# Mask Set Errata for Mask 1N06M

This report applies to mask 1N06M for these products:

- MPC5746C
- MPC5746B
- MPC5746D
- MPC5745B
- MPC5745C
- MPC5745D

Mask Specific Information

| | |
|---|---|
| Major mask revision number | 0x0 |
| Minor mask revision number | 0x1 |
| JTAG identifier | 0x1988_401D |

**Table 1. Errata and Information Summary**

| Erratum ID | Erratum Title |
|---|---|
| e10327 | ADC: Incorrect channel under measure is indicated after sampling phase of conversion until conversion completes |
| e9996 | DSPI0 and DSPI1: Frame transfer does not restart after DSI frame matches preprogrammed value |
| e9995 | DSPI0 and DSPI1: Frame transfer does not restart in case of DSI parity error in master mode |
| e9656 | DSPI: Frame transfer does not restart in case of SPI parity error in master mode |
| e9978 | eMIOS: Unexpected channel flag assertion during GPIO to MCB mode transition |
| e7991 | FLASH: Rapid Program or Erase Suspend fail status |
| e8759 | FlexCAN: FD frame format not compliant to the new ISO/CD 11898-1: 2014-12-11 |
| e10368 | FlexCAN: Transition of the CAN FD operation enable bit may lead FlexCAN logic to an inconsistent state. |
| e8770 | FlexRAY: Missing TX frames on Channel B when in dual channel mode and Channel A is disabled |
| e8180 | HSM: e200z0 Nexus interface DQTAG implemented as variable length field in DQM message |
| e10117 | HSM: The True Random Number Generator (TRNG) can only use the16MHz internal RC oscillator (FIRC) as a clock source |
| e8951 | I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse |
| e7274 | LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state |
| e8933 | LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set |

*Table continues on the next page...*

**Table 1. Errata and Information Summary (continued)**

| Erratum ID | Erratum Title |
|---|---|
| e8970 | LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State |
| e10141 | LPU: LPU_RUN mode system clock must be preconfigured for undivided FIRC prior to LPU_STANDBY entry |
| e10132 | LPU: Mode transition to LPU_STOP or LPU_STANDBY may not complete |
| e10447 | MC_ME & LPU: The transition between DRUN/RUN mode to STANDBY or DRUN/RUN mode to LPU_RUN may not complete if a reset is asserted. |
| e10361 | MC_ME & LPU: The transition between DRUN/RUN mode to STANDBY or DRUN/RUN mode to LPU_RUN may not complete if EXR is asserted. |
| e10362 | MC_ME & LPU: The transition between DRUN/RUN mode to STANDBY or DRUN/RUN mode to LPU_RUN may not complete if any LVD is asserted or PORST goes low |
| e10323 | MC_ME: The transition from DRUN/RUN mode to STANDBY will not complete if a wake-up is triggered in a 50nS window. |
| e10340 | NEXUS: Trace messages following reset are corrupted when the system is reset during Nexus trace operation |
| e10058 | [ACMP] The output of Analog Comparator 0 (ACMP0) always Hi-Z during exit from STANDBY mode. |
| e9993 | [STCU]: If PLL is the source of STCU for online or offline self test execution and it suffers an unrecoverable loss of lock, the device would hang. |

**Table 2. Revision History**

| Revision | Changes |
|---|---|
| 0 | Initial revision |
| 1 | The following errata were revised.<br>• e8951 |
| 2 | The following errata were added.<br>• e9978<br>• e9996<br>• e9994<br>• e9995<br>• e9993<br>• e9992<br>• e9656<br>• e10058<br>The following errata were revised.<br>• e8933 |
| 3 | The following errata were removed.<br>• e9992<br>The following errata were added.<br>• e10132<br>• e10141<br>• e10117 |

*Table continues on the next page...*

**Table 2. Revision History (continued)**

| Revision | Changes |
|---|---|
| 3.1 04/16 | The following errata were added.<br>• e8759 |
| 4 July 2016 | The following errata were removed.<br>• e9994<br>The following errata were added.<br>• e10327<br>• e10323<br>• e10447<br>• e10340<br>• e10362<br>• e10361<br>• e10368<br>The following errata were revised.<br>• e10132 |

## e10327: ADC: Incorrect channel under measure is indicated after sampling phase of conversion until conversion completes

**Description:** The Main Status Register Channel under measure address field (ADC_MSR[CHADDR]) indicates which ADC channel is currently performing a conversion. This field indicates the correct channel during the sampling phase of conversion, but will display an incorrect value in the subsequent phases until conversion is complete.

**Workaround:** User must only consider ADC_MSR[CHADDR] to be valid when the ADC is in the sample phase of conversion. The Main Status Register Status of the ADC field shows when the ADC is in the sample phase (ADC_MSR[ADCSTATUS] = 0b100).

## e9996: DSPI0 and DSPI1: Frame transfer does not restart after DSI frame matches preprogrammed value

**Description:** In the Deserial Serial Peripheral Interface module, in the scenario when:

1. Master/slave mode select bit of module configuration register is set (MCR[MSTR]=0b1) to configure the module in master mode

2. Deserial Serial Interface (DSI) communication is selected via DSPI Configuration field (DCONF) in the Module Configuration Register (MCR [DCONF] = 0b01)

3. Preprogrammed value for data match with received DSI frame is configured using DSI De-serialized Data Polarity Interrupt Register (DPIR) and DSI De-serialized Data Interrupt Mask Register (DIMR)

4. Data Match Stop (DMS) bit of DSI configuration register0 is set (DSICR0 [DMS] =0b1) which stops DSI frame transfer in case of a data match with a preprogrammed value

5. DSI frame is received with bits matching preprogrammed value.

Under these conditions, the next frame transfer is stopped, DSI Data Received with Active Bits bit of status register is set (SR [DDIF] =0b1) and the corresponding DDIF interrupt is asserted. Even after the interrupt is serviced and SR [DDIF] is reset, the frame transfer does not restart.

**Workaround:** DSI frame transfer stop in case of DSI data match condition should be disabled. For this, keep the data match stop bit of DSI configuration register 0 de-asserted (DSICR0 [DMS]=0b0)

## e9995: DSPI0 and DSPI1: Frame transfer does not restart in case of DSI parity error in master mode

**Description:** In the Serial Peripheral Interface module, in the scenario when:

1. Master/slave mode select bit of module configuration register is set (MCR[MSTR]=0b1) to configure the module in master mode

2. Deserial Serial Interface (DSI) communication is selected via DSPI Configuration field (DCONF) in MCR (MCR[DCONF] = 0b01)

3. Parity reception check on received DSI frame is enabled by setting Parity Enable bit (PE) of DSI configuration register 0 (DSICR0[PE]=0b1)

4. Parity Error Stop (PES) bit of DSI configuration register0 is set (DSICR0[PES]=0b1) which stops DSI frame transfer in case of parity error

5. Parity error is detected on received frame

Then the next frame transfer is stopped, DSI parity error flag bit of status register is set (SR[DPEF] =0b1) and the corresponding DSI parity error interrupt is asserted. Even after the interrupt is serviced and SR [DPEF] is reset, the frame transfer does not restart.

**Workaround:** DSI frame transfer stop in case of parity error detection should be disabled. For this, keep the parity error stop bit of DSI configuration register0 de-asserted (DSICR0 [PES]=0b0).

## e9656: DSPI: Frame transfer does not restart in case of SPI parity error in master mode

**Description:** In the Deserial Serial Peripheral Interface (DSPI) module, in the scenario when:

1. Master/slave mode select bit (MTSR) of Module Configuration register (MCR) is set (MCR[MSTR]=0b1) to configure the module in master mode

2. SPI communication is selected via DSPI Configuration field (DCONF) in MCR (MCR[DCONF] = 0b00)

3. Parity reception check on received frame is enabled by setting the Parity Enable or Mask tASC delay (PE_MASC) bit of DSPI PUSH FIFO Register In Master Mode (PUSHR), i.e. PUSHR[PE]=0b1.

4. Parity Error Stop bit (PES) of MCR is set (MCR[PES]=0b1) which stops SPI frame transfer in case of parity error

5. Parity error is detected on received frame.

Then the next frame transfer is stopped, the SPI Parity Error Flag bit (SPEF) of the DSPI Status Register (DSPI_SR) is set (SR[SPEF] =0b1) and the corresponding SPI parity error interrupt is asserted. Even after the interrupt is serviced and SR[SPEF] is reset, the frame transfer does not restart.

**Workaround:** Do not use SPI frame transfer stop in case of parity error detection for SPI transmission in master mode. For this, keep the Parity Error Stop bit of Module Configuration Register de-asserted (MCR[PES] = 0b0).

## e9978: eMIOS: Unexpected channel flag assertion during GPIO to MCB mode transition

**Description:** When changing an Enhanced Modular IO Subsystem (eMIOS) channel mode from General Purpose Input/Output (GPIO) to Modulus Counter Buffered (MCB) mode, the channel flag in the eMIOS Channel Status register (eMIOS_Sn[FLAG]) may incorrectly be asserted. This will cause an unexpected interrupt or DMA request if enabled for that channel.

**Workaround:** In order to change the channel mode from GPIO to MCB without causing an unexpected interrupt or DMA request, perform the following steps:

(1) Clear the FLAG enable bit in the eMIOS Control register (eMIOS_Cn[FEN] = 0).

(2) Change the channel mode (eMIOS_Cn[MODE]) to the desired MCB mode.

(3) Clear the channel FLAG bit by writing ‘1’ to the eMIOS Channel Status register FLAG field (eMIOS_Sn[FLAG] = 1).

(4) Set the FLAG enable bit (eMIOS_Cn[FEN] = 1) to re-enable the channel interrupt or DMA request reaction.

## e7991: FLASH: Rapid Program or Erase Suspend fail status

**Description:** If a flash suspend operation occurs during a 5us window during a verify operation being executed by the internal flash program and erase state machine, and the suspend rate continues at a consistent 20us rate after that, it is possible that the flash will not exit the program or erase operation. A single suspend during a single program or erase event will not cause this issue to occur.

Per the flash specification, a flash program or erase operation should not be suspended more than once every 20 us, therefore, if this requirement is met, no issue will be seen. IF the suspend rate is faster than 20 us continuously, a failure to program/erase could occur.

**Workaround:** When doing repeated suspends during program or erase ensure that suspend period is greater than 20us.

## e8759: FlexCAN: FD frame format not compliant to the new ISO/CD 11898-1: 2014-12-11

**Description:** This version of the device implements a Flexible Controller Area Network (FlexCAN) module version that implements a Flexible Data (CAN-FD) frame format according to ISO/WD 11898-1: 2013-12-13. However, it is not compliant with the new ISO/CD 11898-1: 2014-12-11 format. The frame format was updated during the ISO standardization process.

The limitations are the following:

- the FD frame format is incompatible, the Cyclic Redundancy Check [CRC] does not include the added stuff bit count field
- the FD CRC computation is incompatible, a different seed value is used.

As a consequence this device is not suitable for use in CAN-FD networks that use the new FD frame format according to ISO/CD 11898-1: 2014-12-11.

FlexCAN3 with CAN FD feature enabled is affected by this defect.

**Workaround:** Use CAN-FD mode in networks that only includes devices that conform to the ISO/WD 11898-1: 2013-12-13 frame format.

The Classic CAN mode is unaffected and can be used without restrictions.

## e10368: FlexCAN: Transition of the CAN FD operation enable bit may lead FlexCAN logic to an inconsistent state.

**Description:** The activation or deactivation of the CAN FD operation by setting or clearing the FDEN bit of the CAN_MCR register or by setting the FlexCAN soft reset bit (SOFTRST) of the CAN_MCR register when the FDEN bit is enabled may cause an internal FlexCAN register to become metastable. As result, the first CAN frame, transmitted or received, may have corrupted data (ID and payload). However, even though the data is corrupted, a valid CAN frame is transmitted because the Cyclic Redundancy Check (CRC) calculation is based on the corrupted data. During reception the data is corrupted internally after the CRC bits have been checked and therefore this corrupted data may be stored in a reception message buffer. After the first CAN frame, all subsequent frames are transmitted and received correctly.

**Workaround:** Perform the following steps to set the FDEN bit:

1. If FlexCAN is already in freeze mode, go to step 3, otherwise set the HALT and FRZ bits of the CAN_MCR register.

2. Wait the FRZACK bit of the CAN_MCR register to be set by the hardware.

3. Set the LPB (Loop Back Mode) bit of the CAN_CTRL1 register.

4. Configure only one message buffer to be transmitted. The frame should be a classical one (non-FD) with IDE =0, RTR =1 DLC =0x5 and STD_ID =0x682.

5. Set the FDEN bit of the CAN_MCR register.

6. Clear the HALT bit of the MCR register to leave freeze mode.

7. Wait the FRZACK bit of the CAN_MCR register to be cleared by the hardware.

8. Wait the respective bit of the CAN_IFLAG register to be set (successfully transmission in loop back mode).

9. Clear the respective bit of the CAN_IFLAG register by writing 1.

10. Set the HALT and FRZ bits of the CAN_MCR register.

11. Wait the FRZACK bit of the CAN_MCR register to be set by the hardware.

12. Clear the LPB (Loop Back Mode) bit of the CAN_CTRL1 register.

Perform the following steps to apply a soft reset or clear the FDEN bit:

1. If FlexCAN is already in freeze mode, go to step 3, otherwise set the HALT and FRZ bits of the CAN_MCR register.

2. Wait the FRZACK bit of the CAN_MCR register to be set by the hardware.

3. Set the SOFTRST bit of the CAN_MCR register.

4. Wait the SOFTRST bit of the CAN_MCR register to be cleared by the hardware.

5. Set again the SOFTRST bit of the CAN_MCR register.

6. Wait the SOFTRST bit of the CAN_MCR register to be cleared by the hardware.

## e8770: FlexRAY: Missing TX frames on Channel B when in dual channel mode and Channel A is disabled

**Description:** If the FlexRay module is configured in Dual Channel mode, by clearing the Single Channel Device Mode bit (SCM) of the Module Control register (FR_MCR[SCM]=0), and Channel A is disabled, by clearing the Channel A Enable bit (FR_MCR[CHA]=0) and Channel B is enabled, by setting the Channel B enable bit (FR_MCR[CHB]=1), there will be a missing transmit (TX) frame in adjacent minislots (even/odd combinations in Dynamic Segment) on Channel B for certain communication cycles. Which channel handles the Dynamic Segment or Static Segment TX message buffers (MBs) is controlled by the Channel Assignment bits (CHA, CHB) of the Message Buffer Cycle Counter Filter Register (FR_MBCCFRn). The internal Static Segment boundary indicator actually only uses the Channel A slot counter to identify the Static Segment boundary even if the module configures the Static Segment to Channel B (FR_MBCCFRn[CHA]=0 and FR_MBCCFRn[CHB]=1). This results in the Buffer Control Unit waiting for a corresponding data acknowledge signal for minislot:N in the Dynamic Segment and misses the required TX frame transmission within the immediate next minislot:N+1.

**Workaround:** 1. Configure the FlexRay module in Single Channel mode (FR_MCR[SCM]=1) and enable Channel B (FR_MCR[CHB]=1) and disable Channel A (FR_MCR[CHA]=0). In this mode the internal Channel A behaves as FlexRay Channel B. Note that in this mode only the internal channel A and the FlexRay Port A is used. So externally you must connect to FlexRay Port A.

2. Enable both Channel A and Channel B when in Dual Channel mode (FR_MCR[CHA=1] and FR_MCR[CHB]=1). This will allow all configured TX frames to be transmitted correctly on Channel B.

## e8180: HSM: e200z0 Nexus interface DQTAG implemented as variable length field in DQM message

**Description:** The Hardware Security Module (HSM) core (e200z0) implements the Data Tag (DQTAG) field of the Nexus Data Acquisition Message (DQM) as a variable length packet instead of an 8-bit fixed length packet. This may result in an extra clock ("beat") in the DQM trace message depending on the Nexus port width selected for the device.

**Workaround:** Tools should decode the DQTAG field as a variable length packet instead of a fixed length packet.

## e10117: HSM: The True Random Number Generator (TRNG) can only use the16MHz internal RC oscillator (FIRC) as a clock source

**Description:** The user must not select the Fast External Oscillator (FXOSC) as a clock source for the HSM TRNG. The HSM TRNG clock source is selected at the Auxilliary Clock 3 Select Control Register, selection control bit (MC_CGM_AC3_SC[SELCTL]).

**Workaround:** The user should select the FIRC for the HSM TRNG clock by clearing MC_CGM_AC3_SC[SELCTL] (this is the default setting).

## e8951: I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse

**Description:** When the I2C (Inter-Integrated Circuit) is operating in a multi-master network and a start cycle is attempted by the I2C device when the bus is busy, the attempting master will lose arbitration as expected but a short extra clock cycle is generated in the bus. After losing arbitration, the master switches to slave mode but it does not detect the short clock pulse. The acknowledge signal is expected at the ninth clock by the current bus master but it is not sent as expected due to the undetected short clock pulse.

**Workaround:** Software must ensure that the I2C BUS is idle by checking the bus busy bit in the I2C Bus Status Register (I2C_IBSR.IBB) before switching to master mode and attempting a Start cycle.

## e7274: LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state

**Description:** As per the Local Interconnect Network (LIN) specification, the processing of one frame should be aborted by the detection of a new header sequence and the LIN Finite State Machine (FSM) should move to the protected identifier (PID) state. In the PID state, the LIN FSM waits for the detection of an eight bit frame identifier value.

In LINFlexD, if the LIN Slave receives a new header instead of data response corresponding to a previous header received, it triggers a framing error during the new header's reception and returns to IDLE state.

**Workaround:** The following three steps should be followed -

1) Configure slave to Set the MODE bit in the LIN Time-Out Control Status Register (LINTCSR[MODE]) to '0'.

2) Configure slave to Set Idle on Timeout in the LINTCSR[IOT] register to '1'. This causes the LIN Slave to go to an IDLE state before the next header arrives, which will be accepted without any framing error.

3) Configure master to wait for Frame maximum time (T Frame_Maximum as per LIN specifications) before sending the next header.

Note:

THeader_Nominal = 34 * TBit

TResponse_Nominal = 10 * (NData + 1) * TBit

THeader_Maximum = 1.4 * THeader_Nominal

TResponse_Maximum = 1.4 * TResponse_Nominal

TFrame_Maximum = THeader_Maximum + TResponse_Maximum

where TBit is the nominal time required to transmit a bit and NData is number of bits sent.

## e8933: LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set

**Description:** When the LINFlexD module is configured as follows:

1. LIN (Local Interconnect Network) slave mode is enabled by clearing the Master Mode Enable bit in the LIN Control Register 1 (LINCR1[MME] = 0b0)

2. Auto synchronization is enabled by setting LIN Auto Synchronization Enable (LINCR1[LASE] = 0b1)

The LINFlexD module may automatically synchronize to an incorrect baud rate without setting the Sync Field Error Flag in the LIN Error Status register (LINESR[SFEF]) in case Sync Field value is not equal to 0x55, as per the Local Interconnect Network (LIN) specification.

The auto synchronization is only required when the baud-rate in the slave node can not be programmed directly in software and the slave node must synchronize to the master node baud rate.

**Workaround:** There are 2 possible workarounds.

Workaround 1:

When the LIN time-out counter is configured in LIN Mode by clearing the MODE bit of the LIN Time-Out Control Status register (LINTCSR[MODE]= 0x0]):

1. Set the LIN state Interrupt enable bit in the LIN Interrupt Enable register (LINIER[LSIE] = 0b1)

2. When the Data Reception Completed Flag is asserted in the LIN Status Register (LINSR[DRF] = 0b1) read the LIN State field (LINSR[LINS])

3. If LINSR[LINS]= 0b0101, read the Counter Value field of the LIN Time-Out Control Status register (LINTCSR[CNT]), otherwise repeat step 2

4. If LINTCSR[CNT] is greater than 0xA, discard the frame.

When the LIN Time-out counter is configured in Output Compare Mode by setting the LINTCSR[MODE] bit:

1. Set the LIN State Interrupt Enable bit in the LIN Interrupt Enable register (LINIER[LSIE])

2. When the Data Reception Completed flag bit is asserted in the LIN Status Register (LINSR[DRF] = 0b1), read the LINSR[LINS] field

3. If LINSR[LINS]= 0b0101, store LINTCSR[CNT] value in a variable (ValueA), otherwise repeat step 2

4. Clear LINSR[DRF] flag by writing LINSR[LINS] field with 0xF

5. Wait for LINSR[DRF] to become asserted again and read LINSR[LINS] field

6. If LINSR[LINS] = 0b0101, store LINTCSR[CNT] value in a variable (ValueB), else repeat step 4

7. If ValueB – ValueA is greater than 0xA, discard the frame

Workaround 2:

Do not use the auto synchronization feature (disable with LINCR1[LASE] = 0b0) in LIN slave mode.

## e8970: LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State

**Description:** The LINFlexD module may set a spurious Bit Error Flag (BEF) in the LIN Error Status Register (LINESR), when the LINFlexD module is configured as follows:

- Data Size greater than eight data bytes (extended frames) by configuring the Data Field Length (DFL) bitfield in the Buffer Identifier Register (BIDR) with a value greater than seven (eight data bytes)
- Bit error is able to reset the LIN state machine by setting Idle on Bit Error (IOBE) bit in the LIN Control Register 2 (LINCR2)

As consequence, the state machine may go to the Idle State when the LINFlexD module tries the transmission of the next eight bytes, after the first ones have been successfully transmitted and Data Buffer Empty Flag (DBEF) was set in the LIN Status Register (LINSR).

**Workaround:** Do not use the extended frame mode by configuring Data Field Length (DFL) bit-field with a value less than eight in the Buffer Identifier Register (BIDR) (BIDR[DFL] < 8)

## e10141: LPU: LPU_RUN mode system clock must be preconfigured for undivided FIRC prior to LPU_STANDBY entry

**Description:** If the LPU_RUN mode system clock is selected to be FXOSC or divided-FIRC when LPU_STANDBY mode is entered then the MCU may not return to LPU_RUN mode on a wake-up event.

In LPU_RUN mode the FXOSC or divided-FIRC can be used as the system clock, but the user must ensure that the undivided FIRC is selected as the system clock before the LPU_STANDBY mode transition is initiated.

**Workaround:** Prior to entering LPU_STANDBY select undivided FIRC as the LPU System Clock by configuring LPU_RUN_CF[SYS_CLK_SEL] = 0 and FIRC_CTL[FIRCDIV] = 5’b0.

## e10132: LPU: Mode transition to LPU_STOP or LPU_STANDBY may not complete

**Description:** A mode transition from LPU_RUN to LPU_STOP or LPU_RUN to LPU_STANDBY may not complete if a wake-up or interrupt is received in a 5 FIRC clock window after the mode transition is requested. This is only applicable if the FIRC is disabled in LPU_STOP and LPU_STANDBY. In this scenario, the z2 core is stopped and if the System Watchdog Timer (SWT) is enabled, the SWT continues to run, the SWT will timeout and a SWT destructive reset will be triggered.

**Workaround:** The user can select one of the following workarounds:

1. Enable the SWT during LPU modes to enable recovery through destructive reset

2. Enable the FIRC in LPU_STOP and LPU_STANDBY

## e10447: MC_ME & LPU: The transition between DRUN/RUN mode to STANDBY or DRUN/RUN mode to LPU_RUN may not complete if a reset is asserted.

**Description:** The following reset sources can cause the errata condition but all can either be disabled via a control register or avoided as they are triggered by software.

Destructive Resets indicated at MC_RGM_DES:

• Software Watchdog Timer 0, MC_RGM_DES[F_SWT0_RES]

• Software Watchdog Timer 1, MC_RGM_DES[F_SWT1_RES]

Functional Resets indicated at MC_RGM_FES:

• Non Maskable Interrupt from Wakeup Unit, MC_RGM_FES[F_NMI_WKPU]

• Clock Monitor Unit FXOSC less than FIRC, MC_RGM_FES[F_CMU_OLR]

• Fault Collection and Control Unit Long Functional Reset, MC_RGM_FES[F_FCCU_LONG]

• Fault Collection and Control Unit Short Functional Reset, MC_RGM_FES[F_FCCU_SHORT]

• VDD_HV_A Low Voltage Detect, MC_RGM_FES[F_LVD_IO_A_HI]

• High Voltage Detect, MC_RGM_FES[F_HVD_LV_cold]

• Power Domain 2 Low Voltage Detect, MC_RGM_FES[F_LVD_LV_PD2_cold]

At the DRUN/RUNx to STANDBY transition there are 2 windows (each 50nS typical) at which time if any of the listed resets are asserted, the mode transition will not complete. The 2 windows occur in the STANDBY entry transition period (20uS typical) - this period is from the mode transition request at the MC_ME_MCTL register to a toggle of the EXTREGC (External Regulator Control) pin. The EXTREG pin signals the low power transition is complete.

At the DRUN/RUN to LPU_RUN transition there are 3 windows:

1. A single window, 686nS (typical) for DRUN-FIRC or 236nS (typical) for DRUN-FMPLL160MHZ.

2. Plus 2 other windows each 50nS typical.

If any of the listed resets are asserted in any of the 3 windows the mode transition will not complete. The 3 windows occur in the LPU_RUN entry transition period (20uS typical) - this period is from the mode transition request at the MC_ME_MCTL register to a toggle of the EXTREGC pin.

For the case the mode transition does not complete the MCU will be stuck in reset or stuck in STANDBY and will only recover via a power-cycle of VDD_HVA.

**Workaround:** Prior to transitioning to STANDBY or LPU_RUN mode the application should:

Configure the following reset sources so they cannot be triggered in the window of susceptibility:

• Software Watchdog Timer 0, MC_RGM_DES[F_SWT0_RES]

• Software Watchdog Timer 1, MC_RGM_DES[F_SWT1_RES]

Disable the following reset sources:

• Functional Reset Escalation, MC_RGM_FES[F_FUNC_ESC]

• Non Maskable Interrupt from Wakeup Unit, MC_RGM_FES[F_NMI_WKPU]

• Clock Monitor Unit FXOSC less than FIRC, MC_RGM_FES[F_CMU_OLR]

• Fault Collection and Control Unit Long Functional Reset, MC_RGM_FES[F_FCCU_LONG]

• Fault Collection and Control Unit Short Functional Reset, MC_RGM_FES[F_FCCU_SHORT]

• VDD_HV_A Low Voltage Detect, MC_RGM_FES[F_LVD_IO_A_HI]

• High Voltage Detect, MC_RGM_FES[F_HVD_LV_cold]

• Power Domain 2 Low Voltage Detect, MC_RGM_FES[F_LVD_LV_PD2_cold]

## e10361: MC_ME & LPU: The transition between DRUN/RUN mode to STANDBY or DRUN/RUN mode to LPU_RUN may not complete if EXR is asserted.

**Description:** At the DRUN/RUNx to STANDBY transition there are 2 windows (each 50nS typical) at which time if the External Reset (EXR) is asserted, the mode transition will not complete. The 2 windows occur in the STANDBY entry transition period (20uS typical) - this period is from the mode transition request at the MC_ME_MCTL register to a toggle of the EXTREGC (External Regulator Control) pin. The EXTREG pin signals the low power transition is complete.

At the DRUN/RUN to LPU_RUN transition there are 3 windows:

1. A single window, 686nS (typical) for DRUN-FIRC or 236nS (typical) for DRUN-FMPLL160MHZ.

2. Plus 2 other windows each 50nS typical.

If EXR is asserted in any of the 3 windows the mode transition will not complete. The 3 windows occur in the LPU_RUN entry transition period (20uS typical) - this period is from the mode transition request at the MC_ME_MCTL register to a toggle of the EXTREGC pin.

For the case the mode transition does not complete the MCU will be stuck in reset or stuck in STANDBY and will only recover via a power-cycle of VDD_HVA.

**Workaround:** 1. Ensure that External Reset (EXR) is not triggered during the windows of susceptibility at the entry to STANDBY mode or at the entry to LPU_RUN mode.

2. Alternatively, use the unaffected low power mode STOP.

## e10362: MC_ME & LPU: The transition between DRUN/RUN mode to STANDBY or DRUN/RUN mode to LPU_RUN may not complete if any LVD is asserted or PORST goes low

**Description:** At power-up to DRUN there is a window (50nS typical) at which time if any of the Low Voltage Detects (LVDs) are asserted or PORST goes low, the mode transition will not complete.

At the DRUN/RUNx to STANDBY transition or DRUN/RUN to LPU_RUN there are 2 windows (each 50nS typical) at which time if any of the LVDs are asserted, the mode transition will not complete. The 2 windows occur in the STANDBY/LPU entry transition period (20uS typical) - this period is from the mode transition request at the MC_ME_MCTL register to a toggle of the EXTREGC (External Regulator Control) pin. The EXTREG pin signals the low power transition is complete.

For the case the mode transition does not complete the MCU will be stuck in reset and will only recover via a power-cycle of VDD_HVA.

Upon wake-up from STANDBY or LPU_RUN modes there is a single window (50nS typical) at which time if any of the LVDs are asserted or PORST is asserted, the mode transition will not complete. This window occurs in the STANDBY/LPU exit transition period (12uS typical) immediately after assertion of the wake-up signal. For this case when the mode transition does not complete the MCU will be stuck in reset and recover via a power-cycle of VDD_HVA.

**Workaround:** 1. Ensure that no Low Voltage Detect (LVD) is triggered or PORST goes low during the windows of susceptibility at Power-up, at the entry to STANDBY mode, at the exit of STANDBY, at the exit of LPU modes, or at the entry to LPU_RUN mode.

2. Alternatively, use the unaffected low power mode STOP.

## e10323: MC_ME: The transition from DRUN/RUN mode to STANDBY will not complete if a wake-up is triggered in a 50nS window.

**Description:** At the DRUN/RUNx to STANDBY mode transition there are 2 windows (each 50nS typical) at which time if a WKPU (wake-up) occurs the mode transition will not complete. The 2 windows occur in the STANDBY entry transition period (20uS typical) - this period is from the mode transition request at the MC_ME_MCTL register to a toggle of the EXTREGC (External Regulator Control) pin. The EXTREGC pin signals the low power transition is complete.

For the case the mode transition does not complete the MCU will be stuck in reset (window #1) or stuck in STANDBY (window #2) and will only recover via a power-cycle of VDD_HVA.

**Workaround:** Ensure wake-ups are not triggered in the STANDBY entry transition period during the DRUN to STANDBY mode transition, by adhering to all of the following:

- If the application cannot guarantee to avoid triggering an external wake-up during the STANDBY entry transition period, prior to entering STANDBY mode all external wake-ups must be disabled.
- Application SW should use a periodic wake-up (RTC-API) and poll WKPU_WISR. If a wake-up is recorded at WKPU_WISR this signals to the application SW that an external wake-up has occurred whilst in STANDBY mode.
- The RTC-API timer must not timeout during the STANDBY entry transition period.

Alternatively, use an unaffected mode

a) LPU_STANDBY(FIRC-on) rather than STANDBY. In LPU_STANDBY mode, external wake-ups can be enabled (see also e10132).

b) STOP mode.

## e10340: NEXUS: Trace messages following reset are corrupted when the system is reset during Nexus trace operation

**Description:** If the device is reset (either a software initiated reset or an external reset) the first 256 Nexus trace messages generated after the device exits the reset sequence will be corrupted.

**Workaround:** For a software initiated reset, the issue can be avoided by executing an indirect branch instruction (for example, using a “while(1);” C statement) immediately after the reset is triggered. The reset operation will be triggered by a register transfer to the Reset Generation Module (MC_RGM) and will take time to propagate to the core that is executing code, giving time for the instruction to be executed immediately after this transfer.

## e10058: [ACMP] The output of Analog Comparator 0 (ACMP0) always Hi-Z during exit from STANDBY mode.

**Description:** When the CMP0_STDBY bit in the General Programmable Control Register (GPR_CTL[14]) is set to a 1, the output of Analog Comparator 0 (ACMP0) should drive continuously on PAD 42 whilst the device is in Standby mode and during Standby mode exit. Instead, during STANDBY mode exit , the output pad of ACMP0 is set to a Hi-Z state.

**Workaround:** Enable PAD keeping functionality on all Standby Pads. Program the PAD Keeper Enable (PAD_KEEP_EN) bit in the Power Management Controller Digital Interface RAM domain configuration register PMCDIG_RDCR[PAD_KEEP_EN] to a 1 to enable it.

## e9993: [STCU]: If PLL is the source of STCU for online or offline self test execution and it suffers an unrecoverable loss of lock, the device would hang.

**Description:** During online or offline self-test execution, a PLL loss of lock event is non recoverable resulting in the device stuck in reset. This could potentially happen if the PLL is using the external oscillator as it’s clock source and the external crystal oscillator becomes non functional.

**Workaround:** Use an alternative clock source for offline or online self test execution. This could be either the FIRC or PLL with FIRC as reference clock.