Offset

Register	Offset
VERID	0h

Function

Indicates the version integrated for this instance on the chip.

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 Identification Number Returns the feature set number.

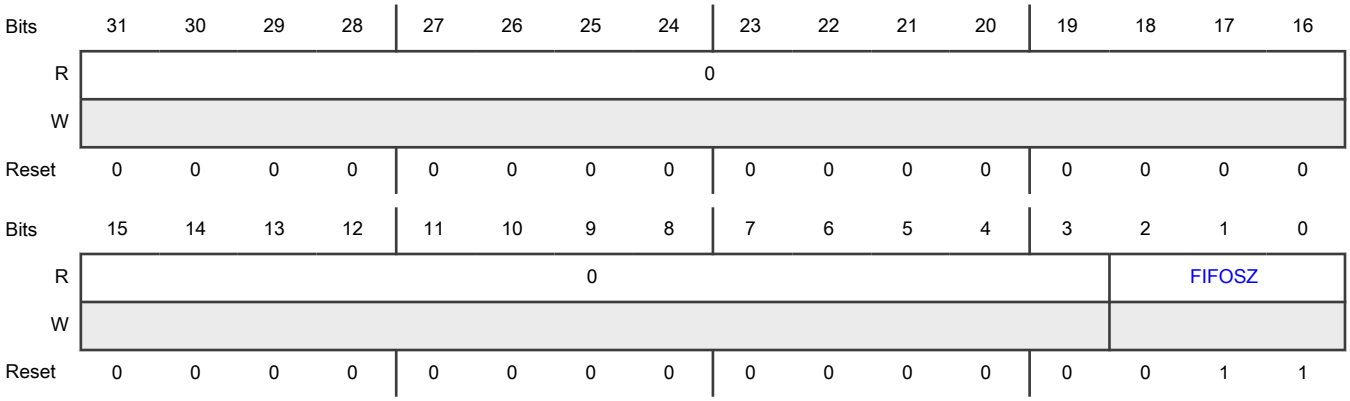
44.7.1.3 Parameter (PARAM)

Offset

Register	Offset
PARAM	4h

Function
Indicates the feature parameters for this instance on the chip.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 FIFOSZ	FIFO Size Indicates that the FIFO depth is 2^(FIFOSZ+1) words.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Reserved 001b - FIFO depth is 4 010b - FIFO depth is 8 011b - FIFO depth is 16 100b - FIFO depth is 32 101b - FIFO depth is 64 110b - FIFO depth is 128 111b - FIFO depth is 256

44.7.1.4 Data (DATA)

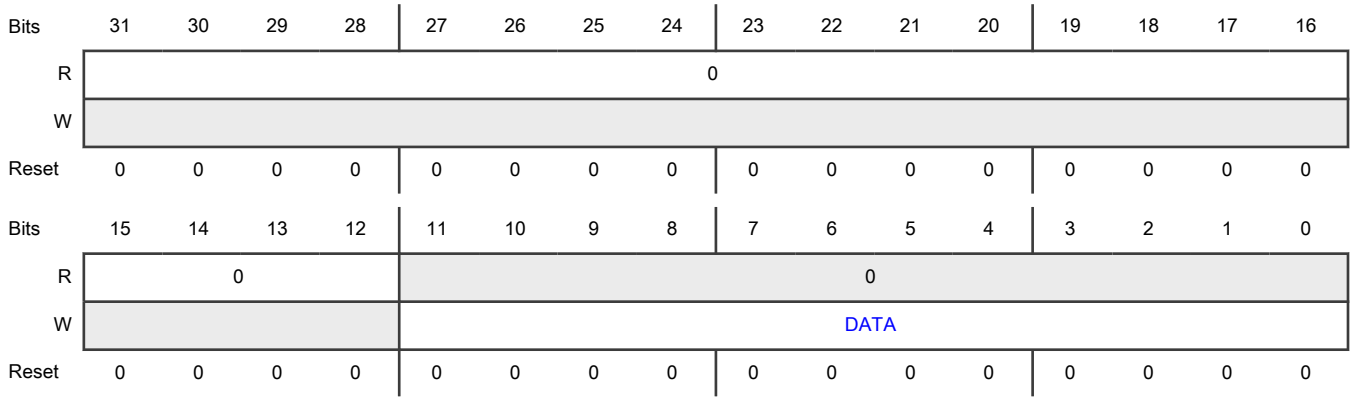
Offset

Register	Offset
DATA	8h

Function

Holds the FIFO entry in FIFO mode and serves as the buffer in Buffer mode.

Diagram



Fields

Field	Function
31-12 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-0 DATA	FIFO Entry or Buffer Entry Specifies that in FIFO mode, Swing Back mode, or Periodic Trigger mode, this is the FIFO data entry. In Buffer mode, write to this field pushes the data to analog without trigger support.

44.7.1.5 Global Control (GCR)

Offset

Register	Offset
GCR	Ch

Function

Contains configuration options for DAC function, such as mode select, trigger select, and so on. It also includes the reference current select for the analog DAC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								BUF_SPD...	0	IREF_ZT...	IREF_PT...	0		BUF_EN	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				LATCH_CYC				0	PTGEN	TRGSEL	SWMD	FIFOEN	DACRFS		DACEN
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23 BUF_SPD_CTRL	OPAMP as Buffer, Speed Control Signal **NOTE** See the DAC electrical characteristics of the chip datasheet for more information on the impact of the modes below. 0b - Lower Low-Power mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Low-Power mode
22 —	Reserved
21 IREF_ZTC_EXT_SEL	External On-Chip ZTC Current Reference Select Selects the external ZTC current reference, which is from the VREF module. See the Chip-specific section for more information. 0b - Not selected 1b - Selected
20 IREF_PTAT_EXT_SEL	External On-Chip PTAT Current Reference Select Selects the external PTAT current reference, which is from the VREF module. See the Chip-specific section for more information. 0b - Not selected 1b - Selected
19-18 —	Reserved
17 BUF_EN	Buffer Enable Uses OPAMP as the DAC analog buffer. 0b - Not used 1b - Used
16-12 —	Reserved
11-8 LATCH_CYC	RCLK Cycles Before Data Latch Configures the DAC sync cycles, which reduces the glitch on the output. The sync time is (LATCH_CYC+1) RCLK cycles. You must configure this register per the RCLK frequency. The recommended sync time is at least 40 ns. The recommended LATCH_CYC configuration is: Sync time is (LATCH_CYC+1) RCLK cycles, when (25×LATCH_CYC) MHz < RCLK ≤ (25×LATCH_CYC+25) MHz. Example 1, if f_{RCLK} is 48 MHz (period is 20.8 ns), you can configure LATCH_CYC to 0x1 to generate a 41.6 ns sync time. Example 2, if f_{RCLK} is 6 MHz (period is 166.7 ns), you can configure LATCH_CYC to 0x0 to generate a 166.7 ns sync time. See the "Clocking" chapter or chip-specific DAC information for more on the RCLK frequency.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE 12-bit DAC adds (LATCH_CYC+1) RCLK cycles before the analog starts to set up the output. Therefore, when you write to this field, NXP recommends using the smallest possible value that still meets the requirement of 40 ns sync time.
7 —	Reserved
6 PTGEN	DAC Periodic Trigger Mode Enable Enables Periodic Trigger Mode when FIFOEN becomes 1. 0b - Disables 1b - Enables
5 TRGSEL	DAC Trigger Select Selects the trigger source for FIFO mode. 0b - Hardware trigger 1b - Software trigger
4 SWMD	Swing Back Mode Specifies whether Swing Back mode is enabled. Selects the Swing Back mode when FIFOEN becomes 1. 0b - Disables 1b - Enables
3 FIFOEN	FIFO Enable Selects the FIFO mode or Buffer mode. 0b - Disables FIFO mode and enables Buffer mode. Any data written to DATA[DATA] goes to buffer then goes to conversion. 1b - Enables FIFO mode. Data will be first read from FIFO to buffer and then goes to conversion.
2-1 DACRFS	DAC Reference Select Selects the source of reference voltage high. 00b - Selects VREFH0 as the reference voltage. 01b - Selects VREFH1 as the reference voltage. 10b - Selects VREFH2 as the reference voltage. 11b - Reserved.
0 DACEN	DAC Enable Enables the DAC system. Starts the programmable reference generator operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE For an 8-bit or 16-bit register write, enable this field after configuring other bytes of GCR. You must configure Periodic Trigger Control (PCR) and DAC FIFO Control (FCR) before enabling this field. 0b - Disables 1b - Enables

44.7.1.6 DAC FIFO Control (FCR)

Offset

Register	Offset
FCR	10h

Function

Contains watermark level control for the FIFO.

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												WML			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 WML	Watermark Level Specifies that when the number of data that remains in FIFO is equal to or less than this level, WM becomes 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE To enable the watermark feature, do not write 0 to this field.

44.7.1.7 DAC FIFO Pointer (FPR)

Offset

Register	Offset
FPR	14h

Function

Contains the read pointer and write pointer of the FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FIFO_WPT			
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												FIFO_RPT			
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 FIFO_WPT	FIFO Write Pointer Shows the current write pointer of the FIFO.
15-4 —	Reserved
3-0 FIFO_RPT	FIFO Read Pointer Shows the current read pointer of the FIFO.

44.7.1.8 FIFO Status (FSR)

Offset

Register	Offset
FSR	18h

Function
Contains the status flags of the FIFO.

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							PTGCOCO	UF	OF	0		SWBK	WM	EMPTY	FULL
W								W1C	W1C	W1C			W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-9 —	Reserved
8 PTGCOCO	Period Trigger Mode Conversion Complete Flag Indicates that PTG mode conversion is completed. This flag becomes 0 by writing a 1 to it. 0b - Not completed or not started 1b - Completed
7 UF	FIFO Underflow Flag Specifies that there is a new trigger after EMPTY becomes 1. The FIFO read pointer do not increase in this case and the data sent to DAC analog conversion will not changed. This flag becomes 0 by writing a 1 to it. 0b - No underflow has occurred since the last time the flag was cleared 1b - At least one trigger underflow has occurred since the last time the flag was cleared
6 OF	FIFO Overflow Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates that data is intended to write into FIFO after FULL becomes 1. The writer pointer do not increase in this case. Do not write the extra data into the FIFO. This flag clears by writing a 1 to it. 0b - No overflow has occurred since the last time the flag was cleared 1b - At least one FIFO overflow has occurred since the last time the flag was cleared
5-4 —	Reserved
3 SWBK	Swing Back One Cycle Complete Flag Indicates that the DAC has completed one period of conversion in Swing Back mode. The read pointer increases to the top (write pointer) once and then decreases to zero. For example, after writing three data to FIFO, the writer pointer is now 3. Then, if continually triggered, the read pointer will swing like: 0-1-2-1-0-1-2-, and so on. After the fourth trigger, the flag sets. Write a 1 to clear this flag. 0b - No swing back cycle has completed since the last time the flag was cleared 1b - At least one swing back cycle has occurred since the last time the flag was cleared
2 WM	FIFO Watermark Status Flag Write 1 to this field if the remaining data in FIFO is less than or equal to WML. By writing data into FIFO by DMA or CPU, this flag clears automatically when the data in FIFO is more than WML. 0b - Data in FIFO is more than watermark level 1b - Data in FIFO is less than or equal to watermark level
1 EMPTY	FIFO Empty Flag Indicates whether the FIFO is empty or not. 12-bit DAC automatically clears this flag when CPU or DMA writes data to the FIFO. 0b - Not empty 1b - Empty
0 FULL	FIFO Full Flag Indicates whether the FIFO is full or not. FIFO read with software trigger or hardware trigger automatically clears this flag. 0b - Not full 1b - Full

44.7.1.9 Interrupt Enable (IER)

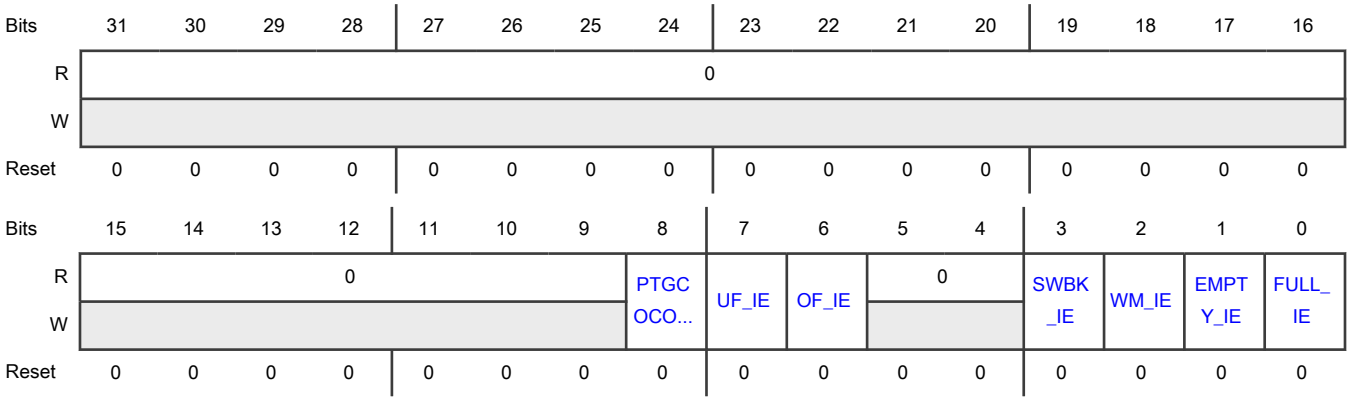
Offset

Register	Offset
IER	1Ch

Function

Contains the interrupt enable fields.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 PTGCOCO_IE	PTG Mode Conversion Complete Interrupt Enable Enables interrupt after the PTG mode conversion completes. 0b - Disables 1b - Enables
7 UF_IE	FIFO Underflow Interrupt Enable Enables interrupt when FIFO underflows. 0b - Disables 1b - Enables
6 OF_IE	FIFO Overflow Interrupt Enable Enables interrupt when FIFO overflows. 0b - Disables 1b - Enables
5-4 —	Reserved
3 SWBK_IE	Swing Back One Cycle Complete Interrupt Enable Enables interrupt after the swing back one cycle completes. 0b - Disables

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enables
2 WM_IE	FIFO Watermark Interrupt Enable Enables interrupt when data in FIFO is less than or equal to watermark level. 0b - Disables 1b - Enables
1 EMPTY_IE	FIFO Empty Interrupt Enable Enables interrupt when FIFO is empty. 0b - Disables 1b - Enables
0 FULL_IE	FIFO Full Interrupt Enable Enables interrupt when FIFO is full. 0b - Disables 1b - Enables

44.7.1.10 DMA Enable (DER)

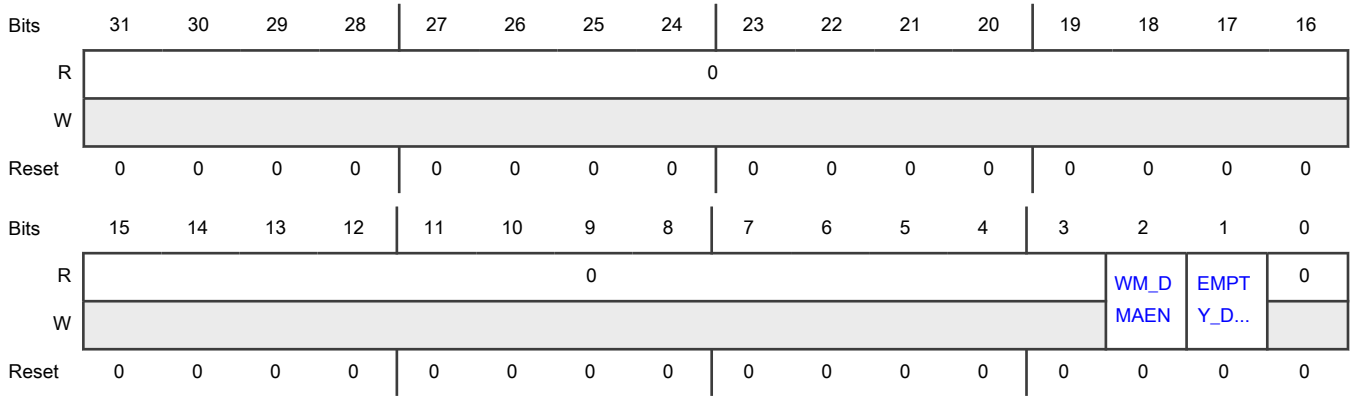
Offset

Register	Offset
DER	20h

Function

Contains the DMA enable fields.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 WM_DMAEN	FIFO Watermark DMA Enable Enables DMA request when data in FIFO is less than or equal to watermark level. 0b - Disables 1b - Enables
1 EMPTY_DMAEN	FIFO Empty DMA Enable Enables DMA request when FIFO is empty. 0b - Disables 1b - Enables
0 —	Reserved

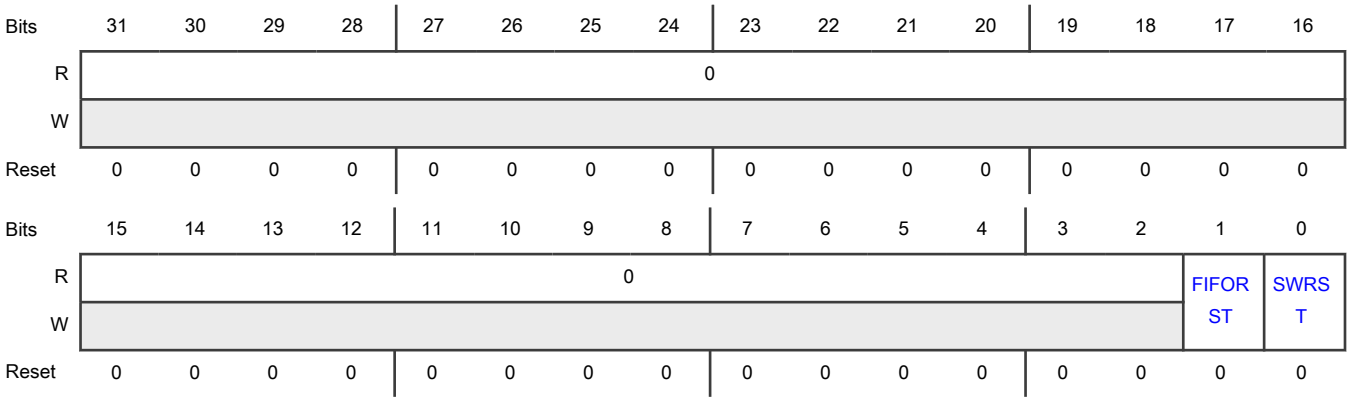
44.7.1.11 Reset Control (RCR)

Offset

Register	Offset
RCR	24h

Function
Contains the software reset and FIFO reset fields.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 FIFORST	FIFO Reset Resets the FIFO pointers and flags in FIFO Status (FSR). Remains 1 until you write 0 to it. 0b - No effect 1b - FIFO reset
0 SWRST	Software Reset Resets all DAC registers and internal logic. Remains 1 until you write 0 to it. 0b - No effect 1b - Software reset

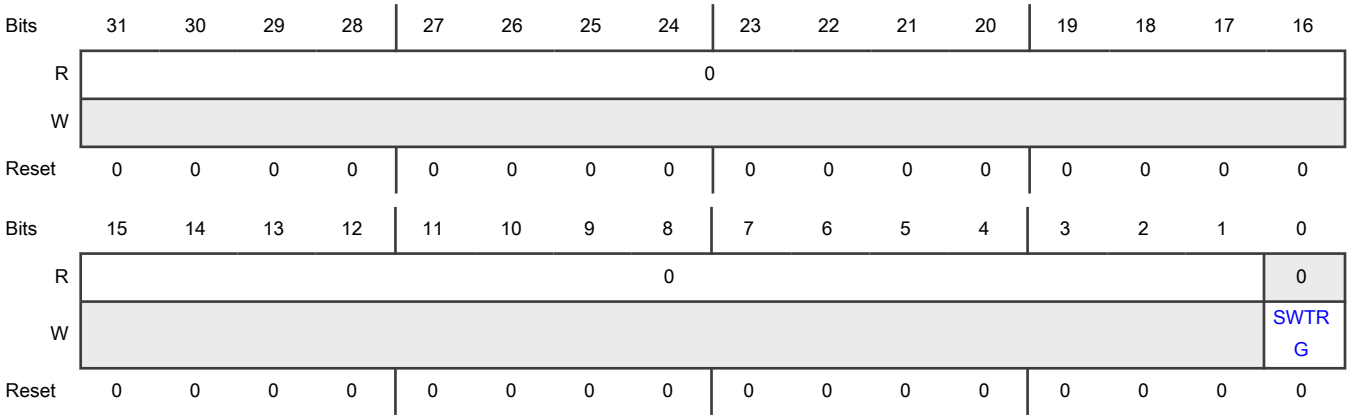
44.7.1.12 Trigger Control (TCR)

Offset

Register	Offset
TCR	28h

Function
Contains the software trigger field.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 SWTRG	Software Trigger Indicates whether the DAC soft trigger is valid. Active high. This is a write-only field, which always reads 0. If TRGSEL is 1 and FIFO is enabled and not empty, writing 1 to this field advances the FIFO read pointer once. **NOTE** During continuous conversions, do not adopt hardware trigger and software trigger in a mixed use. 0b - Not valid 1b - Valid

44.7.1.13 Periodic Trigger Control (PCR)

Offset

Register	Offset
PCR	2Ch

Function
Contains the periodic trigger period and periodic trigger number fields.

Diagram

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

Fields

Field	Function
31-16 PTG_PERIOD	Periodic Trigger Period Width Uses only in Periodic Trigger mode. It controls the periodic trigger frequency. There are [PTG_PERIOD+1] RCLK cycles between each periodic trigger. See the chip-specific DAC information for more on the RCLK frequency. Example 1, if f_{RCLK} is 48 MHz and the conversion speed is 0.5 MHz, there should be 96 RCLK cycles between each periodic trigger. So you must configure the PTG_PERIOD as 0x5F(95). Example 2, if f_{RCLK} is 6 MHz and the conversion speed is 0.5 MHz, there should be 12 RCLK cycles between each periodic trigger. So you must configure the PTG_PERIOD as 0xB(11). **NOTE** The PTG_PERIOD should not be less than 0x1. **NOTE** The periodic trigger frequency should not be larger than the analog conversion speed.
15-0 PTG_NUM	Periodic Trigger Number Uses only in Periodic Trigger mode. There are [PTG_NUM] internal triggers following the first hardware/software trigger. So, there are [PTG_NUM+1] conversions in total. **NOTE** If you write 0 to it, there will be infinite triggers following the first hardware/software trigger, until PTGEN becomes 0.

Chapter 45
Operational Amplifier (OPAMP)

45.1 Chip-specific OPAMP information

Do not configure OPAMP to use INP1. It cannot be used on this device.

45.1.1 Positive voltage references

The positive voltage reference can be selected using the OPAMP_CTR[PREF] register. The PREF inputs correspond to following signals.

Table 339. PREF inputs

Register bit field value	PREF input[1]	Corresponds to
00b	Input 0	DACx_OUT • DAC0_OUT for OPAMP0
01b	Input 1	VREFH/2
10b	Input 2	VREFI
11b	Input 3	520 mv. PMC Bandgap provides VREF1V, which is divided inside OPAMP to generate 0.52 V.

1. The voltage range is between 0 - VDD_ANA-0.8.

45.1.2 OPAMP initialization

1. Release peripheral reset, see peripheral reset control information in SYSCON.MRCC
2. Enable OPAMP power mode, see SOC_CNTRL fields.
3. Configure OPAMP control parameters in OPAMP_CTR register.
4. Enable OPAMP by setting EN bit in OPAMP_CTR register.

45.2 Overview

The OPAMP is an advanced amplifier that supports programmable gain (PGA), inverting, non-inverting, buffer mode and standalone mode. Positive and negative inputs of an amplifier connect to internal channels. The module is applicable to the signal-processing stage before ADC.

45.2.1 Block diagram

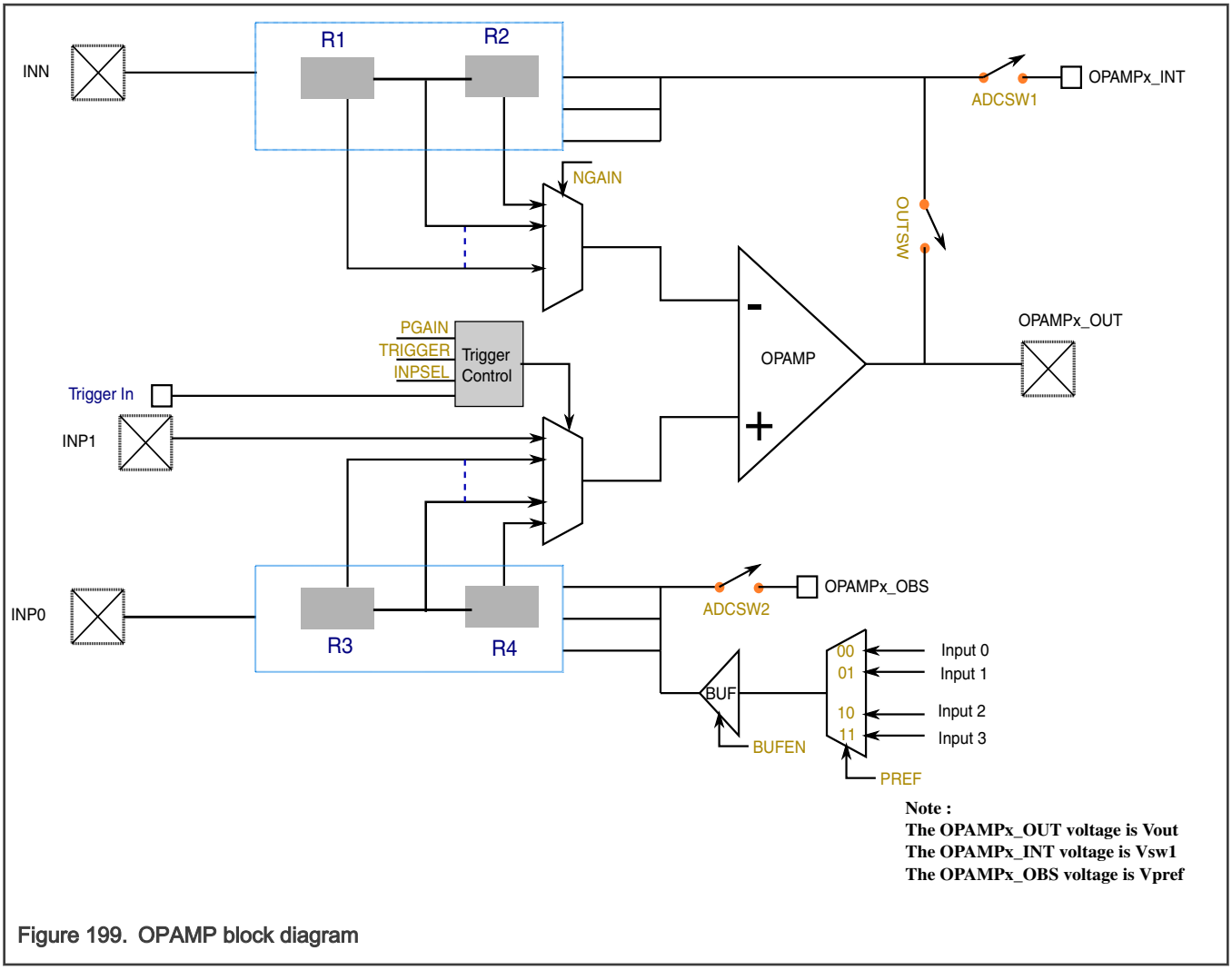


Figure 199. OPAMP block diagram

See chip-specific for supported PREF inputs.

45.2.2 Features

- High performance mode and low power mode
- PGA mode
- Noninverting Amplifier mode
- Inverting Amplifier mode
- Unity gain buffer mode
- Standalone OPAMP mode

45.3 Functional description

This module supports optional gains. See the following table for details where Ngain=R2/R1 and Pgain=R4/R3.

OPAMP_CTR[NGAIN]	Description
000b	Buffer mode
001b	Ngain=1
010b	Ngain=2
011b	Ngain=4
100b	Ngain=8
101b	Ngain=16
110b	Ngain=33
111b	Ngain=64

OPAMP_CTR[PGAIN]	Description
000b	Positive input 1 (INP1)
001b	Pgain=1
010b	Pgain=2
011b	Pgain=4
100b	Pgain=8
101b	Pgain=16
110b	Pgain=33
111b	Pgain=64

NOTE

The actual gain is determined by different usage methods. Some gain calculation examples have been shown in Application Information

45.3.1 Modes

This module supports high performance mode and low power mode selected by configuring OPAMP_CTR[MODE].

The table below shows all supported modes and the related configurations:

Table 340. OPAMP modes configuration

Mode	ADCSW1	OUTSW	ADCSW2	NGAIN	PGAIN	INPSEL	PREF	BUFEN
PGA	ON	ON	ON	x	x	INP0	00b/01b/10b/11b	ON
Noninverting	ON	ON	OFF	x	x	INP0, INP1	-	OFF
Inverting[1]	ON	ON	OFF	x	0	INP1	-	OFF
Standalone	OFF	OFF	OFF	011b	011b	INP0, INP1	-	OFF
Buffer	OFF	OFF	OFF	0	0	INP1	-	OFF

1. Vinp0 must be floating

45.3.1.1 Trigger mode

On the chip, there are up to two pads that can be connected to the OPAMP positive input. In trigger mode, hardware can automatically switch the pad which is connected to OPAMP positive input. It recommends that trigger mode is used when OPAMP is configured to noninverting amplifier. Before enabling trigger mode, the INPSEL bit in the CTR register should be set. Write 0 to BUFEN to disable reference buffer. Configure NGAIN to required gain. After finishing all the configurations, enable the trigger mode. In trigger mode, whenever there is a positive edge of trigger, hardware will switch the pad which is connected to OPAMP positive input. You can read INPF register bit field to monitor which pad is connected to OPAMP positive input in trigger mode.

45.3.1.2 PGA Mode

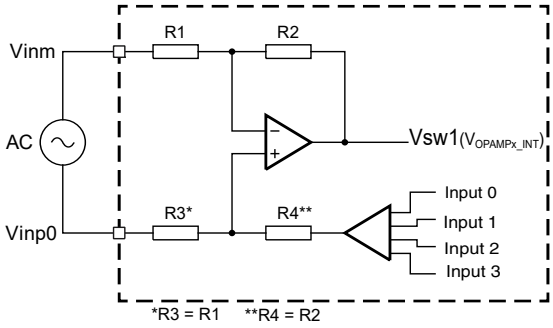


Figure 200. PGA Mode

See chip-specific for supported PREF inputs.

In this mode, configure the registers OPAMP_CTR[NGAIN] and OPAMP_CTR[PGAIN] to select different gains. Enable OPAMP_CTR[BUFEN] and configure the registers OPAMP_CTR[PREF] to select positive reference voltage. Configure OPAMP_CTR[OUTSW] to connect R2 with OPAMP output. Enable OPAMP_CTR[ADCSW1] and OPAMP_CTR[ADCSW2], measure the results from OPAMPx_INT and OPAMPx_OBS by ADC.

45.3.1.3 Noninverting Amplifier mode

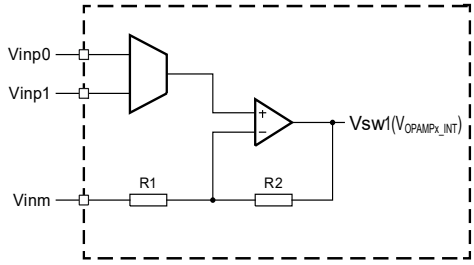


Figure 201. Noninverting Amplifier mode

This mode output a forward gain voltage and the output voltage is connected to the ADC. Configure the registers **OPAMP_CTR[NGAIN]** to select different gains. The positive input can be from Vinp0 or Vinp1. If Vinp1 is used, configure **OPAMP_CTR[PGAIN]** to 0. Configure OPAMP_CTR[BUFEN] to 0 to disable reference buffer. Enable OPAMP_CTR[OUTSW] to connect R2 with OPAMP output (OPAMPx_OUT). Enable OPAMP_CTR[ADCSW1], and measure results from OPAMPx_INT.

45.3.1.4 Inverting Amplifier Mode

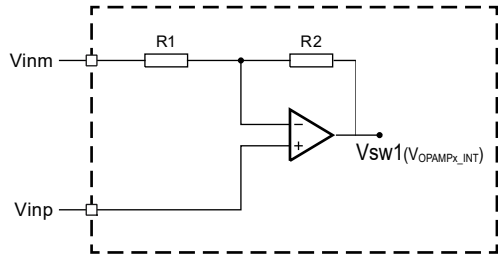


Figure 202. Inverting Amplifier Mode

This mode output a reverse gain voltage and the output voltage is connected to the ADC. Configure the registers **OPAMP_CTR[NGAIN]** to select different gains. The positive input can be either from Vinp1 or reference buffer. Configure **OPAMP_CTR[PGAIN]** to 0 to select Vinp1. Enable reference buffer by **OPAMP_CTR[BUFEN]** to select reference buffer as the source of positive input. Vinp0 must be floating. Enable **OPAMP_CTR[OUTSW]** to connect R2 with OPAMP output (OPAMPx_OUT). Enable **OPAMP_CTR[ADCSW1]**, and measure results from OPAMPx_INT.

45.3.1.5 Standalone OPAMP Mode

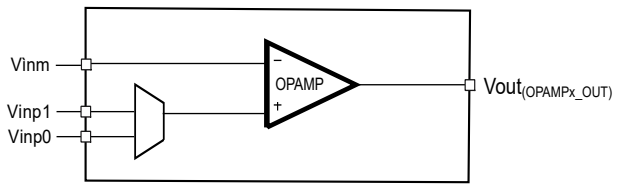


Figure 203. Standalone OPAMP Mode

To use standalone OPAMP mode, disconnect R2 with OPAMP output by disabling **OPAMP_CTR[OUTSW]**. Configure **OPAMP_CTR[NGAIN]=011b** and **OPAMP_CTR[PGAIN]=011b**. Disable reference buffer by **OPAMP_CTR[BUFEN]**. Disconnect ADCSW1 and ADCSW2.

45.3.1.6 Buffer Mode

To select buffer mode, configure the value of registers **OPAMP_CTR[NGAIN]** and **OPAMP_CTR[PGAIN]** to 000b. The output voltage is connected to the **OPAMPx_OUT**. Configure **OPAMP_CTR[BUFEN]** to 0 to disable reference buffer. Enable **OPAMP_CTR[OUTSW]** to connect R2 with OPAMP output. Enable **OPAMP_CTR[ADCSW1]**, and measure results from **OPAMPx_INT**.

45.3.2 Clock

The global clock is used for OPAMP.

45.3.3 Reset

OPAMP is reset by the global chip reset signal.

45.3.4 Interrupts

This module has no interrupts.

45.4 External Signals

Table 341. External Signal Description

Signal	Description	I/O
INN	Analog input channels(Core amplifier negative input channel)	I
INP0	Analog input channels(Core amplifier positive input channel0)	I
INP1	Analog input channels(Core amplifier positive input channel1)	I
V_{OPAMPx_OUT}	Output of the core amplifier	O

45.5 Initialization

This module does not require initialization.

45.6 Application Information

Below is the OPAMP output equation :

$$V_{sw1} = \frac{Ngain+1}{1+\frac{1}{Pgain}} V_{INP0} - Ngain * V_{INN} + \frac{1+Ngain}{1+Pgain} V_{pref}$$

45.6.1 PGA mode Application Information

- Both OUTSW and ADCSW1/2 switches should be on.
- OPAMP_CTR[NGAIN] and OPAMP_CTR[PGAIN] should be set to same value.
- Measurement method:
  - Two steps method
    1. Measure OPAMPx_OBS voltage, defined as Vpref (voltage input selected by PREF bit) first with ADC single end mode, and the measured Vpref will be used in following calculation.
    2. Measure OPAMPx_INT voltage, defined as Vsw1 with ADC single end mode real time.
  - Differential method
    1. Use ADC differential channel measure Vsw1 and Vpref at same time.
- ADCSW1 could be on to fine-tune the results
- Result :

  Recommend Ngain=Pgain,
  - Vsw1 –Vpref = (Vinp0-Vinm) * Ngain

45.6.2 Noninverting mode Application Information

- ADCSW1 should be closed
- Disable the OPAMP_CTR[BUFEN]
- Result
  - Vsw1 + Ngain * Vinm= (1+Ngain) * Vinp

NXP recommends you to connect Vinm to GND, if you do not need to provide a DC offset, and Vinp is recommended to be connected to Vinp1.

45.6.3 Inverting mode Application Information

- ADCSW1 should be closed
- Disable the OPAMP_CTR[BUFEN]
- Result
  - Vsw1 = (Ngain+1) * Vinp-Ngain * Vinm
- OPAMP_CTR[PGAIN] is recommended to be set to 000b.

  NXP recommends you to connect Vinp to Vinp1. The recommended voltage of Vinp0/1 is half of VDD_ANA. Vinp cannot be 0.

45.6.4 Buffer mode Application Information

- The unity gain buffer mode supports OPAMPx_OUT or OPAMPx_INT as the output, with Vout defined as the voltage.
- Disable the OPAMP_CTR[BUFEN]
- Result :-
  - Vout=Vinp1

  Vinp is recommended to be connected to Vinp1.

45.7 Memory map and register definition

This section includes the OPAMP module memory map and detailed descriptions of all registers. You can select different gains by configuring registers. Registers have options for selecting noninverting and inverting gain application.

45.7.1 OPAMP register descriptions

45.7.1.1 OPAMP memory map

OPAMP0 base address: 400B_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0000_0000h
4h	Parameter (PARAM)	32	R	0000_0001h
8h	OPAMP Control (OPAMP_CTR)	32	RW	0100_0000h

45.7.1.2 Version ID (VERID)

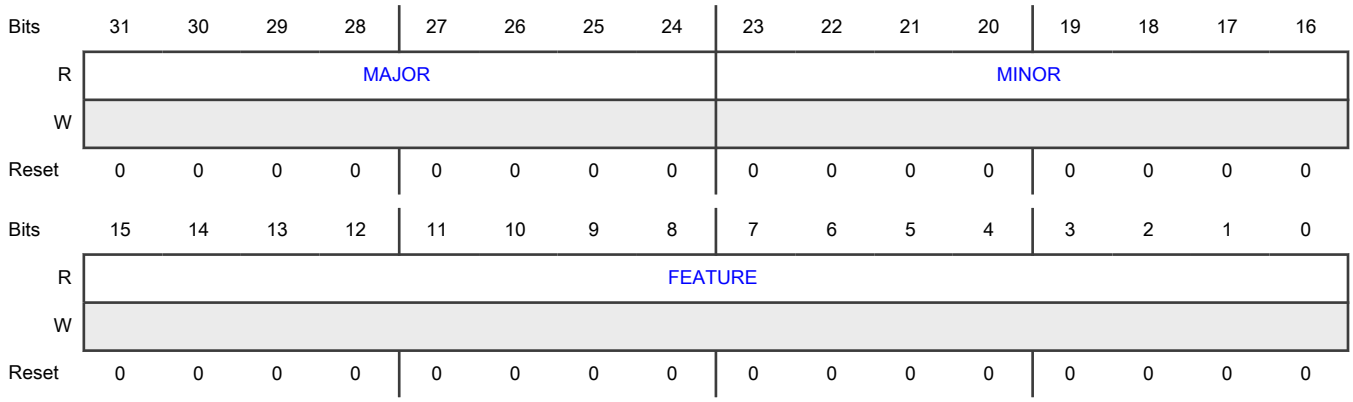
Offset

Register	Offset
VERID	0h

Function

Version ID indicates the version integrated for this instance on the chip. It also indicates inclusion/exclusion of several optional features.

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.

45.7.1.3 Parameter (PARAM)

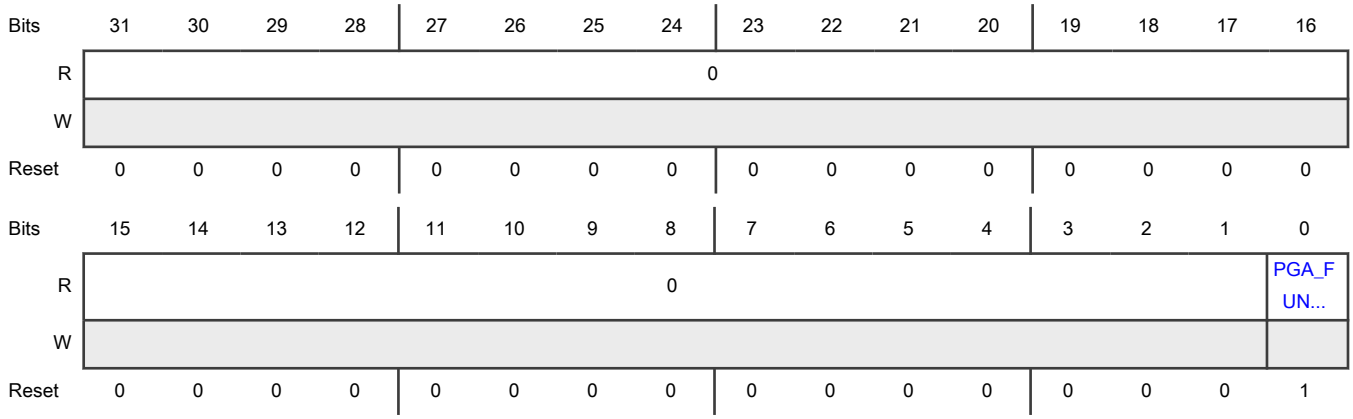
Offset

Register	Offset
PARAM	4h

Function

PARAM indicates optional PGA functions or core amplifier for this instance on the chip.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 PGA_FUNCTION	PGA Function Option Selects different PGA functions. 0b - Core amplifier enabled 1b - PGA function enabled

45.7.1.4 OPAMP Control (OPAMP_CTR)

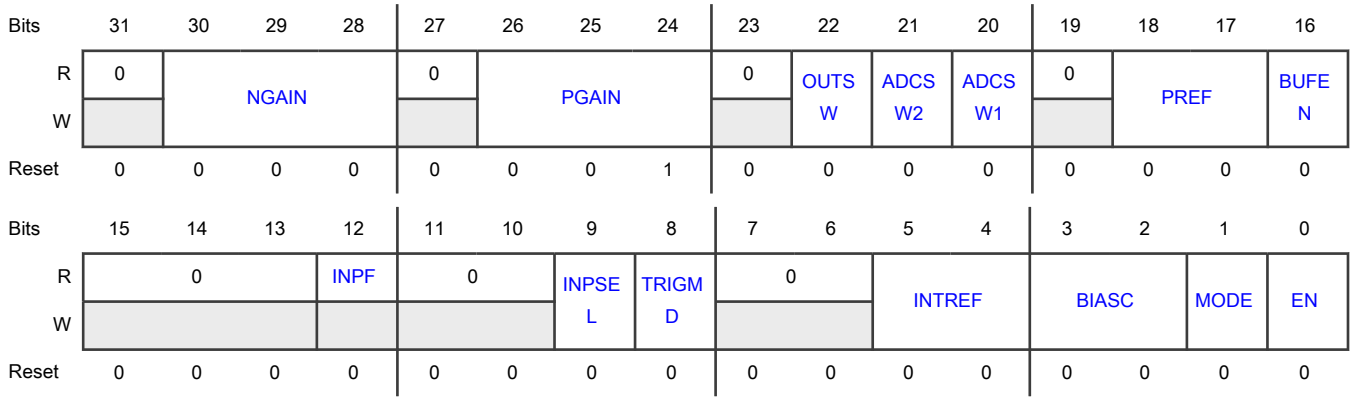
Offset

Register	Offset
OPAMP_CTR	8h

Function

OPAMP_CTR is used to configure the OPAMP module.

Diagram



Fields

Field	Function
31 —	Reserved
30-28 NGAIN	Negative PGA Selection Selects negative programmable gains. 000b - Buffer 001b - Ngain=1 010b - Ngain=2 011b - Ngain=4 100b - Ngain=8 101b - Ngain=16 110b - Ngain=33 111b - Ngain=64
27 —	Reserved
26-24 PGAIN	Positive PGA Selection Selects positive programmable gains. OPAMP cannot write a value to this field when in trigger mode. 000b - Positive input 1 (INP1) 001b - Pgain=1 010b - Pgain=2 011b - Pgain=4 100b - Pgain=8 101b - Pgain=16

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Pgain=33 111b - Pgain=64
23 —	Reserved
22 OUTSW	Output Switch Provides output of OPAMP to the negative gain resistor ladder switch. 0b - OPAMP out to negative gain resistor ladder switch off 1b - OPAMP out to negative gain resistor ladder switch on
21 ADCSW2	Measure Switch 2 Specifies whether the measure positive gain resistor ladder reference voltage switch is on or off for ADC channel. 0b - Measure positive gain resistor ladder reference voltage switch off 1b - Measure positive gain resistor ladder reference voltage switch on
20 ADCSW1	Measure Switch 1 Specifies whether the measure negative gain resistor ladder voltage switch is on or off for ADC channel. 0b - Measure negative gain resistor ladder voltage switch off 1b - Measure negative gain resistor ladder voltage switch on
19 —	Reserved
18-17 PREF	Positive Reference Voltage Selection Selects positive reference voltage. See chip-specific for supported PREF inputs. 00b - Input 0 01b - Input 1 10b - Input 2 11b - Input 3
16 BUFEN	Reference Buffer Enables reference buffer. 0b - Disables 1b - Enables
15-13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 INPF	Positive Input Connection Status The channel which is connected to OPAMP positive input. In trigger mode, it can use INPF to monitor OPAMP positive input connection. 0b - Positive input 0 (INP0) 1b - Positive input 1 (INP1)
11-10 —	Reserved
9 INPSEL	Positive Input Channel Selection When OPAMP is not in trigger mode, use this field to select positive input channel. Before enabling trigger mode, INPSEL can be used to select the initial connection. After trigger mode is enabled, INPSEL does not change when there are triggers. If PGAIN is configured to 0, and INPSEL is configured to 1, INP1 is connected to OPAMP positive input. OPAMP cannot write a value to this field when in trigger mode. 0b - When OPAMP is not in trigger mode, select positive input 0 (INP0) 1b - When OPAMP is not in trigger mode, select positive input 1 (INP1)
8 TRIGMD	Trigger Mode Enables trigger mode 0b - Disable 1b - Enable
7-6 —	Reserved
5-4 INTREF	Provide OPAMP rail to rail voltage selection 00b - Select OPAMP input rail to rail voltage from 0 to VDD_ANA 01b - Select OPAMP input rail to rail voltage from 0 to VDD_ANA-0.8V 10b - Select OPAMP input rail to rail voltage from 0.8V to VDD_ANA 11b - Not allowed
3-2 BIASC	Bias Current Trim Selection 00b - Default 01b - Increase current 10b - Decrease current 11b - Further decrease current
1 MODE	Mode Selection Selects one mode between High performance mode and Low power mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - High performance mode 1b - Low power mode
0 EN	OPAMP Enable Enables OPAMP. 0b - Disable 1b - Enable

Chapter 46
Debug

46.1 Introduction

Cortex-M33 debug functionality is based on ARM CoreSight debug infrastructure, and includes processor halt, single-step, processor core register access, Vector Catch, unlimited software breakpoints, and full system memory access. The processor also includes support for hardware breakpoints and watchpoints configured during implementation:

- A breakpoint unit supporting four instruction comparators.
- A watchpoint unit supporting two watchpoint comparators.

The Cortex-M33 processor supports system level debug authentication to control access from a debugger to resources and memory.

Features:

- The ARM CoreSight Architecture adapted for software trace and debug.
- Serial wire debug (SWD) and Debug Access Port (DAP) to support debugger tools.
- The following are supported for the Cortex-M33 subsystem:
  - Instrumentation Trace Macrocell (ITM)
  - Data Watchpoint and Trace (DWT)
  - Breakpoint Unit (BPU)
  - Software trace and full instruction trace
- SWO trace port for efficiently accessing CM33 trace information from the system

The following figures show the system level debug architecture.

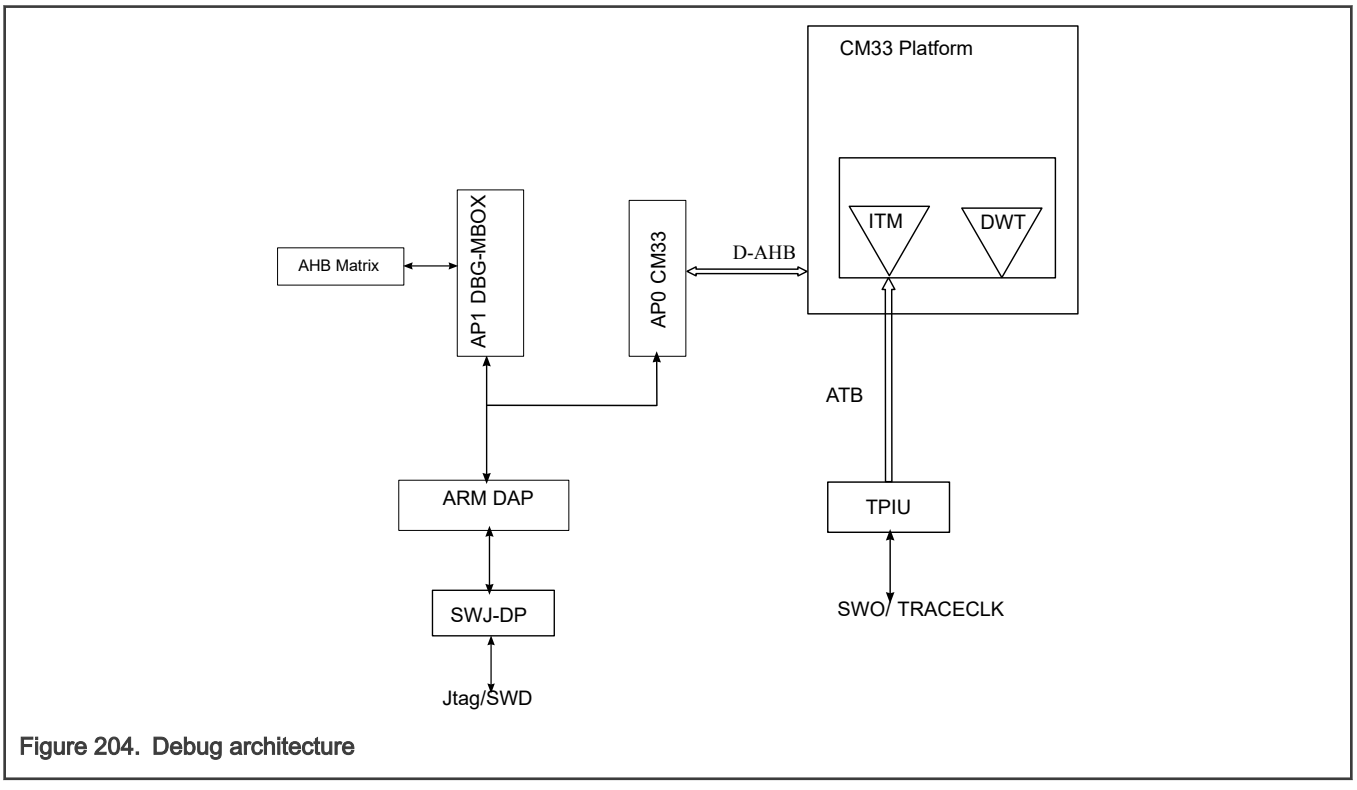


Figure 204. Debug architecture

46.2 Debug/Trace component

46.2.1 D-AHB interface

The 32-bit Debug AHB (D-AHB) interface implements the AMBA 5 AHB protocol. It can be used with a CoreSight AHB-AP to provide debugger access to all processor control and debug resources, and a view of memory that is consistent with that observed by load/store instructions acting on the processor.

46.2.2 AHB-AP debug access port

The AHB-AP is an optional debug access port into the processor system that provides access to all memory and registers in the system, including processor registers through the SCS. System access is independent of the processor status. SWJ-DP is used to access the AHB-AP.

46.2.3 Data Watchpoint and Trace

A Reduced DWT contains two comparators (DWT_COMP0 to DWT_COMP1). These comparators support the following features:

- Hardware watchpoint support.
- Hardware trace packet support, only if it includes an ITM.
- Cycle counter matching support (DWT_COMP0 only).
- Instruction address matching support.
- Data address matching support.
- Data value matching support (DWT_COMP1 only).
- Linked/limit matching support (DWT_COMP1 only).

The DWT contains counters for:

- Cycles (CYCCNT)
- Folded Instructions (FOLDCNT).
- Additional cycles required to execute all load or store instructions (LSUCNT).
- Processor sleep cycles (SLEEPCNT).
- Additional cycles required to execute multi-cycle instructions and instruction fetch stalls (CPICNT)
- Cycles spent in exception processing (EXCCNT).

The user can configure the DWT to generate PC samples at defined intervals, and to generate interrupt event information. The DWT provides periodic requests for protocol synchronization to the ITM and the TPIU.

46.2.4 In System Programming - Access Port (ISP-AP)

The ISP-AP provides a useful interface between debugger and the core and controls the debug information.

46.2.5 Cortex M33 debug

This section covers the debug mechanisms associated with the CM33 processor.

46.2.5.1 CM33 software and hardware trace

Refer Figure 204 to see data flow of software debug and trace information for the Cortex-M33 core on this device.

46.2.5.2 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.

46.2.5.3 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.
- Global timestamping: External timestamps can be generated from OS Timer.

Trace data from ITM will be forwarded to either the TPIO or the SWO-TPIU and streamed out via the trace port. Trace data from ITM can also be collected to the embedded trace buffer.

46.3 Test and debug port connectivity

The following figure shows the connectivity of test and debug port on this device.

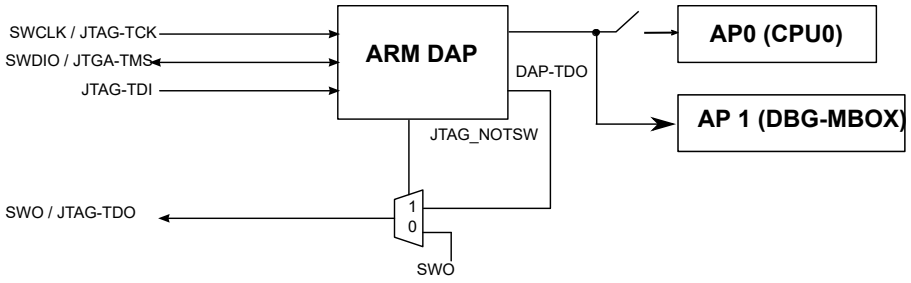


Figure 205. Test and debug port connectivity

The JTAG and SWD signals are summarized in the following table.

Table 342. JTAG/SWD signals summary

Pin Name	JTAG		SWD		Internal pull up/down
	Type	Description	Type	Description	
TMS/SWDIO	I/O	Mode selection	I/O	Data	Pull-up
TCLK/SWCLK/ TRACECLKIN	I	Clock	I	Clock	Pull-down
TDI	I	Data input	--	--	Pull-up
TDO	O	Data output	--	--	N/A

JTAG mode is the default connection of the JTAG/SWD port. The following sequence of events would switch the debug port to SWD mode.

JTAG-to-SWD change sequence has following steps:

- Send more than 50 TCLK cycles with TMS(SWDIO) = 1
- Send the 16-bit sequence on TMS(SWDIO) = 0111_1001_1110_0111 (MSB transmitted first)
- Send more than 50 TCLK cycles with TMS(SWDIO) = 1

Following is the JTAG ID table for this device.

JTAG ID	Version	Part number	Manufacture	mandatory
Value	0000b	TBD	000 0001 0101b	1b

46.3.1 Joint Test Access Group (JTAG)

46.3.1.1 JTAG instruction registers

The JTAG TEST TAP is connected in parallel with Arm DAP controller, that is, the JTAG-DP port of the DAP. The Instruction Register (IR) length is 4-bits. The JTAGC IR codes overlay the DAP controller IR codes. JTAGC uses twelve instructions and the DAP uses the remaining four instructions. The outputs of the TAPs (TDO) are multiplexed based on the IR code which is selected. This design is fully JTAG compliant and appears to the JTAG chain as a single TAP.

The following table gives the IR codes for the system JTAG controller.

Table 343. JTAG TEST TAP

Code	JTAG IR
4’b0000	IR_EXTEST
4’b0001	SAMPLE/PRELOAD
4’b0010	IR_HIGHZ
4’b0011	IR_CLAMP
4’b0100	IR_IDCODE
4’b0101	Reserved
4’b0110	Reserved
4’b0111	Reserved
4’b1000	ARM-DAP IR_ABORT
4’b1001	Reserved
4’b1010	ARM-DAP IR_DPACC
4’b1011	ARM-DAP IR_APACC
4’b1100	Reserved
4’b1101	Reserved
4’b1110	ARM-DAP IR_DAP_ID
4’b1111	ARM-DAP BYPASS

46.3.2 Debug access port

Debug Access Port (DAP) is a standard ARM CoreSight component. The DAP provides multiple master driving ports, all 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 344. 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. It offers JTAG AP to allow adding auxiliary JTAG TAPs. For more information on DAP or JTAG devices under JTAG AP, refer to ARM Debug Interface v5 Architecture specification on arm.com

The AHB AP's AHB transfers are burst size of 1 only, no out of order transactions, and no multiple outstanding accesses. The APB AP is used to access debug components of the system trace like TPIU and ETB.

The exported DAP bus is used to host AHB AP (integrated as part of CM33 integrations) and the Miscellaneous Debug Module Access Port (MDM AP). The MDM-AP implement registers which are used for Test mode control on the device. The MDM-AP hosts system level JTAG status and control registers (refer to Debug Status and Control Registers) which can be used for cross triggering, synchronized debug, low power and other miscellaneous control and status. 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 exported DAP bus is used to host AHB AP (integrated as part of CM33 integrations). The selection of different access ports in the DAP is done based on the APSEL value set in the SELECT register of SWJ-DP.

46.4 Debug ROM tables

The debug ROM tables hold information about different debug components and help identify them. In this chip, this table resides in the CM33 core and contains entries for CM33 debug components.

Table 345. Debug ROM table

Processor		Start address
CPU ROM		0xE00FE000
	CM33 processor ROM	0xE00FF000
	TPIU	0xE0040000

46.5 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

46.5.1 Clock gated modes

If the SWJDAP debug power up request is high, and CMC_DBGCTL[SOD] is clear, when the system attempts to enter STOP mode, the DAP clock and the FCLK 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.

46.6 Halting execution immediately following ROM execution

Traditionally, debug systems may set a vector catch at the reset vector to break code execution, but the chip ROM prevents this method. To allow the debug system to halt execution immediately after the ROM completes preparations following a debug session request, set a watchpoint on a read access to address 4009123Ch.

The debugger should use this reset sequence:

1. Set the data watchpoint to halt the core on a read of address location 4009123Ch.
2. If all data watchpoint comparators are occupied, back-up one of the watchpoint settings and replace it with the above watchpoint location.
3. Reset the core and peripherals by setting AIRCR[SYSRESETREQ].
4. Wait for 100 msec to allow ROM to re-enable debug access.
5. Verify that the core has halted at the watchpoint by checking the DHCSR register.
6. If DHCSR read times out or returns an error response, the ROM has entered an ISP command-handling loop due to an invalid image in boot media.
    a. Execute start debug session sequence.
    b. Wait for 10 msec.
    c. Enable the Cortex-M33 AP.
    d. Read DHCSR and issue HALT to Cortex-M33 AP.
7. Clear data watchpoint added in step 1. If watchpoint was set as described in step 2, then restore the watchpoint configuration.

Chapter 47
Debug Mailbox

47.1 Chip-specific Debug Mailbox information

Table 346. Reference links to related information

Topic	Related module	Reference
Full description	Debug Mailbox	Debug Mailbox
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

47.1.1 Module instances

This device has one instance of the Debug Mailbox module.

47.2 Overview

This chapter outlines the debug capabilities implemented in this chip through DBGMB. Debugging uses the Arm Serial Wire Debug (SWD) interface. This chapter is for debug tool developers and assumes that you know the Arm SWD interface and CoreSight (TM) debug and trace technology.

DBGMB is accessible through two paths:

- By software running on the chip using the DBGMB register interface as the access method.
- By tools software running on a PC connected to a device over SWD pins. These tools access DBGMB using the Arm Debug Interface (ADIv5) specification where the individual nodes or ports are called Access Ports (AP). In this context, we are using the DBGMB AP (DM-AP).

47.2.1 Block diagram

This figure shows the top-level debug ports and connections.

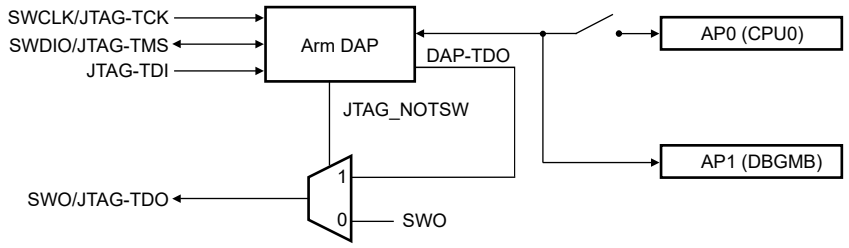


Figure 206. Serial Wire Debug (SWD) internal connections

47.2.2 Ports

For JTAG identifier details, see the "System Controller (SYSCON)" chapter.

47.2.3 Features

- Support for Arm SWD mode
- Cortex-M33 CPU instruction trace capability via trace port, with output via a serial wire viewer
- Direct debug access to all memories, registers, and peripherals
- No target resource requirements for a debugging session
- Support for setting instruction breakpoints
- Support for setting data watchpoints, which you can use as triggers
- Optional additional software-controlled trace for the CPU via the instrumentation trace macrocell (ITM)

47.3 Functional description

47.3.1 Basic configuration

This chip supports Arm's serial wire debug (SWD) interface. SWD is the default function for pins PIO0_1 (SWCLK) and PIO0_0 (SWDIO) after a reset. If you use serial wire output (SWO), you must enable it in the application code by selecting the SWO function on PIO0_2 or PIO0_6 and enabling the trace clock (see External signals).

The ROM controls debug access via a remote host, and it is enabled only when permitted through the chip configuration and when you follow the correct protocol to initiate a debug session. If you have configured the chip for debug authentication, you must initiate a debug session following the correct authentication sequence. When a chip is in the development life-cycle state, use the debug session protocol described in Debug session protocol.

47.3.2 Debug Access Port (DAP)

The DAP with a serial wire port (SWJ-DP) interprets incoming data and routes it to the appropriate AP. External signals describes the external I/O pins that interface with DAP. The DAP block is always enabled, but you can use the I/O pins that provide access to the SWD signals for other functions controlled by software.

47.3.3 Arm Cortex-M33 access port

The debug access port (DAP) for the Arm Cortex-M33 processor is disabled during power-on reset (POR) or during the assertion of the reset pin. The ROM enables the DAP when you follow the correct debug initiation procedures. If you are not using the DAP, you can use the debug enablement protocol to initiate a debug session.

The debug authentication process allows control of the DBGEN, NIDEN, SPIDEN, and SPNIDEN authentication signals connected to Cortex-M33 as described below.

Lifecycle/RoP state management controls these signals. See Lifecycle/RoP chapter.

Table 347. Authentication signals

Signal	Description
DBGEN	Invasive debugging • Breakpoints and watch points halt the processor on a specific activity. • A debug connection examines and modifies registers and memory, and it provides single-step execution.
NIDEN	Noninvasive debugging • Collects information on instruction execution and data transfers. • Delivers trace in real time to off-chip tools to merge data with source code on a development workstation for future analysis.

47.3.4 Debugger mailbox access port (DM-AP)

The DM-AP provides a register-based mailbox accessible by CPU0 and the device debug port (DP) of the MCU. This port is always enabled. An external host communicating through the SWD interface can exchange messages and data with the boot code executing from the ROM on CPU0. This port implements the NXP debug authentication protocol.

The boot ROM implements a debug mailbox protocol to interact with tools via the SWD interface.

The debug mailbox protocol has the following features:

- Request-and-response based, where the requests and responses use the same basic structure
- Support for relatively large command and response data
- 32-bit word alignment for all commands and responses
- Support for data above 32 bits via an ACK_TOKEN that moderates the transfer in 32-bit value chunks

47.3.5 Debug session protocol

When DBGMB fetches instructions from the ROM address range during boot, the DAP of CPU0 is disabled, regardless of device life-cycle state settings. This document refers to this mechanism as Boot-ROM protection. The method to initiate a debug session varies depending on the device state and intended debugging scenario. The scenarios described in the rest of these sections are:

Table 348. Debug session scenarios

Scenario	Description
Debug session with uninitialized or invalid image or ISP mode	If DBGMB detects no valid image, the ROM proceeds into In-System Programming (ISP) mode and waits to be booted via one of its serial interfaces. In ISP mode, the debug interface is disabled.
Debug session with valid application	Program control is in ISP mode, initiated because the ISP pin is asserted on the device at reset.
Debug session attaching to a running target	Connecting to a device with the intent to debug without writing an image (also called a debug attach).
Halting execution immediately following ROM execution	Connecting to a device running a valid application, with the intent to write a new application.

47.3.5.1 Debug session with uninitialized or invalid image or ISP mode

When the chip boots and the boot media does not contain a valid image, the ROM-based program control enters ISP mode. DBGMB disables debug access for security reasons. The chip also enters into ISP mode when the ISP has been asserted as the chip leaves reset. This section describes how to establish a debug session for these scenarios.

To ensure that the state machine controlling debug mailbox commands is in a known state, the debugger can reset this logic via the following steps:

1. Write 1 to CSW[RESYNCH_REQ] to request resynchronization.
2. Write 1 to CSW[CHIP_RESET_REQ] to reset the chip.
3. Read Command and Status Word (CSW).
4. Wait until the chip has completed resynchronization, identified by reading a value of 0 from the lower 16 bits of CSW.

To start a debug session and control the exchange of debug information, the debugger uses the DM-AP commands:

1. Following a successful initial resynchronization, communicate with the chip via 32-bit DM-AP commands to the REQUEST register in the debug mailbox.
2. Read results via the RETURN register. To ensure that the transactions have completed successfully, the debugger must poll the RETURN register as it polled the CSW register following a resynchronization request.
3. View the results in Response packet.

To initiate a debug session over SWD while debug is disabled:

1. The debug system must issue a Start Debug Session command to the ROM-based boot code, following the Debug session protocol.
2. Upon receiving the command, the boot code disables any unwanted peripheral and manages NXP secrets before enabling debug access.
3. After enabling debug access, the ROM enters a while(1) loop.

Once the Start Debug Session command and chip reset have executed successfully, the AP for CPU0 is accessible. You can use it to set breakpoints and perform other tasks, as with other Arm Cortex-M devices.

Following a successful debug connection, a flashloader is loaded into RAM to program the application to be debugged and to set required breakpoints in the code. After this process, a SYSRESET_REQ command must be issued to verify that the ROM fully executes (similar to a deployed end application) before reaching the downloaded application.

The code sample below shows how to initiate a debug session for the scenarios described above:

```
// 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 1
WriteDP 1 0x020000F0
// The returned AP ID should be 0x002A0000
value = ReadAP 3
print "AP ID: ", value
```

```
// Select DM AP index 1
WriteDP 1 0x02000000
// Write DM RESYNC_REQ + CHIP_RESET_REQ
WriteAP 0 0x21
// Poll CSW register (0) for zero return, indicating success
value = -1
while value != 0 {
value = ReadAP 0
}
print "RESYNC_REQ + CHIP_RESET_REQ: ", value
// 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
```

47.3.5.2 Debug session with valid application

In this case, the application has not disabled debug. You can access the CPU0 AP and set breakpoints without resynchronizing the mailbox hardware or issuing a Debug Session Request. You can use the methods described in Debug session with uninitialized or invalid image or ISP mode to simplify debug support implementations.

47.3.5.3 Debug session attaching to a running target

In this scenario, the device has booted and is running an application that has not disabled debug. The host system is attempting to connect to the device without resetting it and without updating the application. In this case, the CPU0 AP should be accessible and you can set breakpoints without resynchronizing the mailbox hardware.

47.3.5.4 Halting execution immediately following ROM execution

Traditionally, debug systems can set a vector catch at the reset vector to break code execution, but the chip ROM prevents this method.

The debugger must use this reset sequence:

Table 349. Reset procedure

Step	Purpose	Programming	Notes
1	Allow the debug system to halt execution immediately after the ROM completes the preparations associated with a debug session request.	Set the data watchpoint on a read to address location 0x40091040.	—
2	—	If all data watchpoint comparators are occupied, backup one of the watchpoint settings and replace it with the above watchpoint location.	—
3	Reset the core and peripherals.	Write 1 to the Arm Cortex-M33 field AIRCR[SYSRESETREQ].	—
4	Allow the ROM to reenable debug access.	Wait for 100 ms.	—
5	Verify that the core has halted at the watchpoint.	Read the Arm Cortex-M33 register DHCSR.	If the DHCSR read times out or returns an error response, the ROM has entered an ISP command-handling loop due to an invalid image in boot media. Proceed to Step 6, otherwise, proceed to Step 10.
6	—	Execute the start debug session sequence described in Debug session with uninitialized or invalid image or ISP mode.	—
7	—	Wait for 10 ms.	—
8	—	Enable the Cortex-M33 AP.	—
9	—	Read DHCSR and issue a HALT command to Cortex-M33 AP.	—
10	—	Clear data watchpoint added in step 1. If you set the watchpoint as described in step 2, restore the watchpoint configuration.	—

47.3.6 Reset handling

The debug domain (DP, Cortex-M33 AP, DM-AP) is reset upon POR or pin reset (assertion of external nRESET). On other resets, the debug domain retains its state. The defined breakpoints and watchpoints persist even when the debugging tool issues a reset to the device.

47.3.7 Mailbox commands

This section describes the request and response message formats and available mailbox commands.

47.3.7.1 Request packet layout

The first word transmitted in a request is a header word containing the command ID and the number of following data words. The command packet is sent to the device by writing 32 bits at a time to the REQUEST register. When sending command packets greater than 32 bits, the debugger must read an ACK_TOKEN in the RETURN register before writing the next 32 bits.

The 32-bit words quantified by the header follow the header itself.

Table 350. 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:

```
struct dm_request {
        uint16_t commandID;
        uint16_t dataWordCount;
        uint32_t data[ ];
        };
```

47.3.7.1.1 DM-AP commands

DM-AP commands are written to the REQUEST register.

See the Boot ROM chapter of this chip for DM-AP commands.

47.3.7.2 Response packet layout

The first word transmitted in a response is a header word containing the command status and the number of following data words. The command response can be read 32 bits at a time through Return Value (RETURN). The initial 32 bits contains the response header, as shown in the table below. When reading a response longer than 32 bits, the debugger writes the ACK_TOKEN to REQUEST after every read until the full response packet is received.

NOTE

To support legacy LPC command and response values, Bit_31 in the header indicates that the response follows the new protocol structure (see Table 352). This bit is 1 when using the new protocol.

Table 351. Response register byte description

Word	Byte 0	Byte 1	Byte 2	Byte 3
0	bits[7:0]:commandStatus[7:0]	bits[7:0]:commandStatus[15:8]	bits[7:0]:dataWordCount[7:0]	bits[6:0]:dataWordCount[14:8]

Table continues on the next page...

Table 351. Response register byte description (continued)

Word	Byte 0	Byte 1	Byte 2	Byte 3
				bits[7]:new_protocol[15]
1	*data*	—	—	—

The C structure definition for a response is:

```
struct dm_response {
        uint16_t commandStatus;
        uint16_t dataWordCount;
        uint32_t data[];
        };
```

47.3.7.2.1 Response packet

You can read the command response 32 bits at a time through Return Value (RETURN). The initial 32 bits contain the response header, as shown in the table below. When reading a response greater than 32 bits, the debugger writes the ACK_TOKEN to Request Value (REQUEST) after every read of RETURN until the full response packet is received.

Table 352. Response Packet

Bits	Field	Description
31	LONG_RESP	Long response packet indicator; also called the new protocol indicator
[30:16]	REMAIN_TRANS	Number of times the debugger must read RETURN to receive remaining response data. The debugger writes ACK_TOKEN to REQUEST after every read of RETURN until the full response packet is received. **NOTE** REMAIN_TRANS is valid only when ERROR_SRESP is 0 and LONG_RESP is 1.
[15:0]	ERROR_SRESP	Short response data or error code If bit_20 is not 1, this field is interpreted as short response data. For example, this field returns the CRP level for a GET_CRP_LEVEL command. Error codes • 0000h: Command succeeded. • 0001h: Debug mode not entered. This value is returned when other commands are sent prior to the Enter DM-AP command. • 0002h: Command is not supported. • 0003h: Communication failure. ACK_TOKEN is missing during data transactions.

47.3.7.3 ACK_TOKEN

The ACK_TOKEN provides an acknowledgment to the sender during debug mailbox data transactions. DBGMB uses the acknowledgment in the following ways:

- When the debugger issues a command with parameter data, it waits for the ACK_TOKEN (read through Return Value (RETURN)) before writing the next 32-bit value to Request Value (REQUEST).

- When the debugger receives a long response packet, it writes the ACK_TOKEN to REQUEST before reading the next 32-bit value RETURN.

Table 353. ACK_TOKEN

Bits	Field	Description
[31:16]	REMAIN_TRANS	Number of remaining data transactions
[15:0]	ACK_MARKER	Acknowledgment marker; must always be A5A5h

The C structure definition for the ACK_TOKEN is:

```
struct dm_ack_token {
        uint16_t token; /* always set to A5A5h */
        uint16_t remainCount; /* count of remaining word */
        };
```

47.3.7.4 Error handling

When an overrun occurs from either side of the communication, DBGMB sets the appropriate error flag in Command and Status Word (CSW). The state machine hardware prevents further communication in either direction. The debugger must start with a new resynchronization request to clear the error flag.

47.4 External signals

Table 354 shows the signals related to the debug process. A trace using the serial wire output has limited bandwidth.

Table 354. Serial wire debug signals

Signal	I/O	Description
SWCLK	I	Serial wire clock. Provides the clock for the SWD debug logic in Serial Wire Debug (SWD) mode. SWCLK is the default signal for its pin. At the release of reset, the pin is pulled down internally.
SWDIO	I/O	Serial wire debug data input and output. Used by an external debug tool to communicate with and control the part. SWDIO is the default signal of its pin. At the release of reset, the pin is pulled up internally.
SWO	O	Serial wire output. Optionally provides data from the Instrumentation Trace Macrocell (ITM) for an external debug tool to evaluate. See the chip-specific DBGMB information for the clocking required to enable SWO.

47.5 Memory map and register definition

47.5.1 DBGMB register descriptions

47.5.1.1 DBGMB memory map

DebugMailbox0 base address: 4010_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Command and Status Word (CSW)	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	Request Value (REQUEST)	32	RW	0000_0000h
8h	Return Value (RETURN)	32	RW	0000_0000h
FCh	Identification (ID)	32	R	002A_0000h

47.5.1.2 Command and Status Word (CSW)

Offset

Register	Offset
CSW	0h

Function

Contains command and status bits to facilitate communication between DBGMB and the chip.

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										CHIP_RE...	SOFT_RE...	AHB_OR_...	DBG_OR_...	REQ_PEN...	RESYNCH...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 CHIP_RESET_REQ	Chip Reset Request Causes the chip (but not the DM-AP) to be reset by generating SYSRESET_REQ. This field is write only. 0b - No effect 1b - Reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SOFT_RESET	Soft Reset Resets the DM-AP. This field is write only by the chip. 0b - No effect 1b - Reset
3 AHB_OR_ERR	AHB Overrun Error Indicates whether an AHB overrun has occurred: the chip has overwritten a RETURN value before DBGMB read the RETURN value. 0b - No overrun 1b - Overrun occurred
2 DBG_OR_ERR	DBGMB Overrun Error Indicates whether a DBGMB overrun has occurred: DBGMB has overwritten a REQUEST value before the chip read the REQUEST value. 0b - No overrun 1b - Overrun occurred
1 REQ_PENDING	Request Pending Indicates a pending request for DBGMB: a value is waiting to be read from Request Value (REQUEST). 0b - No request pending 1b - Request for resynchronization pending
0 RESYNCH_REQ	Resynchronization Request Requests a resynchronization. 0b - No request 1b - Request for resynchronization

47.5.1.3 Request Value (REQUEST)

Offset

Register	Offset
REQUEST	4h

Function
Used by DBGMB to send action requests to the chip.

Diagram

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

Fields

Field	Function
31-0 REQUEST	Request Value Indicates the request value. This field reads 0 when no new request is present. The chip clears this field. DBGMB can read back this field to confirm communication.

47.5.1.4 Return Value (RETURN)

Offset

Register	Offset
RETURN	8h

Function

Provides the responses from the chip to DBGMB.

Diagram

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

Fields

Field	Function
31-0 RET	Return Value Indicates the return value, which is a response from the chip to DBGMB. If no new data is present, DBGMB reading is stalled until new data is available.

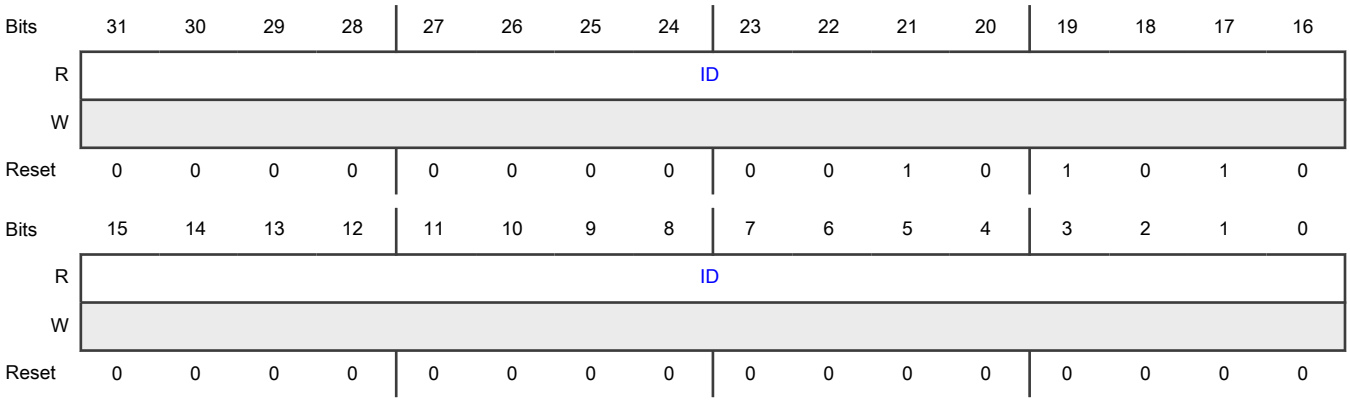
47.5.1.5 Identification (ID)

Offset

Register	Offset
ID	FCh

Function
Provides an identification of the DM-AP interface.

Diagram



Fields

Field	Function
31-0 ID	Identification Value Provides an identification of the DM-AP interface. The Arm Debug Interface Specification version 5.0 (ADIv5) requires every AP to implement an AP identification register, at offset FCh. This register is the last register in the AP register space. This ID register is only readable by external tools (a debug dongle) when identifying the Access Port (AP). For the debug mailbox AP, the identification value is 002A_0000h. This register is not readable from on-chip software. If you attempt a read, you receive a value of 0000_0000h.

Chapter 48
Cyclic Redundancy Check (CRC)

48.1 Chip-specific CRC information

Table 355. Reference links to related information

Topic	Related module	Reference
Full description	CRC	CRC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

48.1.1 Module instances

This device has one instance of the CRC module, CRC0.

48.2 Overview

CRC generates 16-bit or 32-bit CRC codes output for error detection. You can calculate these codes for up to 32 bits input data at a time.

CRC provides a programmable polynomial and other parameters to meet 16-bit or 32-bit CRC standards.

48.2.1 Block diagram

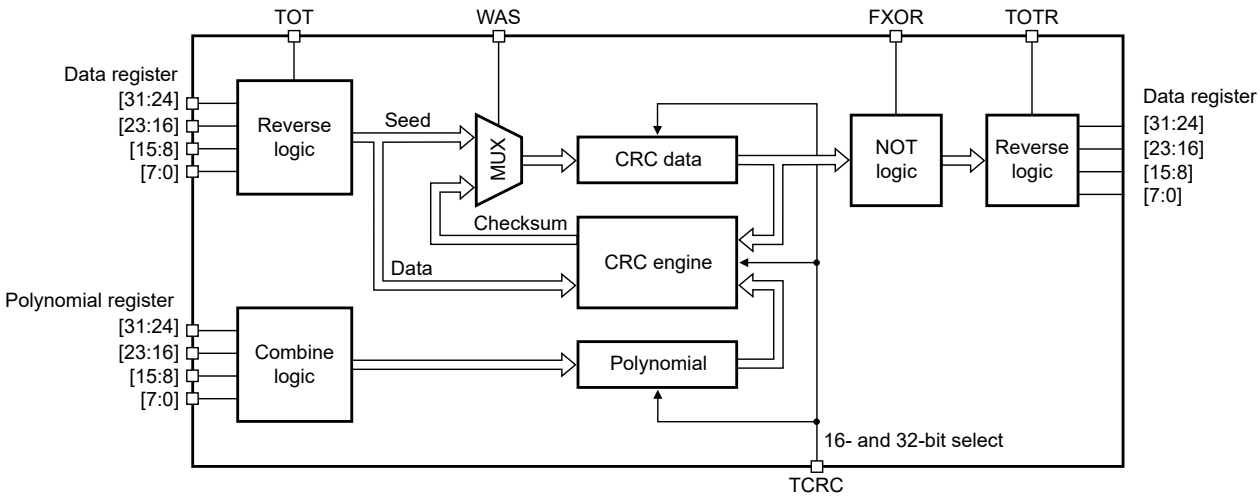


Figure 207. Block diagram

48.2.2 Features

CRC has the following features:

- Hardware CRC generator circuit using 16-bit or 32-bit programmable shift registers
- Programmable initial seed value and polynomial
- Transpose of input or output data (CRC result) in bitwise or bytewise (this option is required for certain CRC standards. You cannot perform a bytewise transpose operation when accessing DATA via 8-bit access.)
- Invert final CRC result
- 32-bit CPU register programming interface

48.3 Functional description

48.3.1 Modes of operation

The following sections describe various modes of operation that affect the functionality of CRC: Run mode and Low power mode.

48.3.1.1 Run mode

Run mode is the basic mode of operation.

48.3.1.2 Low power mode

When the chip enters the lower power mode, the CRC module clock (ipg_clk and ipg_clk_s) is disabled and the in-progress CRC calculation stops. The calculation resumes after the CRC module clock is enabled or the chip exits low power mode via system reset.

48.3.2 CRC calculations

In 16-bit and 32-bit CRC modes, you can program data values as 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes lead to an incorrect CRC calculation.

48.3.2.1 Calculating a 16-bit CRC

Perform these steps to calculate a 16-bit CRC:

1. Write 0 to CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in Control (CTRL) as required for the CRC calculation.
3. Write a 16-bit polynomial to GPOLY[LOW].

   GPOLY[HIGH] 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 start writing data values.
7. Write data values into DATA[LU], and DATA[LL]

   CRC is calculated on every data value write and the intermediate CRC result is stored back into DATA[LU] and DATA[LL]
8. After writing all the data values, read the final CRC result from 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.

48.3.2.2 Calculating a 32-bit CRC

Perform these steps to calculate a 32-bit CRC:

1. Write 1 to CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in 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 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 .
8. After writing all the values, 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.

48.3.3 Transpose feature

Transpose is not enabled by default. However, CRC requires input data and/or final checksum to be transposed. You have an option to configure each transpose operation separately to meet CRC standards. The data is transposed on-the-fly while being read or written.

Some protocols use the little-endian format for data stream to calculateCRC. In this case, transpose flips bits.

48.3.3.1 Types of transpose

CRC provides several types of transpose to flip bits and/or bytes for both writing input data and reading result separately using the CTRL[TOT] and CTRL[TOTR] according to the CRC calculation being used.

The following types of transpose are available for writing to and reading from 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]}.

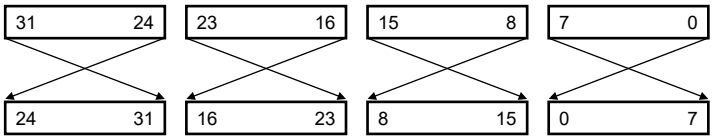


Figure 208. Transpose type 1b

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]}.

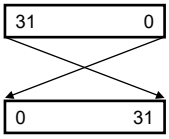


Figure 209. Transpose type 10b

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]}.

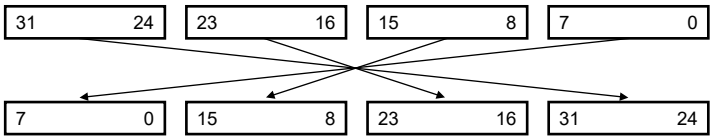


Figure 210. Transpose type 11b

NOTE

For 8-bit and 16-bit write accesses to 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 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.

48.3.4 Result complement

When CTRL[FXOR] = 1, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in Data (DATA) every time Data (DATA) is read. When CTRL[FXOR] = 0,, reading Data (DATA) accesses the raw checksum value.

48.3.5 Clocking

Table 356. CRC clocks

Type of clock	Description
Bus clock (ipg_clk/ipg_clk_s)	ipg_clk_s controls the access to the CRC registers. ipg_clk and ipg_clk_s function the CRC module.

48.3.6 Interrupts

This module has no interrupts.

48.4 External signals

There is no CRC signal that connects off chip.

48.5 Initialization

To enable CRC calculation, you must program:

- CTRL[WAS] .
- 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 registers.

After writing all the data, you must wait for at least two clock cycles to read the data from CRC Data (DATA) register.

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.

48.6 Use cases

The following tables use the little-endian format.

48.6.1 CTRL programming

The following table shows Control (CTRL) programming for 16-bit CRC.

Table 357. 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_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	0h	0h	1h
CRC-16_GENIBUS	1021h	FFFFh	0	0	FFFFh	0h	0h	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 Control (CTRL) programming for 32-bit CRC.

Table 358. 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
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

48.6.2 Expected read data fields

The following table shows the expected read data fields for 16-bit CRC.

Table 359. Expected read data fields for 16-bit CRC

Algorithm	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_X	[31:16] = Unknown [15:0] = Valid data
CRC16_DNP	[31:16] = Valid data [15:0] = Unknown
CRC-16_EN_13757	[31:16] = Unknown [15:0] = Valid data
CRC-16_GENIBUS	[31:16] = Unknown [15:0] = Valid data
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 359. Expected read data fields for 16-bit CRC (continued)

Algorithm	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 360. Expected read data fields for 32-bit CRC

Algorithm	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

48.7 Memory map and register descriptions

NOTE

You must reconfigure CRC engine in case an IPS transfer error occurs (ips_xfr_err).

The CRC module generates a transfer error in the following cases:

- Write accesses to the register addresses that are not mapped to the peripherals but included in the address spaces of the peripherals.

48.7.1 CRC register descriptions

48.7.1.1 CRC memory map

CRC0 base address: 4008_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Data (DATA)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Polynomial (GPOLY)	32	RW	0000_1021h
8h	Control (CTRL)	32	RW	0000_0000h

48.7.1.2 Data (DATA)

Offset

Register	Offset
DATA	0h

Function

Configures the value of 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 calculation.

In 16-bit CRC mode, 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 program to write 8 bits, 16 bits, or 32 bits in big endian order, 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 DATA 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. After writing all data, you must wait for at least two clock cycles to read the data from CRC data (DATA) register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HU								HL							
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	LU								LL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-24	Upper Part of High Byte

Table continues on the next page...

Table continued from the previous page...

Field	Function
HU	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), the values written to this field are part of the seed value when CTRL[WAS] = 1.
23-16 HL	Lower Part of High 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), the values written to this field are part of the seed value when CTRL[WAS] = 1.
15-8 LU	Upper Part of Low Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 1,the values written to this field are part of the seed value.
7-0 LL	Lower Part of Low Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 1, the values written to this field are part of the seed value.

48.7.1.3 Polynomial (GPOLY)

Offset

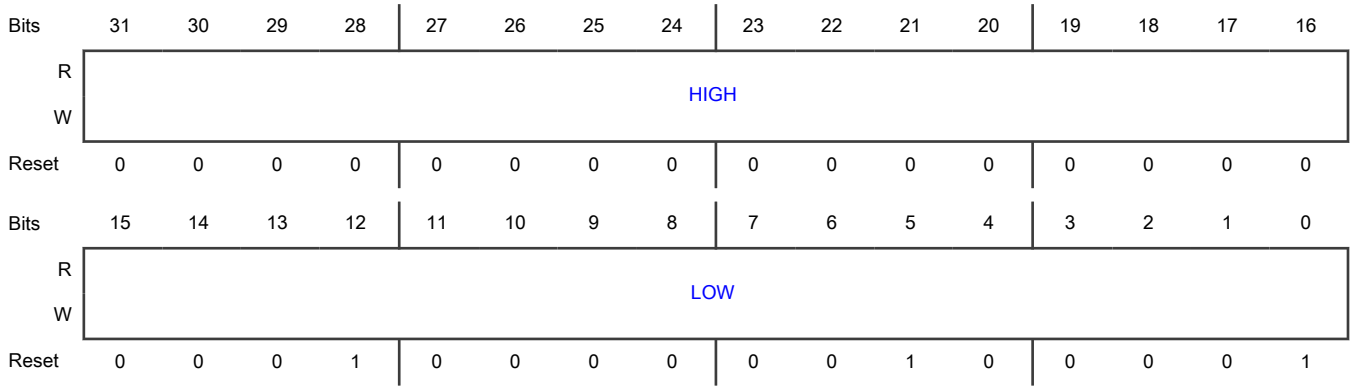
Register	Offset
GPOLY	4h

Function

Configures polynomial value for CRC calculation.

- Sets the upper 16 bits of polynomial that are used only in 32-bit CRC mode. Writes to this field are ignored in 16-bit CRC mode.
- Sets the lower 16 bits of polynomial that are used in both 16-bit and 32-bit CRC modes.

Diagram



Fields

Field	Function
31-16 HIGH	High 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 Half-Word Writable and readable in both 16-bit and 32-bit CRC modes.

48.7.1.4 Control (CTRL)

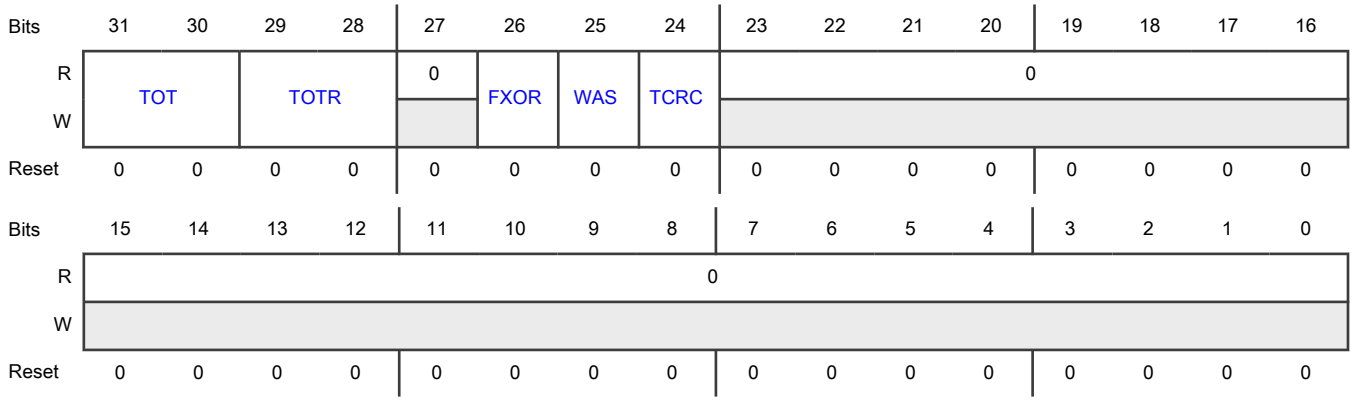
Offset

Register	Offset
CTRL	8h

Function

Sets control for CRC. 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 DATA.

Diagram



Fields

Field	Function
31-30 TOT	Transpose Type for Write Sets transpose type for the values written to DATA. See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed, but 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 Sets transpose type for the values read from DATA . See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed, but 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 - Disables XOR on reading data. 1b - Inverts or complements the read value of the CRC Data.
25 WAS	Write as Seed Specifies whether writes to DATA are data values or seed values. When this field = 1, the value that you write to is considered as seed value. When this field = 0, the value that you write to is considered as data for CRC calculation. 0b - Data values 1b - Seed values
24 TCRC	TCRC Defines the width of CRC. 0b - 16 bits 1b - 32 bits
23-0 —	Reserved

Chapter 49
Memory Block Checker(MBC)

49.1 Chip-specific MBC information

Table 361. Reference links to related information

Topic	Related module	Reference
Full description	MBC (TRDC)	MBC
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

The Memory Block Checker (MBC) is a sub-module of Trusted Resource Domain Controller (TRDC). TRDC is typically distributed across various sub-modules like MGR, DAC, MBC and MRC. However, sub-modules MGR, DAC, and MRC do not exist on this device and any references to them in TRDC chapter should be ignored. This chapter covers TRDC sub-module MBC and its features in detail.

49.1.1 Module instances

This device has one instance of the MBC module and one Domain on this chip. m and d in MBCm_DOMd_BLK_CFG_W are 0. The MBC is instanced between Flash Memory Controller (FMC) and AHB Matrix. MBC can control the Read, Write and Execute permission of each flash block, and the configuration can be locked.

49.1.2 MBC memory blocks

The granularity of MBC is described in the following table.

MBC0_MEMn	Memory	Block size
MEM0	Main flash	16 KB
MEM1	IFR0	8 KB
MEM2	IFR1	2 KB

49.1.3 Security considerations

MBC on this chip can't distinguish secure privilege, secure user, non-secure privilege, and non- secure user, when there is an access to flash, and MBC treat all access to flash as secure user level. Because of it, SUR/SUW/SUW bits in MBC Global Access Control registers control the permission to flash. User should not set the NonSecure Enable (NSE) bits in in MBC Global Access Control registers.

MBC controls the read, write and execute permission to flash. If an access doesn't have the permission to the flash region, MBC will generate interrupt. The domainX error registers are physically in the block checker.

NOTE

TZ-M feature of TRDC is not supported on this device.

49.2 Overview

The Memory Block Checker (MBC) implements access controls for on-chip internal memories based on a fixed-sized block format. MBC provides hardware access control for system bus references targeted at on-chip memory spaces.

Using the pre-programmed region descriptors with fixed-sized block format and their associated access rights, the MBC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references with sufficient access control rights are completed, while the references that are not mapped to any region descriptor or have insufficient rights are terminated with an access error response.

49.2.1 Block diagram

The Memory Block Checker (MBC) implements access controls for on-chip internal memories and slave peripherals based on a fixed-sized block format.

See Figure 211 for MBC block diagram focusing on topology and connections.

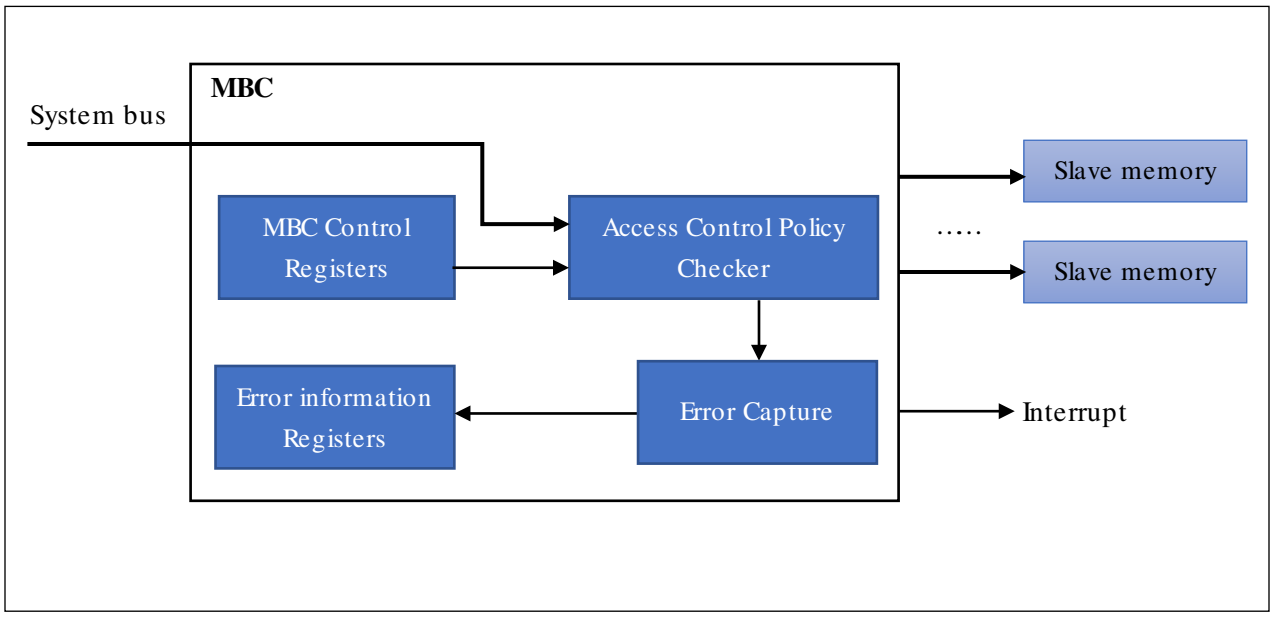


Figure 211. MBC block diagram

49.2.2 Features

Defines access rights to slave targets defined in MBCs for on-chip memories.

49.3 Functional description

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

49.3.1 Memory Block Checker (MBC)

The Memory Block Checker provides access control for system bus references targeted to on-chip internal memories . Using programmed block configuration registers which define the access rights per 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.

Refer to the chip-specific section to determine the number and size of blocks for each sub memory supported on this device.

Each MBCm_DOMd_BLK_CFG_W contains eight access control structures. Each structure is comprised of a 3-bit MBACSEL (Memory Block Access Control Select). The MBACSEL field selects a MBCm_MEMN_GLBACr register, which controls the read/write/execute/lock access to the corresponding block. Below table provides the eight possible access control combinations. In case you are looking for different access combination, such as write and execute allowed but read blocked, you can modify one of the unlocked configurations registers (MBC0_MEMN_GLBAC0 or MBC0_MEMN_GLBAC4 or MBC0_MEMN_GLBAC5) to 0x0000_3300.

Table below depicts eight possible access control combinations.

Table 362. Access control combinations

GAC Index	Register name	Read	Write	Execute	Lock	Description
0	MBC0_MEMN_GLBAC0 (0x0000_6600)	1	1	0	0	**Data flash and unlocked** (RW__). Default flash behavior with read and write access only. Instruction fetches (execute access) are blocked and generates security violation. Blocks assigned with this access control index can be changed to another index by software. This configuration register is unlocked and updatable by application.
1	MBC0_MEMN_GLBAC1 (0x8000_6600)	1	1	0	1	**Data flash and locked** (RW_L). Read and write operations are allowed but execution is blocked. Blocks assigned with this access control index cannot be changed to another index until next reset.
2	MBC0_MEMN_GLBAC2 (0x8000_5500)	1	0	1	1	**Read only Memory (ROM) and locked** (R_XL) Read & execute operations are allowed but write is blocked. Blocks assigned with this access control index cannot be changed to another index until next reset.
3	MBC0_MEMN_GLBAC3 (0x8000_4400)	1	0	0	1	**Data Read only memory (DROM) and locked** (R__L) Read operations is allowed but write & execute are blocked. Blocks assigned with this access control index cannot be changed to another index until next reset.
4	MBC0_MEMN_GLBAC4 (0x0000_5500)	1	0	1	0	**Read-Only-Memory (ROM) and un-locked** (R_X_) Read & execute operations are allowed but write is blocked. Blocks assigned with this access control index can be changed to another index by software. This configuration register is unlocked and updatable by application.
5	MBC0_MEMN_GLBAC5 (0x0000_1100)	0	0	1	0	**eXecute-only Memory (XOM) and un-locked** (__X_)

Table continues on the next page...

Table 362. Access control combinations (continued)

GAC Index	Register name	Read	Write	Execute	Lock	Description
						Execute operations are allowed but read & write are blocked. Blocks assigned with this access control index can be changed to another index by software. This configuration register is unlocked and updatable by application.
6	MBC0_MEMN_GLBAC6 (0x8000_1100)	0	0	1	1	**eXecute-Only-Memory (XOM) and locked** (__XL) Execute operations are allowed but read & write are blocked. Blocks assigned with this access control index cannot be changed to another index until next reset.
7	MBC0_MEMN_GLBAC7 (0x8000_0000)	0	0	0	1	**Hidden** (___L) Read, Write & execute operations are blocked. Once this access control index is assigned to a block, the memory range associated with block is hidden until next reset.

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.

49.3.1.1 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.

49.3.1.2 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 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.

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.

49.3.2 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 MBCn_MEM3_GLBCFG[CLRE] is cleared. This interrupt is in addition to interrupt generated by master after receiving bus error due to access violation by memory block checker . So, system may receive interrupt through different channels but from the same access violation at checker.

49.4 External signals

This module has no external signals.

49.5 Register descriptions

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.

49.5.1 MBC register descriptions

49.5.1.1 MBC memory map

MBC0.MBC base address: 4008_E000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MBC Global Configuration Register (MBC0_MEM0_GLBCFG)	32	R	000E_0040h
4h	MBC Global Configuration Register (MBC0_MEM1_GLBCFG)	32	R	000D_0004h
8h	MBC Global Configuration Register (MBC0_MEM2_GLBCFG)	32	R	000B_0004h
Ch	MBC Global Configuration Register (MBC0_MEM3_GLBCFG)	32	RW	0000_0000h
20h	MBC Global Access Control (MBC0_MEMN_GLBAC0)	32	RW	0000_6600h
24h	MBC Global Access Control (MBC0_MEMN_GLBAC1)	32	RW	8000_6600h
28h	MBC Global Access Control (MBC0_MEMN_GLBAC2)	32	RW	8000_5500h
2Ch	MBC Global Access Control (MBC0_MEMN_GLBAC3)	32	RW	8000_4400h
30h	MBC Global Access Control (MBC0_MEMN_GLBAC4)	32	RW	0000_5500h
34h	MBC Global Access Control (MBC0_MEMN_GLBAC5)	32	RW	0000_1100h
38h	MBC Global Access Control (MBC0_MEMN_GLBAC6)	32	RW	8000_1100h
3Ch	MBC Global Access Control (MBC0_MEMN_GLBAC7)	32	RW	8000_0000h
40h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0)	32	RW	0000_0000h
44h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W1)	32	RW	0000_0000h
48h	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
4Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W3)	32	RW	0000_0000h
50h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W4)	32	RW	0000_0000h
54h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W5)	32	RW	0000_0000h
58h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W6)	32	RW	0000_0000h
5Ch	MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W7)	32	RW	0000_0000h
180h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM1_BLK_CFG_W0)	32	RW	0000_0000h
1A8h	MBC Memory Block Configuration Word (MBC0_DOM0_MEM2_BLK_CFG_W0)	32	RW	0000_0000h

49.5.1.2 MBC Global Configuration Register (MBC0_MEM0_GLBCFG - MBC0_MEM3_GLBCFG)

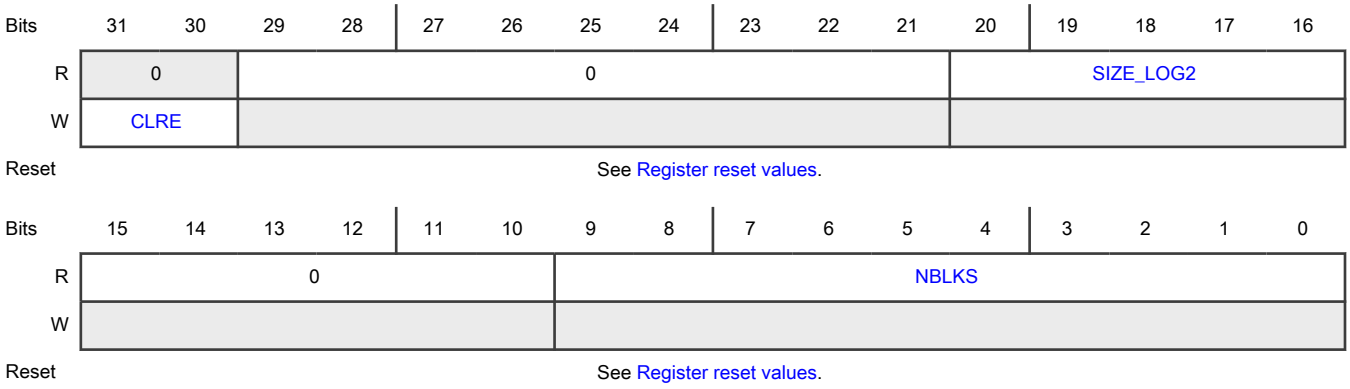
Offset

Register	Offset
MBC0_MEM0_GLBCFG	0h
MBC0_MEM1_GLBCFG	4h
MBC0_MEM2_GLBCFG	8h
MBC0_MEM3_GLBCFG	Ch

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	000E_0040h
MBC0_MEM1_GLBCFG	000D_0004h
MBC0_MEM2_GLBCFG	000B_0004h
MBC0_MEM3_GLBCFG	0000_0000h

Fields

Field	Function
31-30 CLRE	Clear Error This 2-bit, write-only field controls the clearing of an access violation error and any interrupt due to error, on writing 01b to this field. A write of any value other than 01b has no effect. **NOTE** This field is not supported in every instance. The following table includes only supported registers.
29-21 —	Reserved
20-16 SIZE_LOG2	Log2 size per block For example SIZE_LOG2=0x0C is 2^12=4 KB blocks.

Instance	Field supported in	Field not supported in
MBC0.MBC	MBC0_MEM3_GLBCFG	MBC0_MEM0_GLBCFG– MBC0_MEM2_GLBCFG

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-10 —	Reserved
9-0 NBLKS	Number of blocks in this memory

49.5.1.3 MBC Global Access Control (MBC0_MEMN_GLBAC0)

Offset

Register	Offset
MBC0_MEMN_GLBAC0	20h

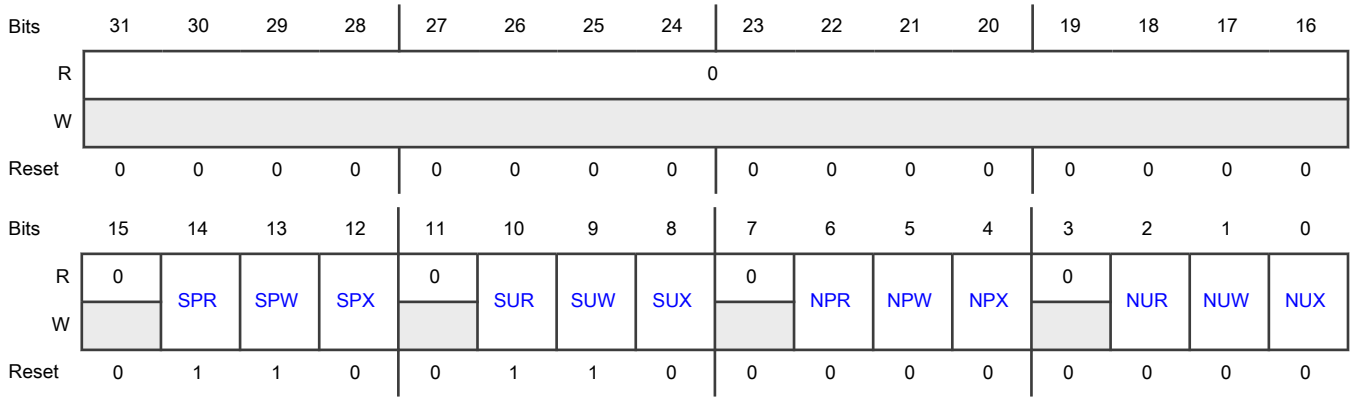
Function

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.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

Diagram



Fields

Field	Function
31-15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
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 0b - Execute access is not allowed in Secure User mode. 1b - Execute access is allowed in Secure User mode.
7 —	Reserved
6	NonsecurePriv Read NonsecurePriv read access control flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPR	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.

49.5.1.4 MBC Global Access Control (MBC0_MEMN_GLBAC1)

Offset

Register	Offset
MBC0_MEMN_GLBAC1	24h

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 control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

This register is not writable.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0														
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	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	1	1	0	0	1	1	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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 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	NonsecurePriv Execute

Table continues on the next page...

Table continued from the previous page...

Field	Function
NPX	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.

49.5.1.5 MBC Global Access Control (MBC0_MEMN_GLBAC2)

Offset

Register	Offset
MBC0_MEMN_GLBAC2	28h

Function

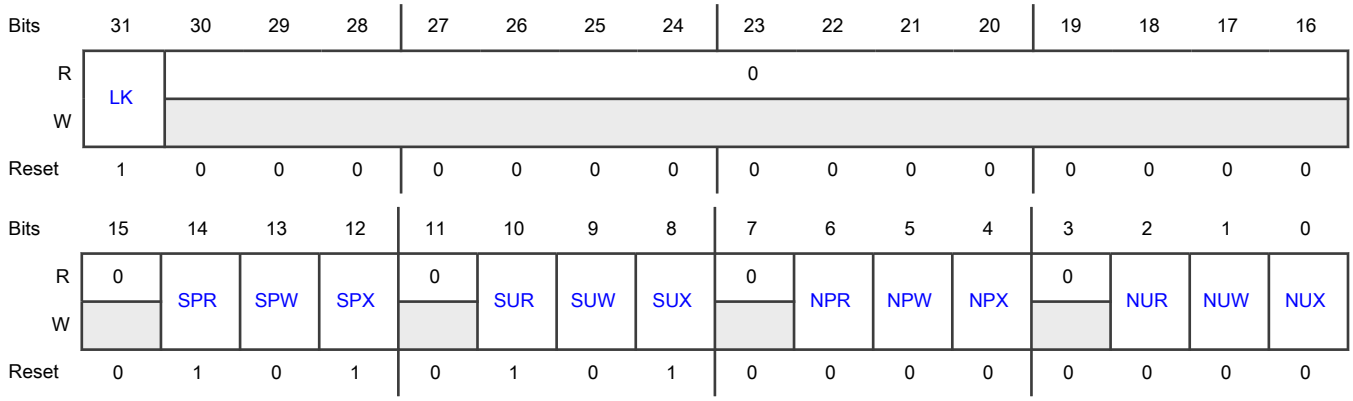
These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these control access flags and to lock the MBC_BLK_CFG[MBACSEL]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

This register is not writable.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

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.

49.5.1.6 MBC Global Access Control (MBC0_MEMN_GLBAC3)

Offset

Register	Offset
MBC0_MEMN_GLBAC3	2Ch

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 control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

This register is not writable.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0														
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	0	SPR	SPW	SPX	0	SUR	SUW	SUX	0	NPR	NPW	NPX	0	NUR	NUW	NUX
W																
Reset	0	1	0	0	0	1	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.

49.5.1.7 MBC Global Access Control (MBC0_MEMN_GLBAC4)

Offset

Register	Offset
MBC0_MEMN_GLBAC4	30h

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 control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

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	1	0	1	0	1	0	1	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	SecurePriv Read

Table continues on the next page...

Table continued from the previous page...

Field	Function
SPR	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 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
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.

49.5.1.8 MBC Global Access Control (MBC0_MEMN_GLBAC5)

Offset

Register	Offset
MBC0_MEMN_GLBAC5	34h

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 control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

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	1	0	0	0	1	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

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.

49.5.1.9 MBC Global Access Control (MBC0_MEMN_GLBAC6)

Offset

Register	Offset
MBC0_MEMN_GLBAC6	38h

Function

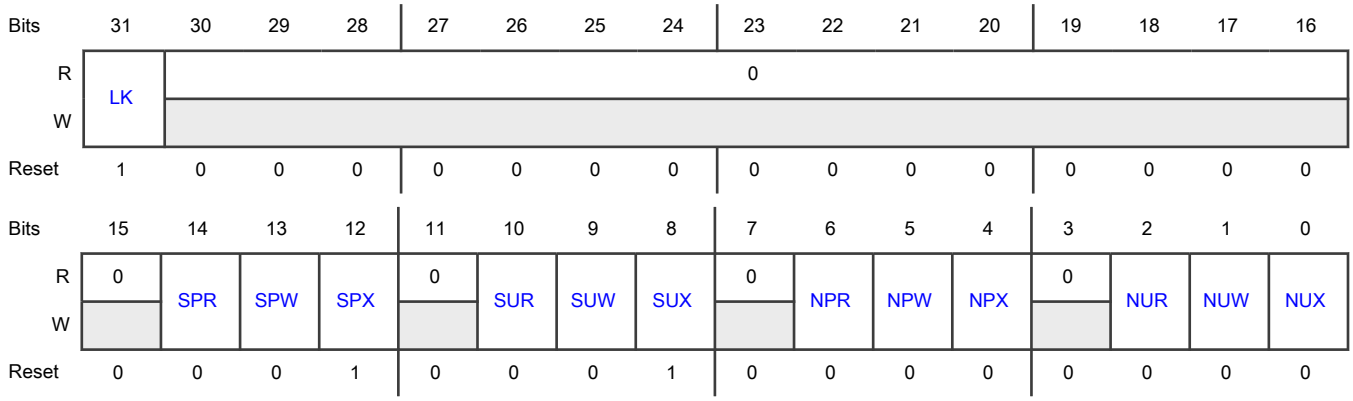
These fully programmable R/W fields define the R/W/X (read/write/execute) access control flags with each of the 4 supported operating modes: SecurePriv, SecureUser, NonsecurePriv, Nonsecure User. The lock bit provides the ability to lock these control access flags and to lock the MBC_BLK_CFG[MBACSEL]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

This register is not writable.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

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.

49.5.1.10 MBC Global Access Control (MBC0_MEMN_GLBAC7)

Offset

Register	Offset
MBC0_MEMN_GLBAC7	3Ch

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 control access flags and to lock the MBC_BLK_CFG[MBACSEL] field when a locked MBC_MEMN_GLBAC register is selected.

This register is not writable.

NOTE

The lock field (LK) in MBC0_MEMN_GLBAC1, MBC0_MEMN_GLBAC2, MBC0_MEMN_GLBAC3, MBC0_MEMN_GLBAC6 and MBC0_MEMN_GLBAC7 registers are set to lock state (LK= 1) at reset. Hence these registers cannot be changed and blocks that are assigned to one of these global access control indexes cannot change the access policy index until next reset.

Diagram

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

49.5.1.11 MBC Memory Block Configuration Word (MBC0_DOM0_MEM0_BLK_CFG_W0 - MBC0_DOM0_MEM2_BLK_CFG_W0)

Offset

Register	Offset
MBC0_DOM0_MEM0_BLK_CFG_W0	40h
MBC0_DOM0_MEM0_BLK_CFG_W1	44h
MBC0_DOM0_MEM0_BLK_CFG_W2	48h
MBC0_DOM0_MEM0_BLK_CFG_W3	4Ch
MBC0_DOM0_MEM0_BLK_CFG_W4	50h
MBC0_DOM0_MEM0_BLK_CFG_W5	54h
MBC0_DOM0_MEM0_BLK_CFG_W6	58h
MBC0_DOM0_MEM0_BLK_CFG_W7	5Ch
MBC0_DOM0_MEM1_BLK_CFG_W0	180h
MBC0_DOM0_MEM2_BLK_CFG_W0	1A8h

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

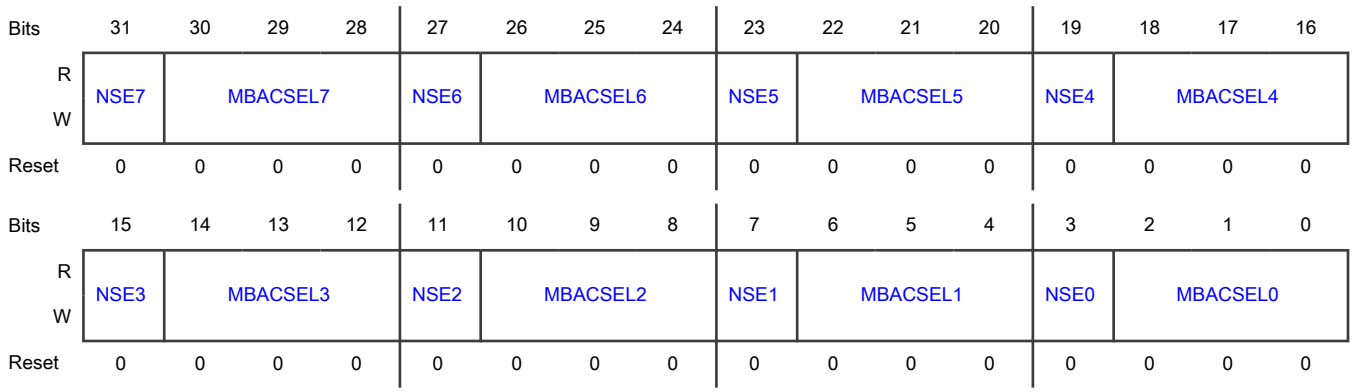
NOTE

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 363. 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								
W8	block 71	block 70	block 69	block 68	block 67	block 66	block 65	block 64
. . .								
W63	block 511	block 510	block 509	block 508	block 507	block 506	block 505	block 504

Diagram



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 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 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 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 MBACSEL6 = 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

Table continues on the next page...

Table continued from the previous page...

Field	Function
	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 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 MBACSEL5 = 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
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 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 MBACSEL4 = 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
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 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 MBACSEL3 = 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
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 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	Memory Block Access Control Select for block B

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBACSEL2	This field selects the global access control associated with block B. **NOTE** This field is locked (read-only) if MBACSEL2 = 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 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 MBACSEL1	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 MBACSEL1 = 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.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Secure accesses to block B 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 MBACSEL0 = 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

Chapter 50
Code Watchdog Timer (CDOG)

50.1 Chip-specific CDOG information

Table 364. Reference links to related information

Topic	Related module	Reference
Full description	CDOG	CDOG
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing

50.1.1 Module instances

This device has one instance of the CDOG module, CDOG0.

50.1.2 Clock gating

CDOG is clock gated when CPU is sleeping or in lower power mode.

50.2 Overview

CDOG helps protect the integrity of software by detecting unexpected changes (faults) in code execution flow. This module can be configured to reset or interrupt the processor core when it detects a fault.

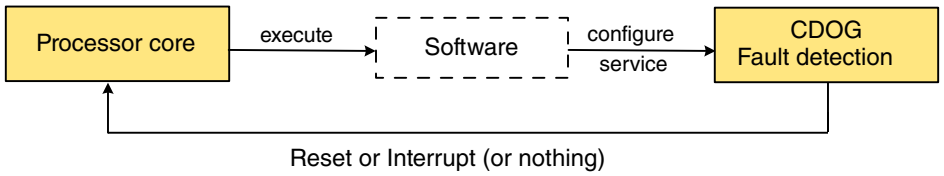


Figure 212. CDOG overview

50.2.1 Block diagram

The following block diagram shows the components of CDOG.

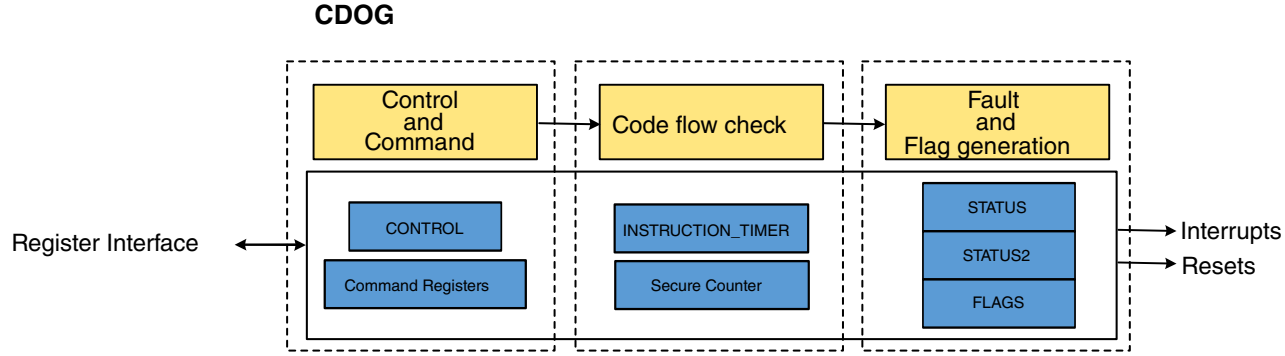


Figure 213. Block diagram of CDOG

50.2.2 Features

CDOG includes the following code flow and data integrity checking:

- Faults and flags configurable to generate a system reset, interrupts, or nothing
- Counters for statistics on code behavior patterns for fault types

50.3 Functional description

The following sections describe functional details of this module.

50.3.1 Code flow checking

CDOG provides two primary mechanisms for detecting low-cost fault attacks and the execution of unexpected instruction sequences:

- Secure Counter
- Instruction Timer (INSTRUCTION_TIMER)

50.3.1.1 Secure counter

The Secure Counter is a 32-bit accumulator that holds a dynamically changing value that can periodically be evaluated to determine if a program is executing as expected. If a mismatch is detected, a fault is generated.

- Secure Counter is an accumulator that when loaded with an initial value lets runtime software issue ADD & SUB commands to increment/decrement the counter.
- Periodically, a Secure Counter value check is initiated by writing the STOP, RESTART, or ASSERT16 command register. The value written to STOP, RESTART, or ASSERT16 is compared with the current value of the Secure Counter.
- If a mismatch is detected between the Secure Counter and the value written to STOP, RESTART, or ASSERT16 command register, the execution flow has potentially been altered by a fault attack or some other suspicious activity.

50.3.1.2 Instruction timer

The Instruction Timer is a 32-bit count-down timer that an application uses to set the number of instructions that software expects to execute. The Instruction Timer counts off the instructions as they are executed (clocks) and if the number is exhausted before the CDOG is serviced, a fault is generated.

The application programmer pre-loads the Instruction Timer with slightly more than the number of instructions in the next execution sequence. The CDOG detects cases where the counter is reset to 0 before all instructions execute, which may indicate that unauthorized instructions are being executed.

- Instruction Timer places a hard upper-limit on the interval between checks of the Secure Counter.

- The Instruction Timer can be programmed to either pause, or keep running while the interrupt service routines are executing.
- The START command loads the internal decremental counter. Before the counter generates an underflow (reaches 0), a STOP or RESTART command must be executed to force a Secure Counter check.

50.3.2 Modes of operation (STATE)

The CDOG has two legal states: IDLE and ACTIVE.

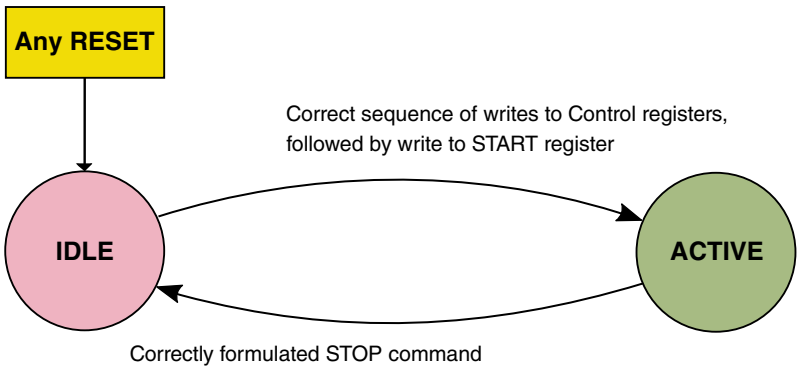


Figure 214. CDOG state diagram

- The two states are encoded in the read-only STATUS[CURST] field.
- After any reset (including a reset generated by the CDOG itself), the module will be in IDLE state.
- To change the module state to ACTIVE, software must execute a correct sequence of writes to the CONTROL group, followed by a START command. See Example use cases.
- Once ACTIVE, a correctly formulated STOP command will change the state back to IDLE.

50.3.3 Faults, flags, and counters

50.3.3.1 Fault types

All fault types (except CONTROL) can be individually controlled to generate either a system reset, an interrupt, or nothing.

Table 365. Fault types

This fault...	Occurs when...
TIMEOUT	The Instruction Timer reaches '0'.
MISCOMPARE	Either a STOP or a RESTART command is issued, and the value passed in the instruction does not match the content of the Secure Counter. ASSERT16 command is issued, and the lower 16-bit value passed in the instruction does not match the lower 16-bit value of the Secure Counter.
SEQUENCE	The prescribed sequence of interactions between software and CDOG is violated.
CONTROL	Any of the CONTROL register's fields contain an illegal value.
STATE	The internal state machine contains any value other than 5h or Ah.
ADDRESS	When an undefined address in the module's register space is accessed.

Table continues on the next page...

Table 365. Fault types (continued)

This fault...	Occurs when...
	NOTE Address 0xC is reserved, but no ADDRESS fault will be generated when 0xC is accessed.

For CONTROL fault, a system reset is requested when FLAGS[CNT_FLAG] = 1 and CONTROL[LOCK_CTRL] != 10b.

Lock the CONTROL register by writing CONTROL[LOCK_CTRL] == 01b at the same time that the desired, valid configuration is written. This blocks additional writes, and prevents CONTROL faults.

Interrupts are available as an alternative to system reset generation, for all fault types (except CONTROL), primarily to facilitate code development and debug operations. In the final application, interrupts can be enabled for some faults, and reset can be enabled for other faults (or disabled completely), at the risk of reduced security.

50.3.3.2 Fault counters

Each fault type has an associated counter that increments when the fault is detected. The counters are cleared by POR, but not by a system reset. Thus, statistics can be built up over many code watchdog resets to reveal the behavior patterns of a specific type of attack.

50.3.3.3 Flags

Each fault type has an associated flag accessible through the FLAGS register. A flag sets whenever its associated fault is detected, and can be cleared by software. The flags themselves (when enabled) generate the module's system reset or interrupt outputs.

The flags retain their value through a system reset (including one generated by the CDOG), but are reset by a POR. Using this mechanism, software can easily answer the 'How did I get here?' question by reading the FLAGS register after any reset.

To facilitate testing and code development, write to the flags directly when the module is not locked (CONTROL[LOCK_CTRL] = 10b). When the module is locked (CONTROL[LOCK_CTRL] = 01b), the flags can be cleared by writing '1' to their bit positions.

Table 366. FLAGS summary

Name	POR	ADDR	STATE	CONTROL	SEQUENCE	MISCOMPARE	TIMEOUT
Bit	16	5	4	3	2	1	0
Reset value after POR?	1	0	0	0	0	0	0
Writeable when unlocked?	Y	Y	Y	Y	Y	Y	Y
W1C when locked?	Y	Y	Y	Y	Y	Y	Y

50.3.3.4 Fault generation

Some faults are related to when and how registers can be accessed. See Figure 215.

The CDOG supports the following faults:

Table 367. Fault generation

This fault...	Is generated when...
SEQUENCE	• Software does not write to RELOAD before the START command is issued. • Software writes to RELOAD in ACTIVE state. Only write to RELOAD in IDLE state. • Software attempts to write to START in ACTIVE state. Only write to START in IDLE state. • In IDLE state, software attempts to write to a COMMAND group register other than START. • In ACTIVE state, software attempts to write to a CONTROL group register.
ADDRESS	• A read access (within the CDOG address space) to any address outside the CONTROL register group. • A write access (within the CDOG address space) to any address outside both the CONTROL and COMMAND register groups. **NOTE** Address 0xC is reserved in the address space of the CONTROL register group.
TIMEOUT	• The countdown is one clock before the Instruction Timer reaches '0'.
MISCOMPARE	• The STOP register is written and the current Secure Counter value does not match the write data. • The RESTART register is written and the current Secure Counter value does not match the write data. • The ASSERT16 register is written and the current Secure Counter lower 16-bit value does not match the 16-bit write data.
CONTROL	• The CONTROL[LOCK_CTRL] field is any value other than 01b or 10b. • The CONTROL bit fields: TIMEOUT_CTRL, MISCOMPARE_CTRL, SEQUENCE_CTRL, STATE_CTRL, or ADDRESS_CTRL contain any value other than 001b, 010b, or 100b. • The CONTROL[DEBUG_HALT_CTRL] field contains any value other than 01b or 10b. • The CONTROL[IRQ_PAUSE] field contains any value other than 01b or 10b.
STATE	• The STATE machine contains any value other than 5h or Ah. See STATUS[CURST].

50.3.4 Clocking

CDOG has only one clock input. See the chip-specific CDOG section for information on available clock inputs and the default option for a specific device.

50.3.5 Reset

CDOG has two hardware reset input sources: power-on reset (POR) and system reset.

CDOG can generate a system reset using the CONTROL register. See Faults, flags, and counters for details on how CDOG generates a system reset.

50.3.5.1 Power on reset (POR)

The logic and all registers of this module are reset to their default states on a POR.

50.3.5.2 System reset

The logic and most of the registers of this module are reset to their default states on a system reset.

The following registers cannot be reset by a system reset:

- Persistent Data Storage (PERSISTENT)
- Flags (FLAGS)
- Status 1 (STATUS), except the STATUS[CURST] field
- Status 2 (STATUS2)

50.3.6 Interrupts

CDOG has only one interrupt output. The interrupt signal output by CDOG can be connected to the system's interrupt controller. The following table lists CDOG's interrupt sources:

Table 368. Interrupt Summary

Interrupt flag	Interrupt enable	Description
FLAGS[TO_FLAG]	CONTROL[TIMEOUT_CTRL] = 3'b010	TIMEOUT fault interrupt
FLAGS[MISCOM_FLAG]	CONTROL[MISCOMPARE_CTRL] = 3'b010	MISCOMPARE fault interrupt
FLAGS[SEQ_FLAG]	CONTROL[SEQUENCE_CTRL] = 3'b010	SEQUENCE fault interrupt
FLAGS[STATE_FLAG]	CONTROL[STATE_CTRL] = 3'b010	STATE fault interrupt
FLAGS[ADDR_FLAG]	CONTROL[ADDRESS_CTRL] = 3'b010	ADDRESS fault interrupt

50.4 Application information

This section gives examples on how to program CDOG to monitor code execution flow.

50.4.1 Example use cases

There is generally a strict sequence to configure the CDOG, activated and serviced as follows:

1. Write an Instruction Timer reload value, corresponding to the current code section, to the Instruction Timer Reload Register (RELOAD) register.
2. Write a control word to the Control Register (CONTROL) register, where each fault type is configured to generate a reset, interrupt, or neither. Then lock the Control register using the CONTROL[LOCK_CTRL] field. Instruction Timer Register (INSTRUCTION_TIMER) may be always kept running, or run only during non-IRQ execution (paused during interrupt processing) based on the value in the CONTROL[IRQ_PAUSE] field.
3. Activate the module by writing to the START Command Register (START) register. The initial value for the Secure Counter is the value written. Instruction Timer Register (INSTRUCTION_TIMER) immediately starts decrementing from the RELOAD value on every clock cycle.
4. At strategically chosen way points in the code flow of the application, update the Secure Counter by issuing ADD or SUB commands, or assert the lower 16-bit value of the Secure Counter by the ASSERT16 command.

5. When the way point that corresponds to the number of executed instructions represented by the RELOAD value (the end of the current code section) is reached, write the expected value of the Secure Counter to the STOP command register. Assuming the written value compares exactly to the contents of the Secure Counter, the instruction timer stops. Then the process can begin again for the next code section, by repeating steps 1 through 5.

Another example of a configuration and start procedure is as follows:

1. Write the Instruction Timer Register (INSTRUCTION_TIMER) reload value to the Instruction Timer Reload Register (RELOAD) register. Write a very large number to provide a buffer for large values.
2. Write a control word to the Control Register (CONTROL) register, where each fault type is configured to generate a reset, interrupt, or neither. Then lock the Control register using the CONTROL[LOCK_CTRL] field.
3. Activate the module by writing to the START Command Register (START) register. The initial value for the Secure Counter is the value written. INSTRUCTION_TIMER immediately starts decrementing from the RELOAD value on every clock.
4. At strategically chosen way points in the code flow of the application, update the Secure Counter by issuing ADD or SUB commands, or assert the lower 16-bit value of the Secure Counter by the ASSERT16 command.
5. During the execution flow, the application must always be aware of the INSTRUCTION_TIMER register value approach toward 0.
    a. Read the INSTRUCTION_TIMER value regularly to determine the expended time.
    b. Before the INSTRUCTION_TIMER value reaches 0, software must service the CDOG by writing the Secure Counter expected value to the RESTART command register.
    c. Assuming the value written compares exactly to the contents of the Secure Counter, INSTRUCTION_TIMER reloads with the value in the RELOAD register as it decrements toward 0.

50.4.2 Using CDOG during code development debug

Following are suggestions for facilitating the code development and debug cycle:

- Maintain control while adjusting the timing and fine-tuning of the counting values by using interrupt-on-fault (instead of reset).
- Trigger faults by direct-writing the bits in the FLAGS register with the same result as hardware-triggered faults.
- Pause the instruction timer during a pause of a debug session by using the CONTROL[DEBUG_HALT_CTRL] field.

50.5 Memory map and register definition

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

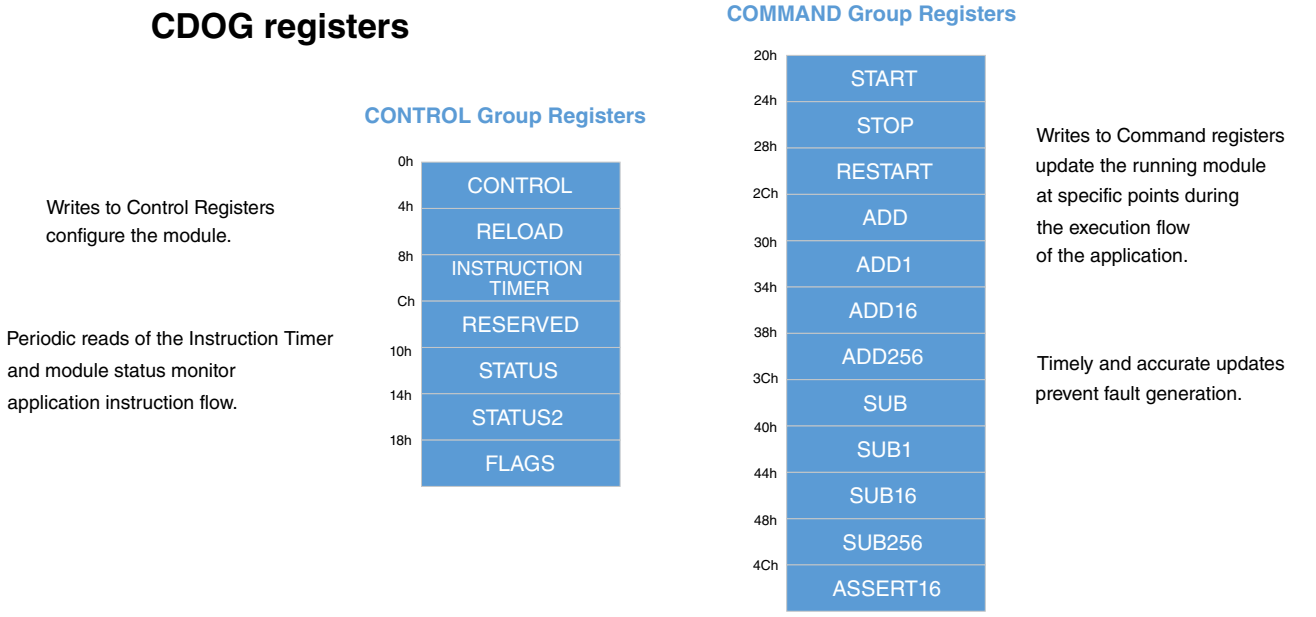


Figure 215. CDOG register groups

50.5.1 CDOG register descriptions

50.5.1.1 CDOG memory map

CDOG0 base address: 4010_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register (CONTROL)	32	RW	5009_2492h
4h	Instruction Timer Reload Register (RELOAD)	32	RW	FFFF_FFFFh
8h	Instruction Timer Register (INSTRUCTION_TIMER)	32	R	FFFF_FFFFh
10h	Status 1 Register (STATUS)	32	R	5000_0000h
14h	Status 2 Register (STATUS2)	32	R	0000_0000h
18h	Flags Register (FLAGS)	32	RW	0001_0000h
1Ch	Persistent Data Storage Register (PERSISTENT)	32	RW	0000_0000h
20h	START Command Register (START)	32	W	0000_0000h
24h	STOP Command Register (STOP)	32	W	0000_0000h
28h	RESTART Command Register (RESTART)	32	W	0000_0000h
2Ch	ADD Command Register (ADD)	32	W	0000_0000h
30h	ADD1 Command Register (ADD1)	32	W	0000_0000h
34h	ADD16 Command Register (ADD16)	32	W	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
38h	ADD256 Command Register (ADD256)	32	W	0000_0000h
3Ch	SUB Command Register (SUB)	32	W	0000_0000h
40h	SUB1 Command Register (SUB1)	32	W	0000_0000h
44h	SUB16 Command Register (SUB16)	32	W	0000_0000h
48h	SUB256 Command Register (SUB256)	32	W	0000_0000h
4Ch	ASSERT16 Command Register (ASSERT16)	32	W	0000_0000h

50.5.1.2 Control Register (CONTROL)

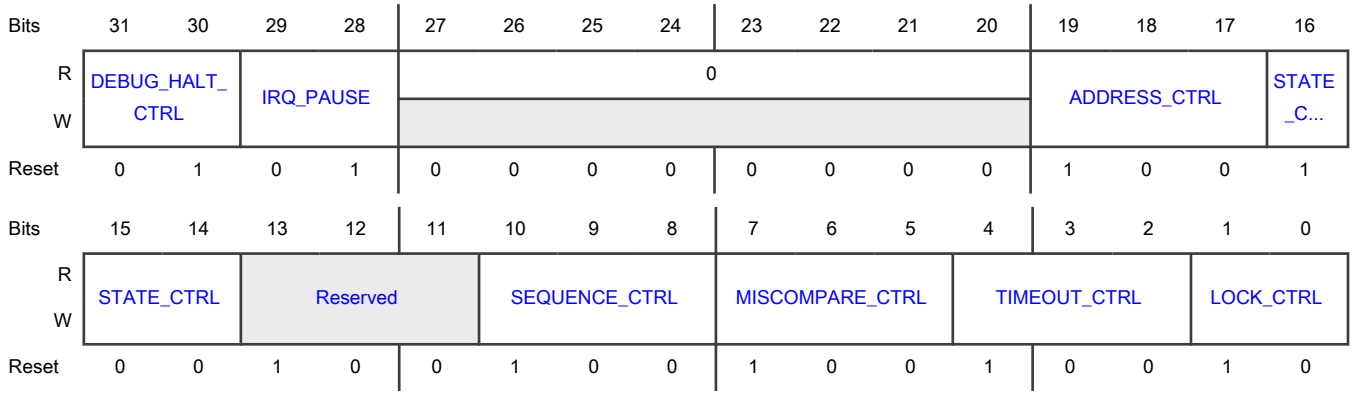
Offset

Register	Offset
CONTROL	0h

Function

The Control register (CONTROL) contains all the controllable attributes of the module, such as how the CDOG module responds when detecting a fault.

Diagram



Fields

Field	Function
31-30 DEBUG_HALT_CTRL	DEBUG_HALT control Controls whether the Instruction Timer runs or stops when the CPU asserts DEBUG_HALT. See chip-specific CDOG section for information on DEBUG_HALT connections for a specific device.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Keep the timer running 10b - Stop the timer
29-28 IRQ_PAUSE	IRQ pause control Controls whether the Instruction Timer runs or stops when the CPU asserts IRQ_PAUSE. See chip-specific CDOG section for information on IRQ_PAUSE connections for a specific device. 01b - Keep the timer running 10b - Stop the timer
27-20 —	Reserved
19-17 ADDRESS_CTRL	ADDRESS fault control Controls how the CDOG module responds when detecting an ADDRESS fault (FLAGS[ADDR_FLAG]). 001b - Enable reset 010b - Enable interrupt 100b - Disable both reset and interrupt
16-14 STATE_CTRL	STATE fault control Controls how the CDOG module responds when detecting a STATE fault (FLAGS[STATE_FLAG]). 001b - Enable reset 010b - Enable interrupt 100b - Disable both reset and interrupt
13-11 —	Reserved Do not change its value.
10-8 SEQUENCE_CTRL	SEQUENCE fault control Controls how the CDOG module responds when detecting a SEQUENCE fault (FLAGS[SEQUENCE_FLAG]). 001b - Enable reset 010b - Enable interrupt 100b - Disable both reset and interrupt
7-5 MISCOMPARE_CTRL	MISCOMPARE fault control Controls how the CDOG module responds when detecting a MISCOMPARE fault (FLAGS[MISCOMPARE_FLAG]). 001b - Enable reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - Enable interrupt 100b - Disable both reset and interrupt
4-2 TIMEOUT_CTRL	TIMEOUT fault control Controls how the CDOG module responds when detecting a TIMEOUT fault (FLAGS[TIMEOUT_FLAG]). 001b - Enable reset 010b - Enable interrupt 100b - Disable both reset and interrupt
1-0 LOCK_CTRL	Lock control Resets to unlocked. Once locked, LOCK_CTRL remains locked until the next system reset. The Control register is writable and readable only when unlocked, LOCK_CTRL = 10. Once locked, the written value to the Control register is ignored, and reading the Control register returns 0. 01b - Locked 10b - Unlocked

50.5.1.3 Instruction Timer Reload Register (RELOAD)

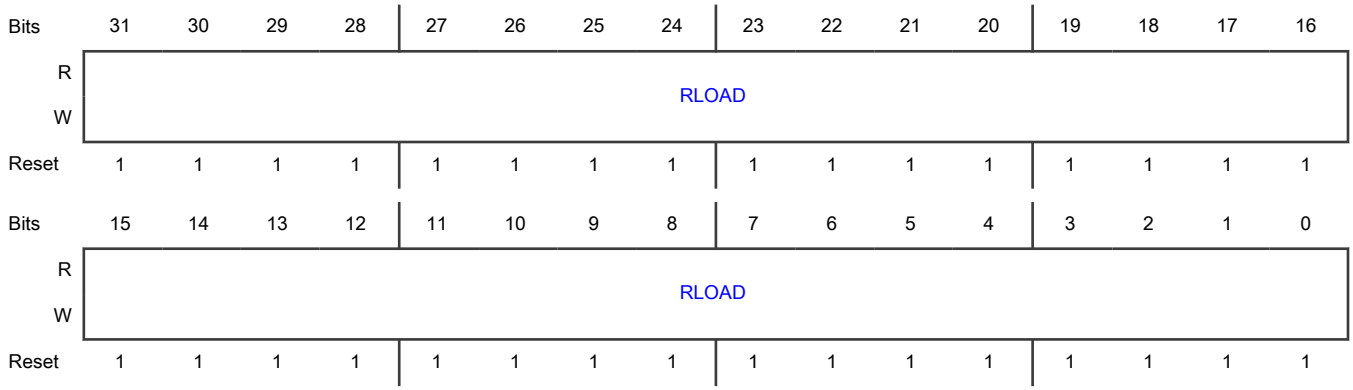
Offset

Register	Offset
RELOAD	4h

Function

The Instruction Timer RELOAD register contains the value from which the Instruction Timer counts down when a valid START or RESTART command is executed. Write to RELOAD only when the module is in the IDLE state. A write to RELOAD must occur before issuing a START command.

Diagram



Fields

Field	Function
31-0 RLOAD	Instruction Timer reload value Contains the starting countdown value used by the Instruction Timer whenever a START or RESTART command is executed.

50.5.1.4 Instruction Timer Register (INSTRUCTION_TIMER)

Offset

Register	Offset
INSTRUCTION_TIMER	8h

Function

The Instruction Timer register (INSTRUCTION_TIMER) contains the current count value of the Instruction Timer.

Diagram

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

Fields

Field	Function
31-0 INSTIM	Current value of the Instruction Timer The current value of the timer can be read from this address. The Software cannot write to INSTRUCTION_TIMER.

50.5.1.5 Status 1 Register (STATUS)

Offset

Register	Offset
STATUS	10h

Function

The Status 1 register (STATUS) holds the current module status and fault counts. The fields (except CURST) in STATUS are reset only by POR.

Diagram

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

Fields

Field	Function
31-28 CURST	Current State Indicates the current state of the module. The only legal states are: • 0x5 IDLE • 0xA ACTIVE Any other value generates a STATE fault (FLAGS[STATE_FLAG]). Resets to 0x5 = IDLE on any reset (unlike other bits in this register that only reset on POR).
27-24 —	Reserved
23-16 NUMILSEQF	Number of SEQUENCE faults (FLAGS[SEQUENCE_FLAG]) since the last POR Resets to 0. Will not increment past 0xFF.
15-8 NUMMISCOMPF	Number of MISCOMPARE faults (FLAGS[MISCOMPARE_FLAG]) since the last POR Resets to 0. Will not increment past 0xFF.
7-0 NUMTOF	Number of TIMEOUT faults (FLAGS[TIMEOUT_FLAG]) since the last POR Resets to 0. Will not increment past 0xFF.

50.5.1.6 Status 2 Register (STATUS2)

Offset

Register	Offset
STATUS2	14h

Function

Status 2 register (STATUS2) holds additional fault counts. The fields in STATUS2 are reset only by POR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								NUMILLA							
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	NUMILLSTF								NUMCNTF							
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 NUMILLA	Number of ADDRESS faults (FLAGS[ADDR_FLAG]) since the last POR Resets to 0. Will not increment past 0xFF.
15-8 NUMILLSTF	Number of STATE faults (FLAGS[STATE_FLAG]) since the last POR Resets to 0. Will not increment past 0xFF.
7-0 NUMCNTF	Number of CONTROL faults (FLAGS[CONTROL_FLAG]) since the last POR Resets to 0. Will not increment past 0xFF.

50.5.1.7 Flags Register (FLAGS)

Offset

Register	Offset
FLAGS	18h

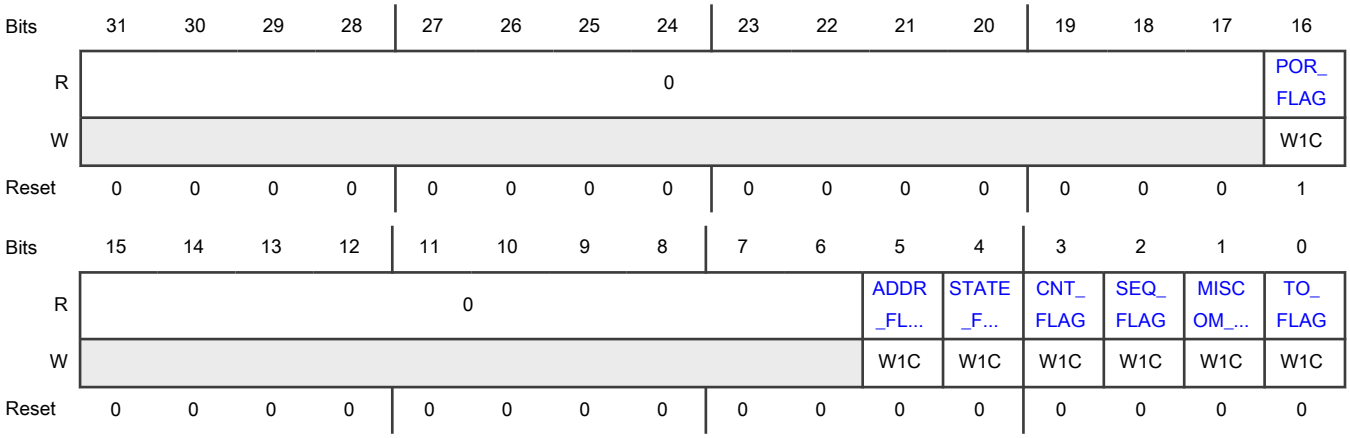
Function

The Flags register (FLAGS) contains the fault detection flags, as well as a POR event flag. The fault detection flags may generate a reset or an interrupt as configured in the CONTROL register.

The FLAGS fields retain their value through a system reset, including one generated by the CDOG. However, FLAGS is reset by a POR.

Writability of the fields depends on the module state. See Flags.

Diagram



Fields

Field	Function
31-17 —	Reserved
16 POR_FLAG	Power-on reset flag POR_FLAG is set by a Power-on reset event. 0b - A Power-on reset event has not occurred 1b - A Power-on reset event has occurred
15-6 —	Reserved
5 ADDR_FLAG	ADDRESS fault flag Hardware sets ADDR_FLAG when CDOG detects an ADDRESS fault. 0b - An ADDRESS fault has not occurred 1b - An ADDRESS fault has occurred
4 STATE_FLAG	STATE fault flag Hardware sets STATE_FLAG when CDOG detects a STATE fault. 0b - A STATE fault has not occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - A STATE fault has occurred
3 CNT_FLAG	CONTROL fault flag Hardware sets CNT_FLAG when CDOG detects a CONTROL fault. 0b - A CONTROL fault has not occurred 1b - A CONTROL fault has occurred
2 SEQ_FLAG	SEQUENCE fault flag Hardware sets the SEQ_FLAG when CDOG detects a SEQUENCE fault. 0b - A SEQUENCE fault has not occurred 1b - A SEQUENCE fault has occurred
1 MISCOM_FLAG	MISCOMPARE fault flag Hardware sets MISCOM_FLAG when CDOG detects a MISCOMPARE fault. 0b - A MISCOMPARE fault has not occurred 1b - A MISCOMPARE fault has occurred
0 TO_FLAG	TIMEOUT fault flag Hardware sets TO_FLAG when CDOG detects a TIMEOUT fault. 0b - A TIMEOUT fault has not occurred 1b - A TIMEOUT fault has occurred

50.5.1.8 Persistent Data Storage Register (PERSISTENT)

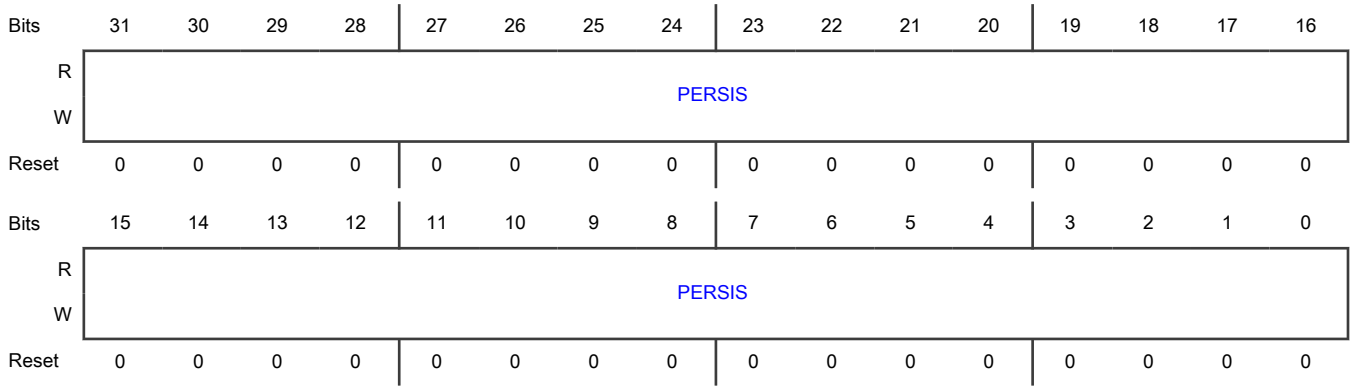
Offset

Register	Offset
PERSISTENT	1Ch

Function

The Persistent Data Storage register (PERSISTENT) provides an application with 32 bits of scratchpad storage for information that needs to persist through resets other than a Power-On Reset (POR).

Diagram



Fields

Field	Function
31-0 PERSIS	Persistent Storage A write value to PERSIS remains unchanged after any reset other than a POR.

50.5.1.9 START Command Register (START)

Offset

Register	Offset
START	20h

Function

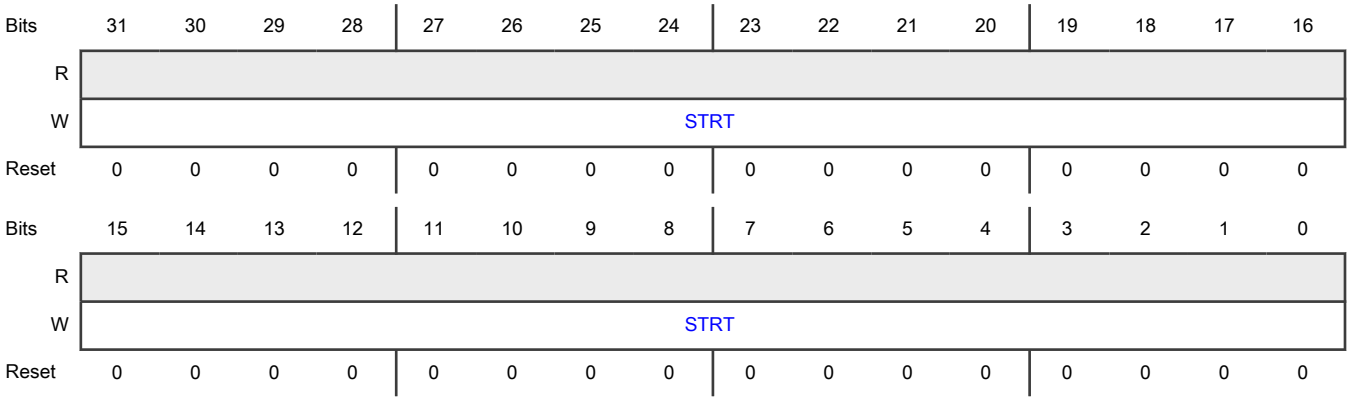
Writing to the Start Command register (START) issues a Start command:

- The value written to STRT is loaded into the Secure Counter as a start value.
- The RELOAD value is loaded into the Instruction Timer.
- The module enters the ACTIVE state in which the Instruction Timer counts down on every clock.

NOTE

Write to START only during the IDLE state, and only after writing to the RELOAD register. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 STRT	Start command The value written to STRT is loaded into the Secure Counter as a start value.

50.5.1.10 STOP Command Register (STOP)

Offset

Register	Offset
STOP	24h

Function

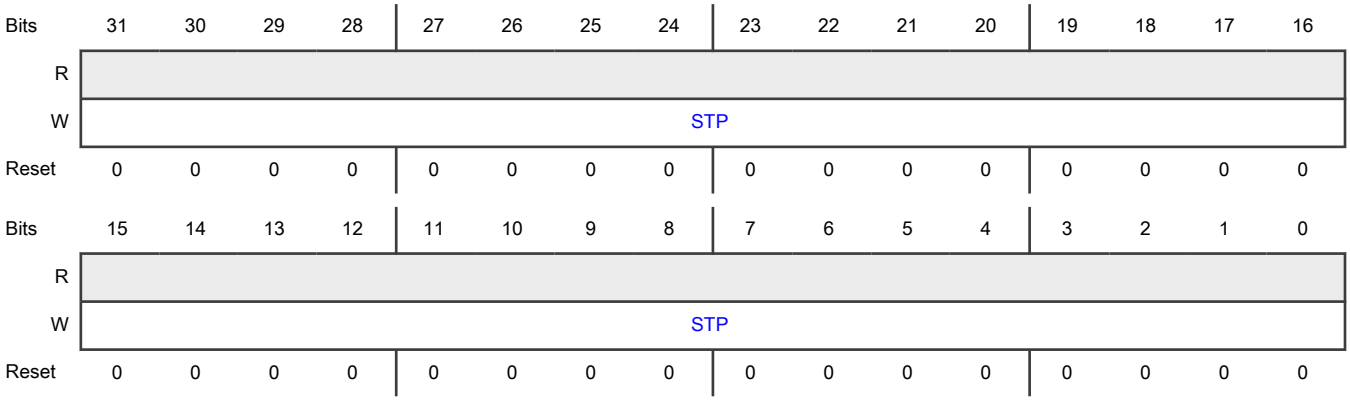
Writing to the Stop Command register (STOP) issues a Stop command:

- The Instruction Timer is stopped.
- The value written to STP is compared with the current value of the Secure Counter.

NOTE

Write to STOP only during the ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 STP	Stop command The value written to STP is compared with the current value of the Secure Counter.

50.5.1.11 RESTART Command Register (RESTART)

Offset

Register	Offset
RESTART	28h

Function

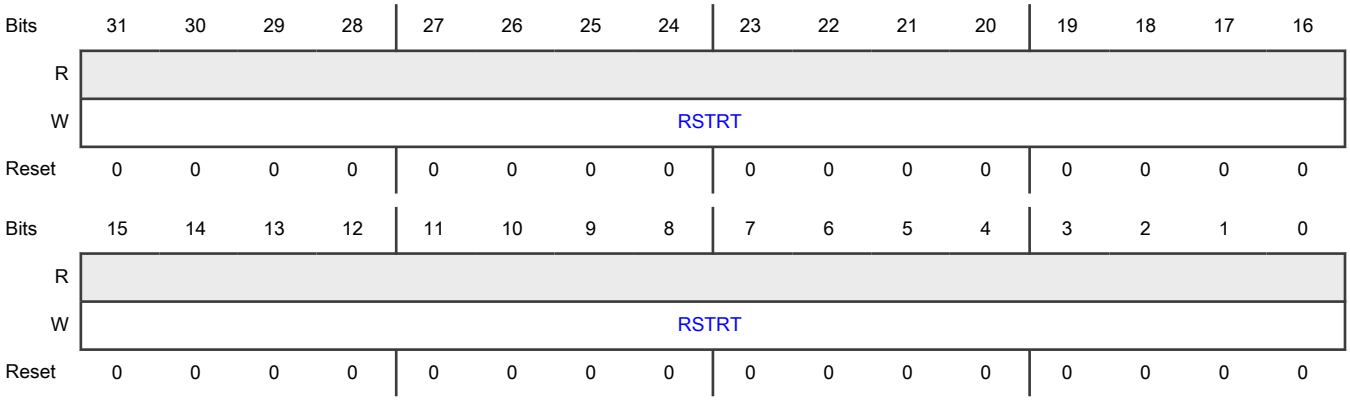
Writing to the Restart Command register (RESTART) issues a Restart command:

- The value written to RSTRT is compared with the current value of the Secure Counter.
- If the values match, the Instruction Timer is reloaded (with the RELOAD value) and starts counting down again.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 RSTRT	Restart command The value written to RSTRT is compared with the current value of the Secure Counter.

50.5.1.12 ADD Command Register (ADD)

Offset

Register	Offset
ADD	2Ch

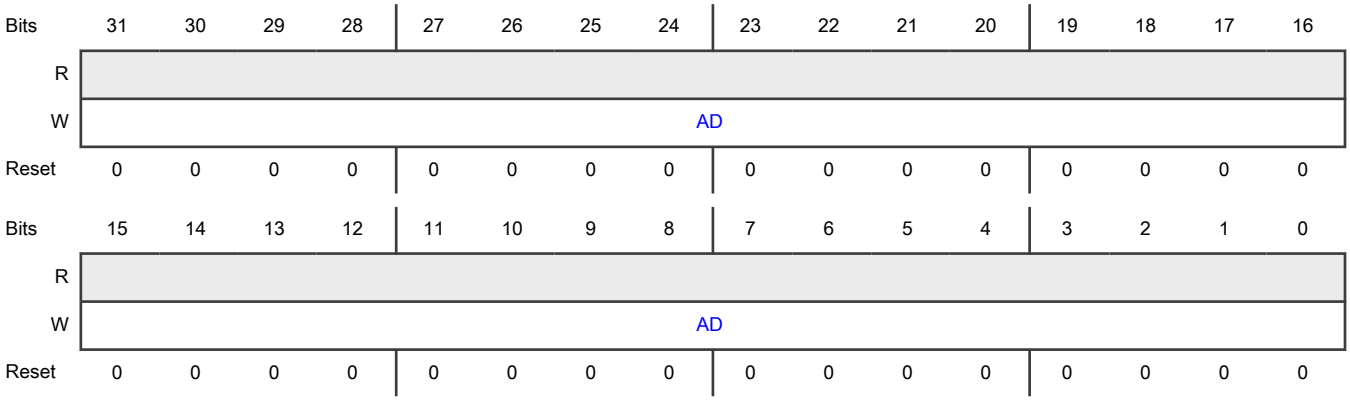
Function

Writing to the Add Command register (ADD) issues an Add command: The value written to AD is added to the current value of the Secure Counter

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 AD	ADD Write Value The value written to AD is added to the current value of the Secure Counter.

50.5.1.13 ADD1 Command Register (ADD1)

Offset

Register	Offset
ADD1	30h

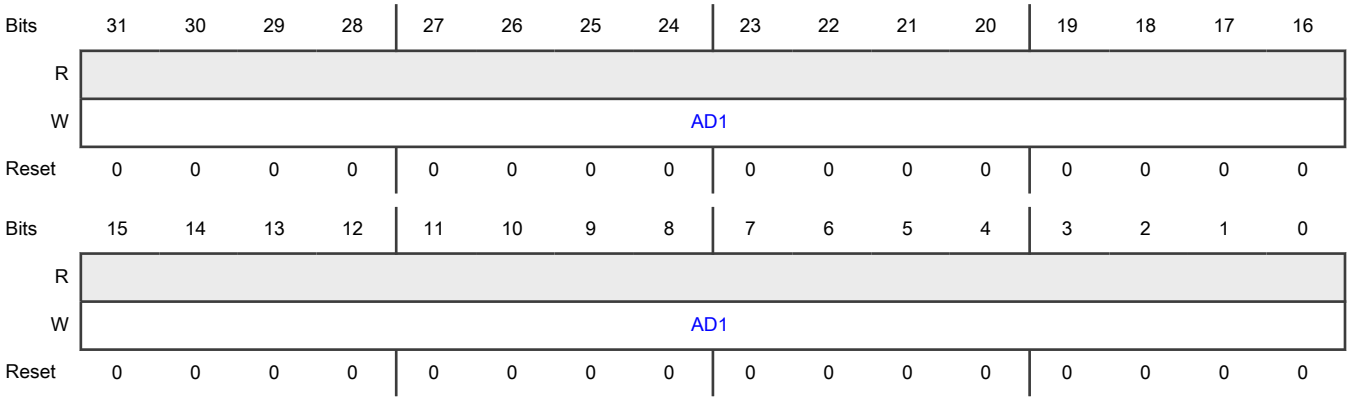
Function

Writing to the ADD1 Command register (ADD1) issues an ADD1 command: The value 1 is added to the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 AD1	ADD 1 Secure Counter is always incremented by 1. The value written to AD1 is ignored.

50.5.1.14 ADD16 Command Register (ADD16)

Offset

Register	Offset
ADD16	34h

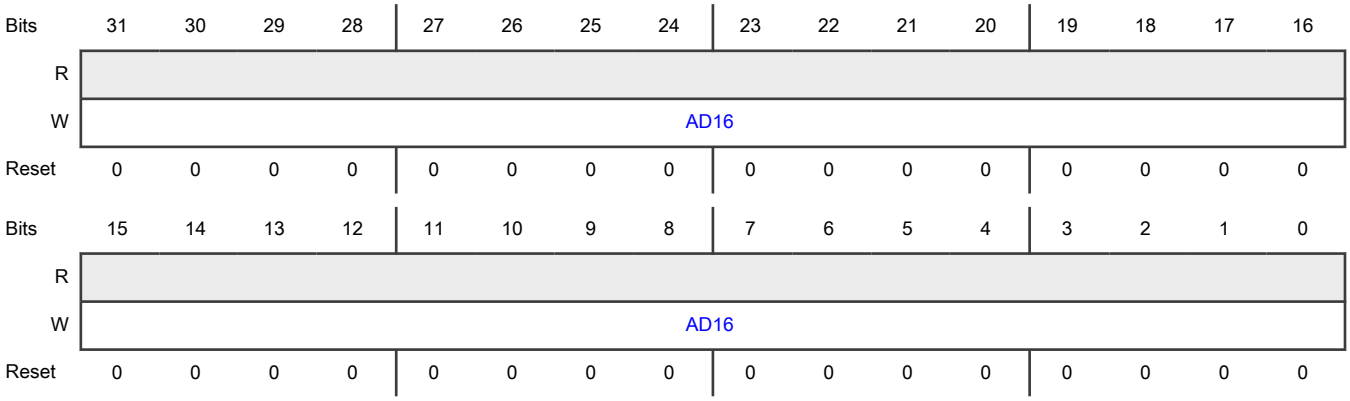
Function

Writing to the ADD16 Command register (ADD16) issues an ADD16 command: The value 16 is added to the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 AD16	ADD 16 Secure Counter is always incremented by 16. The value written to AD16 is ignored.

50.5.1.15 ADD256 Command Register (ADD256)

Offset

Register	Offset
ADD256	38h

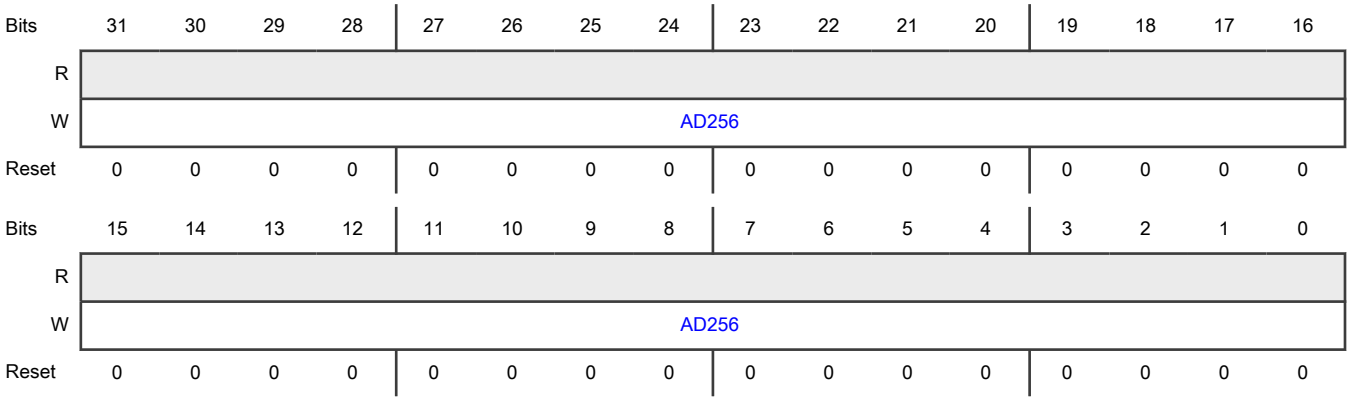
Function

Writing to the ADD256 Command register (ADD256) issues an ADD256 command: The value 256 is added to the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 AD256	ADD 256 Secure Counter is always incremented by 256. The value written to AD256 is ignored.

50.5.1.16 SUB Command Register (SUB)

Offset

Register	Offset
SUB	3Ch

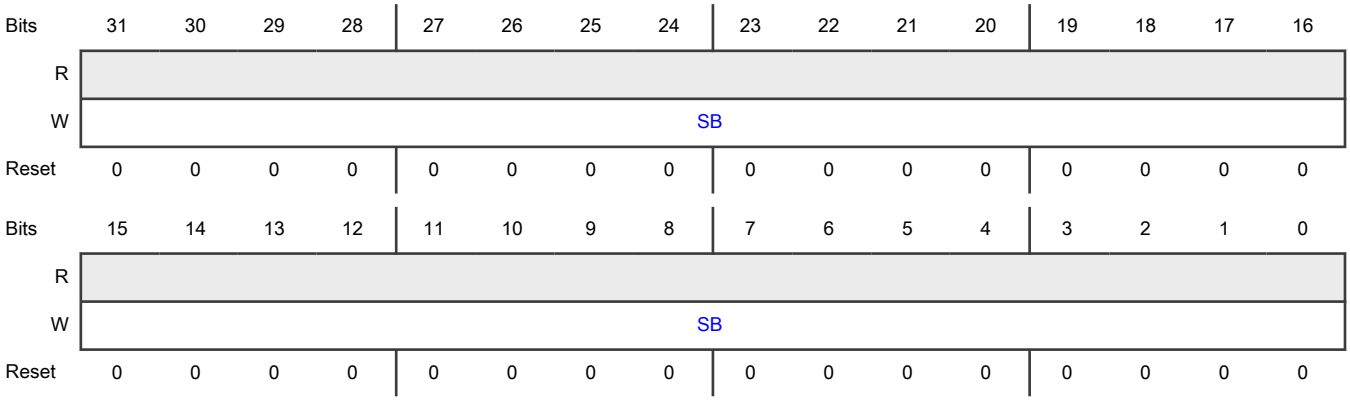
Function

Writing to the SUB Command register (SUB) issues a SUB command: The value written to SB is subtracted from the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 SB	Subtract Write Value The value written to SB is subtracted from the current value of the Secure Counter.

50.5.1.17 SUB1 Command Register (SUB1)

Offset

Register	Offset
SUB1	40h

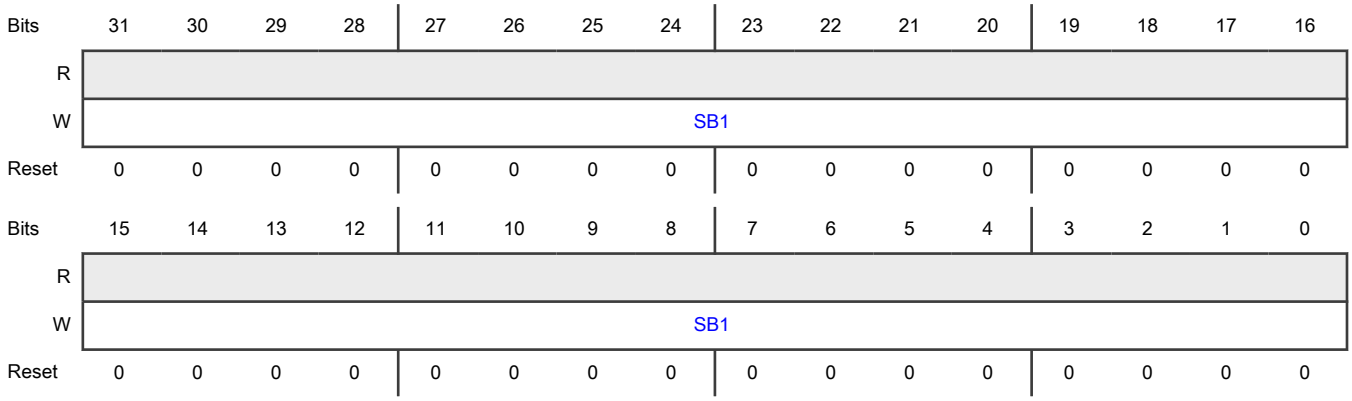
Function

Writing to the SUB1 Command register (SUB1) issues an SUB1 command: The value 1 is subtracted from the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 SB1	Subtract 1 Secure Counter is always decremented by 1. The value written to SB1 is ignored.

50.5.1.18 SUB16 Command Register (SUB16)

Offset

Register	Offset
SUB16	44h

Function

Writing to the SUB16 Command register (SUB16) issues an SUB16 command: The value 16 is subtracted from the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	SB16															
Reset	0	0	0	0	0	0	0	0	0	0	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	SB16															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SB16	Subtract 16 Secure Counter is always decremented by 16. The value written to SB16 is ignored.

50.5.1.19 SUB256 Command Register (SUB256)

Offset

Register	Offset
SUB256	48h

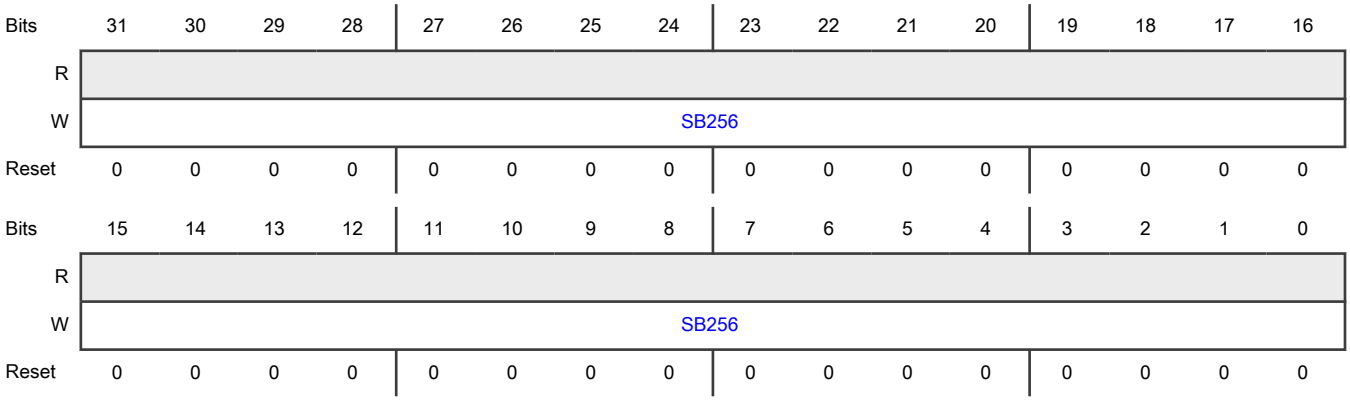
Function

Writing to the SUB256 Command register (SUB256) issues an SUB256 command: The value 256 is subtracted from the current value of the Secure Counter.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 SB256	Subtract 256 Secure Counter is always decremented by 256. The value written to SB256 is ignored.

50.5.1.20 ASSERT16 Command Register (ASSERT16)

Offset

Register	Offset
ASSERT16	4Ch

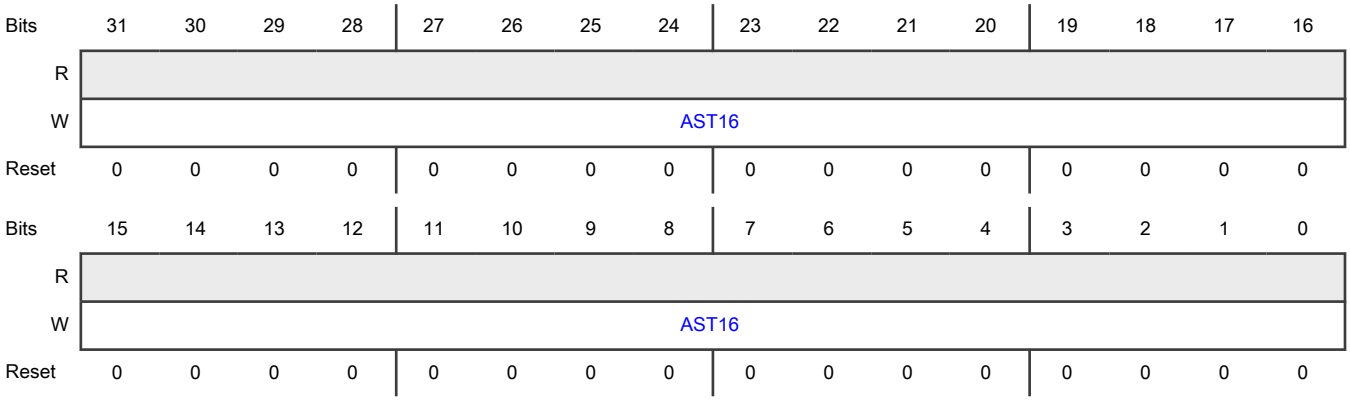
Function

Writing to the ASSERT16 Command register (ASSERT16) issues an ASSERT16 command: The lower 16-bit value written to AST16 is compared with the current lower 16-bit value of the Secure Counter. This command does not affect the Instruction Timer.

NOTE

Can only be written during ACTIVE state. A SEQUENCE fault (FLAGS[SEQUENCE_FLAG]) generates if this specified sequence is not followed.

Diagram



Fields

Field	Function
31-0 AST16	ASSERT16 Command The lower 16-bit value written to AST16 is compared with the current lower 16-bit value of the Secure Counter.

Chapter 51
GLIKEY

51.1 Chip-specific GLIKEY information

Table 369. Reference links to related information

Topic	Related module	Reference
Full description	GLIKEY	GLIKEY
System memory map		Memory map
Clocking		Clock distribution
Power Management		Power Management
Signal multiplexing	Port control	Signal multiplexing
Peripheral Input Mux	INPUTMUX	Peripheral Input multiplexing
System Controller	Syscon	See GLIKEY register section in SYSCON chapter

The registers of GLIKEY module are integrated within the SYSCON module.

51.1.1 Module instances

This device has one instance of GLIKEY module.

51.1.2 GLIKEY Control list

Below registers are controlled by GLIKEY:

GLIKEY Write EN	GLIKEY Reset	Module	Register
wr_en_out[1]	rst[1]	SYSCON	DEBUG_FEATURES_DP
wr_en_out[2]	N/A	SYSCON	SRAM_XEN_DP
wr_en_out[15]	N/A	MBC	All MBC register

51.2 Terms and definitions

Acronym	Explanation
FSM	Finite State Machine
SSR	Security Sensitive Register
SW	Software

51.3 Overview

Glikey IP provides a mechanism to safely access security-sensitive registers. The write-enable and reset's of these registers are controlled via GLIKEY.

51.3.1 Block diagram

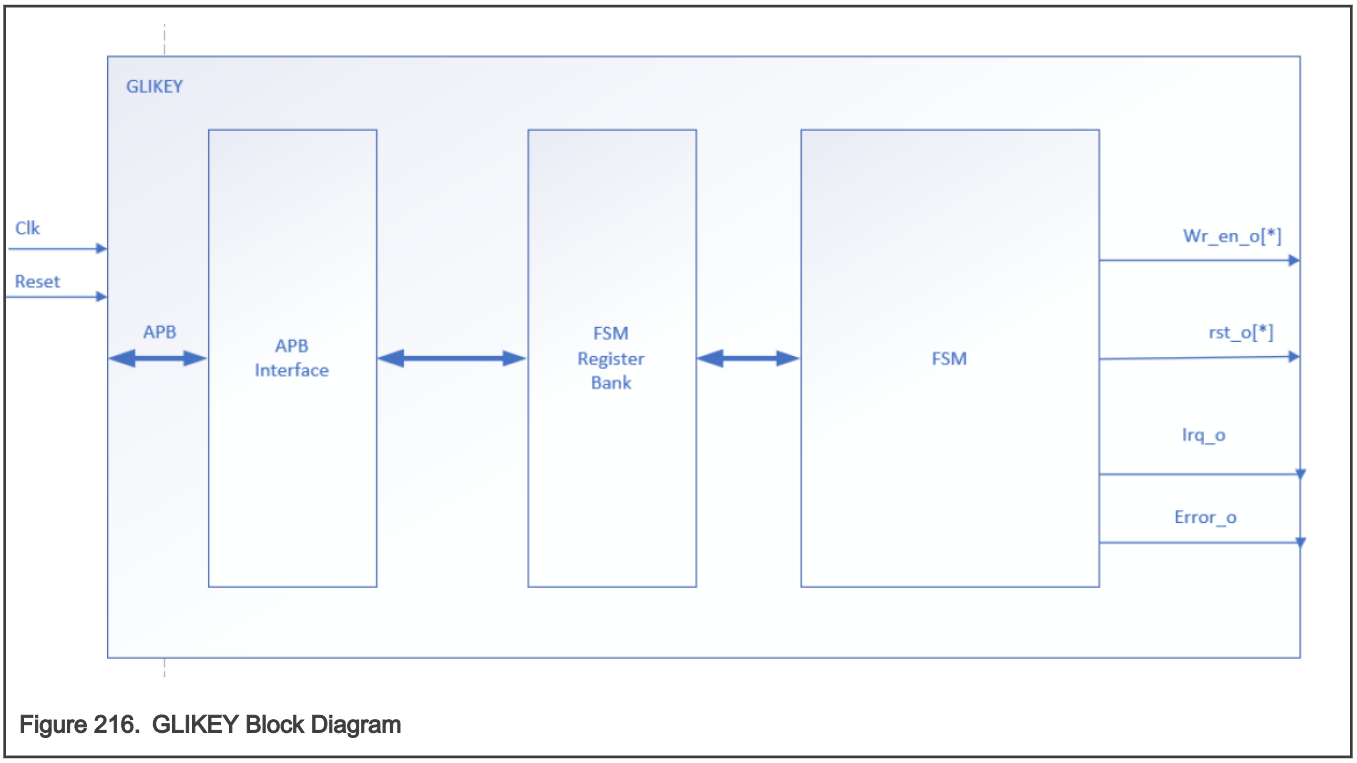


Figure 216. GLIKEY Block Diagram

51.3.2 Features

- Integrity protection of security-sensitive registers during write and at rest against power glitch attacks
- Integrity protection of security-sensitive registers at rest against privilege escalation
- Error Management
  - Dedicated Interrupt
  - Dedicated integrity protection

51.4 Configuration

GLIKEY can be configured in following ways:

- FSM to include 4 steps as mentioned in Figure 217

51.5 Architecture description

51.5.1 Functional description

GLIKEY controls the write-enables and resets for the SSR's through FSM. Software has to follow strict procedure to enable the write-enables and resets for SSR.

GLIKEY safe guards against attacks by constantly monitoring the state which the FSM is in and the expected SW procedure for the current state. If fault is found, then an interrupt is raised and the FSM goes to secure state. Procedure cannot continue until SW resets the GLIKEY.

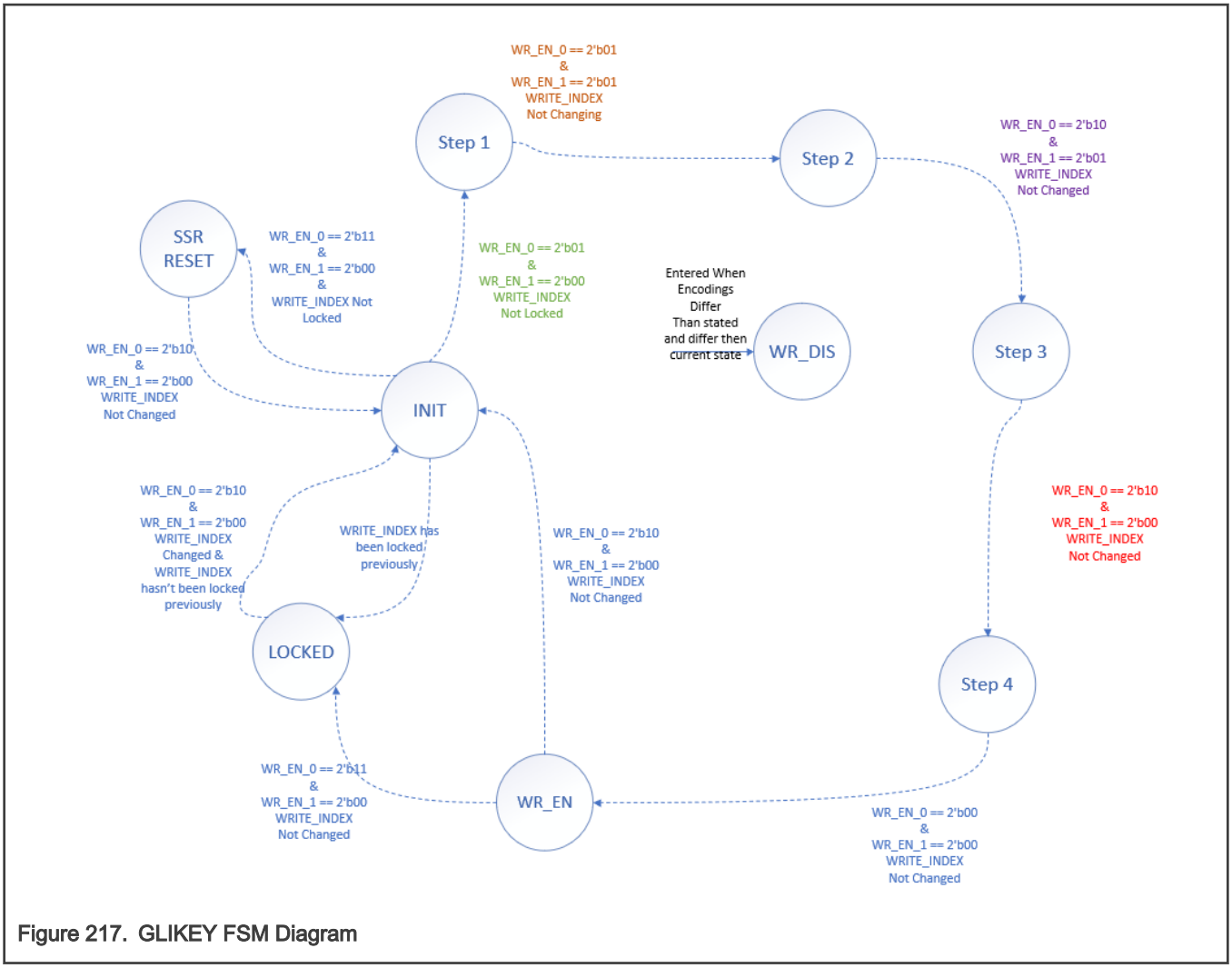


Figure 217. GLIKEY FSM Diagram

51.5.1.1 Progressing Through the FSM

Progressing through the FSM is done through writing to SFR registers: CTRL_0[WR_EN_0] and CTRL_1[WR_EN_1].

Any misconfiguration will lead to the FSM entering the WR_DISABLED state that would require a reset, either a soft reset or a hard reset.

The current state of FSM is read out by reading STATUS[FSM_STATE].

The sequence of progressing through the FSM can be seen in STATUS[FSM_STATE]. If the integrity of FSM encodings is compromised, then the error_o will be triggered.

51.5.1.2 Initial State

Upon reset (either triggered through hardware or software) FSM will enter the INIT state.

The reset value for the CTRL_0[WRITE_INDEX] is 0x0. If software reset is triggered and CTRL_0[WRITE_INDEX] was previously locked, then the FSM enters into the LOCKED state. To progress, the software needs to update the CTRL_0[WRITE_INDEX] to an index that is currently not locked.

In the INIT state if CTRL_0[WRITE_INDEX] is written that is currently LOCKED, then the FSM will enter the LOCKED state irrespective of CTRL_0[WR_EN_0] and CTRL_1[WR_EN_1].

For example, when in the INIT state, one write to the CTRL_0 happens that updates the CTRL_0[WR_EN_0] = 2'b01 and the CTRL_0[WRITE_INDEX] to a locked index, the FSM will enter the LOCKED state. In this case, software must update the CTRL_0[WR_EN_0] = 2'b10 and the CTRL_0[WRITE_INDEX] to an unlocked value to move back into the INIT state.

For normal operation while in the INIT state and CTRL_0[WRITE_INDEX] is written that is not locked and the CTRL_0[WR_EN_0] is set to 2'b01 or 2'b11 the FSM will either enter STEP1 or SSR_RESET and continue normal operation.

51.5.1.3 WR_DIS State

WR_DISABLED state is considered a secured state.

Any misconfiguration of the FSM will lead to this state. Incorrect addressing of the index through CTRL_0[WRITE_INDEX] will also lead to this state.

The FSM monitors the current state of which it is in, if the control signals change and do not match the expected next state encodings the error state is entered.

To leave the WR_DISABLED state, software must perform a soft (Writing one to CTRL_0[SFT_RST]) or hard reset of GLIKEY.

The NON-Secure state is considered a state in which the SSR can be accessed (this is the WR_EN state). This access is allowed via the write-enable outputs of GLIKEY. GLIKEY can enter only the NON-Secure state when the full software procedure has occurred.

Trigger a write-enable out

The write-enable output for any given index is only asserted when the FSM is in WR_ENABLED state.

When the FSM moves out of this state then it is de-asserted. The output wr_en_o[*] outputs are not registered and are driven directly by FSM.

Upon entering the WR_DIS state, all bits of the output rst_o will assert to high causing reset of all SSRs

51.5.1.4 Trigger a Reset output

The reset output for any given index is asserted while the FSM is in SSR_RESET state.

When the FSM moves out of this state, then it is de-asserted.

The output rst_o[*] outputs are not registered and are driven directly by the FSM.

51.5.1.5 Lock SSR access

After accessing the WR_ENABLE state for any given index, further access to this index can be locked. This causes the GLIKEY to lock access for this. If addressed again, FSM jumps to the LOCKED state and cannot trigger a write-enable or reset for this index.

The locked indexes can only be reset by a hardware reset. The lock registers are protected against integrity errors with replication. In case the lock and the inverse lock registers are not matching the error_o port is triggered.

Example:

Assuming software is in the WR_EN state with WRITE_INDEX =1, software writes 2'b11 to CTRL_0[WR_EN_0] and enters the LOCKED State. This Index is now considered locked. To enter back into INIT state, software writes 2'b10 to CTRL_0[WR_EN_0] and the WRITE_INDEX is updated to a WRITE_INDEX not previously locked. If software updates WRITE_INDEX that is previously locked, then the software remains in LOCKED STATE.

This means when all the indexes are locked, software remains in the LOCKED state upon entering the last index that is locked.

To read the status of WRITE_INDEX that are locked, software needs to write the corresponding WRITE_INDEX number to the CTRL_1[INDEX_STATUS] to know the lock status. Software then reads STATUS[LOCK_STATUS] to know the status. 1 = The index is locked, 0 = The index is not locked.

51.5.1.6 Reset FSM

Software has the ability to reset the FSM. Writing one to CTRL_0[SFT_RST] brings the FSM into INIT state. This can be triggered while in any state.

Resetting the FSM will cause the current procedure to be restarted. Resetting the FSM will not reset the indexes that have been locked previously. When all WRITE_INDEX's are locked then the FSM will stay in the LOCKED State and cannot be moved from.

Triggering the CTRL_0[SFT_RST] will also reset the following registers to their reset values:

- CTRL_0[WR_EN_0]
- CTRL_1[WR_EN_1]
- CTRL_0[WRITE_INDEX]

CTRL_0[SFT_RST] takes priority if CTRL_0[WRITE_INDEX] and CTRL_0[WR_EN_0] are written in the same bus access, resulting in these being reset.

51.5.1.7 Disable GLIKEY

GLIKEY can be disabled by writing the value 4'h5 to CTRL_1[SFR_LOCK]. Only hardware reset can re-enable GLIKEY.

Once the GLIKEY is disabled, software has no write permissions to the SFR's only read permissions.

GLIKEY cannot perform any accesses to the security sensitive registers.

51.5.1.8 Interrupt Description

There are 3 interrupt sources, as listed below.

Table 370. Interrupt Sources

Interrupt Source	Description	Register
FSM Interrupt	Upon entering the WR_DIS state this interrupt is raised	STATUS[ERROR_STATUS][0]
WRITE_INDEX out of bounds	If a WRITE_INDEX is addressed that is greater than the number of addressable indexes configured	STATUS[ERROR_STATUS][1]
READ_INDEX out of bounds	If a READ_INDEX is addressed that is greater than the number of addressable indexes configured	STATUS[ERROR_STATUS][2]

Interrupts can be enabled or disabled through INTR_CTRL[INT_EN].

Interrupts can be cleared by writing 1 to INTR_CTRL[INT_CLR].

Interrupts can be set by writing 1 to INTR_CTRL[INT_SET].

Software can monitor the status of the interrupt sources through STATUS[ERROR_STATUS]. Clearing the interrupt will not clear the STATUS[ERROR_STATUS] registers.

Table 371. Interrupt Clearing Methods

Error Status Register	Description

Table continues on the next page...

Table 371. Interrupt Clearing Methods (continued)

STATUS[ERROR_STATUS][0]	Soft or hard reset
STATUS[ERROR_STATUS][1]	Updating WRITE_INDEX to less than number of addressable indexes
STATUS[ERROR_STATUS][2]	Updating READ_INDEX to less than number of addressable indexes

51.5.1.9 Error description

When error_o is high this inidicates that an internal fault has occured.

NOTE

This signal is not driven by any of the interrupt sources

Sources of internal faults

- Duplication Registers

  The internal registers of the locked reigsters are duplicated. These are constantly monitor and when not equal, raises error.
- Entering into the WR_DIS State

  When the FSM enters the WR_DIS state, this will raise the error.
- Attack on the FSM State register

  When the FSM state is not equal to a defined state, this will raise the error.

51.5.2 Registers description (SFRs)

The block has only SFR space that can be accessed directly from the <APB/AHB> slave

51.5.2.1 GLIKEY register descriptions

51.5.2.1.1 GLIKEY memory map

GLIKEY0 base address: 4009_1D00h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register 0 SFR (CTRL_0)	32	RW	See section
4h	Control Register 1 SFR (CTRL_1)	32	RW	See section
8h	Interrupt Control (INTR_CTRL)	32	RW	See section
Ch	Status (STATUS)	32	R	See section
FCh	IP Version (VERSION)	32	R	See section

51.5.2.1.2 Control Register 0 SFR (CTRL_0)

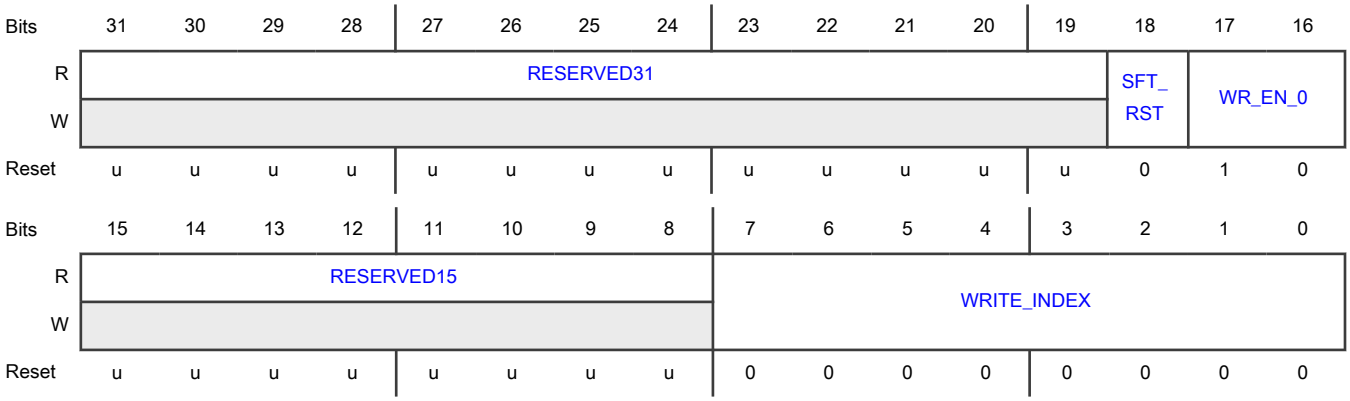
Offset

Register	Offset
CTRL_0	0h

Function

Control Register 0 SFR controls the flow of FSM

Diagram



Fields

Field	Function
31-19 RESERVED31	Reserved for Future Use
18 SFT_RST	Soft reset for the core reset (SFR configuration will be preseved).This register reads as 0 Use this field to bring the logic into a known state. 0b - No effect 1b - Triggers the soft reset
17-16 WR_EN_0	Write Enable 0 Controls the state of FSM. Use these bits with Write Enable 1.
15-8 RESERVED15	Reserved for Future Use
7-0 WRITE_INDEX	Write Index Targets given index for SSR.

51.5.2.1.3 Control Register 1 SFR (CTRL_1)

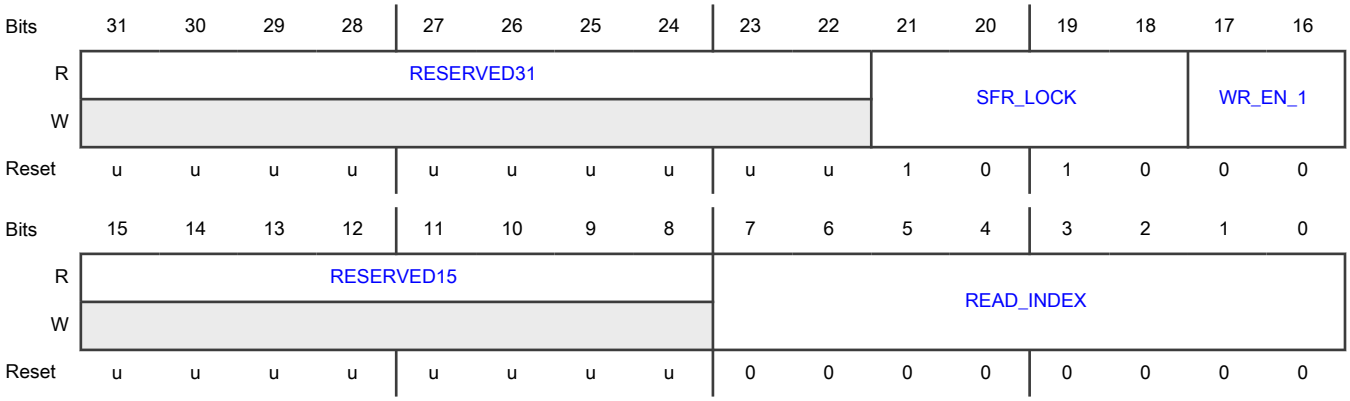
Offset

Register	Offset
CTRL_1	4h

Function

Control Register 1 SFR controls the flow of FSM.

Diagram



Fields

Field	Function
31-22 RESERVED31	Reserved for Future Use
21-18 SFR_LOCK	LOCK register for GLIKEY LOCK register for GLIKEY 4'h5 - Locked 4'hA - Unlocked **NOTE** When IP is locked, Write access to SFR is invalid
17-16 WR_EN_1	Write Enable One
15-8 RESERVED15	Reserved for Future Use

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 READ_INDEX	Index status, Writing an index value to this register will request the block to return the lock status of this index. Use this field to read the status of a target index.

51.5.2.1.4 Interrupt Control (INTR_CTRL)

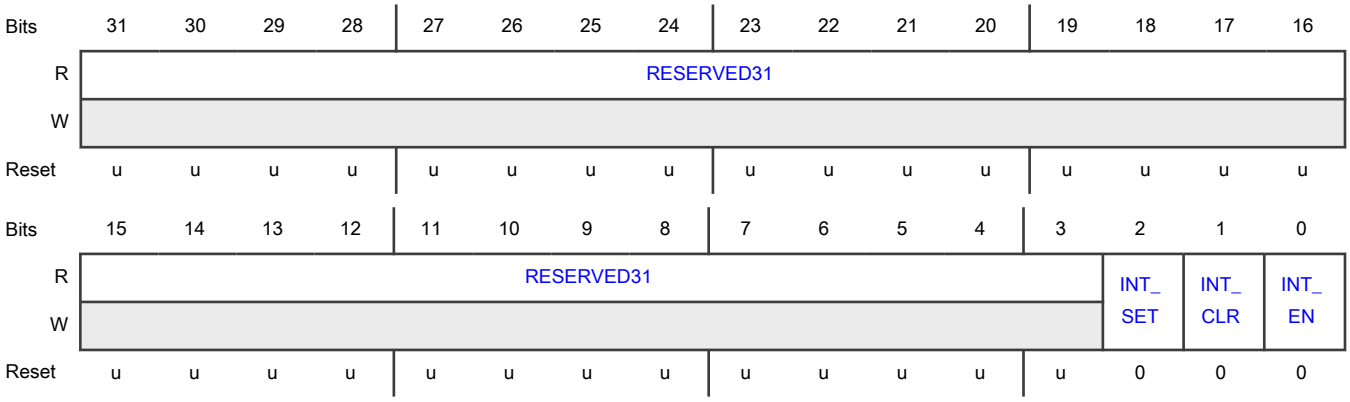
Offset

Register	Offset
INTR_CTRL	8h

Function

Interrupt Control Register

Diagram



Fields

Field	Function
31-3 RESERVED31	Reserved for Future Use
2 INT_SET	Interrupt Set. Writing a 1 to this register asserts the interrupt. This register reads as 0 Use this field to set interrupt. 0b - No effect 1b - Triggers interrupt
1	Interrupt Clear. Writing a 1 to this register creates a single interrupt clear pulse. This register reads as 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
INT_CLR	When the interrupt is set, use this field to clear this interrupt by writing 1 to this register.
0 INT_EN	Interrupt Enable. Writing a 1, Interrupt asserts on Interrupt output port Use this field to control whether to trigger an interrupt.

51.5.2.1.5 Status (STATUS)

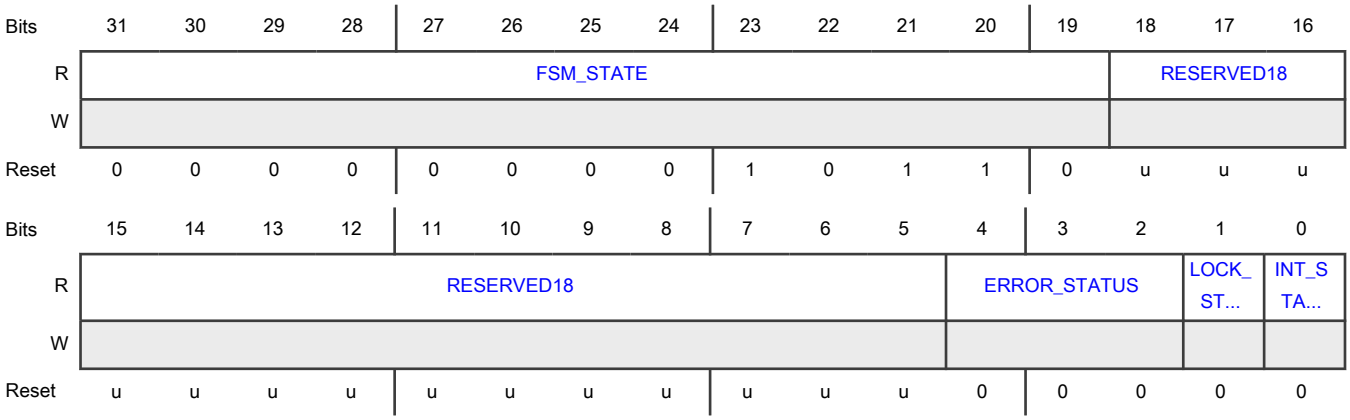
Offset

Register	Offset
STATUS	Ch

Function

Status Register provides the status of GLIKEY

Diagram



Fields

Field	Function
31-19 FSM_STATE	Status of FSM Status of FSM WR_DISABLED 13'b0_0000_0000_1011 INIT 13'b0_0000_0001_0110 STEP1 13'b0_0000_0010_1100 STEP2 13'b0_0000_0101_1000 STEP3 13'b0_0000_1011_0000

Table continues on the next page...

Table continued from the previous page...

Field	Function
	STEP4 13'b0_0001_0110_0000 Reserved 13'b0_0010_1100_0000 Reserved 13'b0_0101_1000_0000 Reserved 13'b0_1011_0000_0000 Reserved 13'b1_0110_0000_0000 LOCKED 13'b0_1100_0000_0001 WR_ENABLED 13'b1_1000_0000_0010 SSR_RESET 13'b1_0000_0000_0101
18-5 RESERVED18	Reserved for Future Use
4-2 ERROR_STATUS	Status of the Error Reports the status of error in GLIKEY, where Bit 0 (1 at position 0) indicates that an FSM error has occurred. Bit 1 (1 at position 1) indicates that software tried to address write index greater than the available value. Bit 2 (1 at position 2) indicates that software tried to address read index greater than the available value. 000b - No error 001b - FSM error has occurred 010b - Write index out of the bound (OOB) error 011b - Write index OOB and FSM error 100b - Read index OOB error 110b - Write index and read index OOB error 111b - Read index OOB, write index OOB, and FSM error
1 LOCK_STATUS	Provides the current lock status of indexes. 0b - Current read index is not locked 1b - Current read index is locked
0 INT_STATUS	Interrupt Status. Reflects the current status of interrupt. 0b - No effect 1b - Triggers interrupt

51.5.2.1.6 IP Version (VERSION)

Offset

Register	Offset
VERSION	FCh

Function

IP Version register

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved31				INDEX_CONFIG									FSM_CON...	MILESTONE	
W																
Reset	u	u	u	u	u	0	0	0	0	1	1	1	1	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved15				Reserved11				Reserved7				Reserved3			
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 Reserved31	Reserved for Future Use
26-19 INDEX_CONFIG	Configured number of addressable indexes
18 FSM_CONFIG	0:4 step, 1:8 step
17-16 MILESTONE	Release milestone. 00-PREL, 01-BR, 10-SI, 11-GO.
15-12 Reserved15	Reserved
11-8 Reserved11	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 Reserved7	Reserved
3-0 Reserved3	Reserved

Appendix A
General changes

A.1 General changes

- Editorial changes
  - Each new release of this document includes editorial improvements such as:
    - Spelling
    - Grammar
    - Punctuation
    - Voice
    - Tense
    - Capitalization
    - Formatting
    - Presentation
    - Navigation

- Updated the RM title to "MCX A156, A155, A154, A146, A145, and A144 Reference Manual" and revised the supporting part as "Supports MCXA144x, MCXA145x, MCXA146x, MCXA154x, MCXA155x, and MCXA156x".

Appendix B
Release notes

B.1 About This Manual changes

• Replaced the 'Master with controller' and 'slave with the target'in the diagram of the "Example of chip-specific information that refers to a different chapter" and "Example: chip-specific information that supersedes content in the same chapter".

B.2 Introduction chapter changes

• In Figure 5, added DSP.
• In Features, added packages HVQFN48 and LQFP64.

B.3 Core overview changes

No substantial content changes.

B.4 Bus and memory architecture chapter changes

• Updated Figure 6.
• In Flash subsystem, updated to "Swap granularity is 8 KB." • In System bus priority and arbitration, removed description of bold. • In Table 16, removed Testport.

B.5 Low Power Cache Controller (LPCAC)

B.5.1 Chip-specific LPCAC information changes

No substantial content changes.

B.5.2 LPCAC changes

No substantial content changes.

B.6 Flash Memory Module (FMU)

B.6.1 Chip-specific FMU information changes

No substantial content changes.

B.6.2 MSF1 changes

No substantial content changes.

B.7 Flash Memory Controller (FMC)

B.7.1 Chip-specific FMC information changes

No substantial content changes.

B.7.2 Flash Memory Controller (FMC-NPX) module changes

No substantial content changes.

B.8 Error Injection Module (EIM)

B.8.1 Chip-specific EIM information changes

No substantial content changes.

B.8.2 EIM module changes

- Added NOTE in Overview section.

B.9 Error Reporting Module (ERM)

B.9.1 Chip-specific ERM information changes

No substantial content changes.

B.9.2 ERM module changes

No substantial content changes.

B.10 Pinouts changes

- Updated Pin Assignments

B.11 Port Control (PORT)

B.11.1 Chip-specific PORT information changes

No substantial content changes.

B.11.2 PORT module changes

No substantial content changes.

B.12 General Purpose Input/Output (GPIO)

B.12.1 Chip-specific GPIO information changes

No substantial content changes.

B.12.2 GPIO module changes

- Removed "Access protection" topic.

B.13 Input Multiplexing (INPUTMUX)

B.13.1 Chip-specific INPUTMUX information changes

No substantial content changes.

B.13.2 INPUTMUX module changes

- Updated description of TRIGIN of ADC1_TRIG0 - ADC1_TRIG3 registers, changed ADC0 trigger inputs to ADC1 trigger inputs.

B.14 System Controller (SYSCON)

B.14.1 Chip-specific SYSCON information changes

No substantial content changes.

B.14.2 SYSCON module changes

No substantial content changes.

B.15 Enhanced DMA Controller (eDMA3)

B.15.1 Chip-specific eDMA3 information changes

No substantial content changes.

B.15.2 eDMA3 module changes

- Updated Clocking

- In Channel preemption updated the formula for calculating Priority vector (replaced CHn_* with CH_NUM).
- In Suspend/resume a DMA channel with active hardware service requests removed the statement "This section provides guidance on how to coherently suspend ------- Analog to Digital Convertor (SDADC), or other module".
- In Suspend an active DMA channel replaced "Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0" with "Disable the DMA service request at the SPI".
- In TCDn_ATTR[SMOD] removed bit field value 0_0001b.

B.16 Wakeup Unit (WUU)

B.16.1 Chip-specific WUU information changes

No substantial content changes.

B.16.2 WUU changes

No substantial content changes.

B.17 AND/OR INVERT (AOI)

B.17.1 Chip-specific AOI information changes

No substantial content changes.

B.17.2 AOI module changes

- Editorial fixes

B.18 Core Mode Controller (CMC)

B.18.1 Chip-specific CMC information changes

No substantial content changes.

B.18.2 CMC module changes

- Updated bit field values for CKCTRL[CKMODE].
- Updated Flash memory description.

- Updated bit field values for CKSTAT[CKMODE].

B.19 Reset changes

No substantial content changes.

B.20 Boot ROM changes

No substantial content changes.

B.21 Clocking chapter changes

No substantial content changes.

B.22 System Clock Generator (SCG)

B.22.1 Chip-specific SCG information changes

No substantial content changes.

B.22.2 SCG module changes

- Removed SOSCCSR[SOSC_SAFE_EN]

B.23 Power Management chapter changes

No substantial content changes.

B.24 VBAT

B.24.1 Chip-specific VBAT information changes

No substantial content changes.

B.24.2 VBAT module changes

- In Overview, updated "VBAT...FRO 16kHZ" to "VBAT...FRO16K".
- In Block diagram, updated "FRO_16K" to "FRO16K".
- In Clocks, updated "FRO16 kHz internal clock" to "FRO16K internal clock".

B.25 System Power Control (SPC)

B.25.1 Chip-specific SPC information changes

No substantial content changes.

B.25.2 SPC module changes

- Updated Power-on reset (POR) section.
- Updated voltage output values of "00b, 01b, 10b, and 11b" in ACTIVE_CFG [CORELDO_VDD_LVL] and LP_CFG[CORELDO_VDD_LVL].

B.26 Standard counter/timers (CTIMER)

B.26.1 Chip-specific CTIMER information changes

No substantial content changes.

B.26.2 CTIMER module changes

- In Functional description added description "Timer mode is selected in this case"
- In Operation sections added description "Counter mode is selected in this case."
- PWM2 for MAT2 waveform shifted left a bit in Figure 68 ."

B.27 Windowed Watchdog Timer (WWDT)

B.27.1 Chip-specific WWDT information changes

No substantial content changes.

B.27.2 WWDT module changes

- Reserved the bit 6 of MOD register.
- Minor update in MOD[WDINT].

- Added one note in MOD[WDRESET].

B.28 Micro-Tick Timer (UTICK)

B.28.1 Chip-specific UTICK information changes

No substantial content changes.

B.28.2 UTICK module changes

No substantial content changes.

B.29 OS Event Timer (OSTIMER)

B.29.1 Chip-specific OSTIMER information changes

No substantial content changes.

B.29.2 OSTIMER module changes

No substantial content changes.

B.30 Wake Timer

B.30.1 Chip-specific Wake Timer information changes

No substantial content changes.

B.30.2 Wake Timer module changes

No substantial content changes.

B.31 Enhanced Flex Pulse Width Modulator (eFlexPWM)

B.31.1 Chip-specific eFlexPWM information changes

- Added Module clocking.

B.31.2 eFlexPWM module changes

No substantial content changes.

B.32 Quadrature Decoder (eQDC)

B.32.1 Chip-specific eQDC information changes

No substantial content changes.

B.32.2 Quadrature Decoder module changes

- Editorial changes.

B.33 Frequency Measurement (FREQME)

B.33.1 Chip-specific FREQME information changes

No substantial content changes.

B.33.2 FREQME module changes

No substantial content changes.

B.34 Low-power Timer (LPTMR)

B.34.1 Chip-specific LPTMR information changes

No substantial content changes.

B.34.2 LPTMR module changes

No substantial content changes.

B.35 USB Full Speed (USBFS) Device Controller

B.35.1 Chip-specific USBFS information changes

No substantial content changes.

B.35.2 USBFS module changes

- Updated USB and VREGIN pin status detection

B.36 Flexible I/O (FLEXIO)

B.36.1 Chip-specific FlexIO information changes

No substantial content changes.

B.36.2 FLEXIO module changes

- Updated Pin operation.

B.37 FlexCAN module changes

- Updated "NOTE" in MCR[LPMACK] and MCR[FRZACK].
- Updated long description of ERFSR[ERFCLR].
- Updated the section Legacy RX FIFO structure (corrected the extended ID range in RXIDB_0 of format B from "296-" to "29-16").
- Updated the section Reset.
- Updated long description of CTRL2[PES].

B.38 Low Power Inter-Integrated Circuit (LPI2C)

B.38.1 Chip-specific LPI2C information changes

No substantial content changes.

B.38.2 LPI2C module changes

- Updated Pin configuration.

B.39 Low Power Universal Asynchronous Receiver/Transmitter (LPUART)

B.39.1 Chip-specific LPUART information changes

No substantial content changes.

B.39.2 LPUART module changes

No substantial content changes.

B.40 Low Power Serial Peripheral Interface (LPSPI)

B.40.1 Chip-specific LPSPI information changes

No substantial content changes.

B.40.2 LPSPI module changes

No substantial content changes.

B.41 Improved Inter-Integrated Circuit (I3C)

B.41.1 Chip-specific I3C module changes

No substantial content changes.

B.41.2 I3C module changes

• Updated Features. • Updated MCONFIG[HKEEP].
• Updated SDATACTRL[TXCOUNT]. • Updated SDATACTRL[RXCOUNT]. • Updated MDATACTRL[TXCOUNT]. • Updated MDATACTRL[RXCOUNT].
• Updated Features.
• Updated I2CBAUD row in Controller configuration. • Updated MCONFIG[I2CBAUD].
• Updated Block diagram. • Updated Protocol modes and states. • Updated External signals.

B.42 Analog-to-Digital Converter (ADC)

B.42.1 Chip-specific ADC information changes

• Module instances, changed from 12-bit ADC to 16-bit ADC.

B.42.2 ADC module changes

No substantial content changes.

B.43 Low Power Comparator (LPCMP)

B.43.1 Chip-specific LPCMP information changes

No substantial content changes.

B.43.2 LPCMP module changes

No substantial content changes.

B.44 12-bit Digital-to-Analog Converter (DAC)

B.44.1 Chip-specific 12-bit DAC information changes

No substantial content changes.

B.44.2 DAC module changes

- Updated bit description of GCR[FIFOEN].

B.45 Operational Amplifier (OPAMP)

B.45.1 Chip-specific OPAMP information changes

No substantial content changes.

B.45.2 OPAMP module changes

No substantial content changes.

B.46 Debug chapter changes

- Updated Figure 204, added arrow signal connectivity from AHB Matrix to AP1 DBG-MBOX.

B.47 Debug Mailbox

B.47.1 Chip-specific Debug Mailbox information changes

No substantial content changes.

B.47.2 DBGMB module changes

No substantial content changes.

B.48 Cyclic Redundancy Check (CRC)

B.48.1 Chip-specific CRC information changes

No substantial content changes.

B.48.2 CRC module changes

- In section Calculating a 16-bit CRC cross reference in step 2 “Program the transpose and complement options in Data (DATA) as required for the CRC calculation”)" changed to "Control (CTRL)".
- In section Calculating a 32-bit CRC cross reference in step 7 “CRC is computed on every data value write and the intermediate CRC result is stored back into DATA[LU] and DATA[LL]” changed to "DATA".
- In register DATA[HL] long description updated to "Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 0."

- Updated the description of Data (DATA) register.
- Updated the description of DATA[LL].

B.49 Memory Block Checker(MBC)

B.49.1 Chip-specific MBC information changes

No substantial content changes.

B.49.2 MBC module changes

- Generic editorial updates done in registers.

B.50 Code Watchdog Timer (CDOG)

B.50.1 Chip-specific CDOG information changes

No substantial content changes.

B.50.2 CDOG module changes

No substantial content changes.

B.51 GLIKEY

B.51.1 Chip-specific GLIKEY information changes

No substantial content changes.

B.51.2 GLIKEY module changes

• Updated GLIKEY register descriptions and Error description.
• Updated Lock SSR access.

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.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

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.

I2C-bus — logo is a trademark of NXP B.V.

For more information, please visit: https://www.nxp.com

Date of release: 9 December 2024
Document identifier: MCXAP100M96FS6RM