Mask Set Errata for Mask 0N50N

This report applies to mask 0N50N for these products:

- MPC5777M

Mask Specific Information

Major mask revision number	0x1
Minor mask revision number	0x1
JTAG identifier	0x1AF0_F01D
GTM Version	GTM104: IP Release v1.5.5-A1

Table 1. Errata and Information Summary

Erratum ID	Erratum Title
ERR010842	CMU: False upper frequency threshold alarm (FHH) signaled to Fault Collection and Control Unit (FCCU)
ERR050159	CMU: Monitoring signal for sudden loss of clock from CMU does not signal the Fault Collection and Control Unit
ERR007116	CRC: AutoSAR 4.0 8-bit CRC8 0x2F is not supported in hardware
ERR007824	DCI: Avoid asserting system reset when switching JTAG operating modes
ERR008343	DCI: EVTO[1:0] outputs remain stuck low if asserted while the system clock source is the IRC
ERR007111	DMA: DMA does not work properly with M_CAN modules
ERR007113	DMA: GTM accesses via DMA may fail
ERR010663	DS/RM: IRC trimming and LVD/HVD clarifications
ERR007351	DSMC: Lockstep-error notification is disabled
ERR050090	DSPI/SPI: Incorrect data may be transmitted in slave mode
ERR009783	DSPI: Frame transfer does not restart after DSI frame matches preprogrammed value
ERR009664	DSPI: Frame transfer does not restart in case of DSI parity error in master mode
ERR009656	DSPI: Frame transfer does not restart in case of SPI parity error in master mode
ERR007115	DSPI: Mixing 16 and 32 bits frame size in XSPI Mode can cause incorrect data to be transmitted
ERR007352	DSPI: reserved bits in slave CTAR are writable
ERR007299	DSPI: SOUT pins are not automatically configured as SOUT when Slave Select is asserted in Slave Mode
ERR010542	DSPI: Transmit, Command, and Receive FIFO fill flags in status register is not cleared when DMA is improperly configured

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR008547	e200z425: Additional cycles required for Load/Store accesses to guarded/precise space
ERR010385	e200z4: Incorrect branch displacement at 16K memory boundaries
ERR007210	e200z710: Nexus timestamps are not available
ERR007259	e200zx: ICNT and branch history information may be incorrect following a nexus overflow
ERR007305	e200zx: JTAG reads of the Performance Monitor Counter registers are not reliable
ERR007408	EBI: Minimum input setup time is 7 ns
ERR006967	eDMA: Possible misbehavior of a preempted channel when using continuous link mode
ERR010452	eDMA: When master ID replication is enabled, the stored ID and privilege level will change if read by another master.
ERR007337	FCCU: Address feedback error disabled for the Safety Core local RAMS and Cache
ERR008034	FCCU: Channel 40 fault (COMP_XBIC_DSMC_Monitor) may be reported if a CPU is reset upon a MC_ME mode change
ERR008229	FCCU: Enabling the programmable glitch filter on EIN may cause a destructive reset
ERR008042	FCCU: EOUT signals are active, even when error out signaling is disabled
ERR007099	FCCU: Error pin signal length is not extended when the next enabled fault, with its alarm timeout disabled, occurs
ERR010900	FCCU: False indication of a fault state for a single safe clock period can be generated on the error output pin
ERR007223	FCCU: FCCU_IRQ_EN register is writeable in all operating modes
ERR007869	FCCU: FOSU monitoring of a fault is blocked for second or later occurrence of the same fault
ERR009584	FCCU: MPC57xxM – FOSU may assert destructive reset when a hardware recoverable fault of width less than one safe clock period occurs
ERR007226	FCCU: the error-out signalling cannot be disabled in non Bi-stable protocols
ERR007325	FCCU: Unsuccessful decorated storage access may cause erroneous signaling of FCCU Channel 40
ERR007638	FCCU: Unsuccessful decorated storage access may cause erroneous signaling of FCCU Channel NCF[39]
ERR010963	Flash: Memory accesses may be corrupted when flash is operating between 33MHz and 75MHz
ERR008647	FLASH: (SPC57xxM) Address Encode False Report (MCR[AEE] and possible FCCU channels)
ERR008004	FLASH: Array Integrity with Breakpoints enabled may skip addresses for certain RWSC and APC combinations
ERR007991	FLASH: Rapid Program or Erase Suspend fail status
ERR050119	FlexRay: Disabling of FlexRay Message Buffer during the STARTUP Protocol State takes longer than expected three Slots
ERR008770	FlexRAY: Missing TX frames on Channel B when in dual channel mode and Channel A is disabled
ERR010647	GTM: (A)TOM asynchronous update in SOMP mode not functional with CM1=0 and selected CMU clock is not equal to the system clock
ERR010646	GTM: (A)TOM output signal postponed by one period for the values CM0=1 and CM1>CM0 if CN0 is reset by the trigger of a preceding channel
ERR008122	GTM: (A)TOM's CCU1 event interrupt is not generated when CM1=0 or 1 and RST_CCU0=1
ERR006639	GTM: A compare match event does not clear WR_REQ when ATOM is in SOMC mode
ERR007085	GTM: A TIM timeout occurs when the TDU is re-enabled
ERR007528	GTM: Action not always calculated immediately by DPLL

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR007084	GTM: An active edge input, that is rejected by the DPLL trigger plausibility check, does not assert a Missing Trigger Interrupt
ERR011364	GTM: An unexpected restart of Signal Output Mode Serial one-shot cycle of Advanced Routing Unit-connected Timer Output Module happens when ATOM Counter Compare Unit 0 register is zero
ERR006409	GTM: ATOM Force Update does not activate a comparison when in SOMC mode
ERR010156	GTM: Behavior of PVT check of DPLL on direction change
ERR007848	GTM: Bit 0 of TIM edge counter register may not indicate the actual signal level
ERR006645	GTM: Clearing of DPLL PCM1/2 bits after the Missing Pulse Correction Values calculations delayed
ERR008688	GTM: Data lost in ATOM when CMU_CLKx is slower than ARU
ERR011215	GTM: Disabling the sub-position generators will not disable output pulses when operating the DPLL in continuous mode
ERR010159	GTM: Discontinuities in DPLL sub-increment in full scale
ERR010149	GTM: DPLL PMT calculation result is incorrect in backward direction
ERR007531	GTM: DPLL Position Minus Time result is not sent to the ARU
ERR010148	GTM: DPLL PWI interrupt request always active
ERR008430	GTM: DPLL sub-inc generation and action calculations are delayed
ERR010158	GTM: DPLL TAXI interrupt not deactivated for THMA = 0
ERR010157	GTM: DPLL wrong result when NUTE/NUSE modified in dedicated time window
ERR010648	GTM: DPLL_RAM1b.PSTC not updated after direction change and DPLL_CTRL_0 has been written
ERR010147	GTM: Erroneous data on CPU read access to an empty FIFO channel in ring buffer
ERR008689	GTM: F2A stream data are not deleted after stream disabling
ERR010989	GTM: Incomplete pulse correction in DPLL on direction change when DPLL_CTRL_1.SMC = 1
ERR006644	GTM: Incorrect duty cycle in TOM PCM mode
ERR006412	GTM: Incorrect Input Signal Characteristics when the TIM channel is in TBCM mode and ECNT is selected as the captured GPR0 value.
ERR006411	GTM: Incorrect Input Signal Characteristics when the TIM channel is in TIEM, TPWM, TIPM, TPIM or TGPS mode, when ECNT is selected as the captured GPRi value.
ERR011363	GTM: Incorrect setting of Digital PLL Lock status on missing trigger/state, trigger/state out of range or Digital PLL direction change event
ERR006643	GTM: Incorrect timestamp captured in CNTS when TIM operates in TPWM or TPIM modes if CMU_CLK is not equal to system clock
ERR010175	GTM: Input events may be missed for some time after the DPLL is enabled
ERR007847	GTM: MCS's CAT status may be incorrect
ERR007530	GTM: New DPLL Position Minus Time data not received
ERR010649	GTM: No DCGI interrupt after direction change and DPLL_CTRL_0 has been written
ERR011217	GTM: Numbers of Digital PLL real and virtual TRIGGER and STATE events for the last increment are updated with bits corresponding to CPU write operation instead of being updated with the previous number of recent events
ERR011216	GTM: Reset of the Digital PLL direction status by disabling the Digital PLL does not remove the unsolicited direction change in every case
ERR010764	GTM: Restoring of F2A read access to FIFO after GTM_HALT condition not functional
ERR010884	GTM: Short FIFO IRQ in level mode when DMA hysteresis is enabled and DMA direction is read

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR007190	GTM: Simultaneous Core and DPLL accesses to RAM Region 2 may lead to the DPLL reading erroneous data
ERR007087	GTM: The DPLL's Address Pointer Extension value is added to the Address Pointer when the Address Pointer Status bit is 0
ERR007191	GTM: The DPLL's SORI and TORI interrupts are not asserted
ERR007083	GTM: The DPLL's SORI, TORI, MTI, and MSI interrupts may not be asserted
ERR010176	GTM: The DPLL_STATUS[BWD1/2] flags are not reset when DPLL is disabled and enabled again
ERR006642	GTM: THVAL not available immediately after inactive trigger in DPLL
ERR007529	GTM: TIM overflow bit is not set and the signal level bit has inverse value when sent to ARU in some cases
ERR007086	GTM: TIM PWM and PIM modes may capture the wrong timestamp
ERR008439	GTM: TOM and ATOM CM0, CM1 and CLK_SRC register updates may not be triggered
ERR011191	GTM: TRIGGER signal will not cause direction change of the Digital PLL when TRIGGER hold minimum value is zero
ERR008429	GTM: Unexpected TIM CNTS register reset in TPWM OSM mode
ERR006640	GTM: Valid edge after Timeout event ignored by TIM
ERR007088	GTM: When ATOM is in SOMP mode the SR0/SR1 registers could be updated twice in one PWM period
ERR006410	GTM: Write to ATOM_CH_CTRL sets WRF if CCU0 compare match has already occurred, but CCU1 compare match is pending, in ATOM SOMC mode
ERR008438	GTM: Wrong signal level when TIM mode is changed from TBCM to any other mode
ERR008047	HSM: Nexus class 3 read/write accesses may be blocked by SMPU
ERR008951	I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse
ERR007109	I2C: In master receive mode, data remains latched in I2C data I/O register (IBDR) until new data is received
ERR007433	JTAGM: Nexus error bit is cleared by successful RWA
ERR008935	JTAGM: write accesses to registers must be 32-bit wide
ERR007398	LBIST: Partitions 2, 4, and 5 during offline self-test mode not functional
ERR007274	LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state
ERR008731	LINFlexD: Corruption of Received Rx data in UART mode
ERR008561	LINFlexD: Corruption of Tx data in LIN mode with DMA feature enabled
ERR007269	LINFlexD: Erroneous timeout interrupt could be generated by LIN in master mode
ERR007228	LINFlexD: Erroneous transmission in LIN master mode for payload greater than eight bytes
ERR008933	LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set
ERR008526	LINFlexD: LIN or UART state may be incorrectly indicated by LINSR[LINS] bitfield
ERR008573	LINFlexD: Pre-mature header/response timeout in LIN mode
ERR007297	LINFlexD: Response timeout values is loaded in LINOCR[OC2] field instead of LINOCR[OC1]
ERR008970	LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State
ERR007589	LINFlexD: Spurious timeout error when switching from UART to LIN mode or when resetting LINTCSR[MODE] bit in LIN mode

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR008602	LINFlexD: Tx through DMA can be re-triggered after abort in LIN/UART modes or can prematurely end on the event of bit error with LINCR2[IOBE] bit being set in LIN mode
ERR006350	LINFlexD: WLS feature cannot be used in buffered mode
ERR010557	MC_CGM: Auxiliary clock divider may get stuck if programmed to divide by 2 and a reset occurs during operation
ERR010555	MC_CGM: Auxiliary Clock Divider Update Status register bits not documented
ERR007103	MC_CGM: Incorrect cause for the latest clock source switch may be reported by the CGM if a safe mode request arrives when the system clock is the IRC
ERR010668	MC_CGM: Progressive Clock Switching (PCS) equations
ERR010559	MC_CGM: The device can become stuck in reset when reset occurs during a system clock switch to a PLL
ERR007211	MC_ME: Core register IAC8 is cleared during a mode change when the core is reset
ERR007911	MC_ME: Decorated accesses not supported to core local memories during reset
ERR003878	MC_ME: Flash low-power modes cannot be activated during MC_ME mode transitions.
ERR008117	MC_ME: Restrictions on enabling FlexRay in low power modes
ERR008145	MEMU: address registers in the uncorrectable error reporting tables can be written when the corresponding valid bit is not asserted
ERR007335	MEMU: ECC errors may be double reported when initiated by the Safety core to local memory of other cores
ERR010613	MEMU: False IMEM RAM fault flag set after reset or core enabling
ERR007126	MEMU: Instead of Byte 1 of MEMU CTRL Register, Byte 3 is currently protected
ERR008885	MEMU: System RAM ECC errors on accesses by some masters report to MEMU Peripheral RAM table
ERR007840	M_CAN: Change of operation mode during start of transmission
ERR009413	M_CAN: Data loss when the storage of a received frame is not complete before EOF
ERR050453	M_CAN: Debug message handling state machine not reset to Idle state when M_CAN_CCCR.INIT is set.
ERR050789	M_CAN: Dedicated Tx Buffers configured with the same Message ID are not transmitted in order of lowest buffer number first
ERR007842	M_CAN: Erroneous Interrupt flag after setting / resetting INIT during frame reception
ERR009070	M_CAN: FD frame abort may cause Protocol exception event and extended Bus Integration state
ERR008923	M_CAN: FD frame format not compliant to the new ISO/CD 11898-1: 2014-12-11
ERR008062	M_CAN: Frame transmission in DAR mode
ERR008904	M_CAN: Incorrect activation of MRAF interrupt
ERR007841	M_CAN: Incorrect frame transmission after recovery from Restricted Operation Mode
ERR009415	M_CAN: Message RAM / RAM arbiter may not respond in time
ERR007796	M_CAN: Message reception and transmission directly after detection of Protocol Exception Event
ERR011470	M_CAN: Message transmitted with wrong arbitration and control fields
ERR050016	M_CAN: Retransmission in DAR mode due to lost arbitration at the first two identifier bits
ERR008655	M_CAN: Setting the Configuration Change Enable (CCE) bit during a transmission scan can halt CAN tranmissions
ERR051044	M_CAN: Transmission order of the Tx Queue buffers with the same Message ID is impacted by the PUT index functionality

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR007498	M_CAN: Transmitted bit in control field is falsified when using extreme bit time configurations
ERR011458	M_CAN: Tx FIFO message sequence inversion
ERR011459	M_CAN: Unexpected High Priority Message (HPM) interrupt
ERR007532	M_TTCAN: Incorrect value of Reference Trigger Offset status for time slaves
ERR007538	M_TTCAN: Switch between CAN operating modes during transmission or reception may be ignored
ERR007932	NAR/SIPI: Part ID for NAR and Debug SIPI does not match the MIDR MINOR_MASK
ERR003970	NAR: Trace messages include a 6-bit Source Identification field instead of 4-bits
ERR011021	NAR: Trace to memory buffer pointers corrupted after break-point
ERR009796	PAD_RING: Increased coupling current during current injection of nearby analog pin
ERR007930	PAD_RING: Pin PM[11] is powered from the FlexRAY and Ethernet I/O power supply instead of the Main I/O supply
ERR010309	PASS: JTAG password does not enable Debug Interface Access during functional reset
ERR010196	PASS: JTAG password match bypasses flash read protection set by PASS.LOCK3[RLx] bits
ERR007904	PASS: Programming Group Lock bit (PGL) can be de-asserted by multiple masters writing the correct password sections to the CINn registers.
ERR007905	PIT: Accessing the PIT by peripheral interface may fail immediately after enabling the PIT peripheral clock
ERR008054	PIT: DMA request stays asserted when initiated by PIT trigger, until PIT is reset
ERR050130	PIT: Temporary incorrect value reported in LMTR64H register in lifetimer mode
ERR007772	PMC: a SWT_2 destructive reset may be forced if the 5V supply is over voltage for longer than 16ms
ERR008019	PMC: Escalation of LVD and HVD functional resets should be expected
ERR010660	PMC: EPR flags may be set after an exit from reset
ERR007365	PRAM: Reset may change RAM content
ERR008141	PSI5-S: Global mode transition interrupt does not work
ERR008075	PSI5-S: Unrecoverable messages are not flagged in the mailbox status register channel 0
ERR007996	PSI5: Incorrect SMC message decoding and timestamp generation in case of late last sensor message overlapping with next SYNC period pulse
ERR006992	PSI5: IS_DEBUG_FREEZE bit is not documented
ERR007234	PSI5: No transfer error generated for accesses within the unused range of the PSI5 peripheral window
ERR007134	RCCU: If any accesses to the I-MEM or D-MEM of the safety and checker core are performed while the cores are disabled, the cores will get out of lockstep when enabled
ERR010556	RGM: ESR0 long functional reset advances to the PHASE1[DEST] reset state, but sets the functional reset only flag
ERR009764	SARADC : DMA interface limitation depending on PBRIDGE/SARADC clock ratio
ERR010171	SARADC: Channel number may show incorrect value for two clock cycles
ERR005947	SARADC: ADC may miss a GTM trigger pulse if width of pulse is less than 1 AD Clk cycle
ERR007245	SARADC: CDATA fields for Left justified data is not documented
ERR007246	SARADC: First conversion after exit from stop mode may be corrupted
ERR010428	SARADC: Interrupted conversions are aborted, but may not be properly restored
ERR007222	SARADC: Minimum value of precharge must be greater than or equal to 2 ADC clock cycles
ERR007138	SARADC: Missed conversion after ABORT of the last channel of an injected chain

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR006197	SARADC: Positive DNL marginal at Cold
ERR007906	SARADC: The Data Overwritten flag bits in the SARADC may not be valid
ERR008126	SARADC_B: Some test channels are no longer available
ERR007362	SDADC: Additional DMA request generated after single read access
ERR010376	SDADC: Additional DMA request generated with slow SDADC SD_CLK clock
ERR008039	SDADC: digital filter and FIFO not disabled when MCR[EN] is cleared
ERR008314	SDADC: Double trigger is generated for conversion when SW trigger is connected to SDADC own HW trigger input
ERR008225	SDADC: FIFO Flush Reset command requires clearing the Data FIFO Full Flag
ERR010378	SDADC: Incorrect data provided when FIFO is disabled and FIFO overwrite is enabled
ERR006906	SDADC: Invalid conversion data when output settling delay value is less than 23
ERR008631	SDADC: low threshold watchdog cannot be used with signed data
ERR005749	SDADC: New conversion data is discarded if the overflow (DFORF) status bit is set
ERR007356	SDADC: The SDADC FIFO does not function correctly when FIFO overwrite option is used
ERR007185	SDADC: Watchdog Crossover event missed if PBRIDGEx_CLK less than SD_CLK
ERR007204	SENT: Number of Expected Edges Error status flag spuriously set when operating with Option 1 of the Successive Calibration Check method
ERR008082	SENT: A message overflow can lead to a loss of frames combined with NUM_EDGES_ERR being set
ERR007203	SENT: In debug mode SENT message data registers appear to lose contents
ERR007425	SENT: Unexpected NUM_EDGES_ERR error in certain conditions when message has a pause pulse
ERR010561	SIPI: Incorrect SIPI_ERR[TOEx] register bit description
ERR007788	SIUL2: A transfer error is not generated for 8-bit accesses to non-existent MSCRs
ERR009780	SIUL2: Open-drain or open-source configured outputs may briefly drive high or low during transitions
ERR007791	SIUL2: Transfer error not generated if reserved addresses within the range of SIUL BASE + 0x100 to 0x23F are accessed
ERR008605	SMPU: debug SIPI accesses may be blocked by SMPU
ERR007781	SPC5777M: Current injection causes leakage path across the DSPI and LFAST LVDS pins
ERR009658	SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event
ERR006850	SSCM: Nexus enable required for mode changes when a debugger is attached
ERR007424	SSCM: The SSCM_STATUS register only contains two NXEN status bits
ERR007339	STCU2: STCU2 fault injected by FCCU is self clearing
ERR010560	STCU: ESR0 resets may not properly abort online self-tests
ERR010667	STCU: LBIST PRPG start value is 32 bits
ERR010607	STCU: STCU watchdog timer timeouts due to low voltage supply excursions
ERR009322	TDM: Erase of TDR flash block may be blocked by the TDM
ERR007948	TDM: Erase protection not enabled by reset
ERR008412	TSENSE: Temperature sensor status flags are incorrect
ERR007801	WKPU: functional NMI filter enable trigger FCCU fault monitor channel #47
ERR008738	XBAR: Masters on peripheral shell bus concentrator may stall or fetch data incorrectly
ERR006994	XBIC: XBIC may trigger false FCCU alarm

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR008310	XBIC: Crossbar Integrity Checker may miss recording information from an initial fault event in the case of back-to-back faults
ERR008730	XBIC: XBIC may store incorrect fault information when a fault occurs
ERR004136	XOSC and IRCOSC: Bus access errors are generated in only half of non-implemented address space of XOSC and IRCOSC, and the other half of address space is mirrored
ERR007947	XOSC: Incorrect external oscillator status flag after CMU event clear
ERR009709	XOSC: Oscillator jitter may be impacted by noise on VDD_HV_OSC and may cause PLL jitter
ERR007862	ZIPWIRE: Incorrect Device ID returned from SIPI
ERR010436	ZipWire: SIPI can have only one initiator with one outstanding write frame at time

Table 2. Revision History

Revision	Changes
August 2015	Revision 5 The following errata were removed. e7196: Moved to the device Reference Manual. The following errata were added. e9322, e9415, e9584, e9070 The following errata were revised. e7274, e8951 --- Previous revision history is shown below. --- Revision 1.0 Release date: 2014 November 05 The following errata were removed. e7351: This errata was replaced with e8028. e7652: This erratum was cancelled. This operation was already described in the device documentation. e7355: Data sheet has been updated with the new specification. e7363: Data sheet has been updated with the new specification. The following errata were added. e8438, e8439, e8225, e8229, e8526, e8429, e8430, e8343, e8561, e8605, e8602, e8075, e8573, e8145, e8141, e8126, e8655, e8028 The following errata were revised. e8122, e7589, e7365, e8042, e7842, e7841, e6350, e7269 --- Revision 2.0

Table continues on the next page...

Table 2. Revision History (continued)

Revision	Changes
	Release Date: December 19,21014 The following errata were removed. e8028: Replaced by e7351. The following errata were added. e8310, e8314, e8904, e8647, e7351, e8689, e8688, e8412, e8885, e8738, e8731, e8730 The following errata were revised. e7841, e8429 --- Revision 3.0 Release date: 03 February 2015 The following errata were added. e8970, e8770, e8547, e8933, e8935, e8923 The following errata were revised. e8688, e8412 --- Revision 4.0 Release date: 08 April 2015 The following errata were added. e8951, e9060 The following errata were revised. e8412: Wording updated to provide additional information. e8904
January 2016	The following erratum was removed. • ERR007356: This is now documented in the device Reference Manual. The following errata were added. • ERR009413 • ERR009656 • ERR009658 • ERR009664 • ERR009780 • ERR009783 • ERR009796 The following errata were revised. • ERR008412 • ERR008933: Minor corrections in the grammar. • ERR009415: Description updated to allow operation for up to 500 k bits/second.
June 2016	Revision 7.0 The following erratum was removed. • ERR008056: This is now documented in the device Reference Manual.

Table continues on the next page...

Table 2. Revision History (continued)

Revision	Changes
	The following errata were added. • ERR007356 • ERR007638 • ERR008631 • ERR009709 • ERR010147 • ERR010148 • ERR010149 • ERR010156 • ERR010157 • ERR010158 • ERR010159 • ERR010171 • ERR010175 • ERR010176 • ERR010196 • ERR010269 • ERR010309 • ERR010376 • ERR010385 The following erratum was revised. • ERR009796: Additional pin groups were added.
June 2016, version 8	Revision 8.0 The following erratum was removed. • ERR010269: Replaced by ERR010428 The following errata were added. • ERR010378 • ERR010428 The following erratum was revised. • ERR007362
December 2016	Revision 9b The following errata were added. • ERR010452 • ERR010542 • ERR010555 • ERR010556 • ERR010557 • ERR010559 • ERR010560 • ERR010561 • ERR010607 • ERR010613 • ERR010646 • ERR010647 • ERR010648 • ERR010649 • ERR010660 • ERR010663

Table continues on the next page...

Table 2. Revision History (continued)

Revision	Changes
	• ERR010667 • ERR010668 The following errata were revised. • ERR007305: Corrected 1 register (changed PMC4 to PMC3) • ERR009780: Fixed typographical error (removed extra s)
August 2017	Revision 10 The following errata were added. • ERR010436 • ERR010764 • ERR010842 • ERR010884 • ERR010900 • ERR010989 The following erratum was revised. • ERR010560: Additional condition added and an additional FCCU flag may be set.
May 2018	Revision 11a The following errata were added. • ERR009764 • ERR011021 • ERR011191 • ERR011215 • ERR011216 • ERR011217 The following errata were revised. • ERR010436: Errata description was clarified. • ERR010660: Clarification added for the workaround. • ERR010900: Errata description was clarified.
October 2018	Revision 12 The following errata were added. • ERR011363 • ERR011364 • ERR011407 • ERR011458 • ERR011459 • ERR011470 • ERR050016
June 2019	Revision 13 The following erratum was removed. • ERR011407: Replaced by e50159 The following errata were added. • ERR050119 • ERR050159 The following errata were revised.

Table continues on the next page...

Table 2. Revision History (continued)

Revision	Changes
	• ERR007869 • ERR008731
December 2019	Revision 14 Revision history is now shown in reverse chronological order (newest first). Summary table now shows the complete erratum number. The following errata were added. • ERR050090 • ERR050130
August 2021	The following errata were added. • ERR010963 • ERR050453 • ERR050789 • ERR051044 The following errata were revised. • ERR007498 • ERR007538 • ERR050119

ERR010842: CMU: False upper frequency threshold alarm (FHH) signaled to Fault Collection and Control Unit (FCCU)

Description: A Clock Monitor Unit (CMU) can be configured to monitor the frequency of a clock signal (CLKMN1) and generate an event if CLKMN1 is greater than a high frequency boundary defined by the CMU High Frequency Reference Register (CMU_HFREFR). If the monitored CLKMN1 clock is above 300MHz, then there is the potential that the CMU_ISR[FHHI] may be falsely set but the frequency is in fact correct. The false CMU_ISR[FHHI] flag may they trigger a false alarm to the FCCU, if so configured. If the measured clock frequency is actually above the upper threshold or below the lower threshold, a proper alarm is always correctly triggered.

Workaround: Prevent false FHH events by disabling the CMU by clearing the CMU_CSR[CME]. The PLL0-PHI frequency integrity can be verified by using other CMU modules that monitor the divided versions of PLL0-PHI. Please refer to the Reference Manual for the complete list of available CMU monitors.

The CMU may still be enabled to utilize the other clock monitoring features (such as FLLI and OLRI) and in this case the FHHI false flag can be ignored. If FHHI is set, the FHHI can be cleared and re-read to determine a real permanent clock fault compared to just a transient false flag. A false FHHI flag being set has no adverse effects on the integrity of the FLLI or OLRI functions or on capturing true FHHI events. Disable the FCCU FHHI trigger event to prevent the FCCU from inadvertently reacting to a false FHHI event.

ERR050159: CMU: Monitoring signal for sudden loss of clock from CMU does not signal the Fault Collection and Control Unit

Description: The Clock Monitor Unit (CMU) detects when the frequency of a monitored clock drops below a programmed threshold and asserts the Frequency Less than Low Threshold (FLL) signal if this occurs. The FLL signal is routed to the Fault Collection and Control Unit (FCCU) providing a mechanism to react to the clock fault but due to the monitoring implementation the FLL signal will not be triggered when the monitored clock suddenly stops.

Workaround: Each of the CMU monitored clocks has been analysed for the system level failure effect upon loss of the monitored clock and the safety mechanisms present to detect this. From this analysis it is concluded that loss of all the monitored clocks can be detected by other existing safety mechanisms in the system.

Further, since the CMU monitored clocks are derived from one of the system clock sources (XOSC_CLK, PLL0_PHI_CLK, PLL1_PHI_CLK) if the loss of the monitored clock is caused by the loss of the source clock then this will be detected and reported to FCCU by existing source clock loss detection mechanisms.

ERR007116: CRC: AutoSAR 4.0 8-bit CRC8 0x2F is not supported in hardware

Description: The Cyclic Redundancy Check (CRC) module does not implement the 8-bit CRC-8-H2F required to support the Autosar 4.0 specification. The CRC-8-H2F uses a polynomial generator seed of 0x2F and an equation of $x^5 + x^3 + x^2+x+1$.

Workaround: Do not set the Polynomial selection to 0b11 in the CRC Configuration register (CRC_CFG). The 8-bit CRC-8-H2F function must be written in software to support AuroSAR 4.0.

ERR007824: DCI: Avoid asserting system reset when switching JTAG operating modes

Description: Assertion of system reset during the transition of the debug pin operating mode, either from JTAG pin mode to the LVDS Fast Asynchronous Serial Transmission (LFAST) pin mode, or from LFAST pin mode to JTAG pin mode, could result in a loss of synchronization with the debugger or a reset of the debug system.

Workaround: Tools should not assert system reset while the Debug and Calibration Interface (DCI) is switching the pin operating mode from JTAG to LFAST, or from LFAST to JTAG. If a system reset occurs due to any other conditions, the tool may lose communication with the microcontroller. If this occurs, the tool should reset the JTAG interface by toggling JCOMP (DEBUG_RXN) low (ground) while holding the TDO (DEBUG_RXP) pin either high or low. This forces the interface operation back to JTAG operating mode. This requires that the Enable Escape mode feature be enabled in the DCI Control Register (DCI_CR[EN_ESC_MODE] = 1).

ERR008343: DCI: EVTO[1:0] outputs remain stuck low if asserted while the system clock source is the IRC

Description: While the system clock source is the Internal Resistor Capacitor oscillator (IRC), if the Event Output pins (EVTO[1:0]) are asserted by the Debug and Calibration Interface (DCI)), they will never negate. This applies to all functions on EVTO[1:0] except the timer functions controlled

by the EVTO Output selection (EOS0/EOS1) fields of the Generic Timer Module debug interface (GTMDI) Development Control (DC) register. This includes when the EVTO pins are being used by the Development Trigger Semaphore (DTS) module.

Workaround: For proper operation of EVTO[1:0] outputs, program clocks to select a system clock source other than the IRC before enabling and using EVTO[1:0].

ERR007111: DMA: DMA does not work properly with M_CAN modules

Description: The Modular Controller Area Network (M_CAN) Direct Memory Access (DMA) acknowledge may assert prior to the actual data being read from the buffers. The DMA request is only required for Debug on CAN to read 3 debug messages in Receive buffers #61, #62, and #63. For more information, see the Debug on CAN support section in the M_CAN chapter of the Reference Manual.

This early acknowledgement event causes the message buffers to be unlocked, and potentially overwritten by another incoming message to the same buffers.

Workaround: Do not allow the transmitter to send adjacent messages after the 3rd in-order transmission to these buffers. Use the recommended Debug over CAN sequence.

ERR007113: DMA: GTM accesses via DMA may fail

Description: The Generic Timer Module (GTM) Direct Memory Access (DMA) flags can be cleared prior to the completion of data read or data write. On DMA transfers where the GTM is the data source, the GTM will receive a transfer completion indication before the data is actually read from the GTM. This may lead to overwritten data, without any error indication. On DMA transfers where the GTM is the data destination, the GTM will receive a transfer completion indication before the data is actually written into the GTM. The GTM may use incorrect data.

The following GTM channels can be affected: Sensor Pattern Evaluation Module (SPE), Parameter Storage Module (PSM), Multi-Channel Sequencer (MCS), Timer Output Module (TOM), Timer Input Module (TIM), and Advanced Routing Unit (ARU) controlled TOM (ATOM). Additionally, the TIM channel's GPRz data overflow (GPROFL) bit in the TIM[i] Channel[x] IRQ Notification Register (TIM[i]_CH[x]_IRQ_NOTIFY) will not be set if a DMA overrun event occurs.

Workaround: Use interrupts to control data flow to and from the GTM.

ERR010663: DS/RM: IRC trimming and LVD/HVD clarifications

Description: The MPC5777M Reference Manual (RM, revision 4.2) indicates that the user programmable frequency trimming of the IRCOSC is controlled by IRCOSC_CTL[USER_TRIM] which is based on the dFvar_SW value as defined in the MPC5777M Data Sheet (DS, dated 06/2016). However, the IRCOSC_CTL[USER_TRIM] should be based on dfTRIM instead of dFvar_SW as defined in the "Internal RC Oscillator electrical specifications" table in the Data Sheet.

Clarify the wording in the Data Sheet of table note #6 of the "Device operating conditions" table to change from "In the LVD/HVD disabled case, it is necessary for the system to be within a higher voltage range during destructive reset events." to "In the LVD/HVD disabled case, it is necessary for the system to be within a valid voltage range to correctly recover from power-on, destructive, or ESR0 long reset events."

Workaround: Update Data Sheet and Reference Manual with these clarifications.

ERR007351: DSMC: Lockstep-error notification is disabled

Description: The Decorated Storage Memory Controller (DSMC) associated with Safety-core local memory is included in the sphere of replication for lockstep checking.

The checking of the DSMC outputs with its replicated checker DSMC by an RCCU (Redundancy Control Checker Unit) has been disabled.

The disabling only affects accesses of masters other than the Safety-core through the Safety-core backdoor (slave port s2) to the Safety-core local memory (read and write).

Accesses of the Safety-core to its own local memory are not affected by this errata.

Failures occurring during a backdoor write access are still covered by other RCCU mechanisms, signaling to either Fault Collection and Control Unit (FCCU) Fault Channel 10 or 11 (depending on where the failure emerges).

Failures occurring during a backdoor read access are covered with end-to-end ECC protection (which has slightly less diagnostic coverage than lockstep checking) or are covered by another RCCU, depending on where the failure occurs.

Workaround: The customer safety assessment in the system Fault Mode Effects and Diagnostic Analysis (FMEDA) should take in to consideration that the end-to-end Error Correcting Code (ECC) is used instead of the depending on a lock-step read access mechanism when the Safety-core local memory is read via the backdoor by other cores.

ERR050090: DSPI/SPI: Incorrect data may be transmitted in slave mode

Description: If the Serial Peripheral Interface (SPI or the Deserial/Serial Peripheral Interface) is operating in slave mode, incorrect or stale data may be transmitted in next transaction without underflow interrupt generation if the set up time of the Peripheral Chip Select (PCS) to the SPI Serial Clock (SCLK) is short and the transmit FIFO may become empty after one transaction.

This can occur if the PCS to SCK is less than:

4 X IPG_CLOCK_PERIOD + 4 X DSPI_CLOCK_PERIOD + 0.5 x SCK_CLOCK_PERIOD

Where:

IPG_CLOCK is the internal bus clock (“system” clock)

DSPI_CLOCK is the protocol clock.

SCK_CLOCK is the Line-Side Serial Communication Clock.

Workaround: When operating in slave mode, software must ensure that the time interval between PCS assertion to start of SCK Clock is greater than 4 X IPG_CLOCK_PERIOD + 4 X DSPI_CLOCK_PERIOD + 0.5 x SCK_CLOCK_PERIOD.

To meet this requirement, the Master SPI can either lengthen the PCS to SCK assertion time or decrease the frequency of the communication interface, or both.

ERR009783: DSPI: 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).

ERR009664: DSPI: 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).

ERR009656: 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).

ERR007115: DSPI: Mixing 16 and 32 bits frame size in XSPI Mode can cause incorrect data to be transmitted

Description: The Deserial Serial Peripheral Interface (DSPI) features an Extended SPI mode (XSPI) supporting frames of up to 32 bits.

When the XSPI Mode is enabled, transferring a mixture of frames having a size up to 16 bits and those having size above 16 bits can cause an incorrect data transmission to occur. This happens when the First In/First Out (FIFO) queue read pointers roll-over and a frame needs to be extracted from both the bottom of the FIFO and the top of the FIFO when the Frame Size is greater than 16 bits.

Workaround: Even number of Transmit FIFO Register (TXFR) registers:

Do not mix frames that have data sizes of less than 16 bits with those having a size more than 16 bits in XSPI Mode.

Odd number of TXFR registers :

Do not mix frames that have data sizes of less than 16 bits with those having a size more than 16 bits in XSPI Mode.

If the frame size is greater than 16, initially send a dummy frame (a frame with no chip select, but containing data) of less than or equal to 16 bits. Continue sending a dummy frame after each (number of TXFR Registers – 1) / 2 frames.

ERR007352: DSPI: reserved bits in slave CTAR are writable

Description: When the Deserial/Serial Peripheral Interface (DSPI) module is operating in slave mode (the Master [MSTR] bit of the DSPI Module Configuration Register [DSPIx_MCR] is cleared), bits 10 to 31 (31 = least significant bit) of the Clock and Transfer Attributes Registers (DSPIx_CTARx) should be read only (and always read 0). However, these bits are writable, but setting any of these bits to a 1 does not change the operation of the module.

Workaround: There are two possible workarounds.

Workaround 1: Always write zeros to the reserved bits of the DSPIx_CTARn_SLAVE (when operating in slave mode).

Workaround 2: Mask the reserved bits of DSPIx_CTARn_SLAVE when reading the register in slave mode.

ERR007299: DSPI: SOUT pins are not automatically configured as SOUT when Slave Select is asserted in Slave Mode

Description: The Deserial Serial Peripheral Interface (DSPI) Serial Data Out (SOUT) pins are not automatically configured as SOUT when the Slave Select (SS) signal is asserted. Instead the pins must be configured as SOUT using the Source Signal Select (SSS) field of the pertinent Multiplexed Signal Configuration Register (MSCR) in the System Integration Unit Lite 2 (SIUL2) module.

Workaround: In DSPI slave mode operation, the SOUT function must be selected in the MSCR. If the device is the only slave device then no functional issues will occur.

If the device is configured in DSPI slave mode and it is not the only slave then an external multiplexing scheme can be used based upon the DSPI Slave Select Signals.

ERR010542: DSPI: Transmit, Command, and Receive FIFO fill flags in status register is not cleared when DMA is improperly configured

Description: The Deserial/Serial Peripheral Interface Transmit, Receive, and Command First In/First Out (FIFO) buffers can request additional information to be transferred via the Direct Memory Access (DMA) module when either the Transmit, Receive, or Command FIFO Fill/Drain Flags are set in the DSPI Status Register (SR[TFFF/RFDF/CMDFFF]). However, the Command/ Transmit Fill Flag only indicates that at least 1 location in the FIFO is available to be written. It does not indicate that the FIFO is empty. Similarly, Receive FIFO fill flag only indicates at least 1 location of the FIFO is available to be read. It does not indicate that the FIFO is full. If the DMA is configured to transfer more than 1 FIFO location size of data, the FIFO Fill Flags may not be properly cleared indicating that the FIFO is not full even when the FIFO is actually full (for Transmit and Command FIFO) and not empty when the FIFO is actually empty (for Receive FIFO).

Workaround: Properly configure the DMA to fill the Transmit, Receive, and Command FIFOs only one FIFO location, in other words, up to 2 bytes, at a time to each of the FIFOs.

Use the DMA loop to transfer more data if needed.

ERR008547: e200z425: Additional cycles required for Load/Store accesses to guarded/ precise space

Description: If an access by the Load/Store unit is to a space that is guarded/precise then it will be subject to additional cycles for completion because there is no pipeline bypass for the Bus Interface Unit. These spaces and accesses are characterized by any one of the following conditions:

- The Memory Protection Unit (MPU) Guarded bit is set for the address space
- The access is a store with the Store Buffer disabled
- The access is a Decorated Load or Store
- The access is a Bypass Store
- The access is an Atomic Load or Store

On this device, the peripheral address space, by default, is set to be guarded/precise space.

Each scenario below will cause a 2 cycle increase:

- Idle to guarded/precise load
- Idle to guarded/precise store
- Store to guarded/precise load
- Store to guarded/precise store

The following scenarios may or may not add extra cycles depending on wait states and back to back requests:

- Load to guarded/precise store
- Load to guarded/precise load

Workaround: Expect additional cycles when guarded/precise accesses occur.

ERR010385: e200z4: Incorrect branch displacement at 16K memory boundaries

Description: The branch target address will be incorrectly calculated in the e200z4 core under the following conditions (all conditions must be matched):

- The first full instruction in a 16 Kbyte section/page of code is a 32-bit long branch with a branch displacement value with the lower 14 bits of the displacement exactly 0x3FFE
- And this branch instruction is located at byte offset 0x0002 in the section/page
- And the preceding instruction is a 32-bit length instruction which is misaligned across the 16K boundary
- And both instructions are dual-issued

Under these conditions, the branch target address will be too small by 32Kbytes.

Workaround: After software is compiled and linked, code should be checked to ensure that there are no branch instructions located at address 0x2 of any 16K memory boundary with the lower 14 bits of the displacement equal to 0x3FFE if preceded a 32-bit instruction that crosses the 16K memory boundary. If this sequence occurs, add a NOP instruction or otherwise force a change to the instruction addresses to remove the condition.

A tool is available on nxp.com that can be run to examine code for this condition, search for branch_displacement_erratum_10385_checker.

ERR007210: e200z710: Nexus timestamps are not available

Description: The optional timestamps on the Nexus trace messages from the e200z710 cores are not implemented and cannot be enabled.

Workaround: Timestamps of Nexus trace messages from the e200z425 core are available. Tools can correlate the e200z425 timestamps to the e200z710 messages for a rough approximation of time.

ERR007259: e200zx: ICNT and branch history information may be incorrect following a nexus overflow

Description: If an internal Nexus message queue over-flow occurs when the e200zx core is running in branch history mode (Branch Method bit [BTM] in the Development Control register 1 [DC1] is set [1]), the instruction Count (ICNT) and branch history (HIST) information in the first program trace message following the Program Correlation message caused by an over-flow of the internal trace buffers, will contain incorrect ICNT and HIST information.

This can also occur following an overflow of the internal Nexus message queues in the traditional branch mode (BTM in the DC1 is cleared [0]). Traditional branch mode Nexus messages do not include HIST information, since all branches generate a trace message.

Workaround: There are two methods for dealing with this situation.

1) Avoid overflows of the Nexus internal FIFOs by reducing the amount of trace data being generated by limiting the range of the trace area by utilizing watchpoint enabled trace windows or by disabling unneeded trace information, or by utilizing the stall feature of the cores.

2) After receiving an overflow ERROR message in Branch History mode, the ICNT and HIST information from the first Program Trace Synchronization message and the next Program Trace message with a relative address should be discarded. The address information is correct, however, the ICNT and previous branch history are not correct. All subsequent messages will be correct.

In traditional branch mode, the ICNT information should be discarded from the Program Trace Sync message and the next direct branch message.

ERR007305: e200zx: JTAG reads of the Performance Monitor Counter registers are not reliable

Description: Reads of the Performance Monitor Counter (PMC0, PMC1, PMC2, and PMC3) registers through the IEEE 1149.1 or IEEE 1149.7 (JTAG) interfaces may return occasional corrupted values.

Workaround: To ensure proper performance monitor counter data at all times, software can be modified to periodically read the PMCx values and store them into memory. JTAG accesses could then be used to read the latest values from memory using Nexus Read/Write Access or the tool could enable Nexus data trace for the stored locations for the information to be transmitted through the Nexus Trace port.

ERR007408: EBI: Minimum input setup time is 7 ns

Description: For the External bus Interface (EBI), the Input Signal Valid to the clock out signal (CLKOUT) positive edge (Setup Time) specification of 6 ns is not met over the full operating conditions of the MCU.

Workaround: Expect that the setup time for the EBI to be 7 ns minimum. Use an external memory whose read data drive time is fast enough to meet desired CLKOUT frequency and meet the MCU Input Setup time, OR slow down the CLKOUT frequency such that timing can be met.

ERR006967: eDMA: Possible misbehavior of a preempted channel when using continuous link mode

Description: When using Direct Memory Access (DMA) continuous link mode Control Register Continuous Link Mode (DMA_CR[CLM]) = 1) with a high priority channel linking to itself, if the high priority channel preempts a lower priority channel on the cycle before its last read/write sequence, the counters for the preempted channel (the lower priority channel) are corrupted. When the preempted channel is restored, it continues to transfer data past its “done” point (that is the byte transfer counter wraps past zero and it transfers more data than indicated by the byte transfer count (NBYTES)) instead of performing a single read/write sequence and retiring.

The preempting channel (the higher priority channel) will execute as expected.

Workaround: Disable continuous link mode (DMA_CR[CLM]=0) if a high priority channel is using minor loop channel linking to itself and preemption is enabled. The second activation of the preempting channel will experience the normal startup latency (one read/write sequence + startup) instead of the shortened latency (startup only) provided by continuous link mode.

ERR010452: eDMA: When master ID replication is enabled, the stored ID and privilege level will change if read by another master.

Description: When master ID replication is enabled (DMA_DCHMIDn[EMI]=1), the DMA_DCHMIDn[PAL] and DMA_DCHMIDn[MID] fields should reflect the privilege level and master ID respectively of the master that wrote the DMA_TCDn_CSR[DONE:START] byte. However, if a different master reads the DMA_TCDn_CSR[DONE:START] byte, the master ID and privilege level will incorrectly change to this read access.

Workaround: Only allow the intended master ID replication core to access the DMA_TCDn_CSR[DONE:START] byte.

ERR007337: FCCU: Address feedback error disabled for the Safety Core local RAMS and Cache

Description: The additional, dedicated measures against addressing and control faults (such as address/control feedback) have been disabled. This applies to the following memories of the Safety main core 0:

Memory type	Name
Instruction SRAM	I-MEM

Data SRAM	D-MEM
Instruction CACHE	I-CACHE
Data CACHE	D-CACHE
Instruction TAG	I-TAG
Data TAG	D-TAG

This, in turn, has disabled the Fault Collection and Control Unit (FCCU) Failure Input Channel 27 which Indicates an addressing error (address-feedback) inside the above memories.

Workaround: Checks for address errors should be performed using the Memory Built-In Self Test features of the device. FCCU input channel 27 will not be asserted.

ERR008034: FCCU: Channel 40 fault (COMP_XBIC_DSMC_Monitor) may be reported if a CPU is reset upon a MC_ME mode change

Description: The Fault Collection and Control Unit (FCCU) channel 40 may report a fault during or after completion of a Mode Entry module (MC_ME) mode change where a CPU is reset in the mode change (setting MC_ME_CADDRn[RMC]=1). FCCU channel 40 (COMP_XBIC_DSMC_Monitor) indicates an addressing or control fault resulting in corrupted transaction through the computational XBAR or interference by system RAM Decorated Storage Memory Controller (DSMC).

Workaround: Disable the computational XBAR slave port 4 parking control (XBAR_0_CRS4[PCTL]=10) prior to a mode change where a CPU will be reset, or expect and clear a FCCU channel 40 fault following the mode change if it occurs. Fault is cleared by clearing the FCCU channel status, FCCU_RF_Sn[RFSm].

ERR008229: FCCU: Enabling the programmable glitch filter on EIN may cause a destructive reset

Description: The Fault Collection and Control Unit (FCCU) external error input (EIN) can be filtered by the FCCU on-chip programmable glitch filter. However, the EIN is routed directly to the FCCU Output Supervision Unit (FOSU) without passing through the glitch filter. Additionally, when the glitch filter is programmed using the FCCU Control Register (FCCU_CTRL) FILTER_WIDTH field, the effective duration may vary depending on time at which the EIN signal arrives, which can cause a missed or false EIN signal. As a result, it is possible for the FOSU to recognize an event on EIN that is ignored by the FCCU resulting in a destructive reset when the FOSU timeout period expires.

Workaround: Bypass the FCCU on-chip glitch filter by writing a 1 to the FCCU Control Register FILTER_BYPASS field (FCCU_CTRL[FILTER_BYPASS]) and use an external glitch filter to ensure that only valid external error events are recognized by the FCCU. The impact on safety can be avoided by using a feedback based signaling on Error In (EIN), wherein the EIN signal is kept asserted until the MCU acknowledges the event to the source.

ERR008042: FCCU: EOUT signals are active, even when error out signaling is disabled

Description: Every time the Fault Collection and Control Unit (FCCU) moves into fault state caused by an input fault for which the error out reaction is disabled (FCCU_EOUT_SIG_ENn[EOUTENx]=0), the Error Out 1 and 2 (EOUT[0] and EOUT[1]) will become active for a duration of 250 us plus the value programmed into the FCCU Delta Time register (FCCU_DELTA_T[DELTA_T]). EOUT is not affected if the FCCU moves into the alarm state that generates an interrupt (IRQ), if the Fault is cleared before the alarm timeout.

This erratum does not affect the outputs of other pins (for example, for communication modules like CAN/Flexray). Only the EOUT signal is impacted.

Workaround: There are three possible workarounds:

1) Enable EOUT signaling for all enabled error sources.

2) In case external device (which evaluates EOUT) can communicate with the MCU, the following procedure could be used:

a) Program any duration of EOUT as per application needs (FCCU_DELTA_T[DELTA_T])

b) For faults requiring error out reaction, the software shall validate EOUT via separate communication channel (like I2C) while EOUT is asserted.

c) External device shall implement a timeout mechanism to monitor EOUT validation by separate channel.

d) Following scenarios shall be considered as valid EOUT reactions:

d1) Validation is performed while EOUT is asserted

d2) Timeout occurs but no validation and EOUT is still asserted.

3) In case external device (which evaluates EOUT) cannot communicate with the MCU, following procedure could be used:

a) Program the error out duration to a duration x (FCCU_DELTA_T[DELTA_T]).

b) For faults requiring error out reaction, clear the fault after the pin has continued to be asserted for a longer duration (for example 2*duration x). This will artificially create a long pulse on EOUT.

c) For faults which do not require error out reaction, clear the fault within duration x. This will artificially create a short pulse on EOUT.

d) External device should ignore short pulse of duration x while recognizing longer pulses as valid reaction.

e) While clearing the fault, the associated software shall check the pending faults.

ERR007099: FCCU: Error pin signal length is not extended when the next enabled fault, with its alarm timeout disabled, occurs

Description: In the Fault Collection and Control Unit (FCCU), when the following conditions are met:

- two faults occur
- the second fault arrises with a delay (T_delay) from the first error
- the second fault has its alarm timeout disabled
- T_delay is lower than the FCCU error pin minimum active time (T_min, defined in the Delta T register (FCCU_DELTA_T))

Then the error output signal is not extended and its duration is only T_min, if the faults are cleared before the timer expires.

The expected behavior is to have the error output signal duration of T_min + T_delay, if the faults are cleared before the timer expires.

Workaround: Take into account that the error out signal duration will only be T_min, if the faults are cleared before the timer expires.

The timer count is meaningful only when the Error pin is driven low, which can be checked by reading the pin status FCCU_STAT[ESTAT].

ERR010900: FCCU: False indication of a fault state for a single safe clock period can be generated on the error output pin

Description: The error out pin from the Fault Collection and Control Unit (FCCU) may pulse to a logic low (0b0) when the following conditions are fulfilled:

- software changes the error out protocol from a toggling protocol to a not-toggling protocol, and programs the FCCU_CFG.FCCU_SET_AFTER_RESET bit to 0b1
- software switches the Fault Collection and Control Unit (FCCU) state machine from CONFIG to NORMAL state

The duration of the glitch is equal to a single clock period of the Internal RC oscillator and there is a 50% of probability of the pulse occurring.

Workaround: Split the configuration of the FCCU in 2 phases.

During the first phase, software should do the following:

1) move the FCCU to the CONFIG state

2) configure the FCCU including the error out protocol, but without setting the FCCU_CFG.FCCU_SET_AFTER_RESET flag to 0b1 (leave as 0b0)

3) exits to the NORMAL state

During the second phase, software should do the following:

4) move the FCCU to the CONFIG state

5) set the FCCU_CFG.FCCU_SET_AFTER_RESET flag to 0b1

6) exit to the NORMAL state

Note: The default (after reset) error out protocol is the Dual Rail. Since this is a toggling protocol, the software must execute the above steps each time the user wants to switch to a non-toggling error out protocol.

ERR007223: FCCU: FCCU_IRQ_EN register is writeable in all operating modes

Description: In the Fault Collection and Control Unit (FCCU), the FCCU Interrupt Enable register (FCCU_IRQ_EN) is writable (and readable) in all states (NORMAL, CONFIG, FAULT and ALARM) while in some revisions of the documentation it is stated “This register is writable only in the CONFIG state”.

Workaround: Take into account that FCCU_IRQ_EN register can be written in all the states of the FCCU.

Please ignore the following text in the description of FCCU_IRQ_EN register if you find it in the documentation revision in hand:

"This register is writable only in the CONFIG state."

ERR007869: FCCU: FOSU monitoring of a fault is blocked for second or later occurrence of the same fault

Description: The Fault Collection and Control Unit (FCCU) Output Supervision Unit (FOSU) will not monitor the FCCU for the second or later occurrence of a given fault in the following cases:

1. Reset is programmed as the only reaction for the fault.

2. Assertion of the fault coincides with the long/short functional reset reaction to a fault previously asserted.

Workaround: Enable either Alarm state (NCFTOEx) or at least one other type of Fault-state reaction: Non-maskable Interrupt (NMI) or error out (EOUT) signaling reaction for the faults that have a reset reaction enabled only. Restrictions of combining reset reaction with additional reactions may be written in the chip specific sub-section of the FCCU chapter.

ERR009584: FCCU: MPC57xxM – FOSU may assert destructive reset when a hardware recoverable fault of width less than one safe clock period occurs

Description: The Fault Collection and Control Unit Output Supervision Unit (FOSU) may issue a destructive reset if all of the following conditions are present:

• An input fault is programmed as hardware recoverable in a FCCU Non-Critical Fault Configuration Register (FCCU_RF_CFGn)

• The only reaction programmed for this fault is FCCU Error Output signaling (FCCU_EOUT_SIG_ENn)

• The source of the fault signal is asserted for less than one safe clock period. The safe clock for this device is the internal RC oscillator (IRC).

Workaround: Always configure faults as software recoverable in the FCCU_RF_CFGn. Set the Recoverable Fault Configuration (RFCx) bit to a ‘1’, this is the default condition for implemented faults after reset.

ERR007226: FCCU: the error-out signalling cannot be disabled in non Bi-stable protocols

Description: In the Fault Collection and Control Unit (FCCU) module, the error out signalling can only be disabled when using the Bi-stable protocol and not all of the protocols.

The Error Out (EOUT) Signaling Enable Register (FCCU_EOUT_SIG_ENx) registers, which can be used for disabling error out signalling, are applicable only for bi-stable mode and do not affect the other protocols. (Various protocols like Bi-stable, Dual rail and other protocols are selected by programming the Fault Output Mode (FOM) field of the Configuration Register (FCCU_CFG)).

Workaround: Do not expect and program the EOUT_SIGN_ENx register to support anything other than bi-stable protocol.

The EOUT Signaling Enable (EOUTENx) field description of FCCU_EOUT_SIG_EN should be read as follows:

0 = EOUT signaling is disabled for error Recoverable Fault (RF) source x for bi-stable protocol. Error pins behave as if in Non-Faulty state.

1 = EOUT signaling is enabled for error RF source x for bi-stable protocol.

ERR007325: FCCU: Unsuccessful decorated storage access may cause erroneous signaling of FCCU Channel 40

Description: In rare conditions, a decorated memory reference targeting the system RAM may interfere with the Decorated Storage Memory Controller (DSMC) safety monitor. The result is an erroneous signalling of Channel 40 in the Fault Collection and Control Unit (FCCU). This will only occur if the decorated memory reference is unsuccessful due to either an illegal decoration encoding or a non-correctable Error Correction Code (ECC) event. Although this usually coincides with an exception taken by the core, no exact procedure is known to detect an erroneous signalling of Channel 40.

Workaround: No special workaround required. Software should handle any fault of FCCU Channel 40 according to the chosen fault reaction for this channel.

ERR007638: FCCU: Unsuccessful decorated storage access may cause erroneous signaling of FCCU Channel NCF[39]

Description: In rare conditions, a decorated memory reference targeting the system RAM may interfere with the Decorated Storage Memory Controller (DSMC) safety monitor. The result is an erroneous signalling of NCF[39] in the Fault Collection and Control Unit (FCCU). This will only occur if the decorated memory reference is unsuccessful due to either an illegal decoration encoding or a non-correctable Error Correction Code (ECC) event. Although this usually coincides with an exception taken by the core, no exact procedure is known to detect an erroneous signalling of NCF[39].

Workaround: No special workaround required. Software should handle any fault according to the chosen fault reaction for this channel.

ERR010963: Flash: Memory accesses may be corrupted when flash is operating between 33MHz and 75MHz

Description: Error Correction Code (ECC) errors may be generated in the flash due to corrupted data when all of the following conditions are met:

- The flash operating frequency is greater than 33MHz and less than or equal to 75MHz
- Read Wait State Control (PFLASH_PFCR1[RWSC]) = 2 and Address Pipeline Control (PFLASH_PFCR1[APC]) = 1
- Pipelined reads which are executed between the UTEST flash block and any other flash block. Pipelined reads are overlapping reads, where before the previous read is completed a new read is requested.

Device has UTEST memory space and remaining flash area is the main array memory space. This issue condition occurs if pipelined reads are executed between UTEST and the main array space. If pipelined reads are executed between UTEST and UTEST memory space or between main array and main array memory space, this issue condition will not be seen.

Workaround: When the flash clock frequency is greater than 33MHz and less than or equal to 75MHz configure PFLASH_PFCR1[RWSC] = 2 and PFLASH_PFCR1[APC] = 0. The configuration for all other flash clock frequencies is specified in the MCU datasheet.

ERR008647: FLASH: (SPC57xxM) Address Encode False Report (MCR[AEE] and possible FCCU channels)

Description: During Flash Read while Write operations (RWW), it is possible for a Program or Erase operation to corrupt the Address Encode feature of the flash, and falsely give an Address Encode Error (AEE) event. The false AEE event only occurs for RWW operations to partitions in the Low, Mid or High address spaces, and may only occur if the read and write occur both in Even RWW partitions (i.e. 0, 2, or 4), or if the read and write occur both in Odd RWW partitions (i.e. 1, 3, 5).

Reads to even numbered RWW partitions while writing to odd numbered RWW partitions will not trigger this false AEE condition. Likewise, reads to odd numbered RWW partitions while writing to even numbered RWW partitions will not trigger this false AEE condition.

Reads and Writes to 256K blocks, do not show the address encode corruption issue.

Read Data and ECC Parity bits returned for these Reads while writing are valid and not corrupted.

Workaround: Disable the Fault Collection and Control Unit (FCCU) Failure input channels 38 and 39 by clearing the corresponding bits in the Recoverable Fault Enable Register 1 (FCCU_RF_E1). FCCU input channel 38 is the Encoding Error Flash, and covers the C55FMC_MCR[AEE] event indication in addition to Flash Read Voltage Error (C55FMC_MCR[RVE]) and Flash Read Reference Error (C55FMC_MCR[RRE]) indications and is controlled by the FCCU_RF_E1[bit 25]. FCCU input channel 39 is the flash controller address feedback event indication and is controlled by the FCCU_RF_E1[bit 24].

Address Encode Error (AEE) fault recognition is a safety mechanism used to detect potential permanent and transient faults in the flash address decode logic. Even with AEE detection disabled, most faults that would be detected by AEE are still detected by other mechanisms as part of the MPC57xxM redundant safety concept. Consequently, disabling the AEE fault notifications in the FCCU has no impact to the overall functional safety integrity of the device, and the ISO26262 ASIL D target is achieved.

With the FCCU channels 38 and 39 disabled, the Read Voltage Error and Read Reference error may be monitored by reading the fields from the Flash Module Configuration register (C55FMC_MCR[RVE], C55FMC_MCR[RRE]).

ERR008004: FLASH: Array Integrity with Breakpoints enabled may skip addresses for certain RWSC and APC combinations

Description: For certain combinations of the Flash Read Wait State Control (RWSC) and Address Pipeline Control (APC) settings in the Platform Flash Configuration Register (PFLASH_PCFR1) the Flash's array integrity (AI) check when run with breakpoints enabled may skip addresses

resulting in an incorrect Multiple Input Signature Register (MISR) value or in the case of back to back ECC event errors (EER) or Single Bit Correction (SBC) events, a skipped breakpoint. This occurs for the following combinations:

RWSC=1 and APC=1

RWSC=3 and APC=2

RWSC=5 and APC=3

If breakpoints are enabled and an EER or SBC cause a breakpoint to occur the address after the breakpoint will be skipped, and the resulting MISR will not match expectations. Likewise, if there are back to back errors (EER or SBC) during AI with the above RWSC/APC combinations the 2nd error (and breakpoint) will be missed.

Margin Read (which by specification is a self timed event and is independent of wait states selected) is not affected by this erratum. This erratum only applies to Array Integrity.

Workaround: One workaround is to follow the recommended RWSC and APC combinations for given frequencies. If this is done, Array Integrity with Breakpoints feature works as expected. Valid RWSC/APC combinations listed in the specification are:

Flash Operating Frequency	RWSC	APC
30 MHz	0	0
100 MHz	2	1
133 MHz	3	1
167 MHz	4	1
200 MHz	5	2

A second workaround is if the above RWSC and APC combinations (listed in the description) are desired to be checked, do so without enabling breakpoints. In this case, the first EER or SBC event will be logged, and the MISR will correctly reflect the result of all reads being executed.

ERR007991: 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.

ERR050119: FlexRay: Disabling of FlexRay Message Buffer during the STARTUP Protocol State takes longer than expected three Slots

Description: Disabling of FlexRay Message Buffer takes longer than the expected three Slots. This is observed, when software application tries to disable the Message Buffer during the FlexRay STARTUP protocol state (vPOC!State = POC:startup) when vPOC!StartupState = “initialize schedule” or “integration consistency check”.

In this scenario, FlexRay Communication Controller keeps the specific Message buffer search results until the availability of next cycle start/segment start/slot start events and therefore prevent the disabling of Message Buffer.

Note:

1.All Message Buffers can be disabled immediately if FlexRay protocol state (vPOC!State) is in following States: “POC:default config”, “POC:config”, “POC:wakeup”, “POC:ready”, “POC:halt”, “POC:startup” and (vPOC!StartupState = "POC:integration listen” or "POC: ColdStart-Listen”).

2.All Message Buffers can be disabled within three slots, if FlexRay protocol state (vPOC! State) is in following states: ”POC: Normal-Active” or ”POC: Normal-Passive”.

Workaround: Do not disable Message Buffer, while FlexRay is in STARTUP protocol State

ERR008770: 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.

ERR010647: GTM: (A)TOM asynchronous update in SOMP mode not functional with CM1=0 and selected CMU clock is not equal to the system clock

Description: (GTM-IP-260)

In the Generic Timer Module (GTM), an asynchronous update of the duty cycle by writing the value 0 to the Compare Register 1 (CM1) while the clock reference selected is not equal to the GTM system clock does not cause the signal to go to the inactive level. The output signal remains at its active level.

This behavior impacts both the Timer Output Module (TOM) and the Advanced Router Unit (ARU) connected TOM (ATOM) when used in Signal Output Mode Pule Width Modulation [PWM] (SOMP).

Workaround: Write 1 to CM1 instead of 0.

ERR010646: GTM: (A)TOM output signal postponed by one period for the values CM0=1 and CM1>CM0 if CN0 is reset by the trigger of a preceding channel

Description: (GTM-IP-270)

The following behavior impacts both the Timer Output Module (TOM) and the Advanced Router Unit (ARU) connected TOM (ATOM) when used in Signal Output Mode Pule Width Modulation [PWM] (SOMP) of the Generic Timer Module (GTM).

When the channel counter register (CN0) is reset by the trigger of a preceding channel, in other words the bit Reset source of Counter Compare Unit 0 [CCU0] (RST_CCU0) of the (A)TOM channel control register {(A)TOM[i]_CH[x]_CTRL} is set, the value of the CCU0 Compare register (CM0) defines the assertion of the output signal level to the value of the Signal Level bit (SL) whereas the value of the CCU1 Compare register (CM1) defines the assertion to the inverted value of SL bit (!SL).

In the case where CM0 = 1 and CM1 is greater than CM0, the expected output edge is postponed by one period.

Workaround: Instead of configuring CM0=1, configure CM1=1 and invert SL to get the expected edge with a counter value of 1 (CN0=1) the expected edge (of configuration CM0=1).

ERR008122: GTM: (A)TOM's CCU1 event interrupt is not generated when CM1=0 or 1 and RST_CCU0=1

Description: (GTM-IP-202)

If a Generic Timer Module (GTM) Timer Output Module (TOM) or Advanced Router Unit (ARU) connected TOM (ATOM) channel is configured with the reset source of the channel as the previous channels trigger (CHn_CTRL[RST_CCU0]=1) and the counter 0 (CN0) counts from 0 to MAX, the Counter Compare Unit 1 (CCU1) event interrupt is not generated if the Compare Register 1 (CM1) is 0 and Compare Register 0 (CM0) is greater than 0. If the Compare Register (CM1) is 1 and Compare Register 0 (CM0) is MAX+1 only one CCU1 interrupt will be generated.

Workaround: To trigger channel x+1 which is the channel that triggers when channel x counter CN0 is reset use the configuration of CM0=MAX and CM1=1.

- When the duty cycle configuration is CM1=0 and CM0>0 on channel x use the CCU0 interrupt of triggering channel x+1 instead of CCU1 interrupt.
- When the duty cycle configuration is CM1=1 and CM0=MAX+1 on channel x use the CCU1 interrupt of triggering channel x+1 instead of CCU1 interrupt on channel x.

ERR006639: GTM: A compare match event does not clear WR_REQ when ATOM is in SOMC mode

Description: (GTM-IP-146)

If a Generic Timer Module (GTM) ARU Connected Timer Output Module (ATOM) channel is operating in Signal Output Mode Compare (SOMC) mode, with the Advanced Routing Unit (ARU) enabled and a late compare register update is requested by the CPU by setting WR_REQ (CPU Write request bit for late compare register update in ATOM[i]_CH[x]_CTRL), a late update of the Counter Compare Unit (CCU0/1) Compare registers (CM0/CM1) and/or the compare strategy (i.e. ARU Control Bits (ACB) bits altered) is successfully done, but after final compare match the WR_REQ bit is not reset. As a result no new ARU read request is set up after final compare match.

Workaround: CPU should clear WR_REQ by software after the late update.

ERR007085: GTM: A TIM timeout occurs when the TDU is re-enabled

Description: (GTM-IP-163)

The Generic Timer Module (GTM) Timer Input Module (TIM) Timeout Detection Unit (TDU) indicates a timeout event when it is re-enabled and the new TDU Timeout Value (TOV) is lower than the previous TOV.

After stopping the TDU, the Time Out Counter (TO_CNT) field will have an arbitrary value less than or equal to (<=) the timeout value. When TOV is reconfigured to a value less than or equal to (<=) TO_CNT, and the TDU re-enabled an incorrect timeout is signaled. This is because if at the same time as the TDU is enabled the selected clock has an active edge, TO_CNT is greater than or equal to (>=) TOV because TO_CNT is not reset to zero.

Workaround: If the TDU needs to be changed to a TOV value which is less than the previous value either:

1) Postpone disabling the TDU until the TO_CNT is less than (<) the new TOV. Then disable the TDU, configure TOV, and re-enable TDU Unit.

2) Disable the TDU and then write TOV with the value 0xFF. Then enable the TDU unit and re-configure TOV to the desired value.

ERR007528: GTM: Action not always calculated immediately by DPLL

Description: (GTM-IP-170)

If the Generic Timer Module (GTM) Digital PLL (DPLL) action calculation is interrupted by a new input event, the next TRIGGER/STATE input event action calculation (after the sub increment calculation is finished) starts at the previous internal action number. If new action data arrives during the sub increment calculation it is only used after the next input event. New Position Minus Time (PMT) data for an action with a higher action number is not recognized immediately.

Normally, the calculation of sub increments and PMT are not done in parallel because of resource sharing. When the DPLL is doing the action calculation it has exclusive access rights to RAM Region 1a which contains the PMT request values, so the DPLL cannot accept new PMT requests via the Advanced Routing Unit (ARU). Therefore requested actions are not calculated regularly with every tooth.

Workaround: The GTM should only request actions which are not “past” with every new tooth. The synchronization of the Multi Channel Sequencer (MCS) task to Timer Input Module (TIM) input event can be done by routing the TIM edge capture event value via ARU to MCS. If new PMT data arrives after the action number has reached the value zero, the action is calculated immediately starting with the highest action number again.

You can request the action calculation tooth by tooth until an action runs in to the past. Additional PMT requests can be placed earlier while the DPLL is performing sub increment calculations because RAM Region 1a is exclusively used for PMT requests via ARU.

Send PMT requests at least 3 teeth before the action has to be executed. This ensures that the MCS and ARU Connected Timer Output Module (ATOM) get action results from a calculation an input event cycle before.

ERR007084: GTM: An active edge input, that is rejected by the DPLL trigger plausibility check, does not assert a Missing Trigger Interrupt

Description: (GTM-IP-162)

The Generic Timer Module (GTM) Digital PLL (DPLL) Missing Trigger Interrupt (MTI) is not asserted during a gap in the trigger profile, stored in RAM region 2c, when an active input signal edge is rejected by the Plausibility Value of Trigger (PVT) check.

In this case, the DPLL’s internal check for MTI is performed when the invalid active input occurs. The check for the first and valid active inputs is not done for this gap. As a result, when monitoring the DPLL synchronization using the MTI in a gap, the application may report a synchronization problem which is not real.

Workaround: The activated PVT check is reported by the activation of the Plausibility Window Violation of TRIGGER (PWI) interrupt. This interrupt can be used to check if a gap condition in the profile has occurred. This information can be used to correct the incorrect synchronization information out of the DPLL.

ERR011364: GTM: An unexpected restart of Signal Output Mode Serial one-shot cycle of Advanced Routing Unit-connected Timer Output Module happens when ATOM Counter Compare Unit 0 register is zero

Description: (GTM-IP-320)

If Advanced Routing Unit(ARU)-connected Timer Output Module (ATOM) of the Generic Timer Module (GTM) is configured for Signal Output Mode Serial (SOMS) one-shot mode (ATOM[i]_CH[x]_CTRL.MODE = 0b11 and ATOM[i]_CH[x]_CTRL.OSM = 0b1), a one-shot cycle is being restarted immediately by writing a value differing from zero to ATOM channel shadow register (ATOM[i]_CH[x]_SR0) while the value of ATOM Counter Compare Unit 0 (CCU0) compare register is zero (ATOM[i]_CH[x]_CM0 = 0).

Workaround: Avoid having value 0 in ATOM Counter Compare Unit 0 compare register (ATOM[i]_CH[x]_CM0) when Signal Output Mode Serial (SOMS) one-shot mode is enabled (ATOM[i]_CH[x]_CTRL.OSM = 0b1).

ERR006409: GTM: ATOM Force Update does not activate a comparison when in SOMC mode

Description: (GTM-IP-139)

When the Generic Timer Module (GTM) ARU Connected Timer Output Unit (ATOM) is configured in Signal Output Mode Compare (SOMC) mode, with the Advanced Routing Unit (ARU) Enabled, and if no comparison is active (no valid data was received by ARU, ATOM[i]_CH[x]_STAT.B.DV = 0) a Force Update Request can cause the ATOM to remain in a state waiting for the update event to happen.

A Force Update is requested by first setting CPU Write Request field (WR_REQ) in ATOM[i]_CH[x]_CTRL, then updating the Shadow Counter Registers (ATOM[i]_CH[x]_SRx) and optionally the ATOM Mode Control Bits (ACB) in ATOM[i]_CH[x]_CTRL, and finally updating the Counter Registers (ATOM[i]_CH[x]_CMx) via a Forced Update (ATOM[i]_TGCx_FUPD_CTRnL = 0b11).

Under the above conditions:

- The registers CMx are updated correctly but no new comparison is activated.
- The ACBO bits are erroneously not cleared.
- The ARU read request is canceled because of WR_REQ=1.

Workaround: After the Forced Update, re-write the desired values to CM0 or CM1 to activate the comparison and to reset the ACBO bits.

ERR010156: GTM: Behavior of PVT check of DPLL on direction change

Description: (GTM-IP-220)

In the Generic Timer Module (GTM) documentation, the behavior of the check for Plausibility Value of next active TRIGGER slope (PVT) inside the Digital PLL (DPLL) sub-module is not described.

Workaround: Take into account the following behavior.

After a direction change, the PVT parameter is set to zero.

ERR007848: GTM: Bit 0 of TIM edge counter register may not indicate the actual signal level

Description: (GTM-IP-181)

When a Generic Timer Module (GTM) Timer Input Module (TIM) channel is enabled, bit 0 of the Edge Counter register (ECNT) may not reflect the current signal level of the filtered input TIM[i]_CH[x]_FOUT until the next input edge occurs. This issue occurs when the ECNT register is not read before re-enabling the channel.

This erratum does not affect TIM Bit Compression Mode (TBCM).

Workaround: After disabling the TIM channel, ensure that the ECNT register is read at least once before the TIM channel is re-enabled. Alternatively. before re-enabling a TIM channel, issue a TIM channel reset and reconfigure the TIM channel control registers.

ERR006645: GTM: Clearing of DPLL PCM1/2 bits after the Missing Pulse Correction Values calculations delayed

Description: (GTM-IP-158)

The Generic Timer Module (GTM) Digital PLL (DPLL) Pulse Correction Mode bits (PCM1/2 in DPLL_CTRL_1) are expected to be cleared after the Missing Pulse Correction Values (MPVAL1/2) are used to calculate the number of sub_incs for the next increment and to calculate the add_in values by the DPLL State Machine, however the PCM1/2 bit is transferred with an active edge into the dedicated shadow registers, but cleared some time later. If the PCM1/2 bits are written by the CPU in between the point of time of the transfer to the shadow register and the point of time were the PCM1/2 bits are cleared, the bits are cleared and never used.

Workaround: Do not allow the CPU to write to the PCM1/2 bits until at least 750 system clocks have passed since the previous Trigger Active Slope Detected (TASI) interrupt. This time could be derived by an GTM resource like an ARU connected TOM (ATOM) channel.

ERR008688: GTM: Data lost in ATOM when CMU_CLKx is slower than ARU

Description: (GTM-IP-210)

The Generic Timer Module (GTM) Advanced Routing Unit (connected Timer Output Module (ATOM) in Signal Output Mode Serial (SOMS) one-shot mode (ATOM[i]_CHn_CTRL[OSM=1]) starts to request new data from the Advanced Routing Unit (ARU) when the channel control (ATOM[i]_CHn_CTRL[ARU_EN]) is set to 1. When new data is delivered by the ARU and stored into the Shadow Registers (ATOM[i]_CHn_SR0 and ATOM[i]_CHn_SR1), the data is immediately transferred to the Compare Match Registers (ATOM[i]_CHn_CM0 and ATOM[i]_CHn_CM1) and the ATOM starts to shift with the next clock tick of the selected CMU_CLKx. In parallel, the ATOM immediately requests new data from the ARU. If the ARU delivers new data before the first bit of the previous data is shifted out, which means before the CMU_CLKx clock ticks, the data is stored into Shadow Registers but it is not marked as valid (ATOM[i]_CHn_STAT[DV] bit is not set) and therefore the data is ignored.

Workaround: Software programming must ensure that the time between two consecutive data being delivered from ARU is greater than two periods of the selected CMU_CLKx clock. This can be accomplished by sourcing the data from the Multi-Channel Sequencer (MCS) instead of the First-In-First-Out (FIFO) buffer.

Ensure that the CMU_CLKx is twice the speed of the ARU round trip time. The data loss cannot occur if the ARU round trip time is greater than two CMU_CLKx periods.

ERR011215: GTM: Disabling the sub-position generators will not disable output pulses when operating the DPLL in continuous mode

Description: (GTM-IP-302)

When the Digital PLL(DPLL) sub-module of the Generic Timer Module (GTM) is configured in a continuous mode (DPLL_CTRL_1.DMO = 0b1) the pulse generation cannot be switched off by disabling sub-position increment generators SUB_INC1 and SUB_INC2 in DPLL Control

Register 1 (DPLL_CTRL_1.SGE1/2 = 0). The pulse generation is ongoing independently of sub-position increment generator mode configured in DPLL Control Register 0 and Control Register 1 (DPLL_CTRL_0.RMO, DPLL_CTRL_1.SMC).

Workaround: Configure number of pulses in Counter for number of SUB_INC1/2 pulses to be zero (DPLL_CNT_NUM1/2 = 0) to suppress pulse generation when DPLL is in continuous mode.

ERR010159: GTM: Discontinuities in DPLL sub-increment in full scale

Description: (GTM-IP-223)

In the Digital PLL sub-module (DPLL) of the Generic Timer Module (GTM), when:

- Number of recent TRIGGER events used (NUTE) is set to full scale, that is, the Full Scale of TRIGGER (FST) bit is set in the DPLL Number of Recent TRIGGER events used for Calculations (DPLL_NUTC) register

or

- Number of recent STATE events used (NUSE) is set to full scale, that is, the Full Scale of STATE (FSS) bit is set in DPLL Number of Recent STATE events used for Calculations (DPLL_NUSC) register

and

- Physical deviations are used, meaning the Adapt mode TRIGGER (AMT) bit or the Adapt mode STATE (AMS) bit of the DPLL Control register 0 (DPLL_CTRL_0) are set

or

- High speed accelerations are occurring

Then the sub-increment generation will show an irregular behavior, meaning that the pulse generator frequency is not correct and the micro-ticks resulting are too fast or too slow.

Workaround: Do not use the DPLL in “full scale” mode when NUTE/NUSE is set to the maximum and FST/FSS bit is set to one, when stronger accelerations exist or physical deviation with significant deviation is used.

ERR010149: GTM: DPLL PMT calculation result is incorrect in backward direction

Description: (GTM-IP-219)

In the Generic Timer Module (GTM) Digital Phase Lock Loop (DPLL), an internal pointer register is incorrectly calculated when running in the backwards direction. As a consequence, the Position Minus Time (PMT) computation of the action channels leads to incorrect results, such as a PMT result that is in the past.

Workaround: Use one of the following options.

Option 1: Do not use PMT calculation in backwards direction.

Option 2: In case PMT calculations are needed in the backwards direction, the PMT results must be sent from the DPLL to a Multi Channel Sequencer (MCS) channel in order to verify the result is not in the past. The MCS channel should therefore only provide valid PMT results to the target ARU-connected Timer Output Module (ATOM) channel.

ERR007531: GTM: DPLL Position Minus Time result is not sent to the ARU

Description: (GTM-IP-174)

The Generic Timer Module (GTM) Digital PLL (DPLL) has a state where there is a delay between when the New Output Data Values Concerning To Action t bit (DPLL_ACT_STA[ACT_N(t)]) is reset and when the corresponding shadow bit (DPLL_ACT_STA_shadow[ACT_N(t)]) is set to “1”, which starts the transfer of the output data via the Advance Routing Unit (ARU).

If during this delay a new input event occurs (DPLL_ACT_STA[ACT_N(t)]) and the internal state controller changes to process this new input event, DPLL_ACT_STA_shadow[ACT_N(t)] is not yet set, so there is no request to transmit the output data to the ARU. In this case, an action calculation is finished without transferring the data via the ARU. PMT calculations where the result is not “past” are not affected by this issue. The time frame in which a incoming input signal causes the issue is about 25 system clock cycles.

Workaround: Use a Multi Channel Sequencer (MCS) channel to read the PMT data from DPLL via non blocking ARU reads using the NARD or NARDI instructions. The data that is read should be tested to check whether the requested action is ‘out of time’ by reading the Time Base Unit Time Stamp Channel 0 (TBU_TS0), or ‘out of angle’ by reading the Time Base Unit Time Stamp Channel 1 or 2 (TBU_TS1/2). If the TBU_TSx values are not within an acceptable window of the PMT value, the MCS can request the old value again, or if the requested event is in the past, request a new value.

Additionally, the Timer Input Module 0 (TIM0) interrupt can be routed to the MCS to check if an active edge occurred. Each action which delivers a PMT result should be checked only once by the MCS. If the PMT result is transferred directly from the DPLL to the ARU connected Timer Output Module (ATOM), the MCS should be prepared to send a default value to the ATOM which is not in the past to ensure that even if the DPLL fails to send the PMT result to the ATOM, the ATOM does not miss the event completely.

ERR010148: GTM: DPLL PWI interrupt request always active

Description: (GTM-IP-218)

When the Generic Timer Module (GTM) Digital PLL (DPLL) is activated by setting the DPLL enable bit (DEN) of the DPLL Control Register 1 (DPLL_CTRL_1) and after that the following conditions occur:

a) a write is performed to the DPLL Control Register 0 (DPLL_CTRL_0)

b) the STATE input is activated

Then, once the Plausibility window (PVT) violation interrupt for the TRIGGER flag (PWI) is set in the DPLL Interrupt Notify register (DPLL_IRQ_NOTIFY), the interrupt will be constantly activated.

Workaround: Do not write to DPLL_CTRL_0 after enabling the DPLL (DPLL_CTRL_1.DEN = 1).

ERR008430: GTM: DPLL sub-inc generation and action calculations are delayed

Description: (GTM-IP-208)

The Generic Timer Module (GTM) Digital Phase Lock Loop (DPLL) delays the start of sub-increment (SUB_INC) generation and the action calculation (PMT) by one input event cycle if the DPLL starts after activation (DPLL_CTRL1[DEN]= 0 ->1) while the first TRIGGER detected flag (DPLL_STATUS[FTD]) is not raised and Pulse Correction Mode (PCM) is enabled (DPLL_CTRL1[PCMn]= 1). This results in an incorrect angle clock (TBU_TS1).

Workaround: Set DPLL_CTRL1[PCMn] bits to '0' before the DPLL is activated.

ERR010158: GTM: DPLL TAXI interrupt not deactivated for THMA = 0

Description: (GTM-IP-222)

In the Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM), the TRIGGER maximum hold time violation interrupt (TAXI) event is not deactivated when the TRIGGER hold time maximum value (THMA) is equal to zero (THMA = 0x0).

The TAXI bit of the DPLL interrupt register (DPLL_IRQ_NOTIFY) can only be reset if the TAXI interrupt is deactivated with the next input event not causing an interrupt.

Workaround: When THMA = 0, the TAXI interrupt is not used and must be disabled by clearing the TRIGGER maximum hold time violation interrupt enable bit (TAXI_IRQ_EN) of the DPLL interrupt enable register (DPLL_IRQ_EN).

ERR010157: GTM: DPLL wrong result when NUTE/NUSE modified in dedicated time window

Description: (GTM-IP-221)

In the Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM), the following parameters:

- NUTE = Number of recent TRIGGER events used
- NUSE = Number of recent STATE events used

are used internally to the DPLL to modify pointers as well as to decide which data should be used for doing the prediction of the next increment or the selection of the algorithm of the Position Minus Time (PMT) calculation should be used. After a new input signal reaches the DPLL, either on a TRIGGER or a STATE processing unit, the internal pointers are updated shortly after the TRIGGER active slope interrupt (TASI) or STATE active slope interrupt (SASI). If the NUTE/NUSE parameter is changed after that point of time, the pointers are not updated until the next input event resulting in an inconsistency. This inconsistency may lead to the use of the wrong data which can corrupt the results of the increment prediction and the frequency calculation as well as the calculation of PMT.

Workaround: Modify NUSE/NUTE and the number of virtual increments in the current NUTE region (VTN) / number of virtual state increments in the current NUSE region (VSN) parameters in uncritical time windows. There are 2 uncritical windows:

- Window 1: after a new input signal (interrupt on Timer Input Module (TIM) channel) and before the TASI/SASI event
- Window 2: if the number n of active PMT calculation is such that THVAL > 10us +n*3us, where:

THVAL is the measured Trigger Hold Time value, meaning the time from the last valid slope to the next inactive slope # n is the number of bits being set to "1" in the Action Status register

(DPPL_ACT_STA) Then the parameters NUTE/VTN may be modified after the TRIGGER inactive slope interrupt (TISI).

ERR010648: GTM: DPLL_RAM1b.PSTC not updated after direction change and DPLL_CTRL_0 has been written

Description: (GTM-IP-272)

In the Generic Timer Module (GTM), if the Digital Phase Lock-Loop (DPLL) is running in normal mode and a direction change is detected after the DPLL Control register 0 (DPLL_CTRL_0) has been written, the Accurate calculated position stamp of last Trigger input (PSTC) value, stored in the DPPL RAM region 1b (DPLL_RAM1b), is not updated for the following active input signal and keeps the difference for the following input signals.

Workaround: Workaround 1

Do not write to DPLL_CTRL_0 until both an active TRIGGER, STATE input signal has occurred in case of an direction change within this time frame.

The write operation to DPLL_CTRL_0 can be replaced by the following operations:

a) Deactivation of Trigger/state input signal (replacing writes to State ENable [SEN] and Trigger ENable [TEN] bits):

Switch the corresponding Timer Input Module (TIM) input channels receiving the TRIGGER/ STATE input signal by configuring the TIM Auxiliary Input source selection register (TIM[i]_IN_SRC) as follow TIM[i]_CH[x]_IN:SRC.MODE[i]=“10”, VAL[i]

b) Changes from normal mode to emergency mode, write to Reference Mode bit (RMO):

If this is necessary, the write operation to DPLL_CTRL_0 cannot be prevented.

c) Changes of TRIGGER Number (TNU), STATE number (SNU) and Multiplier for TRIGGER (MLT) bit-fields:

Such changes should not be necessary, if not the write operation to DPLL_CTRL_0 cannot be prevented.

d) Changes of Adaption modes TRIGGER (AMT), STATE (AMS) bits

Activate AMT, AMS after power up, then activate/deactivate the adaptation mode by writing adapt data to the Physical Deviation (PD) of the Adapt and Profile Values of the TRIGGER (DPLL_ADT_T) or STATE (DPLL_ADT_S) memory location in respectively the RAM2 (DPLL_ADT_T.PD_T) and RAM1c (DPLL_ADT_S.PD_S).

e) Changes of Input Delay TRIGGER (IDT), STATE (IDS) bits

Enable or disable filter input data by writing to the Filter Enable (FLT_EN) bit of the corresponding TIM channel control register(TIM[i]_CH[x]CTRL).

f) Changes of Input filter position (IFP) bit

If such changes are necessary the write operation to DPLL_CTRL_0 cannot be prevented.

In case Workaround 1 cannot be used:

Workaround 2

In this case (direction change after DPLL_CTRL_0 must be written before both a TRIGGER and STATE input event has occurred) the PSTC value must be corrected via CPU access from outside the DPLL.

To achieve this the calculation

PSTC_new = PSTC_old + or – nmb_t_tar (+ forward(dir1=0); - backward(dir1=0))

must be performed and stored to RAM1b.PSTC less than 1000 system clock cycles after the active input event, or 850 system clock cycles after the TRIGGER active slope interrupt (TASI) has occurred.

The internal DPLL PSTC value is used for Position Minus Time (PMT) calculations. If no PMT calculation is ongoing in the tooth after direction change and the PSTC value is not needed in GTM external processes the described timing constraint for the PSTC correction can be relaxed until before the next PMT calculations are requested or the PSTC value is needed otherwise.

ERR010147: GTM: Erroneous data on CPU read access to an empty FIFO channel in ring buffer

Description: (GTM-IP-215)

In the Generic Timer Module (GTM), when a First-In-First-Out (FIFO) channel is configured in ring buffer mode and a read access to the GTM Bus interface (AEI) to FIFO Data Interface (AFD) FIFO x buffer access register (AFD_CH[x]_ACC) is performed while the channel is empty, then the read pointer is incremented. As a consequence the channel will deliver undefined data on the next read access by the Advanced Routing Unit (ARU).

Workaround: Use one of the following options.

Option 1: Do not execute a read access to the AFD_CH[x]_BUF_ACC when the corresponding FIFO channel is in ring buffer mode and the FIFO channel is empty.

Option 2: Do not set a FIFO channel in ring buffer mode while the FIFO channel is empty. Fill the FIFO channel first and then configure it into ring buffer mode.

ERR008689: GTM: F2A stream data are not deleted after stream disabling

Description: (GTM-IP-212)

When the Generic Timer Module (GTM) First-In-First-Out to Advanced Routing Unit (FIFO-to-ARU, F2A) data stream is disabled in the F2A enable register (F2A[i]_ENABLE[STRn_EN] = 0b01), the existing valid data inside the stream is not deleted. After re-enabling this stream (F2A[i]_ENABLE[STRn_EN] = 0b10), the F2A delivers the old data, independent of the configured data transfer direction (F2A[i]_CHn_STR_CFG[DIR]).

Workaround: Before re-enabling an F2A data stream, the old data must to be removed from the FIFO. This can be done after disabling the stream by performing the following steps.

1. Set the ARU Read Address Register (F2A[i]_CHn_ARU_RD_FIFO) to the reset value 0x1FE.

2. Configure the F2A stream direction into ARU to FIFO (F2A[i]_CHn_STR_CFG[DIR] = 0).

3. Enable the stream (F2A[i]_ENABLE[STRn_EN] = 0b10), so that the old data are transported into FIFO.

4. Finally, flush the FIFO channel (FIFO[i]_CHn_CTRL[FLUSH] = 1).

ERR010989: GTM: Incomplete pulse correction in DPLL on direction change when DPLL_CTRL_1.SMC = 1

Description: (GTM-IP-292)

In the Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM), when the Synchronous Motor Control (SMC) mode is enabled by setting the corresponding bit in the DPLL Control 1 register (DPLL_CTRL_1), the pulse correction at direction change is done incompletely; some pulses may not be placed immediately after the change of direction.

Workaround: Enable the Automatic End mode by clearing the DPLL Mode select (DMO) bit of the DPLL_CTRL_1: the missing pulses will be added at the next active input event.

ERR006644: GTM: Incorrect duty cycle in TOM PCM mode

Description: (GTM-IP-154)

The Generic Timer Module (GTM) Timer Output Module (TOM) duty cycle output in Pulse Count Modulation (PCM) mode is always one count less than the configured value in the Compare Match 1 (CM1) register. For example, if the value 1 is written to CM1, a duty cycle of 0% will be generated and if a duty cycle of 100% was configured by writing the maximum value (0xFFFF) in to the CM1 register, the resultant waveform would be a signal with duty cycle less than 100%. A value of zero in the CM1 register results in 100 % duty cycle.

Workaround: Configure the CM1 value for the targeted duty cycle in the CM1 register with target duty cycle + 1.

To get 0 % duty cycle, CM1 = 1.

To get 100 % duty cycle, CM1 = 0 and CM0 = 0xFFFF. (Setting CM0 = 0x1000 and CM1 = 0xFFFF will also result in a duty cycle of 100%)

ERR006412: GTM: Incorrect Input Signal Characteristics when the TIM channel is in TBCM mode and ECNT is selected as the captured GPR0 value.

Description: (GTM-IP-142)

When a Generic Timer Module (GTM) Timer Input Module (TIM) channel captures an input pattern match condition in TIM Bit Compression Mode (TBCM), the values captured by the General Purpose Register TIM[i]_CH0_GPR0 are incorrect under the condition EGPR0_SEL=1 with GPR0_SEL= 0 (use ECNT as input).

Starting at t=0 with counter value ECNT(t=0), the captured values of two consecutive edges can be ECNT(t=0)+2 followed by ECNT(t=0)+2 instead of ECNT(t=0)+1 followed by ECNT(t=0)+2. The captured ECNTs do not increment by 1 as expected, and reading TIM[i]_CH0_GPR0 shows an inconsistency when comparing ECNT [bits 31:24] to GPR0 [bits 7:0].

Workaround: Ignore the captured data via the Advanced Routing Unit (ARU) and construct the edge count value with an independent Multi Channel Sequencer (MCS) counter which increments on each ARU transfer, or when reading TIM[i]_CH0_GPR0 use only TIM[i]_CH0_GPR0[31:24] as EDGE counter; do not use TIM[i]_CH0_GPR0[23:0].

ERR006411: GTM: Incorrect Input Signal Characteristics when the TIM channel is in TIEM, TPWM, TIPM, TPIM or TGPS mode, when ECNT is selected as the captured GPRi value.

Description: (GTM-IP-141)

When a Generic Timer Module (GTM) Timer Input Module (TIM) channel captures a valid rising edge event at TIM[i]_CH[x]_FOUT (post filtering) in TIM Input Event Mode (TIEM), TIM PWM Measurement Mode (TPWM), TIM Input Prescaler Mode (TIPM), TIM Pulse Integration Mode (TPIM) or TIM Gated Periodic Sampling Mode (TGPS), the captured values of the Edge Counter (TIM[i]_CH[x]_ECNT) to the General Purpose Registers (TIM[i]_CH[x]_GPRi) are incorrect. The captured value will be ECNT+2; bit 0 (signal level) will be 0 (Falling Edge). The correct operation would be to capture ECNT+1; bit 1 (signal level) would be 1 (Rising Edge).

This leads to an inconsistency between the ARU signal level bit, bit 0 of the ARU word which shows the captured ECNT value, and TIM[i]_CH[x]_GPRi which shows an inconsistency when comparing GPRi [bits 31:24] to ECNT [bits 7:0].

Workaround: When using the TIMs captured data the correct data can be reconstructed by:

if ARU_SIGNAL_LEVEL ==1 and ARU_DATA[0] == 0 the ARU_DATA = ARU_DATA -1;

When reading TIM[i]_CH[x]_GPRi by the data can be corrected as long as there is no GPR overflow and no new edge by:

if TIM[i]_CH[x]_GPRi[24] == 1 and TIM[i]_CH[x]_GPRi[0] == 0 then TIM[i]_CH[x]_GPRi[23:0] = TIM[i]_CH[x]_GPRi[23:0] -1;

ERR011363: GTM: Incorrect setting of Digital PLL Lock status on missing trigger/state, trigger/state out of range or Digital PLL direction change event

Description: (GTM-IP-317)

When the Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM) is not locked for TRIGGER or STATE signals (DPLL_STATUS.LOCK1/2 = 0) and one of the three following events occurs:

i. Digital PLL direction change

ii. An input signal with unexpectedly missing TRIGGER/STATE (DPLL_STATUS.MT/MS = 1)

iii. TRIGGER/STATE out of range (DPLL_STATUS.TOR/SOR = 1)

the lock status bit is being set (DPLL_STATUS.LOCK1/2 = 1) after two subsequent missing TRIGGER/STATE interrupt events detected instead of this lock status bit being set after two subsequent missing TRIGGER/STATE in either the same direction or without an unexpected missing TRIGGER/STATE interrupt (increment number of TRIGGER/STATE is different form profile thus not plausible DPLL_STATUS.ITN/ISN=1 and missing TRIGGER/STATE interrupt) or a TRIGGER/STATE out of range interrupt occurs in between.

Workaround: When DPLL is not locked for TRIGGER/STATE signals the status of an unexpected missing TRIGGER/STATE, direction change and TRIGGER/STATE out of range must be monitored to make decision if DPLL lock status bit (DPLL_STATUS.LOCK1/2) is being set correctly. For this reason either the interrupt flags DPLL_IRQ_NOTIFY.MTI/MSI, DPLL_IRQ_NOTIFY.CDTI/CDSI, DPLL_IRQ_NOTIFY.TORI/SORI or the TRIGGER/STATE signal direction

(DPLL_STATUS.BWD1/2), increment number of TRIGGER/STATE plausibility (DPLL_STATUS.ITN/ISN) and TRIGGER/STATE events out of range (DPLL_STATUS.TOR/SOR) need to be monitored.

ERR006643: GTM: Incorrect timestamp captured in CNTS when TIM operates in TPWM or TPIM modes if CMU_CLK is not equal to system clock

Description: (GTM-IP-153)

When a Generic Timer Module (GTM) Timer Input Module (TIM) channel operates in the following configuration, the Timebase Unit Channel 0 (TBU_TS0) value is not captured correctly in to the counter register CNTS:

- PWM Measurement Mode (TPWM) or Pulse Integration Mode (TPIM) and
- TBU_TS0 selected as the input to CNTS (CNTS_SEL = 1 in TIM[i]_CH[x]_CTRL) and
- selected clock (CMU_CLKn) of the TIM channel is not the same frequency as the system clock.

Workaround: Configure the TIM channel to operate on a CMU_CLK (Divider =1) which is identical to the system clock when TBU_TS0 is to be captured in TPWM or TPIM modes.

ERR010175: GTM: Input events may be missed for some time after the DPLL is enabled

Description: (GTM-IP-247)

In the Generic Timer Module (GTM), a new input signal on either TRIGGER or STATE is not recognized or stored by the Digital Phase-Lock Loop (DPLL) for some time after the DPLL is enabled in the DPLL Control Register 1 (DPLL_CTRL_1) by setting the DPLL Enable bit (DEN) (DPLL_CTRL_1[DEN] changed from 0 to 1).

After power on reset or a DPLL software reset, the time is approximately 140 clock cycles. When the DPLL is enabled after the module was disabled, the time is 20 clock cycles for the STATE signal and approximately 45 clock cycles for the TRIGGER input signal. The TRIGGER signal time is lengthened by parallel accesses to the RAM1b memory. Each RAM1b access lengthens the time window by 10 clock cycles.

Workaround: When the DPLL is enabled for the first time after a reset, the first event may be neglected and the DPLL calculation may be delayed by one event.

When the DPLL is re-enabled during operation (was enabled, then disabled), there are two possible workarounds.

Workaround 1:

Within the time frame after the DPLL is enabled the Timer Input Module (TIM) inputs must be observed to check for a new event on TRIGGER or STATE signal: The angle clock must then be adapted setting the Pulse Correction Mode for SUB_INC1 / SUB_INC2 bits (PCM1, PCM2) in the DPLL_CTRL_1 register.

Workaround 2:

Within the time frame after the DPLL is enabled the Timer Input Module (TIM) inputs must be observed to check for a new event on TRIGGER or STATE signal: The angle clock must be corrected, for the missing pulses: re-generate a TIM input event by writing to TIM0 Input Source Register (TIM_0_IN_SRC).

ERR007847: GTM: MCS's CAT status may be incorrect

Description: (GTM-IP-178)

The Generic Timer Module (GTM) Multi-channel Sequencer's (MCS) Advance Routing Unit (ARU) blocking read/write instructions, such as ARD, AWR, ARDI, and AWRI, describe the use of the Cancel ARU Transfer (CAT) status field to check whether the last ARU transfer was successful (CAT=0) or canceled (CAT=1). Because CAT can be written by software to cancel an ARU transfer at any time, CAT does not reliably reflect the last ARU transfer status.

Workaround: Check data consistency of the ARU transfer by inspecting the transferred data (for example, check for the linear increment of the edge counter (ECNT) for data transfers from Timer Input Module (TIM) to the MCS) instead of relying on the CAT field.

ERR007530: GTM: New DPLL Position Minus Time data not received

Description: (GTM-IP-173)

When the Generic Timer Module (GTM) Digital PLL (DPLL) receives Position Minus Time (PMT) requests after a TRIGGER/STATE event, only that request can be considered. The DPLL blocks new PMT requests for about 200 ns. New PMT requests are only accepted after the calculation of the pending action calculations are performed. This calculation starts in the state machine, about 10 us after the input event and completes depending on the number of actions (A) to be calculated A*3.7 us later. After this time the PMT request is accepted, but it is not possible to adjust the action calculation with updated data. The "old" value is always calculated.

The PMT result is calculated based on older PMT input data because the pending data transfer with newer input data to the DPLL cannot be executed.

Workaround: When the calculated action is transmitted to the Multi Channel Sequencer (MCS), check if there was an Advanced Routing Unit (ARU) transfer with new data for this action blocked by the ARU because the DPLL was not ready to receive new data within this time. If the ARU transfer was just completed, the corresponding action contains only the older PMT requirements. Ignore this action value and wait for the new value which appears about 3.7 us after the PMT requirement update was transmitted.

ERR010649: GTM: No DCGI interrupt after direction change and DPLL_CTRL_0 has been written

Description: (GTM-IP-271)

In the Generic Timer Module (GTM), if the Digital Phase Lock-Loop (DPLL) is running in normal mode and a direction change is detected after the DPLL Control register 0 (DPLL_CTRL_0) has been written, the Direction change interrupt (DCGI) does not occur for following direction changes until both a TRIGGER and a STATE input signal has been arrived at the DPLL inputs.

Workaround: Workaround 1

Do not write to DPLL_CTRL_0 until both an active TRIGGER, STATE input signal has occurred in case of an direction change within this time frame.

The write operation to DPLL_CTRL_0 can be replaced by the following operations:

a) Deactivation of Trigger/state input signal (replacing writes to State ENable [SEN] and Trigger ENable [TEN] bits):

Switch the corresponding Timer Input Module (TIM) input channels receiving the TRIGGER/ STATE input signal by configuring the TIM Auxiliary Input source selection register (TIM[i]_IN_SRC) as follow TIM[i]_CH[x]_IN:SRC.MODE[i]="10", VAL[i]

b) Changes from normal mode to emergency mode, write to Reference Mode bit (RMO):

If this is necessary the write operation to DPLL_CTRL_0 cannot be prevented.

c) Changes of TRIGGER Number (TNU), STATE number (SNU) and Multiplier for TRIGGER (MLT) bit-fields:

Such changes should not be necessary, if not the write operation to DPLL_CTRL_0 cannot be prevented.

d) Changes of Adaption modes TRIGGER (AMT), STATE (AMS) bits

Activate AMT, AMS after power up, then activate/deactivate the adaptation mode by writing adapt data to the Physical Deviation (PD) of the Adapt and Profile Values of the TRIGGER (DPLL_ADT_T) or STATE (DPLL_ADT_S) memory location in respectively the RAM2 (DPLL_ADT_T.PD_T) and RAM1c (DPLL_ADT_S.PD_S).

e) Changes of Input Delay TRIGGER (IDT), STATE (IDS) bits

Enable or disable filter input data by writing to the Filter Enable (FLT_EN) bit of the corresponding TIM channel control register(TIM[i]_CH[x]CTRL).

f) Changes of Input filter position (IFP) bit

If such changes are necessary the write operation to DPLL_CTRL_0 cannot be prevented.

In case Workaround 1 can't be used:

Workaround 2

After writing to DPLL_CTRL_0: Check for direction change by evaluating the Backwards drive of SUB_INC1 (BWD1) bit of the DPLL Status register (DPLL_STATUS) when an inactive edge occurred on TRIGGER (use the TRIGGER inactive slope detected interrupt [TISI]) until both an active TRIGGER, STATE input signal has occurred.

The pulse corrections and pointer modifications of the direction change are operated correctly.

ERR011217: GTM: Numbers of Digital PLL real and virtual TRIGGER and STATE events for the last increment are updated with bits corresponding to CPU write operation instead of being updated with the previous number of recent events

Description: (GTM-IP-306)

According to the Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM) specification the Number of recent TRIGGER events register configuration DPLL_NUTC.WSYN = 1 makes the SYN_T variable containing the nominal TRIGGER increment value (NT) of the current increment (DPLL_NUTC. SYN_T) writeable, while the SYN_T_old variable of the DPLL_NUTC register (DPLL_NUTC.SYN_T_old) inherits the previous value of SYN_T. Differing from the specified behavior the actual hardware does not update the value of DPLL_NUTC.SYN_T_old with the previous value of DPLL_NUTC. SYN_T. Instead the DPLL_NUTC.SYN_T_old is updated with the corresponding bits of the write operation executed by the CPU.

Similarly, the Number of recent STATE events register configuration DPLL_NUSC.WSYN = 1 makes the SYN_S variable containing the nominal STATE increment value (NS) of the current increment (DPLL_NUSC. SYN_S) writeable, while the SYN_S_old variable of the DPLL_NUSC register (DPLL_NUSC.SYN_S_old) inherits the previous value of SYN_S. Differing from the specified behavior the actual hardware does not update the value of DPLL_NUSC.SYN_S_old with the previous value of DPLL_NUSC. SYN_S. Instead the DPLL_NUSC.SYN_S_old is updated with the corresponding bits of the write operation executed by the CPU.

Workaround: If the update of SYN_T/S_old is to be done as described in the specification, the register DPLL_NU(T/S)C.SYN_(T/S) must be read first, then the DPLL_NU(T/S)C. SYN_(T/S) can be used to modify the bits which are written to DPLL_NU(T/S)C. SYN_(T/S)_old.

ERR011216: GTM: Reset of the Digital PLL direction status by disabling the Digital PLL does not remove the unsolicited direction change in every case

Description: (GTM-IP-301)

Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM) may perform an unwanted direction change when it is operating in normal mode (DPLL_CTRL_0.RMO=0, DPLL_CTRL_1.SMC=0) and the direction of the TRIGGER signal is evaluated by comparing the TRIGGER Hold time Minimum value (THMI) with the duration between active and inactive slope of TRIGGER signal (DPLL_CTRL_1.IDDS = 0). In this configuration, when a direction change happens on the TRIGGER signal (which is not plausible, because the direction change may happen due to e.g. a disturbed signal) such a direction change performed by the DPLL should be removed.

The direction in which the DPLL is operating can be read out by the DPLL status register (DPLL_STATUS.BWD1). Disabling and then enabling the DPLL by setting DPLL_CTRL_1.DEN = 0 followed by DPLL_CTRL_1.DEN = 1 results in resetting the DPLL status register (DPLL_STATUS), so the backward bit of DPLL status register (DPLL_STATUS.BWD1) is cleared. However, this exercise does not remove the unsolicited direction change in every case and the BWD1 bit could be set to the unwanted direction again. The issue occurs when the DPLL has not received yet an active input signal on the STATE input after DPLL was enabled (DPLL_STATUS.FSD = 0) before the DPLL is disabled (DPLL_CTRL_1.DEN = 1 followed by DPLL_CTRL_1.DEN = 0 followed by DPLL_CTRL_1.DEN = 1) and switched to emergency mode (DPLL_CTRL_0.RMO = 1). The issue does not occur when at least one valid STATE event was detected after enabling DPLL (DPLL_STATUS.FSD = 1), or if the DPLL is not switched to emergency mode (DPLL_CTRL_0.RMO = 0) after the DPLL has been disabled/enabled.

The DPLL internal direction remains in current direction state while the backward drive bit of the DPLL status register (DPLL_STATUS.BWD1) is reflecting it's reset value during a toggle sequence (1->0->1) of the DPLL enable bit (DPLL_CTRL_1.DEN). At the end of the toggle sequence the backward drive bit of the DPLL status register (DPLL_STATUS.BWD1) returns to the state of the current internal direction.

Workaround: If the issue occurs under the described conditions the wrong direction could be corrected by:

1) Adding an additional input signal (active edge followed by inactive edge while the TRIGGER hold time minimum limit is not exceeded) to the TRIGGER input which switches the DPLL back to forward direction.

2) Switching the DPLL to the direction control mode (DPLL_CTRL_1.IDDS = 1) and to control the direction by setting the GTM input signal TIM0_IN6 to e.g. zero (forward direction). For combustion engine operation the DPLL direction can be controlled with TRIGGER direction (TDIR) and STATE direction (SDIR) signals having configured Timer Input Module 0 channel 6 input signal (TIM0_IN6) as direction signal TDIR (when MAP_CTRL.TSEL = 0).

ERR010764: GTM: Restoring of F2A read access to FIFO after GTM_HALT condition not functional

Description: (GTM-IP-278)

In the Generic Timer Module (GTM), the GTM_HALT signal is used to disable the clocks for debug purposes.

When the GTM_HALT signal is activated while the FIFO to ARU unit (F2A) sub-module is executing a read access to a FIFO channel buffer, the F2A read access is stopped and then restored after GTM_HALT is deactivated. The restoring of the F2A read access provides erroneous data.

Workaround: Do not halt the GTM during debug, or take into account that erroneous data can be read by the F2A when resuming operation.

ERR010884: GTM: Short FIFO IRQ in level mode when DMA hysteresis is enabled and DMA direction is read

Description: (GTM-IP-283)

When the Generic Timer Module (GTM) First In First Out Module (FIF0) channel interrupt (IRQ) Mode is configured to level mode (FIFO[i]_CHn_IRQ_MODE[IRQ_MODE] = 0b00), DMA hysteresis mode is enabled (FIFO[i]_CHn_IRQ_MODE[DMA_HYSTERESIS] = 0b1) and DMA direction is read (FIFO[i]_CHn_IRQ_MODE[DMA_HYST_DIR] = 0b0), the IRQ output level is set for just one clock period and not held as expected in level mode.

Workaround: Do not use IRQ level mode when DMA hysteresis is enabled and DMA direction is read.

ERR007190: GTM: Simultaneous Core and DPLL accesses to RAM Region 2 may lead to the DPLL reading erroneous data

Description: (GTM-IP-168)

A core or DMA access to the Generic Timer Module (GTM) RAM Region 2 at the same time as a Digital PLL (DPLL) accesses that memory may result in the DPLL reading erroneous data. As a result, calculations of the DPLL may be wrong and this may lead to loss of synchronization.

For example, if the DPLL accesses RAM Region 2 to read a value from the profile (for calculation of TRIGGER) and at the same time the core or DMA has initiated a second read/write operation of RAM Region 2 via the Automotive Electronics Interface (AEI), it is possible that the output data of the core or DMA RAM read/write request is used as read data for the DPLL initiated read instead of the Number of Real and Virtual Events to be Considered for the Current or Last Increment (syn_t and syn_t_old) values.

Workaround: Option 1: Synchronize core and DMA accesses to phases where the DPLL is not accessing RAM Region 2.

Example 1: synchronize DPLL RAM2 accesses to the TRIGGER signal using the TRIGGER Active Slope Interrupt (TASI) and checking that the RAM2 access is finished before the next active TRIGGER edge.

Example 2: use an Advanced Routing Unit (ARU) Connected Timer Output Module (ATOM) channel in Signal Output Mode PWM (SOMP) one shot mode to output a 200 SYS_CLK pulse when the DPLL calculation starts. Use a Timer Input Module (TIM) channel to generate an active edge interrupt based on the DPLL calculation synchronized ATOM output. With the ATOM's Counter Compare Unit 1 (CCU1) interrupt (200 SYS_CLKs later) the DPLL sub increment calculation should be complete. At start of the ATOM interrupt service routine check if any action calculation is ongoing in the DPLL by reading the Calculation of Actions In Progress (CAIP2) flag in the DPLL_STATUS register. If this flag is zero (0), RAM Region 2 access by the core and DMA are safe to do.

Option 2: allow core and DMA accesses in tooth profile phases where DPLL is not accessing RAM2.

For example: use a Multi-channel Sequencer (MCS) to calculate and set flags that indicate the non critical phases the tooth profile when the DPLL does not access RAM2.

ERR007087: GTM: The DPLL's Address Pointer Extension value is added to the Address Pointer when the Address Pointer Status bit is 0

Description: (GTM-IP-166)

If the Generic Timer Module (GTM) Digital PLL (DPLL) Address Pointer Extension field (APT_2b_EXT/APS_1c2_EXT) in the Address Pointer Trigger/State Synchronization Register (DPLL_APT_SYNC) is not zero during synchronization, it is added to the Address Pointer for Trigger/State (APT_2b/APS_1c2) in DPLL_APT/DPLL_APS registers regardless of the state of the status bits (APT_2b_STATUS/APS_1c2_STATUS) in DPLL_APT_SYNC.

Workaround: If the pointers should remain unchanged after synchronization, APT_2b_EXT/APS_1c2_EXT must be set to zero before the synchronization is performed.

ERR007191: GTM: The DPLL's SORI and TORI interrupts are not asserted

Description: (GTM-IP-169)

The Generic Timer Module (GTM) Digital PLL (DPLL) Trigger Out Of Range (TORI) and State Out of Range (SORI) interrupts are not set (1) when the following conditions are met:

- the upper 24 bits of Timebase Timestamp Channel 0 (TBU_TS0) are used as the input (DPLL_STATUS[LOW_RES]=1)
- the trigger/state input time stamps have an 8 times higher resolution than TBU_TS0, and the estimated time point value is multiplied by 8 (DPLL_CTRL1[TS0_HRT/S] = 0)

Workaround: In this configuration use a Timer Output Module (TOM) or ARU Connected TOM (ATOM) to generate an interrupt on the time out of TRIGGER/STATE.

With every TRIGGER/STATE edge, adapt the (A)TOM period to the current speed and reset Counter 0 (CN0). If CN0 is not reset by the next TRIGGER/STATE event, (A)TOM raises an edge interrupt at the end of the period.

ERR007083: GTM: The DPLL's SORI, TORI, MTI, and MSI interrupts may not be asserted

Description: (GTM-IP-161)

The Generic Timer Module (GTM) Digital PLL (DPLL) Missing Trigger(MTI), Missing State (MSI), Trigger Out Of Range (TORI) and State Out of Range (SORI) interrupts are not set (1) when the 3 following conditions are met:

- the upper 24 bits of Timebase Timestamp Channel 0 (TBU_TS0) are used as the input (DPLL_STATUS[LOW_RES]=1)
- the trigger/state input time stamps have an 8 times higher resolution than TBU_TS0 (DPLL_CTRL1[TS0_HRT/S] = 1)
- the upper three bits of TBU_TS0 are not equal to "000".

The DPLL Lock Status bits for SUBINC1 and SUBINC2 (LOCK1/2) and the MTI/MSI flags in the DPLL_STATUS register are incorrect if the above case occurs.

Workaround: Do not use the configuration DPLL_STATUS[LOW_RES]=1 with DPLL_CTRL1[TS0_HRT/S] = 1.

ERR010176: GTM: The DPLL_STATUS[BWD1/2] flags are not reset when DPLL is disabled and enabled again

Description: (GTM-IP-250)

When the Generic Timer Module (GTM) Digital PLL (DPLL) is disabled and enabled again (DPLL_CTRL_1[DEN] = 1 -> 0 -> 1), the following bits of the DPLL Status Register (DPLL_STATUS) may not be reset:

- Backwards drive of sub-increment 1 (BWD1)
- Backwards drive of sub-increment 2 (BWD2)

The minimum time in the disabled state (DPLL_CTRL_1[DEN] = 0), needed to correctly reset DPLL_STATUS register, is either

a) 120 clock cycles if there is no active input event processed, or

b) 860 clock cycles if there is an active input event processed.

Workaround: When the DPLL is disabled (DPLL_CTRL_1[DEN] = 1 -> 0), ensure there is at least a time of

a) 120 clock cycles (1,5us @80MHz or 1,2us @100MHz GTM clock frequency) if no active input event is processed/expected

b) 860 clock cycles (10,75us @80MHz or 8,6us @100MHz GTM clock frequency) if there is an active input event expected

before enabling the DPLL again (DPLL_CTRL_1[DEN] = 0 -> 1).

ERR006642: GTM: THVAL not available immediately after inactive trigger in DPLL

Description: (GTM-IP-152)

The Generic Timer Module (GTM) Digital PLL (DPLL) Measured TRIGGER hold time value (THVAL) is calculated correctly for each INVALID trigger slope, but this value is only stored into the THVAL memory location in RAM Region 1a with every new ACTIVE edge of the trigger signal.

Workaround: If the THVAL value is needed immediately with the inactive trigger edge it is necessary to calculate the THVAL value by using Timer Input Module Channels 0/1 (TIM_CH0/1) to obtain the active and inactive slopes in TIM input event mode (TIEM). With these timestamps the CPU is able to calculate the time span.

ERR007529: GTM: TIM overflow bit is not set and the signal level bit has inverse value when sent to ARU in some cases

Description: (GTM-IP-172)

When the Generic Timer Module (GTM) Timer Input Module (TIM) is in Timer Input Event Mode (TIEM, TIMn_CHx_CTRL[TIM_MODE] = 2), with Advanced Routing Unit (ARU) enabled (TIMn_CHx_CTRL[ARU_EN] = 1), and Input Signal Level high (ISL, TIMn_CHx_CTRL[ISL] = 1, the Overflow Bit (ACB1) might not be set and the signal level bit (ACB0) will be incorrect.

This error occurs when two input signals change in close proximity (faster than the ARU routing time), for example, an edge initiates an ARU transfer and one system clock before the ARU request is serviced the second input signal changes. Note that the Interrupt Request bit associated with the Overflow is set correctly.

Workaround: Workaround 1:

Use the TIM channel input filter to remove signal changes smaller than the ARU routing time by configuring the filter parameters for rising and falling edges (TIMn_CHx_FLT_FE/ TIMn_CHx_FLT_RE) with a delay which is greater than the ARU routing time.

Workaround 2:

Select the Edge Counter (TIMn_CHx_ECNT or TIMn_CHx_CNT) to be transferred in the ARU data to the Mutli Channel Sequencer (MCS) and use the MCS to reconstruct the correct TIM data as follows:

Last_CNT = -1 For each ARU_DATA If ARU_DATA(ACB1) ==0 If Last_CNT != -1 If Last_CNT +1 != ARU_DATA(CNT) Message(Hit on ERRATA: Detected overflow condition) ARU_DATA(ACB1) = 1 ARU_DATA(ACB0) = not ARU_DATA(ACB0) else Message(No signal level present yet, cannot apply workaround) Last_CNT = ARU_DATA(CNT)

ERR007086: GTM: TIM PWM and PIM modes may capture the wrong timestamp

Description: (GTM-IP-164)

When the Generic Timer Module (GTM) Timer Input Module (TIM) channel is configured with Counter Select (CNT_SEL) of the Channel Control Register (CTRL) set, and an input edge is detected by that channel before a rising edge on the clock, the Channel Counter Shadow Register (CNTS) will capture the value of Channel Count Register (CNT) instead of the timebase (TBU_TS0) in PWM measurement mode (TPWM) and pulse integration mode (TPIM).

Workaround: To avoid incorrect timestamp captures in the TIM PWM and PIM mode, the follow steps must be taken:

1) Select a TIM clock source which is identical to SYS_CLK.

2) Use the input event mode (TIEM) to capture TBU_TS0 for rising and falling input edges.

3) In TPWM mode, use CNT register as input (CNTS_SEL=0) with CMU_CLK source selected. With TBU_TS0 selected as the input to General Purpose Register 0 (GPR0), and with CNT selected as the input to General Purpose Register 1 (GPR1), calculate the correct timestamp: GPR0 – GPR1 + CNTS.

ERR008439: GTM: TOM and ATOM CM0, CM1 and CLK_SRC register updates may not be triggered

Description: (GTM-IP-209)

The trigger signal between the Generic Timer Module (GTM) Timer Output Module (TOM) or ARU Connected TOM (ATOM) submodules (e.g. signal TOM_TRIG_[i]) can be stored in a register at the module output to break long combinational paths. When this store register in place, it results in a delay of one system clock period of the trigger signal.

Between module instances TOM[i] / ATOM[i] and TOM[i+1] / ATOM[i+1], when there is a store register in the the trigger path, this trigger is only recognized by the channel of TOM[i+1] / ATOM[i+1] if the channel is running from a source identical to the system clock (i.e. the selected Clock Management Unit Fixed Frequency Clock (CMU_FXCLKx) or Clock Management Unit Clock (CMU_CLKx) period is the system clock (SYS_CLK) 1). If another frequency is chosen to clock the TOM[i+1] / ATOM[i+1] channel, the trigger is not recognized by the Compare Registers (CM0/CM1) or the Clock Source (CLK_SRC) register.

Workaround: TOM Workaround 1:

When there is a register in the trigger path between TOM[i] and TOM[i+1], the channel of TOM[i+1] that should be triggered has to use a clock of period identical to SYS_CLK period. The configuration of the TOM outputs differs between devices, in some cases each TOM has the save trigger register, in some devices every second TOM module has the register. Check the GTM specification for the configuration applicable to the device in use.

TOM Workaround 2:

On TOM[i+1] configure a redundant channel to trigger another channel of TOM[i+1] as it was configured on TOM[i] to trigger the other channel. Then start TOM[i] and TOM[i+1] synchronously by using the Time Base Unit (TBU) comparator of the TOM Global Control (TGCx) unit (TOM[i]_TGC[y]_ACT_TB register).

ATOM Workaround 1:

When there is a register in the trigger path between ATOM[i] and ATOM[i+1], the channel of ATOM[i+1] that should be triggered has to use a clock of period identical to SYS_CLK period. The configuration of the ATOM outputs differs between devices, in some cases each ATOM has the save trigger register, in some devices every second ATOM module has the register. Check the GTM specification for the configuration applicable to the device in use.

ATOM Workaround 2:

On ATOM[i+1] configure a redundant channel to trigger another channel of ATOM[i+1] as it was configured on ATOM[i] to trigger the other channel. Then start ATOM[i] and ATOM[i+1] synchronously by using the Time Base Unit (TBU) comparator of the ATOM Global Control (AGC) unit (ATOM[i]_AGC_ATC_TB register).

ERR011191: GTM: TRIGGER signal will not cause direction change of the Digital PLL when TRIGGER hold minimum value is zero

Description: (GTM-IP-300)

When the Digital PLL (DPLL) sub-module of the Generic Timer Module (GTM) direction control is being set up using the TRIGGER input signal in DPLL Control Register 1 (DPLL_CTRL_1.IDDS = 0, DPLL_CTRL_1.SMC = 0) and the TRIGGER hold time minimum value is set to zero (DPLL_THMI = 0) the DPLL direction is not changed (direction status is reflected in DPLL_STATUS.BWD1). Instead, when DPLL_THMI = 0, the latest direction status is held, it means in case of forward operation state (DPLL_STATUS.BWD1 = 0) the direction will stay in forward state, in case of backward direction of operation (DPLL_STATUS.BWD1 = 1) the direction stays held backward

Workaround: 1. In case of Input direction strategy is controlled by comparison of TRIGGER hold time minimum value with the duration between valid and invalid slope of TRIGGER signal and if the DPLL is operating in forward direction (DPLL_STATUS.BWD1 = 0) the direction can be kept by setting the TRIGGER hold time minimum value to zero (DPLL_THMI = 0). If the DPLL is operating in backward direction the direction can be switched to forward by setting the DPLL_THMI value to the biggest possible value DPLL_THMI = 0x00FFFF. This should set the direction back to forward.

2. Use different mechanism of direction control. In this case the direction can be controlled by Timer Input Module 0 channel 6 input signal (TIM0_IN6) and Timer Input Module 0 channel 0 (TIM0_CH0) is selected as TRIGGER output signal by configuration in TIM0 Input mapping module register (MAP_CTRL.TSEL = 0)

In both cases the direction evaluation is done with the inactive edge of the TRIGGER input signal. The TRIGGER input signal must be active even in emergency mode to handle the direction changes correctly. If the TRIGGER input signal is not in a usable condition the necessary input signal sequence can be generated by a direct modification of the input signal of TIM0_CH0 with the use of TIM0 Configuration registers (TIM[0]_IN_SRC.MAKE_0/VAL_0 and TIM[0]_CH[0]_ECTRL.USE_LUT)

ERR008429: GTM: Unexpected TIM CNTS register reset in TPWM OSM mode

Description: (GTM-IP-205)

When a Generic Timer Module (GTM) Timer Input Module (TIM) channel is configured for One Shot PWM Measurement Mode (TIM[i]_CH[x]_CTRL[TIM_MODE]=0, TIM[i]_CH[x]_CTRL[OSM]=1) an active edge will stop the measurement. If this active edge is followed by an inactive edge 1 GTM system clock later, the Counter Shadow register (TIM[i]_CH[x]_CNTS) is unexpectedly reset.

Workaround: Configure the TIM channel to use a Clock Management Unit (CMU) clock source slower than the GTM system clock so that any subsequent active edge is captured at a time after 1 GTM system clock, and/or enable the channel filter with parameters set so that two consecutive edges will be filtered out.

ERR006640: GTM: Valid edge after Timeout event ignored by TIM

Description: (GTM-IP-150)

When a Generic Timer Module (GTM) Timer Input Module (TIM) timeout event triggers an Advanced Routing unit (ARU) write request with timeout information “timeout detected without valid edge” (ARU Control Bit 2 (ACB2)=1 and ACB1=0), and this request is acknowledged by the ARU at the same time as a new valid edge occurs, the valid edge is neither acknowledged by setting the bits ACB2=1 and ACB1=1 (timeout detected with subsequent valid edge detected) within the acknowledged transfer nor acknowledged by setting up a subsequent ARU write request for the new valid edge with ACB2=0 and ACB1=0 (valid edge detected).

Workaround: The workaround for this issue requires an additional plausibility check within the Multi Channel Sequencer (MCS) or CPU via FIFO:

Step 1) store the received data (ARUDATA(47:0)) and ACB0 in temporary variables.

Step 2) if an ARU transfer with ACB2=1 and ACB1=0 is received also check whether the previously received ARUDATA(47:0) and ACB0 values are the same:

If they are not the same values then a timeout with subsequent valid edge has occurred, which means ACB1 must be corrected to 1.

ERR007088: GTM: When ATOM is in SOMP mode the SR0/SR1 registers could be updated twice in one PWM period

Description: (GTM-IP-167)

When the Generic Timer Module (GTM) Advanced Routing Unit (ARU) Connected Timer Output Module (ATOM) is in Signal Output Mode PWM (SOMP) mode with the ARU enabled, and the channel is configured to be updated by the preceding channel (ARU_EN=1 and RST_CCU0=1 in ATOM[i]_CH[x]_CTRL), an update of the channel’s Shadow Registers (ATOM[i]_CH[x]_SR0/SR1) via ARU is requested when Counter 0 (ATOM[i]_CH[x]_CN0) reaches Compare 0 (ATOM[i]_CH[x]_CM0).

In this case, if CN0 reaches CM0, CN0 is not reset and continues counting until it is reset by the trigger of the preceding channel. As a result, the ATOM channel updates the SR0/SR1 registers after the update of CM0/CM1, which is not synchronous with the counter reset. Depending on the time between CM0 and the value of CN0 when it was reset by the trigger, the SR0/SR1 registers may be updated twice in one period.

Workaround: If new data via ARU is provided by Multi Channel Sequencer (MCS) ensure, through software, that only one value per period of new data for SR0/SR1 register can be read. For example, this can be achieved by starting a ‘master period’ which triggers the reset of CN0 on a time base value and provides the start value and period to the MCS. The MCS can then calculate a minimum time it must wait before providing new ARU data.

If the above workaround is unsuitable (for example if new data via the ARU is provided by the FIFO) do not use SOMP mode with ARU_EN=1 and RST_CCU0=1.

ERR006410: GTM: Write to ATOM_CH_CTRL sets WRF if CCU0 compare match has already occurred, but CCU1 compare match is pending, in ATOM SOMC mode

Description: (GTM-IP-140)

When the Generic Timer Module (GTM) ARU Connected Timer Output Unit (ATOM) is configured in Signal Output Mode Compare (SOMC) mode, with the Advanced Routing Unit (ARU) Enabled, the capture/compare strategy is ‘Serve Last’ (ATOM[i]_CH[x]_CTRL.B.ACB42

= 0b1xx), with Counter Compare Unit 0 (CCU0) matched but the CCU1 match is pending, a write access to the ATOM Channel Control register (ATOM[i]_CH[x]_CTRL) will set the “Write Request of CPU Failed for the Last Update” (WRF) field in the Channel Status Register (ATOM[i]_CH[x]_STAT) independently of the status of the CPU Write Request Bit for Late Compare field (WR_REQ) in ATOM[i]_CH[x]_CTRL.

Workaround: If ATOM[i]_CH[x]_CTRL is written during an active ‘Serve Last’ comparison without the intention of setting WR_REQ, write to ATOM[i]_CH[x]_CTRL and then reset/clear WRF in ATOM[i]_CH[x]_STAT by writing a ‘1’.

ERR008438: GTM: Wrong signal level when TIM mode is changed from TBCM to any other mode

Description: (GTM-IP-204)

When a Generic Timer Module (GTM) Timer Input Module (TIM) channel is disabled (TIM[i]_CH[x]_CTRL[TIM_EN]=0) and in Bit Compression Mode (TBCM, TIM[i]_CH[x]_CTRL[TIM_MODE]=0x4) while the corresponding channel input is high, a mode change will not update the input signal level indication bit (Least Significant Bit of TIM[i]_CH[x]_GPR0[ECNT]).

Workaround: If the input signal level information previously captured in TIM[i]_CH[x]_GPR0[ECNT] is sent to another submodule by the ARU, and is then used by the other submodule to make a decision on what action to take next, do not move from TBCM to any other TIM mode while the channel is disabled.

In general it would be unusual to use one TIM channel for a TBCM function and then reuse the channel for another function, therefore this should not have any implications for most use cases.

ERR008047: HSM: Nexus class 3 read/write accesses may be blocked by SMPU

Description: The Hardware Security Module (HSM) e200z0h core Nexus debug interface is not defined as a logical bus master in the HSM System Memory Protection Unit (SMPU). As a result, Nexus Class 3 read/write accesses will be blocked if the HSM SMPU is enabled.

Workaround: Either ensure that the HSM SMPU is disabled prior to performing any HSM Nexus class 3 read/write accesses. Alternatively, avoid HSM Nexus 3 read/write accesses and instead perform Nexus class 1 accesses through the HSM e200z0 core (stop the core and execute instructions on the core to perform all memory accesses).

ERR008951: 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.

ERR007109: I2C: In master receive mode, data remains latched in I2C data I/O register (IBDR) until new data is received

Description: When the Inter-Integrated Circuit (I2C) is configured in master mode and receiving data from a slave which is transmitting data bytes on an irregular basis, there is no way for the master to know if the data received in the I2C data Input/Output register (IBDR) is the old latched data or the new data received from the slave.

Workaround: When slave is configured to transmit data on an irregular basis, in other words intermittently, it should not send 2 consecutive bytes with the same data. When 2 consecutive data bytes are different, a dummy read of the I2C data I/O register (IBDR) can be initiated before the actual read. These 2 bytes can be compared to know if it is the new data or the old data.

ERR007433: JTAGM: Nexus error bit is cleared by successful RWA

Description: The JTAG Master module status register includes a Nexus error status bit (JTAGM_SR[Nexus_err]) that indicates the status of the last Nexus Read/Write Access (RWA) command. Once this information is latched, it can only be cleared by performing a successful RWA transaction via the same core that caused the error. In addition, if a RWA transaction is performed by a different core, the error bit will not be cleared and it is not possible to determine if the access by the second core RWA was successful or generated another error.

In general, this bit should only be set when the Nexus RWA accesses non-existent or protected memory spaces.

Workaround: If the status information is required from a specific core, the user software or tool should read the error bit (ERR) of the e200zx core's Nexus Read/Write Access Control/Status register. To avoid setting the error bit, do not perform illegal memory accesses.

ERR008935: JTAGM: write accesses to registers must be 32-bit wide

Description: The JTAG Master module (JTAGM) supports only 32-bit write accesses to its registers. A byte write access will be converted into a 32-bit write with the other bytes values at 0x0.

Workaround: Perform only 32-bit write accesses on JTAGM registers. Do not use byte writes.

ERR007398: LBIST: Partitions 2, 4, and 5 during offline self-test mode not functional

Description: Offline Logic Built-in Self-Test (LBIST) executed by the Self-Test Control Unit (STCU) is not functional for the following partitions:

LBIST Partition	Partition Name	Partition Contents
2	Peripheral Shell	MPU, PBRIDGE_B, SWT_3, LINFLEX_2, LINFLEX_15, SENT_1, PSI5_1, IIC_1,

		DSPI_2, DSPI_3, DSPI_5, CRC_1, SARADC_1, SARADC_2, SARADC_3, SARADC_5, SARADC_6, SARADC_7, SARADC_8, SARADC_9, SARADC_10, SDADC_1, SDADC_3, SDADC_5, SDADC_7, SDADC_9, CMU
4	Peripheral Shell Masters	eDMA, DMACHMUX, DMA TCD RAM, FEC, RAM FEC, FlexRay, INTC, FCCU
5	Peripherals	SDADC_0, SDADC_2, SDADC_4, SDADC_6, SDADC_8, SARADC_0, SARADC_4, SARADC_B, BAM ROM, CRC_0, DSPI_0, DSPI_1, DSPI_4, DSPI_6, DSPI_12, IIC_0, PSI5_0, SENT_0, REG_PROT, LINFLEX_0, LINFLEX_1, LINFLEX_14, LINFLEX_16, MCAN, TTCAN, PBRIDGE_A, MEMU, WKPU, NAR, SPU, PIT

Workaround: Do not attempt to run LBIST on partitions 2, 4 and 5 during offline Self-test mode. Use the on-line mode to test these partitions.

ERR007274: 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.

ERR008731: LINFlexD: Corruption of Received Rx data in UART mode

Description: The LINFlexD module is driven by two different clocks. The transmit/reception logic is controlled by the module clock (LIN_CLK) and register accesses are controlled by the peripheral bus clock (PBRIDGEx_CLK).

In the Universal Asynchronous Receive/Transmit (UART) Mode, the resynchronization of the access of the Buffer Data (BDRL and BDRM) registers may reset the internal counter which generates the baud sampling signal to receive the incoming bits. This will occur only if LIN Integer Baud Rate (LINIBRR) register is greater than 1.The data being received may not be correctly sampled:

- Sampling point will be anticipated by maximum of 1 bit period
- The final data may be shifted by one bit in the data BDRM register.

Workaround: When using the LINFlex module in UART mode:

- If possible, LINIBRR should be configured with value ‘1’. This is possible in cases where the baud rate has ratio of 4,5,6,8,16 with respect to clock LIN_CLK. For example with LIN_CLK at 80MHz this allows 20M baud down to 5M baud reception rate.
- If the baud rate is not compatible with a LINIBRR value of ‘1’,
  - In UART full duplex mode, application software must manage systematic detection of the failure and on detection, request that the receive data be resent. See possible implementation below
  - In UART half-duplex mode (receive mode only), systematic detection can be implemented as above. Alternatively, software or DMA should ensure that BDRM is read before the start of the next frame reception. Reception information is available through interrupt, DMA request, or the UART status register (UARTSR). This information is available 5 times the LINIBRR[IBR] period of the LIN_CLK before the completion of the STOP bit reception. Further delay may be available depending on delay in between frame reception.

Systematic detection can be performed by using a ‘0’ logic parity bit and enable the generation of the Framing Error Interrupt. This is done by programming UARTCR[PCE] = ‘1’, UARTCR[PC1] = ‘1’, UARTCR[PC0] = ‘0’, and LINIER[FEIE] = ‘1’

Depending on the timing of the possible glitch occurrence, detection will be indicated by

- Either the setting of UARTSR[FE] bit, in other words, a Framing Error which generates a Framing Error Interrupt,
- Or the setting of the noise flag (UARTSR[NF]). Note: No interrupt is generated in this case.

ERR008561: LINFlexD: Corruption of Tx data in LIN mode with DMA feature enabled

Description: The LINFlexD module is driven by two different clocks. The transmit/reception logic is controlled by the module clock (LIN_CLK) and register accesses are controlled by the peripheral bus clock (PBRIDGEx_CLK). In LIN mode, the re-synchronization of the “Idle on bit

error" between the two clocks may cause the Direct Memory Access (DMA) Finite State Machine inside the LINFlexD module to move to the idle state while a transmission is in process. This unwanted idle state transition could lead trigger a new DMA request, potentially overwriting the Buffer Identifier Register (BIDR) and the Buffer Data Registers (BDRL and BDRM).

Workaround: Do not enable the "Idle on bit error" of LIN Control Register 2 (ILINCR2[IOBE] = 0). Instead of using the "Idle on bit error", use the bit error interrupt of LIN Interrupt Enable Register (LINIER[BEIE] = 1) to trigger an Interrupt service routine and force the LIN into idle mode through software if needed.

ERR007269: LINFlexD: Erroneous timeout interrupt could be generated by LIN in master mode

Description: When the LINFlexD (Local Interconnect Network module) is configured in LIN Master mode by setting the Master Mode Enable bit of the LIN Control Register 1 (LINCR1[MME]=1) and the Time-out counter mode bit in the LIN Time-Out Control Status Register is cleared (LINTCSR [MODE]=0), if the LIN Master transmits an header and expects a response from a slave, it is unable to update the OC2[7:0] (Output Compare Value 2) bits in the LINOCR (LIN Output Compare Register) with a timeout value called 'Response Timeout Value', which is used to trigger a timeout interrupt in case no response is received.

Consequently, an erroneous or no timeout interrupt will be generated by the LIN timer.

Workaround: In order to generate a Response Timeout, and consequently an interrupt, the LIN Master mode should only be used with the MODE (Time-out counter mode) bit in the LINTCSR (LIN Time-Out Control Status Register) set.

In addition, the software must program the OC2[7:0] (Output Compare Value 2) bits in the LINOCR (LIN Output Compare Register) with the value of Header Duration plus Response Timeout value plus the LINTCSR CNT[7:0] before asserting the HTRQ (Header Transmission Request) value in the LINCR2 (LIN Control Register 2).

ERR007228: LINFlexD: Erroneous transmission in LIN master mode for payload greater than eight bytes

Description: When the LINFlexD module is configured as follows:

- LIN (Local interconnect network) master mode is enabled by setting the MME (Master Mode Enable) bit in the LINCR1 (LIN Control Register 1);

if an extended frame is transmitted (DFL (Data Field Length) bits in BIDR (buffer identifier register) is bigger than 7) and the data buffer is refilled immediately after the DBEF (Data Buffer Empty Flag) bit in the LINSR (LIN Status Register) is set, then the DATA7 (DATA byte 7) field of BDRM (Buffer Data Register Most Significant) register is overwritten by the received data byte and sent instead of the desired value.

Workaround: Once the DBEF (Data Buffer Empty Flag) in LINSR (LIN Status Register) is set, add a delay of 3 sample time duration before refilling the data buffer and then clear the DBEF flag.

Note: One sample time duration is (1/16*Baud Rate) seconds.

ERR008933: 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.

ERR008526: LINFlexD: LIN or UART state may be incorrectly indicated by LINSR[LINS] bitfield

Description: The Local Interconnect Network (LIN) or Universal Asynchronous Receiver/Transmitter (UART) state is shown in the read only LIN state bits in the LIN Status Register (LINSR[LINS]). Whenever the LinFLEXD (in either LIN or UART mode) updates these state bits, there is a possibility that the wrong LIN or UART state is indicated for one Peripheral Bridge Clock (PBRIDGEx_CLK) cycle. If software is asynchronously polling the LIN or UART state bits, the state read by the software may be incorrect.

Workaround: The incorrect state in the LINSR[LINS] bit-field is auto-corrected in the next PBRIDGEx_CLK cycle. Therefore, any software polling the LINSR[LINS] bit-field should read the bit-field twice and confirm that the back to back reads are the same value before taking further action based on the state.

ERR008573: LINFlexD: Pre-mature header/response timeout in LIN mode

Description: The LINFlexD instance is driven by two different clocks. The transmit/reception logic is controlled by the module clock (LIN_CLK) and register accesses are controlled by the peripheral bus clock (PBRIDGEx_CLK). The PBRIDGEx_CLK is used to control timer logic during header/response reception, while the LIN_CLK is used to control the header/response reception. In LIN mode, the resynchronization between the two clocks may cause the pre-mature setting of the OCF bit (Output Compare Flag) of LIN Error Status Register (LINESR).

Depending on LIN Timeout, Control and Status Register, IOT = Idle On TimeOut (LINTCSR[IOT]) settings, effects of the pre-mature setting may be:

- if LINTCSR[IOT] is set , termination of the data reception
- if LINTCSR[IOT] is not set, generation of spurious timeout event, though the reception will continue.

Workaround: Always configure LINTCSR[IOT] as ‘0'. In case of a time out event, the spurious event can be detected by comparing the LINOCR[OC1] and LINOCR[OC2] values with the timer value in LINTCSR[CNT]. The difference should not be more than one bit time period.

ERR007297: LINFlexD: Response timeout values is loaded in LINOCR[OC2] field instead of LINOCR[OC1]

Description: In the LINFlex module, the response timeout value calculated by hardware is loaded onto the OC2[7:0] (Output Compare 2) bits of LINOCR (LIN Output Compare Register) instead of being loaded into the OC1[7:0] (Output Compare 1) bits of the same register as stated in the documentation.

This applies when the Time-out counter mode is enabled by clearing the MODE (Time-out counter mode) bit in the LINTCSR (LIN Timeout Control Status Register).

Workaround: Expect that OC2[7:0] (Output Compare 2) bits are loaded by hardware with the response timeout value when the MODE (Time-out counter mode) bit in LINTCSR is cleared.

ERR008970: 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)

ERR007589: LINFlexD: Spurious timeout error when switching from UART to LIN mode or when resetting LINTCSR[MODE] bit in LIN mode

Description: If the LINFlexD module is enabled in Universal Asynchronous Receiver/Transmitter (UART) mode and the value of the MODE bit of the LIN Timeout Control Status register (LINTCSR) is 0 (default value after reset), any activity on the transmit or receive pins will cause an unwanted change in the value of the 8-bit field Output Compare Value 2 (OC2) of the LIN Output Compare register (LINOCR).

If the LINFlexD module is enabled in LIN mode and the value of the MODE bit of the LIN Timeout Control Status register (LINTCSR) is changed from ‘1’ to ‘0’, then the old value of the Output Compare Value 1 (OC1) and Output Compare Value 2 (OC2) of the LIN Output Compare register (LINOCR) is retained.

As a consequence, if the module is reconfigured from UART to Local Interconnect Network (LIN) mode, or LINTCSR MODE bit is changed from ‘1’ to ‘0’, an incorrect timeout exception is generated when the LIN communication starts.

Workaround: If the LINFlexD module needs to be switched from UART mode to LIN mode, before writing UARTCR[UART] to 1, ensure that the LINTCSR[MODE] is first set to 1.

If the LINFlexD module is in LIN mode and LINTCSR[MODE] needs to be switched from 1 to 0 in between frames, the LINOCR must be set to 0xFFFF by software.

ERR008602: LINFlexD: Tx through DMA can be re-triggered after abort in LIN/UART modes or can prematurely end on the event of bit error with LINCR2[IOBE] bit being set in LIN mode

Description: The LINFlexD module is driven by two different clocks. The transmit/reception logic is controlled by the module clock (LIN_CLK) and register accesses are controlled by the peripheral bus clock (PBRIDGEx_CLK). Due to possible synchronization issue between the two clock domains, there is a possibility that DMA transmission get stuck due to DMA Finite State Machine doesn’t go into idle. This may occur in one of the following conditions:

- if an abort request is triggered (LINCR2[ABRQ]=1) in LIN or UART modes
- if idle on bit error feature is enabled (LINCR2[IOBE]=1) in LIN mode, and a bit error occurs.

DMA state machine will not generate any transaction, waiting for data transmission flag LINSR[DTF] to be set which will never occur.

Workaround: If DMA is used:

- Bit error interrupt should be enabled through LINEIER[BEIE]. When an bit error interrupt is triggered, the interrupt service routine must either reset the DMA Tx channel enable (DMATXE) and the DMA Rx channel enable (DMARXE) registers
- if an abort is requested (LINCR2[ABRQ]=1) in LIN/UART mode, either reset DMATXE/ DMARXE of LINFlexD after writing LINCR2 [ABRQ]

ERR006350: LINFlexD: WLS feature cannot be used in buffered mode

Description: The Flexible Local Interconnect Network (LINFlex) module may not operate correctly if the Special Word Length (WLS) for enabling 12-bit data length is selected in the UniversalAsynchronous Receiver/Transmitter (UART) Mode Control Register (UARTCR) and the module is configured in the receive buffered mode.

Workaround: When WLS mode is required, always use the First In, First Out (FIFO) mode of the UART LINFLEX module by setting the Receive FIFO/Buffer mode bit of the UARTCR (UARTCR[RFBM]=1). In addition, the UART word length bits must be set (UARTCR[WL0 = 0b1] and UARTCR[WL1] = 0b1).

ERR010557: MC_CGM: Auxiliary clock divider may get stuck if programmed to divide by 2 and a reset occurs during operation

Description: A reset of any type (except for power-on) when the Auxiliary Clock 0 Divider 0 in the Clock Generation Module (MC_CGM) is programmed for divide by 2 (MC_CGM_AC0_DC0[DIV]=0b0001) and is enabled (MC_CGM_AC0_DC0[DE]=1) and the clock source is selected for Phase Lock Loop PLL0 (MC_CGM_AC0_SC[SELCTL]=0b0010) may cause the divider to get stuck. A stuck clock divider will keep the corresponding MC_CGM_DIV_UPD_STAT[AUX0_UPD_STAT] flag set, stop the clock divider output clock, and prevent MC_CGM_AC0_DC0 from being reprogrammed. The other auxiliary clock dividers have a default reset of MC_CGM_ACn_DCx[DE]=0 (disabled) and are not affected.

Workaround: Avoid programming Auxiliary Clock 0 Divider 0 to a divide by 2 to prevent the possibility of a stuck divider.

A stuck auxiliary clock divider can be detected by reading a stuck high MC_CGM_DIV_UPD_STAT[AUX0_UPD_STAT] flag. A stuck clock divider output clock can also be detected using the corresponding Clock Monitor Unit (CMU).

Recovery from a stuck clock divider is by a change in the clock source selector by programming MC_CGM_AC0_SC[SELCTL]. Recovery is also by a power-on reset. A power-on reset back to the POWERUP reset phase is also possible by software without affecting device voltage by writing PMCDIG_VD_UTST[VD_UTST[5:0]] = 6b011110 (POR085_C) followed by setting PMCDIG_MCR[USER_SELF_TEST_EN].

To avoid a clock divider issue, execute the following software sequence after each non power-on reset for Auxiliary Clock 0 Divider 0 configured to divide by 2 and sourced from PLL0.

• Disable the clock divider by clearing the clock divider enable to 0 (MC_CGM_AC0_DC0[DE] = 0). Note that the clock divider enable will already be 1 since the default reset value of the clock divider enable is 1.

• Change the clock divider source selection (MC_CGM_AC0_SC[SELCTL] = 0b0000). Note that the clock divider source selection is not reset during the reset sequence.

• Re-select the configured PLL0 as the clock divider source selection (MC_CGM_AC0_SC[SELCTL] = 0b0010).

• Re-enable the clock divider by setting the clock divider enable to 1 (MC_CGM_AC0_DCx[DE] = 1).

Note: For any auxiliary clock divider, disable the clock divider (MC_CGM_ACn_DCx[DE] = 0) before switching the corresponding auxiliary clock divider source select (MC_CGM_ACn_SC[SELCTL]) or changing the configuration of a clock source which has already been selected. Changing clock source configuration or clock source select while the auxiliary clock divider is enabled and configured for divide by 2 may cause the divider to get stuck.

ERR010555: MC_CGM: Auxiliary Clock Divider Update Status register bits not documented

Description: The Auxiliary Clock Divider Update Status bits (AUXn_UPD_STAT) in the Divider Update Status register (MC_CGM_DIV_UPD_STAT) in the Clock Generation Module (CGM) are not documented. There is one update status register bit for each of the auxiliary divider groups implemented in the device. Thus, there are 9 AUXn_UPD_STAT register bits where n={10, 9, 8, 7, 6, 5, 2, 1, 0} corresponding to MC_CGM_DIV_UPD_STAT register bits {21, 22, 23, 24, 25, 26, 29, 30, 31} with default reset values of 0b0.

Register Bit	Bit Description
0	System Clock Divider Update Status
1:20	Reserved
21	Auxiliary Clock 10 Divider Update Status
22	Auxiliary Clock 9 Divider Update Status
23	Auxiliary Clock 8 Divider Update Status
24	Auxiliary Clock 7 Divider Update Status
25	Auxiliary Clock 6 Divider Update Status
26	Auxiliary Clock 5 Divider Update Status
27	Reserved
28	Reserved
29	Auxiliary Clock 2 Divider Update Status
30	Auxiliary Clock 1 Divider Update Status
31	Auxiliary Clock 0 Divider Update Status

The bit description for the Auxiliary Clock n Divider Clock status is:

Value	Description
0	the configuration for at least one auxiliary clock n divider is not being updated

1	the configuration for at least one auxiliary clock n divider is being updated

In addition, the AUXn_UPD_STAT bits, including the SYS_UPD_STAT bit, are read only.

Workaround: Reading the MC_CGM_DIV_UPD_STAT[AUXn_UPD_STAT] bits are not required but may be used to indicate when the auxiliary dividers have been successfully updated after a change in divider configuration.

ERR007103: MC_CGM: Incorrect cause for the latest clock source switch may be reported by the CGM if a safe mode request arrives when the system clock is the IRC

Description: If the current system clock source is the Internal RC oscillator (IRC) as reported in the Clock Generation Module System Clock Select Status Register System Clock Source Selection field (CGM_SC_SS.SELSTAT = 0b0000) and the Clock Generation Module System Clock Select Status Register Switch Trigger Cause shows the cause for the latest clock switch as MC_ME succeeded (CGM_SC_SS.SWTRG = 0b001) indicating that a successful Mode Entry mode change was the cause of the last clock change then the CGM_SC_SS.SWTRG will incorrectly continue to show the cause for the latest clock switch as MC_ME succeeded after a safe mode request is generated. If a subsequent safe mode request is generated CGM_SC_SS.SWTRG switches to report the correct status value of 0b100 (switch to system clock source 0 due to SAFE mode request or reset succeeded).

Workaround: If the CGM_SC_SS.SELSTAT shows the system clock as IRC (0b0000), then software should check the Mode Entry Global Status register Current Mode field (ME_GS.CURRENT_MODE) and the Mode Entry Interrupt Status Register Safe mode Interrupt (ME_IS.I_SAFE) to establish the cause of the switch.

ERR010668: MC_CGM: Progressive Clock Switching (PCS) equations

Description: The Reference Manual is inconsistent in the Progressive Clock Switching (PCS) equations. Here are a consistent set of equations.

k = 0.5 + SQRT[0.25 - (2*(1-fSRC/16MHz) / d)]

d = CGM_PCS_DIVCn[RATE] / 1000

PCS_DIVCn[INIT] = d*k*1000

PCS_DIVSn[DIVS] = (1 + d*k*(k+1)/2) * 1000 – 1

PCS_DIVEn[DIVE] = (fSRC / 16MHz) * 1000 – 1

Workaround: Refer to the AN4812 “Initializing the MPC5777M Clock Generation Module and Progressive Clock Switching Feature” application note for more details.

ERR010559: MC_CGM: The device can become stuck in reset when reset occurs during a system clock switch to a PLL

Description: A reset of any type (except for power-on) when the source of the system clock dividers is being switched to an enabled Phase Lock Loop (PLL) source (either PLL1 or PLL0) by programming Mode Entry Module Mode Configuration Register (ME_MC_*_MC[SYSCLK]=0b0010 or 0b0100) may cause the system clock divider alignment logic to get stuck and thus prevent the exit from reset. The stuck in reset is only possible if there are more than one enabled system clock divider not programmed to divide of a common divide multiple. Thus, system clock dividers which are disabled or programmed to divide by 1 are not affected.

When a system clock divider in the Clock Generation Module (CGM) is programmed for divide by 2 (MC_CGM_SC_DCn[DIV]=0b0001) and is enabled (MC_CGM_SC_DCn[DE]=1), the same reset during a source switch to a PLL may cause the divider to get stuck. A stuck clock divider will keep the MC_CGM_DIV_UPD_STAT[SYS_UPD_STAT] flag set, stop the clock divider output clock, and prevent MC_CGM_SC_DCn from being reprogrammed.

Workaround: Recovery from the stuck in reset case or stuck clock divider is by a power-on reset. A power-on reset back to the POWERUP reset phase is also possible by software without affecting device voltage by writing PMCDIG_VD_UTST[VD_UTST[5:0]] = 6b011110 (POR085_C) followed by setting PMCDIG_MCR[USER_SELF_TEST_EN].

Prevent a stuck clock divider by avoiding a system clock divider configuration of divide by 2 for an enabled system clock divider during the system clock source switch to a PLL. A stuck system clock divider can be detected by reading a stuck high MC_CGM_DIV_UPD_STAT[SYS_UPD_STAT] flag. A stuck clock divider output clock can also be detected using a corresponding Clock Monitor Unit (CMU).

Prevent a stuck in reset condition or a stuck divider by avoiding resets during the window of time that a mode switch is used to switch the system clock source to a PLL.

Also a stuck in reset condition or a stuck divider are prevented if avoiding the use of a PLL as the source of a system clock.

Prevent a stuck in reset condition by configuring the PLL maximum clock output frequency to 50MHz and assure that the clock dividers are either disabled or selected to a divide by 1 during the system clock source switch to a PLL. After the switch to the PLL is complete, then configure the system clock dividers as desired while maintaining the lower PLL output clock frequency.

ERR007211: MC_ME: Core register IAC8 is cleared during a mode change when the core is reset

Description: If a core is reset (ME_CADDR[0,1,2].RMC =1) in the Core Address register during a Mode Entry module (MC_ME) mode change then the Instruction Address Compare 8 (IAC8) register within the core which receives the reset will be cleared. In this implementation IAC8 is used as the Security watchdog service address. If a watchdog time-out occurs after this mode change and no valid service address exists, the core will attempt to execute code from the invalid address potentially resulting in an exception.

The watchdog (SWT) associated with that core is not reset by this change and retains its configuration. If fixed address execution is configured by the Service Mode in the software watchdog control register (SWT_CR.SMD= 0b10) when IAC8 is cleared to 0, it will not be possible to update IAC8 with the correct value. For other service modes the IAC8 register will be cleared to 0, but can be updated.

Workaround: If the software watchdog mode is in fixed address execution (SWT_CR.SMD= 0b10), do not reset the corresponding core upon mode change. For all other modes, IAC8 must be updated by software immediately after the mode transition completes.

ERR007911: MC_ME: Decorated accesses not supported to core local memories during reset

Description: Resetting a core through a Mode Entry module (ME) mode change, while a decorated access is being performed to that core's local Instruction (I-MEM) or data memory (D-MEM) by a different core may cause the decorated access to end prematurely (without completion). A reset of a core is performed by setting the Reset on Mode Change bit of the Core Address register (ME_CADDR1[RMC] or ME_CADDR3[RMC] = 1) and then performing a mode change by writing the proper sequence to the Mode Entry Mode Control register.

Workaround: There are 3 possible options for preventing loss of a decorated access:

1. Do not perform decorated accesses of either the safety (core 0) or the computational (core 1) core by a different core. (Allow decorated accesses to only be performed by a core to its own local memory. Do not allow decorated accesses to a different cores local memory.)

2. Do not use the mode entry module to reset or reboot the safety or computational core.

3. Implement a software controlled reset lock prior to performing any decorated access to the computational or safety cores' local memory that prevents the mode change from occurring until after the decorated access completes. This requires that a flag be set and cleared before and after the decorated access occurs. Cores that perform mode change resets of other cores would need to check this flag prior performing a mode change controlled reset of cores.

ERR003878: MC_ME: Flash low-power modes cannot be activated during MC_ME mode transitions.

Description: The flash does not support low-power modes. Therefore, if software configures a device mode in the Mode Entry module (MC_ME) to put the flash into its low-power or power-down mode by setting the FLAON field in the corresponding Mode Configuration register (ME_"mode"_MC) to 0b01 or 0b10, the mode transition to that mode will fail, and the device will hang.

Workaround: Software must always configure the flash to be in its normal mode by setting ME_"mode"_MC[FLAON] = 0b11 for all device modes.

Note that this must be actively done for HALT0 and STOP0 modes, since the default values of ME_HALT0_MC[FLAON] and ME_STOP0_MC[FLAON] after reset are 0b10 and 0b01, respectively. For all other modes, the default value after reset is already 0b11.

ERR008117: MC_ME: Restrictions on enabling FlexRay in low power modes

Description: The FlexRay module is dependent on the Auxiliary (AUX) Clock 2, for which the only source clock is Phase Lock Loop 0 (PLL0). This dependency between FlexRay and PLL0 results in the following restrictions during entry to the low power modes, STOP0 and HALT0:

- Entry into STOP0 with FlexRay enabled (by setting MC_ME_PCTLx[LP_CFG=n] and MC_ME_LP_PCn[STOP0]=1) is not possible, even if the Crystal Oscillator (XOSC) is selected as the FlexRay protocol clock source (FR_MCR[CLKSEL]=0).
- Entry into HALT0 with FlexRay enabled (by setting MC_ME_PCTLx[LP_CFG=n] and MC_ME_LP_PCn[HALT0]=1) is possible only if the PLL0 is enabled during HALT0 (by setting MC_ME_HALT0_MC[PLL0ON]=1).

(For MC_ME_PCTLx, x=107 for FlexRay0 and x=235 for FlexRay1.)

Workaround: To enter STOP0, the FlexRay must be disabled (MC_ME_PCTLx[LP_CFG=n] and MC_ME_LP_PCn[STOP0]=0). To enter HALT0 with FlexRay enabled, enable the PLL0 in HALT0 (by setting MC_ME_HALT0_MC[PLL0ON]=1) prior to entering HALT0.

(For MC_ME_PCTLx, x=107 for FlexRay0 and x=235 for FlexRay1.)

ERR008145: MEMU: address registers in the uncorrectable error reporting tables can be written when the corresponding valid bit is not asserted

Description: The Memory Error Management Unit (MEMU) allows error reporting tables to be written by the CPU to simulate a memory error condition or to disable memory errors from a known faulty location. This requires that the valid bit for the System RAM, Peripheral RAM, or Flash memory in the error reporting table (SYS_RAM_UNCERR_STS[VLD], PERIPH_RAM_UNCERR_STS[VLD], FLASH_UNCERR_STS[VLD]) corresponding to the address register (SYS_RAM_UNCERR_ADDR, PERIPH_RAM_UNCERR_ADDR, FLASH_UNCERR_ADDR) is asserted when the address is written to the register. The uncorrectable error reporting tables allow these address registers to be written when the valid bit is not asserted, when they should be read-only.

Workaround: Before writing to any of the uncorrectable error reporting tables address registers (SYS_RAM_UNCERR_ADDR, PERIPH_RAM_UNCERR_ADDR, FLASH_UNCERR_ADDR), the corresponding valid bit (SYS_RAM_UNCERR_STS[VLD], PERIPH_RAM_UNCERR_STS[VLD], FLASH_UNCERR_STS[VLD]) must be asserted.

ERR007335: MEMU: ECC errors may be double reported when initiated by the Safety core to local memory of other cores

Description: Error Correction Code (ECC) events on memory references initiated by the Safety (Main) Core 0 targeting the local memory of either the Main Core 1 or the Input/Output Peripheral Core 2 (IOP) may be reported twice in the Memory Error Management Unit (MEMU) System RAM Reporting Table with the same address. One entry will include syndrome information, the other entry will not include this information.

Workaround: During operation, if the MEMU contains two entries for the same address of an address which is part of a memory for which ECC syndromes are reported, software should check whether one of the two syndromes is 0xFF. If so, this entry should be ignored and deleted. This entry came from another ECC unit which does not report syndromes. As long as the entry with the correct syndrome is stored in the MEMU, entries for the same address without syndrome will not be stored.

ERR010613: MEMU: False IMEM RAM fault flag set after reset or core enabling

Description: Regardless of the boot address for the e200zx core, the e200zx cores initially perform a speculative access (instruction fetch operation) to the local Instruction Memory (IMEM). If the IMEM has not been initialized, an Error Correcting Code (ECC) error will be flagged by the Memory Error Management Unit (MEMU) and captured as either a correctable fault (MEMU_ERR_FLAG[SR_CE]), uncorrectable fault (MEMU_ERR_FLAG[SR_UCE]), or an overflow fault (MEMU_ERR_FLAG[SR_CEO, SR_UC, SR_EBO]). The e200z425 peripheral core_2 performs this fetch operation after reset and the e200z710 main core_0 and core_1 perform this fetch operation after being enabled via the Mode Entry module (MC_ME_CCTLn/ MC_ME_CADDRn). However, the MEMU_SYS_RAM_CERR_ADDRn or MEMU_SYS_RAM_UNCERR_ADDRn contains the IMEM address flagged which will be equal to the address of the reset vector as defined in MC_ME_CADDRn. The local Data Memory (DMEM) memory is not affected. This is the normal operation of the cores and there are no adverse functional effects besides a potential false MEMU flag being set. This is only a single fetch occurrence for each core and no overflow fault will occur unless the MEMU buffer was already full from prior faults. The set MEMU flag will set the corresponding FCCU input for FCCU channels 16 (MEMU_RAM_CE), 17 (MEMU_RAM_UCE), or 18 (MEMU_RAM_BOV).

Workaround: Initialize the e200z710 IMEM memories before enabling the e200z710 cores to prevent a false MEMU fault.The e200z4 IMEM can not be pre-initialized since will be active by default after reset.

Check for a potential peripheral core_2 MEMU set flag or FCCU flag after reset and if a peripheral core_2 IMEM address is flagged, then clear the associated MEMU and FCCU flags. Check for a potential e200z710 core_0 or core_1 MEMU set flag or FCCU flag after enabling the e200z710 core, and if the corresponding e200z710 IMEM address is flagged, then clear the associated MEMU and FCCU flags. Clear the FCCU flag by clearing the corresponding FCCU_RF_S0[RFSm] register bit. The MEMU flags may be cleared by writing a 1 to the set flag register bit for MEMU_ERR_FLAG[SR_CE], MEMU_ERR_FLAG[SR_UCE], or MEMU_ERR_FLAG[SR_CEO, SR_UC, SR_EBO].

ERR007126: MEMU: Instead of Byte 1 of MEMU CTRL Register, Byte 3 is currently protected

Description: The Memory Error Management Unit (MEMU) Control (CTRL) register has Byte 3 protected instead of Byte 1 (using Byte numbering 3 to 0). Therefore the Software Reset bit (SWR) does not have register protection features.

Workaround: Write software such that it does not rely on register protection features for MEMU CTRL Byte 1 SWR. Note that if the SWR bit is inadvertently set, only the status flags and overflow register will be cleared. The reporting tables are not cleared when SWR is set.

ERR008885: MEMU: System RAM ECC errors on accesses by some masters report to MEMU Peripheral RAM table

Description: As described in the reference manual, Error Correction Code (ECC) errors detected by Crossbar bus Concentrator 0 and Concentrator 1 are reported to the Memory Error Management Unit (MEMU) Peripheral RAM table. ECC errors detected by other system bus masters are reported to the MEMU System RAM table. As a result, System RAM locations may appear in both the System RAM table and the Peripheral RAM table. The system bus

masters which report to the MEMU Peripheral RAM table are the Hardware Security Module (HSM), and the DMA controller(s), Ethernet controller, Flexray controller(s), SIPI_0 (Zipwire/ Interprocessor Interface), and SIPI_1 (debug Zipwire).

Workaround: Expect system RAM ECC errors encountered by masters on Concentrator 0 and Concentrator 1 to be reported to the MEMU peripheral RAM table. The system bus masters which report to the MEMU Peripheral RAM table are the Hardware Security Module (HSM), and the DMA controller(s), Ethernet controller, Flexray controller(s), SIPI_0 (Zipwire/Interprocessor Interface), and SIPI_1 (debug Zipwire). System RAM ECC errors encountered by other system bus masters will result in errors reported to the MEMU System RAM table.

ERR007840: M_CAN: Change of operation mode during start of transmission

Description: In the Modular CAN (M_CAN) module, when the transmit Event FIFO is used and a change of CAN operation mode is requested (writing to the CAN Mode Request (CMR) field of the CAN Core Control Register (CCCR)) during the start of transmission, the following incorrect behaviors will occur.

Case 1: Change from classic CAN frame to Flexible Data rate (FD) frame with bit rate switching

The Extended Data Length (EDL) and Bit Rate Switch (BRS) bits of the related Tx Event FIFO element do not match with the transmitted frame type. They signal a CAN FD frame with bit rate switching (EDL and BRS bits are set) while a classic CAN frame was transmitted.

Case 2: Change from classic CAN to CAN FD without bit rate switching

The EDL bit of the related Tx Event FIFO element does not match with the transmitted frame type. It signals a CAN FD frame without bit rate switching (EDL is set) while a classic CAN frame was transmitted.

Case 3: Change from CAN FD with bit rate switching to CAN FD without bit rate switching

The BRS bit of the related Tx Event FIFO element does not match with the transmitted frame type. It signals a CAN FD frame without bit rate switching while a CAN FD frame with bit rate switching was transmitted.

Case 4: Change from CAN FD without bit rate switching to CAN FD with bit rate switching

The BRS bit of the related Tx Event FIFO element does not match with the transmitted frame type. It signals a CAN FD frame with bit rate switching while a CAN FD frame without bit rate switching was transmitted.

Case 5: Change from CAN FD with/without bit rate switching to Classic

The Message RAM Access Failure flag (MRAF) of the M_CAN module Interrupt Register (IR) is set (IR.MRAF = 1), the M_CAN switches to Restricted Operation Mode and the transmission is aborted.

Workaround: Wait that all the Transmission Request Pending n flags (TRPn) of the Tx Buffer Request Pending register (TXBRP) are cleared (TXBRP.TRPn = ‘0’) before changing the CAN operation mode.

ERR009413: M_CAN: Data loss when the storage of a received frame is not complete before EOF

Description: In the Modular Controller Area Network (M_CAN) module, during a frame reception, the receive handler accesses the Message RAM for acceptance filtering (read access) and storage of accepted messages (write access). The time needed for acceptance filtering and storage of a received message depends on the Host clock frequency, the number of M_CAN modules connected to a single Message RAM, the Message RAM arbitration scheme, and the number of configured filter elements.

In the case where the storage of a receive message has not completed before the end of frame field (EOF) is reached, the following faulty behavior occurs:

- The last write to the Message RAM to complete the storage of the received message is omitted: this data is lost. This applies for data frames with a Data Length Code (DLC) different from 0, the worst case being for a DLC equal to 1
- the FIFO put index field (FnPI) of the Receive FIFO n Status register (RXFnS) is updated although the last FIFO element holds corrupted data
- the corresponding New Data flag (NDxx) of the New Data y Register (NDATy) is set although the receive buffer holds corrupted data
- the Message RAM Access Failure flag (MRAF) of the M_CAN Interrupt Register (IR) is not set

Workaround: In this device the M_CAN Host Clock is the Peripheral Bridge Clock (PBRIDGE_CLK).

For Classic CAN operation up to 1 Mbit/s, limit the number of filter elements to:

- 30 elements for 20 MHz if the PBRIDGE_CLK is less than 40 MHz, but is greater than or equal to 20 MHz
- 63 elements for PBRIDGE_CLK greater than or equal to 40 MHz

For non-ISO CAN FD operation up to 1 Mbit/second for arbitration and 4 Mbit/s for data phases:

- limit the number of filter elements to 15 elements and the number of transmit buffer to 30 for if the PBRIDGE_CLK < 40 MHz, and is greater than or equal to 20 MHz
- limit the number of filter elements to 33 elements and the number of transmit buffer to 30 for PBRIDGE_CLK greater than or equal to 40 MHz

ERR050453: M_CAN: Debug message handling state machine not reset to Idle state when M_CAN_CCCR.INIT is set.

Description: When bit M_CAN_CCCR.INIT is set by the Host by writing to CC Control Register or when the M_CAN enters BusOff state, the debug message handling state machine stays in its current state instead of being reset to Idle state. Setting configuration change enable bit in M_CAN_CCCR.CCE does not change M_CAN_RXF1S.DMS.

Workaround: In case the debug message handling state machine has stopped while M_CAN_RXF1S.DMS=“01” or M_CAN_RXF1S.DMS=“10” it can be reset to Idle state by hardware reset or by reception of debug messages after M_CAN_CCCR.INIT is reset to zero.

In case the debug message handling state machine has stopped while M_CAN_RXF1S.DMS=“11” with a DMA request active, it can be reset to Idle state by hardware reset or by the completion of the DMA request.

ERR050789: M_CAN: Dedicated Tx Buffers configured with the same Message ID are not transmitted in order of lowest buffer number first

Description: When several Tx Buffers are configured with the same Message ID and transmission of these Tx Buffers is requested sequentially with a delay between the individual Tx requests, it may happen that the Tx Buffers are not transmitted in order of the Tx Buffer number (lowest number first) when there will be two or more TX requests with the same Message ID pending.

Workaround: In case a defined order of transmission is required:

1) Use the TXBAR features:

a. Write the group of Tx message buffers with same Message ID to the Message RAM

b. Request transmission of all these messages concurrently by a single write access to TXBAR. Before requesting a group of Tx messages with this Message ID ensure that no message with this Message ID has a pending Tx request.

2) Tx buffers shall be requested in ascending order with lowest buffer number first.

3) A single Tx Buffer can be used to transmit those messages one after the other.

4) The Tx FIFO shall be used for transmission of messages with the same Message ID.

ERR007842: M_CAN: Erroneous Interrupt flag after setting / resetting INIT during frame reception

Description: In the Modular Controller Area Network (M_CAN) module, when the Initialization bit (INIT) is set in the CAN Core Control Register (CCCR) during the reception of a frame, the first reception of a frame after clearing the INIT bit will be correctly received but the Message RAM Access Failure flag (MRAF) of the Interrupt Register (IR) will be set.

Workaround: If the Initialization (INIT) bit of the CC Control Register (CCCR) needs to be set during operation, proceed as follows:

Step 1: Issue a clock stop request by setting the Clock Stop Request (CSR) bit of CCCR register

Step 2: Wait until the M_CAN sets INIT and the Clock Stop Acknowledge (CSA) bits of the CCCR register to one

Before clearing INIT, first clear CSR.

ERR009070: M_CAN: FD frame abort may cause Protocol exception event and extended Bus Integration state

Description: In the Modular Controler Area Network module (M_CAN), when a transmission is aborted shortly before the transmission of the Flexible Data Frame (FDF) bit, a receiver will detect a recessive FDF bit followed by a recessive reserved (res) bit. In this case, the receiving M_CAN modules detect a protocol exception event and will enter Bus Integration state. The receivers should leave the Bus Integration state after 11 consecutive recessive bits.

Instead of starting to count the 11 recessive bits immediately after entering the Bus Integration state, the M_CAN needs to see at least one dominant bit before it starts to count the sequence of 11 recessive bits.

Workaround: Take into account that, in the described condition, the M_CAN module will take more time to recover as it will wait for 11 recessive bits after the first dominant bit on the network and not 11 recessive bits after entering the Bus Integration state.

ERR008923: M_CAN: FD frame format not compliant to the new ISO/CD 11898-1: 2014-12-11

Description: This version of the device implements a Modular Controller Area Network (M_CAN) 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.

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.

ERR008062: M_CAN: Frame transmission in DAR mode

Description: In the Modular CAN (M_CAN) module, when the Disable Automatic Re-transmission bit is set in the CAN Core Control Register CCCR (CCCR[DAR]), the two following incorrect behaviors occur.

1- Transmission of a frame will cause the Event Type (ET) field of the Transmit Event FIFO Element to be incorrectly set to ‘0b01’ (transmit event) instead of ‘0b10’ (transmission in spite of cancellation).

2- When multiple messages are transmitted sequentially using the same Transmit buffer, after a successful transmission, the next transmission will not start if it is requested before the CAN bus becomes idle. This message is then treated as if it had lost arbitration.

Workaround: Do not use the same transmit buffer for consecutive transmissions in DAR mode.

Or wait at least for 4 CAN bit times after successful transmission before requesting the next transmission from the same transmit buffer.

ERR008904: M_CAN: Incorrect activation of MRAF interrupt

Description: In the Modular CAN (M_CAN) module, the Message RAM Access Failure flag (MRAF) of the Interrupt register (IR) may be set although there was no Message RAM access failure.

This behavior occurs when the module is receiving a frame and:

- the module is in the Error Passive sate, and
- the Receive Error counter (REC), field of the Error Counter register (ECR), has the value 127.

Workaround: In processing the interrupt from the M_CAN module, if the Error Passive flag of the Protocol Status register (ECR[RP]) is set (ECR[RP] == 1) and the Receiver Error counter equals 127 (ECR[REC] == 127), clear the IR.MRAF flag and exit the interrupt handler.

ERR007841: M_CAN: Incorrect frame transmission after recovery from Restricted Operation Mode

Description: When the Modular CAN (M_CAN) module detects a Message RAM Access Failure (MRAF) during a frame transmission, it sets the MRAF bit of the Interrupt Register (IR) and enters the Restricted Operation Mode (ASM) bit of the CAN Core Control Register [CCCR] is set.

During the first transmission after leaving the Restricted Operation Mode by resetting the CCCR.ASM bit, a frame with an unexpected identifier and control field may be transmitted and could be accepted and acknowledged by a receiver.

Workaround: Use the following procedure to exit from the Restricted Operation Mode:

Step 1: Cancel all pending transmission requests by writing 0xFFFF_FFFF to Transmit (TX) Buffer Cancellation Request (TXBCR) register

Step 2: Issue a clock stop request by setting the Clock Stop Request (CSR) bit of the CC Control Register (CCCR)

Step 3: Wait until the M_CAN sets the Initialization (INIT) and Clock Stop Acknowledge (CSA) bits of the CC Control Register (CCCR) to one

Step 4: First clear CSR bit

Step 5: Then clear INIT bit

Step 6: Wait until INIT is read as zero

Step 7: Issue a second clock stop request by setting CSR bit

Step 8: Wait until the M_CAN sets INIT and CSA bits to one

Step 9: Set the Configuration Change Enable (CCE) bit of the CC Control Register (CCCR), clear CSR and ASM bits in a single write operation.

Step 10: Restart M_CAN by clearing INIT bit.

Step 11: Configure the CAN operation mode by writing to the CAN Mode Request (CMR) field of the CC Control register.

Step 12: Request the transmissions cancelled by step one

ERR009415: M_CAN: Message RAM / RAM arbiter may not respond in time

Description: If the Modular Controller Area Network (M_CAN) module needs to store a received frame, and the Message RAM / RAM Arbiter does not respond in time, this message cannot be stored completely and it is discarded with the reception of the next message. The Message RAM Access Failure flag (MRAF) of the M_CAN Interrupt Register (IR) gets correctly set (IR[MRAF]=1) but the next received message may be incompletely stored. In this case, the respective receive buffer or receive FIFO element contains inconsistent data. This errata only applies when the bit rate is configured for greater than 500 kBit/s.

Workaround: Configure the RAM Watchdog register (RWD) to the maximum expected Message RAM access delay. In case the Message RAM / RAM Arbiter does not respond within this time, the Watchdog Interrupt flag (WDI) of the Interrupt Register (IR) will be set. The frame received after IR[MRAF] has been set and with IR[WDI] set must be discarded.

ERR007796: M_CAN: Message reception and transmission directly after detection of Protocol Exception Event

Description: In the Modular CAN (M_CAN) module, if a Flexible Data rate (FD) frame is received and the reserved bit (res) following the FD frame format bit (FDF) is recessive, the protocol controller correctly detects a Protocol Exception Event. The reception of the message is not finished and the message is discarded but the first message after this Protocol Exception Event will generate the following wrong behaviors.

Case 1: Message reception directly after Protocol Exception Event

The Message RAM Access Failure flag (MRAF) of the M_CAN Interrupt Register (IR) is set even if there was no incorrect access to the Message RAM and the frame has been correctly received.

Case 2: Message transmission directly after Protocol Exception Event

The first frame transmitted after a Protocol Exception Event is transmitted with a faulty frame format. In this case, the MRAF bit is not set.

The other nodes on the CAN network will react to this faulty format and generate error frames causing the M_CAN cell to resend the frame, with a correct format.

Workaround: For reception of messages: ignore the MRAF error. The MRAF signal is primarily intended to validate the correct integration of the M_CAN within a device.

For transmission of messages: the other node(s) will detect the error and trigger the resend of the frame.

ERR011470: M_CAN: Message transmitted with wrong arbitration and control fields

Description: Under the following conditions, the Modular Controller Area Network (M_CAN) module may transmit a message with wrong ID, format fields, and Data Length Code (DLC):

- M_CAN is in state “Receiver” (M_CAN_PSR[ACT] = 10), with no pending transmission
- A new transmission is requested before the 3rd bit of Intermission is reached
- The CAN bus is sampled dominant at the third bit of Intermission, which is treated as Start of Frame (SoF) (see: ISO11898-1:2015 Section 10.4.2.2). This dominant level at the 3rd bit of Intermission may result from an external disturbance or may be transmitted by another node with a significantly faster clock.

Under the conditions listed above, the following can occur:

- The shift register is not loaded with ID, format fields, and DLC of the requested message
- The M_CAN will start arbitration with wrong ID, format fields, and DLC on the next bit
- In case the ID wins arbitration, a CAN message with wrong ID, format fields, and DLC is transmitted, however this message will have a valid Cyclic Redundancy Check (CRC) and therefore will appear to the receiver as a valid frame with no errors. The incorrect format fields and/or DLC may cause the data field to be truncated or padded, but the CRC will be computed after these changes and will be valid for the transmitted frame.

- In case this message is acknowledged, the ID stored in the Tx Event FIFO is the ID of the requested Tx message and not the ID of the message transmitted on the CAN bus. No error is detected by the transmitting M_CAN
- If the message loses arbitration or is disturbed by an error, it is re-transmitted with correct arbitration and control fields.

Workaround: If another transmission is already pending or the M_CAN is not in state “Receiver” (i.e. When M_CAN_PSR[ACT] != 10), a new transmission may be requested without causing this issue.

If no transmission is pending and the M_CAN is in state “Receiver” (M_CAN_PSR[ACT] = 10) then the application must avoid requesting the new transmission during the critical time window between the sample points of the 2nd and 3rd bit of Intermission.

To accomplish this, the application software can evaluate the Rx Interrupt flags M_CAN_IR[DRX], M_CAN_IR[RF0N], and M_CAN_IR[RF1N] which are set at the last bit of End of Frame (EoF) when a received and accepted message is validated. The last bit of EoF is followed by three bits of Intermission. Therefore the critical time window has safely terminated three bit times after the Rx interrupt. Now a transmission may be requested by writing to M_CAN_TXBAR.

After the interrupt, the application has to take care that the transmission request for the CAN Protocol Controller is activated before the critical window of the following reception is reached.

If the application cannot reliably avoid the critical time interval described above, another option is to implement additional application-defined protocol checks within the data payload of each message. This protocol can be used by the receiving node to detect messages that have been corrupted by this issue, discard them, and request retransmission. Such a protocol is already recommended for safety-relevant communication as described in the Safety Manual for this device (See sub-section “Fault-tolerant communication protocol” in the Software Requirements chapter).

ERR050016: M_CAN: Retransmission in DAR mode due to lost arbitration at the first two identifier bits

Description: When the Modular Controller Area Network module (M_CAN) is configured in DAR mode (CCCR.DAR = ’1’) the Automatic Retransmission for transmitted messages that have been disturbed by an error or have lost arbitration is disabled. When the transmission attempt is not successful, the Tx Buffer’s transmission request bit (TXBRP.TRPxx) shall be cleared and its cancellation finished bit (TXBCF.CFxx) shall be set. When the transmitted message loses arbitration at one of the first two identifier bits, it may happen, that instead of the bits of the actually transmitted Tx Buffer, the TXBRP.TRPxx and TXBCF.CFxx bits of the previously started Tx Buffer (or Tx Buffer 0 if there is no previous transmission attempt) are written (TXBRP.TRPxx = ’0’, TXBCF.CFxx = ’1’). If in this case the TXBRP.TRPxx bit of the Tx Buffer that lost arbitration at the first two identifier bits has not been cleared, retransmission is attempted. When the M_CAN loses arbitration again at the immediately following retransmission, then actually and previously transmitted Tx Buffer are the same and this Tx Buffer’s TXBRP.TRPxx bit is cleared and its TXBCF.CFxx bit is set.

This erratum is limited to the case when the M_CAN loses arbitration at one of the first two transmitted identifier bits while in DAR mode. The problem does not occur when the transmitted message has been disturbed by an error.

Workaround: No workaround is necessary as there is no lasting impact to this erratum. When this issue occurs, only one retransmission attempt will be made.

ERR008655: M_CAN: Setting the Configuration Change Enable (CCE) bit during a transmission scan can halt CAN tranmissions

Description: The Modular CAN (M_CAN) Transmission Handler normally scans for Transmission Buffers with pending transmission requests. Pending transmissions are indicated by Transmit (Tx) Buffer Request Pending (TXBRP) register bits being set. If the user decides to reconfigure the M_CAN module by setting the Configuration Change Enable (CCE) bit of the CAN Core Control Register (CCCR) while the Transmission Handler is scanning for pending transmission requests, the TXBRP register can be cleared and the Transmission Handler FSM is halted. This has the effect of halting any M_CAN message transmission.

After the Initialization (INIT) and CCE bits have been cleared by the Host, the M_CAN is unable to transmit messages. When the Host requests a transmission by writing to the Tx Buffer Add Request (TXBAR) register, the respective Tx Buffer Request Pending bit in register TXBRP is set, but the Transmission Handler will not start the requested transmission.

Workaround: If the M_CAN configuration needs to be changed while in use, place the M_CAN module in a halted state (similar to preparing a Power Down (Sleep Mode) as described in the Reference Manual). The following steps must be used:

Step 1: Cancel all pending transmission requests by writing 0xFFFF_FFFF to Tx Buffer Cancellation Request (TXBCR) register

Step 2: Issue a clock stop request by setting the Clock Stop Request (CSR) bit of the CC Control Register (CCCR)

Step 3: Wait until the M_CAN sets the Initialization (INIT) and Clock Stop Acknowledge (CSA) bits of the CC Control Register (CCCR) to one

Step 4: First clear the CSR bit of the CC Control Register (CCCR)

Step 5: Then, performing a separate write operation, clear the INIT bit of the CC Control Register (CCCR)

Step 6: Wait until INIT bit is read as zero

Step 7: Issue a second clock stop request by setting CSR bit

Step 8: Wait until the M_CAN sets INIT and CSA bits to one

Step 9: Set the Configuration Change Enable (CCE) bit of the CC Control Register (CCCR) and clear CSR bit in a single write operation

Now the M_CAN configuration can be modified.

ERR051044: M_CAN: Transmission order of the Tx Queue buffers with the same Message ID is impacted by the PUT index functionality

Description: When multiple Tx Queue buffers are configured with the same Message ID, the transmission order depends on numbers of the buffers where the messages were stored for transmission (lowest buffer number is transmitted first). An Add request points the PUT Index always to that free buffer of the Tx Queue with the lowest buffer number, and thus the buffer transmission order is not predictable.

Workaround: In case a defined order of transmission is required for message with some ID:

1) Use the TXBAR features:

a. Write the group of Tx message buffers with same Message ID to the Message RAM

b. Request transmission of all these messages concurrently by a single write access to TXBAR. Before requesting a group of Tx messages with this Message ID ensure that no message with this Message ID has a pending Tx request.

2) Tx buffers shall be requested in ascending order with lowest buffer number first.

3) A single Tx Buffer can be used to transmit those messages one after the other.

4) The Tx FIFO shall be used for transmission of messages with the same Message ID.

ERR007498: M_CAN: Transmitted bit in control field is falsified when using extreme bit time configurations

Description: When the Modular CAN (M_CAN) module transmits a frame, and when the values of both the Time segment after sample point (TSEG2) and the Baud Rate Prescaler (BRP) fields of the Bit Timing and Prescaler Register (BTP) are 0 (zero), one bit in the control field will be transmitted with an erroneous value. The effect is different for frames to be transmitted in Classic CAN format or CAN Flexible Data-rate (FD) format.

Case 1: Transmission of a classic CAN frame with the 2-bit wide CAN Mode Enable field (CME) of the CC Control Register (CCCR) is 0b00.

If the frame under transmission has a 29-bit identifier where the most significant bit (bit 28 of identifier) is “1”, then the reserved bit following the RTR bit will be transmitted recessive instead of dominant. As a consequence, this frame will be incorrectly interpreted as a CAN FD frame by any node with the CAN FD mode enabled that will therefore generate an error frame. Nodes supporting only the classic CAN mode will ignore this bit (reserved) and correctly receive the frame.

Case 2: For a transmission of a CAN FD Frame with the CME of the CCCR not equal to 0b00.

If the FD frame under transmission has a 29-bit identifier where the most significant bit is “0”, or has an 11-bit identifier, then the FD Frame Format (FDF) bit of the frame is transmitted dominant instead of recessive and the rest of the frame is transmitted in Classic CAN format with an incorrect Data Length Code (DLC) field.

Workaround: Do not use bit timing configurations where BTP.TSEG2 and BTP.BRP are both zero for CAN FD communication.

ERR011458: M_CAN: Tx FIFO message sequence inversion

Description: A message sequence inversion can occur if the following condition occurs. The condition requires that there be two transmit (TX) messages in the TX FIFO and a higher priority non-Tx FIFO message is added, such as to a TX dedicated buffer. The sequence of events is as follows:

Transmission of TX FIFO message 1 is started (from position 1 of the TX FIFO).

A second message is in the second position of the TX FIFO

The message buffer then contains:

Position 1: TX FIFO message 1 (transmission ongoing)

Position 2: TX FIFO message 2

Position 3: -

A higher priority non-TX FIFO message is input into the CAN module and will be the next message to be transmitted. This message will be inserted into the message flow after the message that is being transmitted and the previous second message is pushed down in the transmission buffer.

After the following two message scans the output pipeline has the following content:

Position 1: TX FIFO message 1 (transmission ongoing)

Position 2: non TX FIFO message with higher CAN priority

Position 3: TX FIFO message 2

If the first message that is being transmitted is not successful, due to lost arbitration or CAN bus error, the higher priority (non-TX FIFO) message will be transmitted. The aborted message is put back in the output pipeline. However, instead of being put behind the non-TX FIFO message, it is placed after the previous second message in the TX FIFO.

The output pipeline will then appear as follows:

Position 1: non TX FIFO message with higher CAN priority (transmission ongoing)

Position 2: TX FIFO message 2

Position 3: TX FIFO message 1

At this point, TX FIFO message 2 is in the output pipeline in prior to TX FIFO message 1 and they are transmitted in that order, resulting in a message sequence inversion.

The scope of erratum describes the case when the M_CAN uses both dedicated TX Buffers and a TX FIFO (TXBC.TQFM = '0') and the messages in the TX FIFO do not have the highest internal CAN priority. The above described sequence inversion may also occur between two non TX FIFO messages (TX Queue or dedicated TX Buffers) that have the same CAN identifier and that should be transmitted in the order of their buffer numbers (not the intended use).

Workaround: When transmitting messages from a dedicated TX Buffer with higher priority than the messages in the TX FIFO, choose one of the following workarounds:

First Workaround

Use two dedicated TX Buffers, for example, use TX Buffers 4 and 5 instead of the TX FIFO. The pseudo-code below replaces the function that fills the TX FIFO.

Write message to TX Buffer 4

Transmit Loop:

- Request TX Buffer 4 by setting TXBAR.AR[27] bit to 1
- Write message to TX Buffer 5
- Wait until transmission of TX Buffer 4 completed by reading IR.TC and TXBTO.TO[27]
- Request TX Buffer 5 by setting TXBAR.AR[26] bit to 1
- Write message to TX Buffer 4
- Wait until transmission of TX Buffer 5 completed by reading IR.TC and TXBTO.TO[26]

Second Workaround

Assure that only one TX FIFO element is pending for transmission at any time. The TX FIFO elements may be filled at any time with messages to be transmitted, but their transmission requests are handled separately. Each time a TX FIFO transmission has completed and the TX FIFO gets empty (IR.TFE = '1') the next TX FIFO element is requested.

Third Workaround

Use only a TX FIFO. Send the message with the higher priority also from TX FIFO. However, this workaround has a drawback: the higher priority message has to wait until the preceding messages in the TX FIFO have been sent.

ERR011459: M_CAN: Unexpected High Priority Message (HPM) interrupt

Description: The High Priority Message (HPM) Interrupt flag IR.HPM is set erroneously in the following cases:

Configuration A:

- At least one Standard Message ID Filter Element is configured with priority flag set (S0.SFEC = “100”/“101”/“110”)
- No Extended Message ID Filter Element configured
- Non-matching extended frames are accepted (GFC.ANFE = “00”/“01”)

In configuration A, the HPM interrupt flag is set erroneously on reception of a non-high-priority extended message under the following conditions:

(1) A standard HPM frame is received, and accepted by a filter with priority flag set. Interrupt flag IR.HPM is set as expected for this message.

(2) Next an extended frame is received and accepted because of GFC.ANFE configuration. Interrupt flag IR.HPM is set erroneously for this message.

Configuration B:

- At least one Extended Message ID Filter Element is configured with priority flag set (F0.EFEC = “100”/“101”/“110”)
- No Standard Message ID Filter Element configured
- Non-matching standard frames are accepted (GFC.ANFS = “00”/“01”)

The HPM interrupt flag is set erroneously on reception of a non-high-priority standard message under the following conditions:

(1) An extended HPM frame is received, and accepted by a filter with priority flag set. Interrupt flag IR.HPM is set as expected for this message.

(2) Next a standard frame is received and accepted because of GFC.ANFS configuration. Interrupt flag IR.HPM is set erroneously for this message.

Workaround: For configuration A:

Set up an Extended Message ID Filter Element with the following configuration:

- F0.EFEC = “001”/“010” - select Rx FIFO for storage of extended frames
- F0.EFID1 = any value – value not relevant as all ID bits are masked out by F1.EFID2
- F1.EFT = “10” - classic filter, F0.EFID1 = filter, F1.EFID2 = mask
- F1.EFID2 = zero – all bits of the received extended ID are masked out

Now all extended frames are stored in Rx FIFO 0 or Rx FIFO 1 depending on the configuration of F0.EFEC.

Configuration B:

Set up a Standard Message ID Filter Element with the following configuration:

- S0.SFEC = “001”/“010” - select Rx FIFO for storage of standard frames
- S0.SFID1 = any value – value not relevant as all ID bits are masked out by S0.SFID2

- S0.SFT = “10” - classic filter, S0.SFID1 = filter, S0.SFID2 = mask
- S0.SFID2 = zero – all bits of the received standard ID are masked out

Now all standard frames are stored in Rx FIFO 0 or Rx FIFO 1 depending on the configuration of S0.SFEC.

ERR007532: M_TTCAN: Incorrect value of Reference Trigger Offset status for time slaves

Description: When the Time Triggered Modular CAN (M_TTCAN) module is configured as time slave, read accesses to the Reference Trigger Offset (RTO) field of the TT Operation Status register (TTOST) always return the value 0x7F when the Error Level (EL) 2-bit field of TTOST is greater than 0b00, signalling the presence of error in the TTCAN operation.

The M_TTCAN should return the value configured in the Initial Reference Trigger Offset (IRTO) field of the TT Operation Configuration register (TTOCF).

Workaround: Ignore the value of the RTO field when reading the TTOST register for time slaves.

ERR007538: M_TTCAN: Switch between CAN operating modes during transmission or reception may be ignored

Description: When the Flexible Data rate (FD) mode is enabled in the Time Triggered Modular CAN (M_TTCAN), in other words, the 2-bit wide CAN Mode Enable field (CME) of the CC Control Register (CCCR) is not set to 0b00, request for change of CAN operation mode may be ignored if a frame reception or transmission is in process.

The M_TTCAN supports three modes of operation:

- A: CAN 2.0 mode
- B: CAN FD mode
- C: CAN FD mode with bit rate switching enabled

A change of operation mode is done by writing the Change Mode Request field (CMR) of the CC Control Register (CCCR).

The affected transitions are between {A and B} or {B and C} modes.

The request is acknowledged (CCCR[CMR] reverts to 0b00) but the M_(TT)CAN remains in its previous operation mode.

Workaround: Verify the successful switch of operation mode by reading the CAN FD Bit Rate Switching (FDBS) and the CAN FD Operation (FDO) bits of the CCCR register. If the values do not match the intended operation mode, repeat the mode change request (write to CCCR[CMR]) and the check operation until the change succeeds.

ERR007932: NAR/SIPI: Part ID for NAR and Debug SIPI does not match the MIDR MINOR_MASK

Description: The revision portion of Device Part Identification (ID) register in the Nexus Aurora Router (NAR) and the Serial Interprocessor Interface (SIPI) modules reads 0x0. However, it should match the mask minor revision (MINOR_MASK) in the System Integration Unit Lite (SIUL) Microcontroller Identification Register (MIDR), which is 0x1.

Workaround: Write software such that it does not depend on the minor revision portion of SIPI Part ID matching between the SIPI ID (value = 0x0) and MIDR MINOR_MASK (value = 0x1). Tools should use the JTAG Identification for the minor revision value instead of the NAR ID.

ERR003970: NAR: Trace messages include a 6-bit Source Identification field instead of 4-bits

Description: The source field (SRC) of trace messages from the Nexus Aurora Router are 6-bits in length. All other clients implement a 4-bit SRC field. Per the IEEE-ISTO 5001 Standard (Nexus) the SRC field of all clients on a device should be the same length. The two most significant bits of the SRC are 0b00.

Workaround: Tools should treat the SRC field as a 4-bit field for all Nexus clients. In addition, tools should ignore the extra 2-bits as an extra field with no meaning. In the case of the NAR Error Message (TCODE=8), these two bits are between the 4-bit SRC field and the 4-bit Error Type (ETYPE) field. For the NAR Watchpoint Message, these bits are between the 4-bit SRC and the 6-bit Watchpoint Hit (WPHIT) field.

ERR011021: NAR: Trace to memory buffer pointers corrupted after break-point

Description: When using the trace to memory feature of the Nexus Aurora Router (NAR), some address pointers are not reset properly following a break-point. Therefore, following the break-point, the full trace memory cannot be used to hold trace information. The amount of memory not available depends on the pointer state after the break-point.

Workaround: To use full trace memory debugger following a break-point, perform the additional steps below before enabling trace after any break-point:

1. Perform a soft-reset of the NAR by setting the NAR Soft Reset bit in the NAR Control register (NCR[NSR]=1)

2. Re-configure all the NAR registers (that were cleared by this soft-reset) along with trace memory address register

3. Resume MCU execution from the break-point with trace enabled in the desired trace mode.

ERR009796: PAD_RING: Increased coupling current during current injection of nearby analog pin

Description: The Data Sheet input leakage current (ILK_INUD) for an ADC input channel pin is a maximum of 150 nA and applies even during an injection current (IINJ) event (maximum +/- 3mA) on another ADC input pin. Under a specific set of conditions, there may be some coupling current introduced from a nearby analog pin that would add to the input leakage and then exceed the ILK_INUD specification. The magnitude of the coupling current increases with a larger injection current. The magnitude will also vary depending on what pin has the injection current and what pin is being measured. For the maximum current injection, the total input leakage current may range from 0 – 300 nA.

All of the following conditions need to apply before a potential increase in pin leakage current is possible:

1) An analog input pin needs a negative current injection (in other words, the pin is pulled below ground). Positive current injection will not cause a coupling current to be seen.

2) The pin with the current injection and the affected pin with increased coupling current both need to be in the same pin grouping as defined below.

3) The Analog Pad Control Enable (SIUL2_MSCR_IO_n[APC]) for both pins need to be set. Either pin APC value at 0 (default) will prevent the issue.

Pin groupings. Only the pins within the groupings are affected by current injection of another pin within the grouping. Not all pins are available in all packages.

(PB0, PB1)

(PB2, PB3)

(PB7, PR5)

(PB14, PY1)

(PB15, PG3, PR12, PR13)

(PG1, PR14)

(PG4, PR15)

(PG5, PG6)

(PR2, PR3, PR4)

(PR8, PR9, PY0)

Workaround: The coupling current may be prevented or reduced by preventing or reducing injection current of corresponding pins in the pin groupings. If current injection is not preventable then isolate to pins that are not in one of the pin groupings. To prevent coupling between the pins, clear SIUL2_MSCR_IO_n[APC]=0 (default) for one or both of the pins. For the affected pins, the SIUL2_MSCR_IO_n[APC] bit controls the pin use as a SAR ADC input channel. For pin use as a SD ADC input channel, then SIUL2_MSCR_IO_n[APC] can be at 0. The APC value may be changed by software during the application.

ERR007930: PAD_RING: Pin PM[11] is powered from the FlexRAY and Ethernet I/O power supply instead of the Main I/O supply

Description: Pin PM[11] is powered from the FlexRay & Ethernet I/O power supply (VDD_HV_IO_FLEX) instead of the specified Main I/O supply (VDD_HV_IO_MAIN). There are no FlexRay or Ethernet functions on PM[11].

Workaround: Input or Output functions on pin PM[11] need to be designed to accommodate the voltage supplied on the FlexRay & Ethernet I/O power supply (VDD_HV_IO_FLEX).

ERR010309: PASS: JTAG password does not enable Debug Interface Access during functional reset

Description: The JTAG and Nexus Debug Interface Access cannot be enabled by supplying the JTAG password during functional reset.

Workaround: To enable the Debug Interface Access, supply the JTAG password after functional reset or wait until exit of functional reset for JTAG password to take effect. If the Debug Interface Access needs to be enabled before exit of functional reset, use the JTAG Data Communication (JDC) interface to clear the PASS.LOCK3[DBL] bit instead.

ERR010196: PASS: JTAG password match bypasses flash read protection set by PASS.LOCK3[RLx] bits

Description: On a censored device, setting the region lock bits in the Pass Lock Register 3 (PASS.LOCK3[RLx]) of the Password and Device Security Module (PASS) should protect the contents of flash memory blocks from being read when a debug tool is attached and the block is enabled for any flash region that has the region lock set. Debug enable may be achieved either through clearing the PASS.LOCK3[DBL] bit or providing a matching JTAG password.

However, providing a matching JTAG password bypasses the region lock control of the PASS.LOCK3[RLx] bits and allows all flash regions to be read, regardless of the PASS.LOCK3[RLx] bit settings.

Workaround: It is not possible to maintain the flash read protection on locked regions if the JTAG password is used to enable the debug interface.

In order to maintain the read protection on locked regions of the flash, only enable the debug interface by clearing the Debug Interface Lock bit (PASS.LOCK3[DBL]) bit. The PASS.LOCK3[DBL] bit can be cleared by providing the correct password sequence to a password challenge/response, either via the hardware Security Module (HSM) and JTAG Communication module (JDC), if available, or other external interface, such as CAN using software executing on the MCU.

For parts that support using specific JTAG passwords to invalidate the JTAG password comparison, an invalid JTAG password can be used in UTEST flash to disable this feature. Otherwise, programming a random value for the JTAG password match, which is not communicated to users, can greatly reduce the chances of enabling the debug interface and unlocking flash read access.

ERR007904: PASS: Programming Group Lock bit (PGL) can be de-asserted by multiple masters writing the correct password sections to the CINn registers.

Description: The eight Challenge Input Registers (CINn) in the Password and Device Security Module (PASS) where the 256-bit unlock lock password (8 × 32-bit registers) is provided, can be written by multiple masters. If the written password is correct even though it has been provided from different masters, the password Group Lock (PASS PGL) in the Password Group n Lock 3 Status register (PASS_LOCK3_PGn) is de-asserted and UnLockMaster (MSTR) is set to 0xF.

Therefore, internal registers would not be writable by any of the master other than master whose ID is 0xF if the Master Only (MO) bit is set PASS_LOCK3_PGn.

If a Master wants to update internal registers, it needs to unlock the PASS by writing into all the 8 Password registers.

Workaround: Set the master only bit inside the PASS (LOCK3_PGn.MSTR) to block other master accesses to the unlocked registers. If the written password has been provided from different masters, a single master should perform the unlock operation again by writing into all the 8 password registers.

ERR007905: PIT: Accessing the PIT by peripheral interface may fail immediately after enabling the PIT peripheral clock

Description: If a write to the Periodic Interrupt Timer (PIT) module enable bit (PIT_MCR[MDIS]) occurs within two bus clock cycles of enabling the PIT clock gate in the MC_CGM (Clock Generation Module) register, the write will be ignored and the PIT will not be enabled.

Workaround: After enabling the PIT clock in the MC_CGM, insert a read of the PIT_MCR register before writing to the PIT_MCR register. This guarantees a minimum delay of two bus clocks to guarantee the write is not ignored.

ERR008054: PIT: DMA request stays asserted when initiated by PIT trigger, until PIT is reset

Description: When a Periodic Interrupt Timer 0 (PIT0) channel trigger is used to initiate a Direct Memory Access (DMA) transfer, the DMA request does not negate at the end of the DMA transfer. The result is that if that DMA channel is re-enabled, a subsequent PIT-triggered DMA transfer will be initiated.

Workaround: Either do not use the PIT0 to initiate DMA transfers, or write software such that anytime a PIT0 channel trigger is used to initiate a DMA transfer, the PIT0 module is then reset after that DMA transfer completes, prior to re-enabling the DMA channel that was used for that transfer. The PIT0 module should be reset by setting the PIT_RTC_0 reset bit in the Peripheral Reset Register 0 in the Reset Generation Module.

Other timer systems, such as the System Timer Module (STM), can be used instead of the PIT to trigger the DMA.

ERR050130: PIT: Temporary incorrect value reported in LMTR64H register in lifetimer mode

Description: When the Programmable interrupt timer (PIT) module is used in lifetimer mode, timer 0 and timer 1 are chained and the timer load start value (LDVAL0[TSV] and LDVAL1[TSV]) are set according to the application need for both timers. When timer 0 current time value (CVAL0[TVL]) reaches 0x0 and subsequently reloads to LDVAL0[TSV], then timer 1 CVAL1[TVL] should decrement by 0x1.

However this decrement does not occur until one cycle later, therefore a read of the PIT upper lifetime timer register (LTMR64H) is followed by a read of the PIT lower lifetime timer register (LTMR64L) at the instant when timer 0 has reloaded to LDVAL0[TSV] and timer 1 is yet to be decremented in next cycle then an incorrect timer value in LTMR64H[LTH] is expected.

Workaround: In lifetimer mode if the read value of LTMR64L[LTL] is equal to LDVAL0[TSV] then read both LTMR64H and LTMR64L registers one additional time to obtain the correct lifetime value.

ERR007772: PMC: a SWT_2 destructive reset may be forced if the 5V supply is over voltage for longer than 16ms

Description: When the High Voltage Detect, HVD600_C, that monitors the high voltage supply, asserts, setting the Voltage Detect 15 flag in the Power Management Controller (PMC) Reset Event Pending Register (PMCDIG_EPR_VD15[HVD15_C] due to an over voltage condition, a reset occurs.

If this reset is configured to be a functional reset and the voltage is over voltage for more than 16ms, the Software Watchdog Timer 2 (SWT_2) will time out while waiting for the voltage to return to a valid range.

The SWT_2 timeout will cause a destructive reset and indicate that there was an issue with Flash Initialization.

Destructive resets are not affected.

Workaround: If HVD600_C is configured to be a functional reset in the PMC Reset Event Select Register for Voltage Detect 15 (PMCDIG_RES_VD15[HVD15_C]), then software should expect that SWT_2 may time out and cause a destructive reset. Destructive resets are not affected.

ERR008019: PMC: Escalation of LVD and HVD functional resets should be expected

Description: There are two conditions where Low Voltage Detect (LVD) and/or High Voltage Detects (HVD) functional resets may become escalated to a destructive reset.

First, as expected, if the ramp rate of the supply slowly exceeds a HVD trip point, the HVD may assert multiple times. These multiple assertions can cause a functional reset escalation to a destructive reset.

The LVDs and HVDs themselves cannot cause a destructive reset escalation.

Second, some LVD/HVDs can be configured so that a functional reset escalation is possible independent of the supply ramp rate.

The list of these LVD/HVDs is

LVD108_P	Low voltage supply PLL medium range low voltage detector
LVD112_C	Low voltage supply core medium range low voltage detector
HVD145_C	Low Voltage supply core high voltage detector
HVD145_F	Flash LV supply high voltage detector (cold point)
LVD270_EBI	High voltage supply I/O Oscillator low range low voltage detector
LVD270_IF	FlexRay I/O Supply low voltage detector
LVD270_IF2	Second FlexRay I/O low voltage detector
LVD360_IM	HV IO main supply low voltage detector
LVD400_A	HV ADC supply low voltage detector
LVD400_IM	HV IO main supply low voltage detector
HVD360_F	HV flash supply high voltage detector
HVD600_A	HV ADC supply high voltage detector

The configuration to cause this issue is if the Device Configuration Format record (DCF) client assigned to the Power Management Controller digital Reset Event Enable register (PMC_DIG REE/PMC_REE_BUS) must disable the LVD/HVDs.

Also, the software must enable the LVD/HVD as a functional reset by writing to the PMC_DIG REE (to enable) and RES (to select functional resets) registers.

Note, that the following LVD/HVDs do not allow an exit from reset until the supply has returned to a valid range:

LVD108_C	LVD on Low voltage internal supply (hot point)
LVD108_F	LVD on Flash Low voltage supply (hot point)
LVD270_C	HV PMC supply low voltage detector
LVD270_F	HV flash supply low voltage detector
LVD270_IJ	JTAG I/O supply low voltage detector
LVD270_IM	HV IO main supply low voltage detector
LVD270_O	Oscillator supply low voltage detector
LVD295_F	HV flash supply low voltage detector
LVD295_A	HV ADC supply low voltage detector
HVD600_C	HV PMC supply high voltage detector

These cannot cause a functional reset escalation even if configured via the PMC_REE_BUS DCF client and software enabling as described.

Workaround: Use LVD/HVD resets as destructive resets only.

Alternatively, if functional resets for the LVD/HVDs are used, the Reset Generation Module (RGM) functional escalation register (MC_RGM_FRET) should be cleared before enabling the LVD/HVD.

Also, if functional resets for the LVD/HVDs are used, the resets should not be enabled until the current status of the LVD/HVD (shown in the PMC_DIG GR_S register) shows that the voltage is currently within a valid range.

ERR010660: PMC: EPR flags may be set after an exit from reset

Description: During parts of the reset sequence, the Low Voltage Detects (LVDs) and High Voltage Detects (HVDs) are masked until the Power Management Controller (PMC) can be trimmed. During this masking, valid voltage levels are assured by Power On Reset (POR) signals and thus the Voltage Out of Range Destructive reset bit in the Reset Generation Module (MC_RGM_DES[F_VOR_DEST]) is not set. However, some register flags of the PMC Digital Module Event Pending registers (PMCDIG_EPR_*) including PMCDIG_EPR_TD[TEMP_0, TEMP_2 & TEMP_3) may be set incorrectly after device reset.

Workaround: The statement in the PMC “Analog PMC interface” section of the Reference Manual which currently states “After a POR, some of the EPR registers may be set due to slow supply ramps or noisy supplies during power up. Clear these flags during initialization after a POR.” needs to be updated to "After any reset (except short functional), some of the EPR registers may be set due to slow supply ramps or noisy supplies during reset events. Check the PMC Supply Gauge Status Register (PMCDIG[GR_S]) to verify there is not a pending voltage event, and then clear the EPR flags during initialization after resets."

For temperature EPR flags, check the EPR flags PMCDIG_EPR_TD[TEMP_0, TEMP_2 & TEMP_3) and if found to be set use the ADC to measure the on chip temperature sensor and determine if it is the associated over temperature event that caused the flag to be set. Only if the ADC confirms the temperature is close to the over temperature condition then the flag can be considered valid. Incorrect EPR_TD flag bits can then be cleared, this also includes clearing the interrupts, resets and FCCU events to avoid any false events. This must be done prior to enabling any Interrupts, Resets, and FCCU events.

ERR007365: PRAM: Reset may change RAM content

Description: When Reset is asserted if a memory access cycle is pending for the platform ram controller, a location in the system ram may be written with an unknown value. For a software initiated short functional reset or a software initiated destructive reset (asserted with maximum system ram controller clock of 40MHz), the system RAM is preserved.

Workaround: 1) Critical data in the RAM which must be preserved across reset should be protected by a checksum/signature, or should be stored redundantly in the memory. Usage of software functional reset or a software initiated destructive reset (asserted with maximum system ram controller clock of 40MHz) will preserve the RAM contents.

2) System RAM can be protected from corruption if application uses a software initiated short functional reset or a software initiated destructive reset (asserted with maximum system ram controller clock of 40MHz). The system ram controller clock is the same as the FXBAR_CLK clock . Thus, reconfigure the system clock PLL and/or dividers to achieve 40MHz. Or use either the internal 16MIHz RCOSC clock or external oscillator clock as the system clock source. These are the only types of resets. This is the only reset and the only RAM's that are covered by this statement.

A short functional reset can be generated by initiating a mode transition through the Mode Entry Module (MC_ME). This is done by writing 0b0000 to the Target Mode bits of the Mode Control Register (MC_ME.MCTL). This is a two step process as this register is protected and requires a special sequence to modify the Target Mode bits.

MC_ME.MCTL.R = 0x00005AF0

MC_ME.MCTL.R = 0x0000A50F

A destructive reset can be generated by initiating a mode transition through the Mode Entry Module (MC_ME). This is done by writing 0b1111 to the Target Mode bits of the Mode Control Register (MC_ME.MCTL). This is a two step process as this register is protected and requires a special sequence to modify the Target Mode bits.

MC_ME.MCTL.R = 0xF0005AF0

MC_ME.MCTL.R = 0xF000A50F

ERR008141: PSI5-S: Global mode transition interrupt does not work

Description: In the Peripheral Sensor Interface Support module (PSI5-S), the Global Mode Transition Done bit (GL_MODETR_DONE) of the Global Status Register (PSI5S_GLSR) does not get set unless there is a read access to the PSI5-S module.

As a consequence, even if enabled, the associated mode transition interrupt will not be triggered.

Workaround: Do not enable the Global Mode transition enable interrupt. Poll the PSI5S_GLSR[GL_MODETR_DONE] to detect the module entered the desired mode.

ERR008075: PSI5-S: Unrecoverable messages are not flagged in the mailbox status register channel 0

Description: In the Peripheral Sensor Interface Support module (PSI5-S), unrecoverable messages are correctly routed to Frame 1 of the special channel 0 but the Frame 1 error bit (F1_ERR) of the Mailbox status register channel 0 (PSI5S_MBOX_SR_CH0) is not set.

Instead, the Frame n error bit (Fn_ERR, n = 0 to 5) of the mailbox status register of the channel y (PSI5S_MBOX_SR_CHy, y = 1 to 7) of the intended channel.

Workaround: Ignore the value of PS_SR_MBOX_CH0[F1_ERR].

Assume that any message that arrives in the Channel 0 Frame 1 always contains some error that can be identified reading the Message Recovery Unit output Buffer 2 register 1 (PSI5S_MRU_BUF2_REG1). Also, when reading the message corresponding to the channel 0 frame 1, the PSI5S_MBOX_SR_CHy[Fn_ERR] of the faulty channel must be cleared.

ERR007996: PSI5: Incorrect SMC message decoding and timestamp generation in case of late last sensor message overlapping with next SYNC period pulse

Description: As stated in section 6.6 of Peripheral Sensor Interface (PSI5) Standard v2.0 and v2.1, PSI5 sensor frames should not overlap with the SYNC pulse. This overlap is considered to be an error condition. In extension to the standard requirement, the PSI5 module implemented in this device allow the possibility to manage this overlap error condition.

In case of such overlap condition:

- PSI5 message extraction happens correctly
- Timing bit error [T] and CRC error [C] flags are correctly set within the PSI5 message
- The serial messaging channel (SMC) frame counter gets reset on Sync pulse and the extraction of SMC message does not happen correctly, resulting in loss of SMC message

The PSI5 module correctly handles this error condition for the PSI5 message, but is not able to handle the SMC message correctly.

Also during this overlap condition, the timestamp appended with the overlapped slot is the new sync pulse timestamp. Whenever a sync pulse comes, the internal sync pulse timestamp capture registers are updated with the timestamp of the new sync pulse. Hence if any message overlaps with the sync pulse and that message slot is configured to capture the timestamp of the SYNC pulse (PSI5_SnFCR[TS_CAPT] = 1), then the timestamp appended for that slot is the timestamp corresponding to the new sync pulse and not of the previous sync pulse belonging to the PSI5 frame.

Workaround: The PSI5 message is properly received and analysed with the appropriate error flags set. Application software can identify from the properly received PSI5 message that an error on the bus occurred. If an SMC message is configured to be present in the slot, application software can request an immediate halt and restart of the SMC or it can wait until the SMC reception has finished and check the CRC of the SMC message

- if incorrect, a resend of the SMC message can be re-started at that point in time. Further, application software can check the timestamp of the previous message in the final slot from the previous SYNC period to identify if the wrong timestamp has been captured

- if the difference between the two messages is equivalent to two SYNC periods, application software can correctly identify that an overlap of the final message with the following SYNC pulse has occurred assuming the slot is configured to capture the SYNC pulse timestamp (PSI5_SnFCR[TS_CAPT] = 1) rather than the message timestamp (PSI5_SnFCR[TS_CAPT] = 0)

ERR006992: PSI5: IS_DEBUG_FREEZE bit is not documented

Description: Bit 0 (the most significant bit of the register, MSB) of the Peripheral Sensor Interface 5 General Interrupt Status Register (PSI5_GISR) is not documented. This bit is the IS_DEBUG_FREEZE bit and is set when the PSI5 module is stopped in debug freeze mode. To enable the debug freeze mode, both PSI5 Debug mode Enable and the Debug Freeze Control bits must be set in the PSI5 Channel Control register (PSI5_PCCR[DEBUG_EN] = 0b1 and PSI5_PCCR[DEBUG_FREEZE_CTRL] = 0b1). When thePSI5 module receives the request to enter the debug mode, it finishes the current processing and is stopped coherently. At this stage, the IS_DEBUG_FREEZE bit is set to “1” . This bit automatically gets cleared by the hardware when the debug mode is exited.

Workaround: Expect bit 0 (MSB) of the PSI5_GISR register to be set when the PSI5 module is stopped in debug freeze mode if both DEBUG_EN and DEBUG_FREEZE_CTRL are set. The documentation will be updated.

ERR007234: PSI5: No transfer error generated for accesses within the unused range of the PSI5 peripheral window

Description: The Peripheral Sensor Interface (PSI5) uses 4 Kbytes of the 16 Kbytes range of the peripheral bridge slot assigned to it.

Accesses after the 4 Kbytes (from offset 0x1000 to offset 0xFFFF) will not generate a transfer error.

Note: accesses to unimplemented locations within the 4 Kbyte window will correctly generate a transfer error.

Workaround: Take into account that no transfer error will be generated outside the 4 Kbyte region used by the PSI5 module.

In case such accesses must be detected, use the memory protection unit (MPU) to limit accesses.

ERR007134: RCCU: If any accesses to the I-MEM or D-MEM of the safety and checker core are performed while the cores are disabled, the cores will get out of lockstep when enabled

Description: If any accesses to the local Instruction Memory (I-MEM) or Data Memory (D-MEM) of the safety (Main Core_0) and checker core (Checker Core_0s) are performed while the cores are disabled, the cores will get out of lockstep when enabled. A disablement of Main Core_0 and Checker Core_0s is defined to be when the corresponding status bits of the Core Status

Register (ME_CS) are set to 0. Most typically, this will happen after a destructive RESET where the system is booted up with the I/O Processor (IOP) and Hardware Security Module (HSM) enabled, but the safety and checker cores are disabled.

Memory Built In Self Test (MBIST) and Logic Built In Self Test (LBIST) are not affected by this errata.

Workaround: Do not read or write the I-MEM or D-MEM of the safety core while it is disabled. Insure that the Main Core_0 and Checker Core_0s are enabled and executing out of some memory (I-MEM / D-MEM, system RAM, Flash, ROM, external memory) when any bus master is accessing the safety core I-MEM / D-MEM.

ERR010556: RGM: ESR0 long functional reset advances to the PHASE1[DEST] reset state, but sets the functional reset only flag

Description: The assertion time/action of the External System Reset 0 (ESR0) pin is configured in the Reset Generation Module (RGM) Functional Event Short Sequence Register (FESS) to trigger either a long functional reset (default) or a short functional reset (MC_RGM_FESS[SS_ESR0] = 0b1). The “Reset Generation Module Reset Sequence” figure in the “Reset and Boot” chapter incorrectly shows an ESR0 functional reset advances to PHASE1[FUNC]. The correct operation is that an ESR0 long functional reset advances to PHASE1[DEST] and an ESR0 short functional reset advances to PHASE3[FUNC]. Both ESR0 long and short functional resets will set the Functional ESR0 bit in the Functional Event Status register (MC_RGM_FES[F_ESR0].

Additional reset information for the Mode Entry DRUN Mode Configuration register (MC_ME_DRUN_MC) is that the system clock selector as controlled by SYSCLK is reset by any type of reset. Additional reset information for the MC_CGM registers is that the MC_CGM_SC_DCn, MC_CGM_ACn_DCx, and MC_CGM_DIV_UPD_* registers are reset in POWERUP, PHASE0, PHASE1, PHASE2, and PHASE3 reset phases but the MC_CGM_ACn_SC clock source are only reset by POWERUP resets only.

Workaround: Assertion of an external ESR0 long functional reset may be used to activate an off-line self-test sequence. The system clock source selector will switch glitch free for any reset type and the auxiliary clock source selector and auxiliary clock divider control are only reset by POWERUP.

ERR009764: SARADC : DMA interface limitation depending on PBRIDGE/SARADC clock ratio

Description: The Successive Approximation Register Analog-to-Digital Converter (SARADC) modules can trigger a Direct Memory Access (DMA) request through the DMA Enable (DMAE) register interface.

When the SARADC clock (SAR_CLK) frequency is slower than half of the peripheral bridge (PBRIDGEx_CLK) clock frequency, the SARADC may trigger a spurious transfer request to the DMA module after the completion of a first valid transfer.

Workaround: Setting the DMA clear sequence enable (DCLR) bit in the DMAE register (DMAE[DCLR] = 1) forces the clearing of the DMA request on read access to the data register and therefore prevents the spurious DMA transfer request.

In case the Internal Channel Data Registers (ICDRn) are only accessed through DMA module (i.e. there are no bus accesses to ICDRn registers triggered by other than DMA bus master when the DMAE[DMAEN] bit is set), it is possible to configure DMAE[DCLR] bit to '1'. This will clear DMA transfer request on the first DMA read access, ensuring both that DMA triggered transfer will complete successfully and that no other spurious DMA request will be triggered.

This work-around can be applied when any of below condition can be met:

- frequency ratio PBRIDGEx_CLK/SAR_CLK <= 8/3
- PBRIDGEx_CLK is 40MHz and SAR_CLK >= 14MHz

ERR010171: SARADC: Channel number may show incorrect value for two clock cycles

Description: The Channel under measure (CHADDR) field of the Successive Approximation Register Analog to Digital Converter (SARADC) Main Status Register may read an incorrect value for 2 ADC clock cycles (SARADC_MSR:CHADDR).

Case 1: During an injected conversion, the CHADDR field will show an incorrect channel number instead of showing the injected channel number. The incorrect value depends on when the injection trigger is applied.

Case 2: When read, the CHADDR field may be incorrect for 2 clocks immediately after it is updated.

Workaround: In order to ensure that the channel address of the current conversion is correct, the MSR:CHADDR field should be read at least three times consecutively. If all three reads are the same, then this is the correct CHADDR. Otherwise read the MSR:CHADDR again and compare until 3 values match.

ERR005947: SARADC: ADC may miss a GTM trigger pulse if width of pulse is less than 1 AD Clk cycle

Description: The Successive Approximation Register Analog to Digital Converter (SARADC) may miss a trigger (and no conversions will be started) from the Generic Timer Module (GTM) if the pulse width from the GTM Timer Output Module (TOM) or Advanced Routing Unit (ARU) Connected TOM (ATOM) is less than one ADC clock. The GTM Counter Compare Unit registers (CM0/ CM1) set the pulse width.

Workaround: The GTM registers Counter Compare Unit registers (CM0 and CM1) should be appropriately programmed such that pulse width of trigger pulses is always greater than one (1) ADC clock cycle.

ERR007245: SARADC: CDATA fields for Left justified data is not documented

Description: In the Successive Approximation Register Analog to Digital Converter (SARADC), the channel converted data field (CDATA) of the Internal, Test and External Channel Data Registers (ICDRx, TCDRx and ECDRx) is a 16-bit field instead of 12-bit as documented.

In addition, depending on the Write left/right-aligned bit of the Main Configuration register (SARADC.MCR[WLSIDE]) and on the Conversion Resolution in the Conversion Timing register (SARADC.CTRx[CRES]), the actual data will use only part of this 16-bit field.

Therefore the complete description of the CDATA field is:

The CDATA field uses the bit-field [16:31] of the I/T/ECDR registers and:

A) For 12-Bit resolution (i.e. SARADC.CTRx[CRES]=0)

If MCR[WLSIDE]=0, i.e. Right Aligned Data, the rightmost 12 bits (I/T/ECDRx[20:31]) contain the converted data and the CDATA upper 4 bits (I/T/ECDRx[16:19]) always return 0b0000.

If MCR[WLSIDE]=1, i.e. Left Aligned Data, the leftmost 12 bits (I/T/ECDRx[16:27]) contain the converted data and the CDATA lower 4 bits (I/T/ECDRx[28:31]) always return 0b0000.

B) For 10-Bit resolution (i.e. SARADC.CTRx[CRES]=1)

If MCR[WLSIDE]=0, the rightmost 10 bits (I/T/ECDRx[22:31]) contain the converted data and the CDATA upper 6 bits return 0b000000.

If MCR[WLSIDE]=1, the leftmost 10 bits (I/T/ECDRx[16:25]) contain the converted data and the CDATA lower 6 bits return 0b000000.

Workaround: Take into account the correct description above for the 10- and 12-bit, left or right justified CDATA field.

ERR007246: SARADC: First conversion after exit from stop mode may be corrupted

Description: In the Successive Approximation Analog to Digital Converter (SARADC), if a chain conversion is on going and a transition to stop mode request is done, the result of the first conversion after exit of the stop mode (in other words, the conversion that was interrupted when going into stop mode) will be corrupted in the following cases:

Case A: the peripheral bridge clock (PBRIDGEx_CLK) becomes lower than the SARADC clock (SAR_CLK)

Note: this might be the case if the input of the PBRDIGEx_CLK divider is changed during the mode transitions (from the output of the PLL to the internal RC Oscillator for example)

Case B: the PBRIDGEx_CLK is resumed after the SAR_CLK, with a delay greater than 10 cycles of SAR_CLK

Workaround: The following workarounds are possible:

1) Disable PBRIDGEx_CLK during stop mode and enable it only with a configuration such that it is greater than SAR_CLK.

OR

2) Verify that no analog conversion is ongoing before issuing a stop mode request by reading the ADC status field of the Main Status Register (SARADC_x.MSR[ADCSTATUS] = 0b000, also known as IDLE).

OR

3) Ignore the result of the first conversion after stop mode exit, considering it as corrupted.

ERR010428: SARADC: Interrupted conversions are aborted, but may not be properly restored

Description: When a triggered conversion interrupts an on-going conversion in the Successive Approximation Analog to Digital Converter (SARADC), it is possible that the aborted conversion does not get restored to the SARADC and is not converted during the chain. The vulnerable configuration is for an injected chain over a normal chain.

When the injected trigger arrives while the SARADC is in the conversion stage of the sample and conversion, the sample is discarded and is not restored. This means that the External Channel Data Register, Internal Channel Data Register, or Test Channel Data Register (SARADC_xCDRn) will not show the channel as being valid and the [External, Internal, or Test] Channel Interrupt Pending Register (SARADC_xCIPRn) End of Conversion field EOC_CHx will not indicate a pending conversion. The sample that was aborted is lost.

If the injection occurs when the finite state machine switches from the sample phase, it is possible that on resuming normal chain, the chain is restored from an incorrect channel. This may lead to a second conversion on one of the channels in the chain.

Workaround: The application should check for valid data using the External Channel Data Register, Internal Channel Data Register or Test Channel Data Register (SARADC_xCDRn) status bits or the [External, Internal, or Test] Channel Interrupt Pending Register (SARADC_xCIPRn) registers to ensure all expected channels have converted. This can be tested by running a bitwise AND and an Exclusive OR (XOR) with either the [External, Internal, or Test] Channel Injected Mask Register (SARADC_xCJCMRn) or the [External, Internal, or Test] Channel Normal Conversion Mask Register (SARADC_xCNCMRn) and the SARADC_xCIPRn registers during the NECH/JECH handler. Any non-zero value for (xCE_CHx & (xCE_CHx ⊕ EOC_CHx)) / (SARADC_xCxCMRn & (SARADC_xCxCMRn ⊕ SARADC_xCIPRn.EOC_CHx)) indicates that a channel has been missed and conversion should be requested again.

ERR007222: SARADC: Minimum value of precharge must be greater than or equal to 2 ADC clock cycles

Description: The Successive Approximation Register Analog-to-Digital Converter (SARADC) requires a minimum valid value of the Precharging phase duration field (PRECHG) of the Conversion Timing Register (SARADC_x.CTRz[PRECHG]) must be 2, meaning the precharge phase duration is two (2) SARADC clock cycles. This is incorrectly defined as ‘1’ in some revisions of the documentation.

Workaround: Take into account the minimum value of 2 when configuring the precharge duration in the SARADC_x.CTRz[PRECHG].

ERR007138: SARADC: Missed conversion after ABORT of the last channel of an injected chain

Description: In the Successive Approximation Register Analog-to-Digital Converter (SARADC), when a chain conversion is injected over a normal chain conversion and an abort conversion command is initiated by setting the abort conversion bit of the Main Configuration Register (SARADC.MCR[ABORT]), when the last channel of the injected chain is in the sampling phase then a conversion will be missed.

The conversion of the next normal channel after the resume is skipped.

Expected behavior: the conversion of the last injected channel is aborted, and the normal chain is resumed.

Errata behavior: the conversion of the last injected channel is correctly aborted, but the conversion of the next normal channel is incorrectly aborted.

For example: If channels 0, 1, 2, 3, 4, and 5 are to be converted in Normal chain and channels 6, 7, 8, and 9 are injected when the channel 3 conversion was ongoing the following behavior occurs:

nch0 --> nch1 --> nch2 --> nch3 (aborted by injected conversion chain) --> jch6 --> jch7 --> jch8 --> jch9 (MCR[ABORT] set at this time) --> nch3 (restarted after end of injected chain) * --> nch5

*The conversion of the normal channel 4 is missing.

Workaround: Do not issue a conversion abort request when the conversion of the last channel of an injected chain is on going.

The user can read the Channel under measure address field (CHADDR) of the SARADC Main Status register (SARADC_MSR) to identify the channel under conversion.

ERR006197: SARADC: Positive DNL marginal at Cold

Description: The Differential Non-Linearity (DNL) specifications for the Successive Approximation Register Digital to Analog Converter (SARADC) quoted in the Data Sheet is +/-1 least significant bit (LSB) with no missing codes. This specification will be changed to +2/-1 LSB in future versions of the data sheet specification. At a temperature of -40C for some devices will be marginally within +1 LSB, however, some devices will exceed the previous +1 LSB specification. Since the negative DNL is well within the specification of -1 LSB, there are no missing codes.

Workaround: Expect the SARADC DNL to be within the new +2/-1 specification.

ERR007906: SARADC: The Data Overwritten flag bits in the SARADC may not be valid

Description: In the Succesive Approximation Analog to Digital converter (SARADC), if the overwrite unread converted data feature is enabled using the overwrite enable bit in the SARADC Main Configuration Register (SARADC_MCR[OWREN]) then the data overwritten flag bit in the SARADC Internal Channel data register (SARADC_ICDR[OVERW]), External Channel Data Register (SARADC_ECDR[OVERW]) and Test Data Registers (SARADC_TCDR[OVERW]) registers may incorrectly show as “1". Software monitoring these OVERW bits could confuse this as an actual overwrite condition occurring.

Workaround: The SARADC_ICDR[OVERW], SARADC_ECDR[OVERW] or SARADC_TCDR[OVERW] flags should not be used by software to identify if an overwrite has occurred.

ERR008126: SARADC_B: Some test channels are no longer available

Description: Connections to some test points in the Power Management Controller (PMC) have been removed. The crude band gap (test channel 111) is no longer supported. In addition, some test points that were available through the PMC generic input to the Successive Approximation Digital to Analog Converter B (SARADC_B) (channel 110) are no longer available. These points are shown in the following table.

Name	Description	PMC ADC Select value
REF_1P0	PMC 1.0V reference	0b111011
VBG_REF_1P22	1.2V crude band gap	0b111110
VREF_SOFT	Reference for 3.3V flash supply regulator	0b000111
BIS_VBG_REF_1P 22	Second crude band gap	0b100100

Aux Reg REF	Reference for Auxiliary regulator	0b100010
REF for LFAST PLL REG	Reference to LFAST PLL regulator	0b100001

Workaround: Do not try to convert SARADC_B test channel 111 or any of the PMC test channels listed above via the PMC ADC Select register.

ERR007362: SDADC: Additional DMA request generated after single read access

Description: The Sigma-Delta Analog-to-Digital Converter (SDADC) issues an extra transfer request when the FIFO full Direct Memory Access (DMA) channel is configured to read only 1 data value from the SDADC.

Therefore, when the FIFO (First-In-First-Out) Threshold (FTHLD) field of the SDADC FIFO Control Register (FCR) is 0 (1 conversion) or when the FIFO Enable bit (FE) of FCR is 0 (FIFO is disabled), the extra read request will return invalid data.

The first DMA read access to the SDADC (correct read) returns good data, the second one (extra access, unwanted) returns the contents of the FIFO (undefined, old conversion results).

Workaround: Workaround 1:

Configure the SDADC FIFO threshold to a number N greater than 0 and its FIFO full DMA channel to read at least 2 conversion results at a time.

Workaround 2:

Use the SDADC watchdog DMA channel, that is not generating extra requests, instead of the SDADC FIFO full channel. Use the following settings:

- FIFO threshold set to 1
- Watchdog high and low threshold set to 0
- FOWEN set to 1

Note: This workaround is not applicable when SDADC watchdog feature is not present.

ERR010376: SDADC: Additional DMA request generated with slow SDADC SD_CLK clock

Description: The Sigma-Delta Analog-to-Digital Converter (SDADC) may issue an extra Direct Memory Access (DMA) transfer request when the ratio between SDADC conversion clock (SD_CLK) period and the DMA clock (SXBAR_CLK) is more than 8:

(PERIOD SD_CLK)/PERIOD SXBAR_CLK)>8.

The DMA transfer may complete before the S/D data comparison cycle is completed, actually triggering a new DMA transfer request.

Workaround: When using Direct Memory Access transfer request including transfer using watchdog threshold, ensure that clock relation between SD_CLK and SXBAR_CLK is granted: (PERIOD SD_CLK)/PERIOD SXBAR_CLK)<=8

ERR008039: SDADC: digital filter and FIFO not disabled when MCR[EN] is cleared

Description: When the Enable bit (EN) of the Sigma-Delta Analog to Digital Converter (SDADC) Module Configuration Register (MCR) is cleared (MCR[EN]=0), the digital part of the SDADC continues operating and does not go to low power mode if the module is disabled while a valid conversion is already in process and the application software continues to initiate conversions. As a consequence, the digital block of the SDADC still produces new conversion results in the Channel Data Register (CDR) and dummy data are transferred to the result First-In, First-Out (FIFO) buffers. In addition, interrupt and/or Direct Memory Access (DMA) events are still generated.

Note: the analog part does enter the power-down mode, reducing the consumption on the ADC high voltage supply domain (VDD_HV_ADV).

Workaround: Do not initiate a conversion prior to enabling the SDADC (MCR[EN]=1). In addition, once the SDADC has been enabled (MCR[EN]=1), if the SDADC needs to be disabled (MCR[EN]=0), prior to clearing the EN bit, either turn off the clock to the SDADC module in the Clock Generation Module (CGM) or Select the External Modulator Mode (EMSEL) by setting the MCR[EMSEL] bit along with the clearing the MCR[EN].

ERR008314: SDADC: Double trigger is generated for conversion when SW trigger is connected to SDADC own HW trigger input

Description: In the Sigma-Delta Analog-to-Digital Converter (SDADC), when the wraparound mechanism is enabled by setting the Wrap-Around Mode bit (WRMODE) of the Module Configuration Register (MCR) register, the number of the channel to be sampled gets incremented based on software (SW) and hardware (HW) triggers.

Each SDADC module is allowed to generate a SW trigger using the Software Trigger Key Register (STKR). The SW trigger generates a pulse which is used to synchronize other SDADCs. To do this, the SW trigger of a SDADC module is connected to the HW trigger input of other SDADCs.

In this device implementation, the SW trigger pulse is also connected to a HW trigger input of the same SDADC module.

Since there is a one cycle delay between the SW and the HW trigger pulses (for synchronization), the SDADC using the SW triggered pulse will be triggered a second time by its own HW trigger input. This causes the channel to be incremented twice with just one trigger.

Workaround: Do not program the SDADC_MCR[TRIGSEL] register of the master SDADC channel to select the HW input from itself.

ERR008225: SDADC: FIFO Flush Reset command requires clearing the Data FIFO Full Flag

Description: When the Sigma-Delta Analog-to-Digital Converter (SDADC) FIFO is flushed by writing ‘1’ to the FIFO Control Register FIFO Flush Reset bit (SDADC_FCR[FRST]), the FIFO is correctly flushed, but the Status Flag Register Data FIFO Full Flag (SDADC_SFR[DFFF]) may be incorrectly asserted, indicating the FIFO is full when it is empty..

Workaround: Clear SDADC_SFR[DFFF] by writing a ‘1’ to this field after performing a FIFO Flush Reset command or after the FIFO is disabled.

ERR010378: SDADC: Incorrect data provided when FIFO is disabled and FIFO overwrite is enabled

Description: The Sigma-Delta Analog-to-Digital Converter (SDADC) allows continuous data acquisition. When the overwrite functionality is enabled (FCR[FOWEN] = 1) in the FIFO Control Register (FCR), previously converted data will eventually be overwritten if it is not read before new data is available. In case the overwrite functionality is enabled (FCR[FOWEN] = 1) in the FIFO Control Register (FCR) together with the disabling of the associated FIFO buffer (FCR[FE] = 0), the Data FIFO Empty flag in the Status Flag register (SFR[DFEF]) will toggle high (Data FIFO is empty) and low (Data FIFO is not empty). If the Converted Data Register (CDR) is read while in the “Data FIFO is empty" state, then the previous converted data is provided rather than newest converted data.

Workaround: Always disable the FIFO overwrite functionality in the FIFO Control Register (FCR[FOWEN]=0) if the FIFO buffer is disabled (FCR[FE]=0).

ERR006906: SDADC: Invalid conversion data when output settling delay value is less than 23

Description: In the Sigma Delta Analog to Digital Converter (SDADC), if the Output Settling Delay field of the Output Settling Delay register (OSDR[OSD]) is programmed to a value less than 23 then the initial converted data from SDADC block is “0000” instead of the correct conversion result.

Workaround: Program the OSDR[OSR] value equal to or greater than 23.

ERR008631: SDADC: low threshold watchdog cannot be used with signed data

Description: Each Sigma Delta Analog to Digital Converter (SDADC) provides a watchdog (WDG) to monitor the converted data range. This watchdog should trigger when a converted value is either higher than the value configured in the WDG Threshold Register Upper Threshold Value bit-field (SDADC_WTHHLR[THRH]), or lower than the value configured in the Lower Threshold Value bitfield (SDADC_WTHHLR[THRL]). Instead, the low WDG threshold acts as a high WDG threshold, triggering when a converted value is greater than the value configured in SDADC_WTHHLR[THRL].

Workaround: There are two workarounds available:

1) Do not use the WDG function by clearing the SDADC Module Control Register Watchdog Enable Bit (SDADC_MCR[WDGEN]).

2) Configure the WDG low threshold SDADC_WTHHLR[THRL] to the value 0x7FFF. This guarantees that a low threshold trigger will not be generated. The WDG high threshold (SDADC_WTHHLR[THRH]) can be used without restriction.

ERR005749: SDADC: New conversion data is discarded if the overflow (DFORF) status bit is set

Description: The Sigma-Delta Analog-to-Digital Converter (SDADC) stops filling the Converted Data Register (CDR) and FIFO (if enabled) when the FIFO overrun bit in the Status Flag Register (SFR[DFORF]) becomes set.

The SFR[DFORF] bit becomes set when a FIFO or CDR overflow condition occurs, and once this happens the SDADC stops filling the CDR and FIFO, causing new converted data results to be discarded until the software clears the overflow bit. After clearing the overflow, normal operation resumes.

Workaround: If the Global DMA/Interrupt gating feature is not being used then the problem can be avoided by either of two methods.

1) If DMA is being used then continuously transferring the the CDR or FIFO data via DMA will prevent overflows.

2) If DMA is not used then use an interrupt service routine (ISR) to clear the overflow bit (SFR[DFORF]) and empty the FIFO/CDR

If the Global DMA/Interrupt gating feature is being used then use an interrupt service routine at the start of the DMA gating window to clear the DFORF bit and empty the FIFO and CDR. This will resume normal filling of the CDR. or FIFO which should then be serviced using DMA or ISR for the duration of the gating window.

ERR007356: SDADC: The SDADC FIFO does not function correctly when FIFO overwrite option is used

Description: In the Sigma-Delta Analog-to-Digital Converter (SDADC), when the FIFO Over Write Enable bit (FOWEN) of the FIFO Control Register (FCR) is set (FCR[FOWEN]=1), the following flags of the Status Flag Register (SFR) may not reflect the correct status:

- Data FIFO Full Flag (DFFF)
- Data FIFO Empty Flag (DFEF)

When the number of entries received by the FIFO reaches 2x the FIFO size (field FSIZE of FIFO Control Register (FCR)):

- SFR[DFFF] is cleared, incorrectly indicating the FIFO is not full
- SFR[DFEF] is set, incorrectly indicating the FIFO is empty

The expected behavior is that:

- SFR[DFFF] remains set until data is read out of the FIFO
- SFR[DFEF] remains clear until all data is read out of the FIFO

Workaround: Do not use the FIFO Overwrite option to overwrite FIFO contents. Software shall clear the FIFO overrun condition (if necessary) and flush the FIFO contents before expecting valid data in the FIFO.

ERR007185: SDADC: Watchdog Crossover event missed if PBRIDGEx_CLK less than SD_CLK

Description: In the Sigma-Delta Analog-to-Digital Converter (SDADC), the watchdog monitor Lower and Higher threshold crossover events may get missed if the peripheral bridge clock (PBRDIGEx_CLK) is lower than the SDADC clock (SD_CLK). Therefore, the Watchdog Upper Threshold Cross Over Event (WTHH) and Watchdog Lower Threshold Cross Over Event (WTHL) bits of the Status Flag Register (SDADC.SFR) may not be set and the corresponding Direct Memory Access (DMA) or interrupts are not triggered.

Workaround: When setting the different clocks in the Clock Generation Module (MC_CGM), ensure that PBRIDGEx_CLK is greater than SD_CLK.

ERR007204: SENT: Number of Expected Edges Error status flag spuriously set when operating with Option 1 of the Successive Calibration Check method

Description: When configuring the Single Edge Nibble Transmission (SENT) Receiver (SRX) to receive message with the Option 1 of the successive calibration pulse check method (CHn_CONFIG[SUCC_CAL_CHK] = 1), the number of expected edges error (CHn_STATUS[NUM[EDGES_ERR]) gets randomly asserted. Option 2 is not affected as the number of expected edges are not checked in this mode.

The error occurs randomly when the channel input (on the MCU pin) goes from idle to toggling of the calibration pulse.

Note: The Successive Calibration Pulse Check Method Option 1 and Option 2 are defined as follows:

Option 2 : Low Latency Option per SAE specification

Option 1 : Preferred but High Latency Option per SAE specification

Workaround: To avoid getting the error, the sensor should be enabled first (by the MCU software) and when it starts sending messages, the SENT module should be enabled in the SENT Global Control register (by making GBL_CTRL[SENT_EN] = 1). The delay in start of the two can be controlled by counting a fixed delay in software between enabling the sensor and enabling the SENT module. The first message will not be received but subsequent messages will get received and there will be no false assertions of the number of expected edges error status bit (CHn_STATUS[NUM[EDGES_ERR]).

Alternatively, software can count the period from SENT enable (GBL_CTRL[SENT_EN] = 1) to the first expected calibration pulse. If the number of expected edges error status bit (CHn_STATUS[NUM[EDGES_ERR]) is asserted, software can simply clear it as there have no messages which have been completely received.

Alternatively, the software can clear this bit at the start and move ahead. When pause pulse is enabled, then NUM_EDGES will not assert spuriously for subsequent messages which do not have errors in them or cause overflows.

ERR008082: SENT: A message overflow can lead to a loss of frames combined with NUM_EDGES_ERR being set

Description: In the case of a Single Edge Nibble Transfer (SENT) receiver (Rx) message overflow (CHn_STATUS[FMSG_OFLW] = 1) and if the following registers are continuously being read without clearing the FMSG_RDY[F_RDYn] bit, there is a possibility that one message will be lost. Additionally, if the pause pulse feature is enabled, the module assert up to two NUM_EDGES_ERR in the status register (CHn_STATUS). In this case up to two frames can be lost.

Note that some debuggers perform a continuous read of memory which can cause this issue to occur.

Register	Register Name
CHn_FMSG_DATA	Channel Fast Message Data Read Register
CHn_FMSG_CRC	Channel Cyclic Redundancy Check Register
CHn_FMSG_TS	Channel Fast Message Time-stamp Register

Workaround: 1. Software should ensure that SENT message overflow does not occur.

If interrupts are used (when the Enable FDMA (FDMA_EN) bit of Fast Message DMA Control Register (SRX_FDMA_CTRL) is set to 0) to read the SENT messages, the interrupt for data reception should be enabled by setting the Enable for Fast Message Ready Interrupt (FRDY_IE[n]) bit of Fast Message Ready Interrupt Control Register (SRX_FRDY_IE) for every channel n and the interrupt priority should be such that the software is able to read the message before the next message arrives.

When using Direct Memory Accesses (eDMA) to access the SENT (when the Enable FDMA (FDMA_EN) bit of Fast Message DMA Control Register (SRX_FDMA_CTRL) is set to 1), the DMA request from the SENT module should be serviced before the next message arrives.

The minimum duration between the reception of two consecutive messages in one channel is 92 times the utick length (time).

2. Ensure that the following registers are not read continuously either in the software code or as a result of a debugger being connected. The following registers should be read once per message and the FMSG_RDY[F_RDYn] bit should be cleared after the reads.

Register	Register Name
CHn_FMSG_DATA	Channel Fast Message Data Read Register
CHn_FMSG_CRC	Channel Cyclic Redundancy Check Register
CHn_FMSG_TS	Channel Fast Message Time-stamp Register

ERR007203: SENT: In debug mode SENT message data registers appear to lose contents

Description: The message read registers [Channel ‘n' Fast Message Data Read Register (n = 0 to (CH-1)) (CHn_FMSG_DATA), Channel ‘n' Fast Message CRC Read Register (n = 0 to (CH-1)) (CHn_FMSG_CRC), Channel ‘n’ Fast Message Time Stamp Read Register (n = 0 to (CH-1)) (CHn_FMSG_TS), Channel ‘n’ Serial Message Read Register (Bit 3) (n = 0 to (CH-1))

(CHn_SMSG_BIT3), Channel ‘n’ Serial Message Read Register (Bit 2) (n = 0 to (CH-1)) (CHn_SMSG_BIT2), Channel ‘n’ Serial Message Time Stamp Read Register (n = 0 to (CH-1)) (CHn_SMSG_TS), DMA Fast Message Data Read Register (DMA_FMSG_DATA), DMA Fast Message CRC Read Register (DMA_FMSG_CRC), DMA Fast Message Time Stamp Read Register (DMA_FMSG_TS), DMA Slow Serial Message Bit3 Read Register (DMA_SMSG_BIT3), DMA Slow Serial Message Bit2 Read Register (DMA_SMSG_BIT2) and DMA Slow Serial Message Time Stamp Read Register (DMA_SMSG_TS)] will appear to lose their contents in the following conditions:

(a) The very first message is being received but not yet completely received and the MCU enters debug or freeze mode, the current message reception will get discarded and message read registers (as mentioned above) will read zeros

(b) Auto clear functionality is enabled (GBL_CTRL[FAST_CLR] = 1). In this case, when first message is read, it will get clear the message read registers (due to auto clear functionality being enabled). On reading again, the message read registers (as mentioned above) might read zeros.

Workaround: If the MCU requests entry to debug or stop mode, the message being received currently by SENT Receiver is discarded and the MCU enters debug/stop mode immediately. This does not affect the messages received completely, prior to entering debug mode and these messages will still be present on the message buffer and registers until they are read out.

Thus allow one message to be received completely and do not enable “Auto Clear” (GBL_CTRL[FAST_CLR] = 0), to allow messages to be read in debug or stop mode.

ERR007425: SENT: Unexpected NUM_EDGES_ERR error in certain conditions when message has a pause pulse

Description: When the Single Edge Nibble Transmission (SENT) Receiver (SRX) is configured to receive a pause pulse (Channel ‘n' Configuration Register – CHn_CONFIG[PAUSE_EN] = 1) the NUM_EDGES error can get asserted spuriously (Channel ‘n' Status Register – CHn_STATUS(NUM_EDGES_ERR] = 1) when there is any diagnostic error (other than number of expected edges error) or overflow in the incoming messages from the sensor.

Workaround: Software can distinguish a spurious NUM_EDGES_ERR error from a real one by monitoring other error bits. The following tables will help distinguish between a false and real assertion of NUM_EDGES_ERR error and other errors. Software should handle the first error detected as per application needs and other bits can be evaluated based on these tables. The additional error may appear in the very next SENT frame. Table 1 contains information due to erratum behavior. Table 2 contains clarification of normal NUM_EDGES_ERR behavior.

Table 1. Erratum behavior of NUM_EDGES_ERR

First Error Detected	Other error bits asserted	Cause for extra error bits getting asserted	Action
NIB_VAL_ERR	NUM_EDGES_ERR asserted twice	Upon detection of the first error, the state machine goes into a state where it waits for a calibration pulse, the first NUM_EDGES_ERR error is for the current message as the state machine does not detect an end of message. The	Ignore both NUM_EDGES_ERR error

		second error comes when both the Pause pulse and the Calibration pulse are seen as back to back calibration pulses and no edges in between.	
FMSG_CRC_ERR	NUM_EDGES_ERR asserted twice	Same as NIB_VAL_ERR.	Ignore both NUM_EDGES_ERR errors
CAL_LEN_ERR	NUM_EDGES_ERR asserted once	Since the calibration pulse is not detected as a valid calibration pulse, the internal edges counter does not detect the end of one message and start of bad message (which has CAL_LEN_ERR); hence the NUM_EDGES_ERR gets asserted.	Ignore NUM_EDGES_ERR error
FMSG_OFLW	NUM_EDGES_ERR asserted once (random occurrence)	A message buffer overflow may lead the state machine to enter a state where it waits for a calibration pulse (behavior also seen in ERR007404). When in this state, the state machine can detect both a Pause pulse and a Calibration pulse as back to back calibration pulses and no edges in between. Then, the NUM_EDGES_ERR can get asserted. Since entry into this state is random, the error can be seen occasionally.	Ignore NUM_EDGES_ERR error

Table 2. Expected behavior, clarification of NUM_EDGES_ERR cases

First Error Detected	Other error bits asserted	Cause for extra error bits getting asserted	Action
NUM_EDGES_ERR (when edges are less than expected)	NIB_VAL_ERR is asserted	When the actual number of edges in the message are less than expected, then a pause pulse gets detected as a nibble since the state machine expects nibbles when actually there is a pause pulse present. This generates NIB_VAL_ERR.	Ignore the NIB_VAL_ERR

NUM_EDGES_ERR (when edges are more than expected)	NIB_VAL_ERR and PP_DIAG_ERR are asserted	When the actual number of edges in a message are more than expected, then after receiving the programmed number of data nibbles, the state machine expects a pause pulse. However, the pause pulse comes later and gets detected as a nibble and hence NIB_VAL_ERR is asserted. Since the message length is not correct, PP_DIAG_ERR is also asserted.	Ignore NIB_VAL_ERR and PP_DIAG_ERR

ERR010561: SIPI: Incorrect SIPI_ERR[TOEx] register bit description

Description: The documented bit descriptions for Serial Interprocessor Interface (SIPI) Error Register Timeout Error bits (SIPI_ERR[TOEx], where x= 0, 1, 2, 3) are incorrect.

Incorrect description:

Timeout Error for Channel x.

0 Timeout error occurred

1 Timeout error didn't occur

Correct description:

Timeout Error for Channel x.

0 Timeout error did not occur

1 Timeout error occurred

Workaround: Software accessing the SIPI_ERR[TOEx] bits need to reflect the corrected descriptions.

Timeout Error for Channel x.

0 Timeout error did not occur

1 Timeout error occurred

ERR007788: SIUL2: A transfer error is not generated for 8-bit accesses to non-existent MSCRs

Description: An 8-bit access attempt to non-existent MSCRs (Multiplexed Signal Configuration Registers) in the SIUL2 (System Integration Unit Light 2) address space does not generate a transfer error. 16-bit or 32-bit accesses to non-existent MSCRs will generate a transfer error.

Workaround: Do not expect transfer errors on 8-bit accesses to non-existent MSCRs in the SIUL2 address space.

ERR009780: SIUL2: Open-drain or open-source configured outputs may briefly drive high or low during transitions

Description: The System Integration Unit Lite 2 (SIUL2) Output Drive Control in the Multiplexed Signal Configuration registers (SIUL2_MSCR_IO_n[ODC]) specifies the type of output drive control for the associated pins. When configured as open-drain, the high drive for the output buffer is disabled and when configured as open-source the low drive for the output buffer is disabled. However, during the transition from the driven to non-driven state, the output may momentarily drive the opposite value. In other words, the open-drain output may momentarily drive high and the open-source output may momentarily drive low. The duration of the contention is 0-10 nanoseconds.

Workaround: Open-drain or open source pins need either internal or external weak pull devices to maintain the default pin state. Strong external drivers connected to the pin need to assure a non-overlap time with the pin driver to prevent contention.

ERR007791: SIUL2: Transfer error not generated if reserved addresses within the range of SIUL BASE + 0x100 to 0x23F are accessed

Description: If any reserved register within the System Integration Unit Lite 2 (SIUL2) register range from SIUL2 BASE + 0x100 to 0x23F is accessed then no transfer error will occur.

Workaround: Software should not be dependent on the indication of a transfer error occurring from an access within the SIUL2 register range from SIUL2 BASE + 0x100 to 0x23F.

ERR008605: SMPU: debug SIPI accesses may be blocked by SMPU

Description: The debug Zipwire interface (logical bus master 6), made up of the debug Serial Interprocessor Interface (SIPI_1) and the Low Voltage Differential Signaling Fast Asynchronous Serial Transmission (LFAST_1), is not defined as a valid logical bus master in either of the System Memory Protection Units (SMPU_0 and SMPU_1). As a result, debug accesses from the debug Zipwire interface will be blocked to any memory subsystem connected through the SMPU if the SMPU is enabled.

Workaround: Disable the SMPU to allow accesses from the debug Zipwire interface (SIPI_1/LFAST_1).

ERR007781: SPC5777M: Current injection causes leakage path across the DSPI and LFAST LVDS pins

Description: The General Purpose Input/Output (GPIO) digital pins (including all digital CMOS input or output functions of the pin) connected to the differential LVDS drivers of the Deserial/Serial Peripheral Interface (DSPI), high-speed debug, and LVDS Fast Asynchronous Serial Transmit Interface (LFAST) do not meet the current injection specification given in the operating conditions of the device electrical specification. When the LVDS transmitter or receiver is disabled and current is positively or negatively injected into one pin of the GPIO pins connected to the differential pair, a leakage path across the internal termination resistor of the receiver or through the output driver occurs potentially corrupting data on the complementary GPIO pin of the differential pair. All LFAST and DSPI LVDS receive and transmit GPIO pairs on the SPC5777M exhibit the current injection issue.

There is an additional leakage path for the LFAST receive pins through the loopback test path when current is negatively injected into a GPIO pin connected to an LFAST pair. In this case current will be injected into the same terminal of the GPIO pin connected through the loopback path (terminal to positive terminal, negative terminal to negative terminal). The pins affected by the loopback path on the SPC5777M are: PD[6] to/from PF[13], and PA[14] to/from PD[7].

There is no leakage issue when the pins are operating in normal LVDS mode (both LVDS pairs of the LFAST interface configured as LVDS).

Workaround: As long as the GPIO pad pins are operated between ground (VSS_HV_IO) and the Input/ Output supply (VDD_HV_IO) then no leakage current between the differential pins occurs. If the GPIO pad is configured as an input buffer then DC current injection must be limited to a maximum of 2.5mA. In this case, the adjacent CMOS pin will see a shift in the VOH/VOL levels and A/C timing if configured as an output. If the GPIO pad is configured as an output care should be taken to prevent undershoot/overshoot/ringing during transient switching of capacitive loads. This can be done by carefully configuring the output drive strength to the capacitive load and ensuring board traces match the characteristic impedance of the output buffer to critically damp the rising and falling edges of the output signal.

ERR009658: SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event

Description: In the Serial Peripheral Interface (SPI) module, when both the receive FIFO and shift register are full (Receive FIFO Overflow Flag bit in Status Register is set (SR [RFOF] = 0b1)) and then the Clear Rx FIFO bit in Module Configuration Register (MCR [CLR_RXF]) is asserted to clear the receive FIFO, shift register data is sometimes loaded into the receive FIFO after the clear operation completes.

Workaround: 1. Avoid a receive FIFO overflow condition (SR[RFOF] should never be 0b1). To do this, monitor the RX FIFO Counter field of the Status Register (SR[RXCTR]) which indicates the number of entries in receive FIFO and clear before the counter equals the FIFO depth.

2. Alternatively, after every receive FIFO clear operation (MCR[CLR_RXF] = 0b1) following a receive FIFO overflow (SR[RFOF] = 0b1) scenario, perform a single read from receive FIFO and discard the read data.

ERR006850: SSCM: Nexus enable required for mode changes when a debugger is attached

Description: If the Nexus interface is enabled in the the e200zx cores, even if trace (program, data, ownership, watchpoint, data acquisition) is not enabled, the Nexus Aurora Router (NAR) must be enabled to allow mode changes via the Mode Entry module if debug mode is enabled (debugger connected to the MCU). In addition, trace options must be set to guarantee that none of the NAR trace buffers become full and to prevent new messages from being input into the NAR since some Nexus trace messages are automatically generated regardless whether any trace mode is disabled. Nexus is enabled in the core if any Nexus feature is accessed by a tool (executing the Nexus_enable command to use the Nexus Read/Write Access feature to access memory).

Workaround: The Nexus enable bit (NEN) must be set in the NAR Control Register (NAR_CR[NEN]=0b1) when a debugger is connected to allow messaged to exit the core Nexus module.

ERR007424: SSCM: The SSCM_STATUS register only contains two NXEN status bits

Description: The System Status and Configuration register (SSCM_STATUS) only contains two Nexus Enable status bits (NXEN) to show the status of the e200zx core Nexus enables.

SSCM_STATUS[NXEN1] corresponds to Core_0 Nexus status. SSCM_STATUS[NXEN] corresponds to Core_1 status.

That status of the Core_2 Nexus interface is not supported in the SSCM_STATUS register.

Workaround: If it is required that software know when a tool is connected to the device and if Nexus has been enabled on each core individually, alternate means of tool detection should be used. Alternate methods include using the JTAG Data Communication (JDC) module or the Development Trigger Semaphore (DTS) module to signal between the tool and software which cores have Nexus enabled on them.

The JDC module supports the tool writing to the JTAG Input (JIN) register a value that software can use to to indicate that the tool is connected. Software can then acknowledge that it recognizes the tool by writing a value to the JTAG Output (JOUT_IPS) register in the peripheral memory space.

The DTS module includes a STARTUP register that can be written only by the tool and can be read by software. This STARTUP register defaults to all zeros after reset.

ERR007339: STCU2: STCU2 fault injected by FCCU is self clearing

Description: In the Self-Test Control Unit (STCU2), a fault can be injected by the Fault Collection and Control Unit (FCCU) in order to verify the correct behavior of the interface (fake fault).

The STCU_LMBIST_USR_ERR signal, which is connected to the FCCU input #8, generates only a pulse when an error is injected to this signal by the FCCU.

This is different to other signals from STCU2, where injected faults remain asserted until explicitly cleared.

Workaround: Use a software recoverable fault (select-able with FCCU_RF_CFG) for FCCU input #8, when a fault is injected into the STCU2.

ERR010560: STCU: ESR0 resets may not properly abort online self-tests

Description: When External System Reset 0 (ESR0) is configured as a short reset (in the Reset Generation Module Functional Event Short Sequence Register (RGM_FESS[SS_ESR0] = 1b1)) and ESR0 is asserted during an online self-test operation, then the Self-Test Control Unit (STCU) may not properly abort the online self-test and will not set the Hardware Abort Flag in the STCU2 Error Status register (STCU2_ERR_STAT[ABORTHW]). The STCU operation may complete normally or may complete incorrectly which may result in a STCU watchdog timeout (STCU_ERR_STAT[WDTOSW] = 1b1).

ESR0 long resets (RGM_FESS[SS_ESR0] = 1b0) during an ongoing online STCU self-test will not set the STCU2_ERR_STAT[ABORTHW] abort flag as expected, but may not properly abort the STCU operation. The online self-test operation will be immediately interrupted and the device proceeds with the PHASE1[DEST] reset sequence. The LBIST partition interrupted by the ESR0 long reset may not properly clear the logic on the destructive reset domain within the LBIST partition, for example PIT1 and MEMU in LBIST5; FCCU in LBIST4.

Note: If the online Memory Built In Self-Test (MBIST) with Phase Lock Loop (PLL) Enabled (MBSWPLLEN) or Logic BIST with PLL Enabled (LBSWPLLEN) are set in the Self-Test Control Unit (STCU2) Run Software Register, the STCU2 monitors the PLL1 lock status (even if the PLL1 is not enabled). If PLL1 is disabled or not locked, the STCU may not exit self-test mode. If PLL1 is enabled and locked, but a reset occurs during the self-test sequence, then PLL1 is disabled and clears the lock status which also may not exit self-test mode. If the input reference to PLL1 is interrupted, PLL1 will lose lock and switch to a free running mode until the input reference returns and this also may interfere with properly exiting self-test mode. An unexpected PLL unlock event may also cause (STCU_ERR_STAT[LOCKESW] = 1b1). Recovery from the STCU loss of lock condition will require a non-External System Reset 0 (ESR0) reset, or a STCU2 Watchdog timeout, or a FCCU reset caused by a PLL loss of lock, or an external Power On Reset (PORST).

Running online self-test of lbist4, which contains the Fault Collection and Control Unit (FCCU) logic, may result in spurious settings of FCCU flags along with potential setting of the MC_RGM_DES FCCU failure to react reset status flag (F_FFRR) and MC_RGM_FES FCCU hard reaction reset flag (F_FCCU_HARD). The F_FFRR event will force a destructive reset which will clear the FCCU.

Workaround: Resets caused by ESR0 short or long resets as indicated by a set Functional ESR0 flag in the RGM Functional Event Status register (MC_RGM_FES[F_ESR0]) with incomplete online self-test operations, will need to perform a non-functional reset (in other words, a Power-On reset [PORST], destructive reset, or externally cause a low-voltage detect [POR/LVD]) to recover from any potential issues from an improperly aborted online self-test. Configure the STCU2 Watchdog timeout timer to better assure recover from improperly aborted online self-tests.

Prevent abort issues altogether by avoiding ESR0 resets during online self-test operation.

Always check and clear any spurious effects of the FCCU before the FCCU is used for safety applications. A destructive reset will always clear the FCCU.

ERR010667: STCU: LBIST PRPG start value is 32 bits

Description: The Logical Built-In Self Test (LBIST) controller Pseudo Random Pattern Generator (PRPG) start values are 32 bits but the Reference Manual incorrectly includes information for 64 bits. Thus, the STCU LBIST PRPG High Registers (STCU2_LB_PRPGH0-STCU2_LB_PRPGH9) registers as listed in the “STCU2 memory map” table do not exist and accesses will generate bus errors. The STCU_LB_PRPGH_0-STCU_LB_PRPGH_9 clients as listed in the “DCF client list” table do not exist and programming will have no effect.

Also, keep the STCU LBIST controller Enable PRPG Loading control bit (STCU_LB_CTRLn[PRPGEN]) at the default value of 0 (use the default LBIST value). Thus there is no need to program the STCU LBIST PRPG Low Registers (STCU2_LB_PRPGL0-STCU2_LB_PRPGL9) registers.

In addition, the default start value of the PRPG is 0xFFFF_FFFF. Therefore, to support a MISR value calculated from a seed start value of 0xFFFF_FFFF, the STCU_LB_PRPGL_0-STCU_LB_PRPGL_9 DCF clients are not required to support either offline or online self-test.

Workaround: To prevent bus errors do not access the STCU2_LB_PRPGH0-STCU2_LB_PRPGH9 registers. Keep STCU_LB_CTRLn[PRPGEN]=0.

ERR010607: STCU: STCU watchdog timer timeouts due to low voltage supply excursions

Description: A Self-Test Control Unit (STCU) watchdog timeout, as indicated by the woatchdog Timeout bit in the STCU Error Status Register (STCU2_ERR_STAT[WDTO]==1), may occur during the reset sequence prior to the start of a STCU offline self-test operation thus skipping the execution of the offline self-test.

Some scenarios which may produce this result:

- Slow power-on ramp of the VDD_LV supply while all other supplies are already powered on (for example a ramp rate of approximately 1.3V/0.2S)

- Slow power-on ramp of the VDD_HV_ADV_S supply while all other supplies are already powered on (for example a ramp rate of approximately 5V/0.5S)

- Low voltage excursions that trigger one of the Low Voltage Detects (LVDs) with a duration that exceeds the default time of the STCU watchdog timeout as defined by STCU2_WDG[WDGEOC]. Note that STCU2_WDG[WDGEOC]==0x0000_FFFF by default which equates to about 4mS.

Workaround: After reset, for applications configured to run a STCU offline self-test sequence, check the STCU2_ERR_STAT[WDTO] flag and if set, perform a reset (either PORST, ESR0 long, or any destructive reset) which will re-run the offline self-test sequence.

ERR009322: TDM: Erase of TDR flash block may be blocked by the TDM

Description: Erase of a flash block associated with a Tamper Detect Region (TDR) may be improperly blocked by the Tamper Detect Module (TDM) in a system if the Hardware Security Module (HSM) is also performing a program or erase operation. When this occurs, the erase operation will be shown as complete with Program/Erase Good (c5FMC_MCR.DONE=1 and C55FMC_MCR.PEG=1), but the block will not be erased.

Workaround: Avoid HSM program/erase operations when erasing a flash block covered by a TDR. Alternatively, when erase of a flash block is attempted, but it completes with C55FMC_MCR.PEG=1 and is not erased, a new diary entry should be written before attempting to erase the flash block again.

ERR007948: TDM: Erase protection not enabled by reset

Description: When writing a record to the Tamper Detection Module (TDM) diary to allow erase of a flash block, if functional reset occurs during the erase operation, further erase of the flash block may be allowed until the next destructive reset.

Workaround: Prior to programming a flash block protected by a TDM diary, verify that the tamper region is locked by reading the TDR Status Register (TDM_TDRSR). If the TDRSRx bit of the region indicates the TDR is unlocked, the TDR should be locked (prior to programming) by performing a destructive reset. Other alternatives are to check the TDM_TDRSR after reset, and issue destructive reset if the TDR is unlocked. Or if the TDM_TDRSR indicates the TDR is unlocked, issue another erase operation.

ERR008412: TSENSE: Temperature sensor status flags are incorrect

Description: The temperature sensor over/under temperature status flags (PMC_EPR_TD[TEMP_x]) in the Power Management Controller Event Pending Register (PMCDIG_EPR_TD) that indicate that a temperature threshold was passed, do not operate correctly. The 3 status flags are TEMP_0 (under the cold temperature range), TEMP_2 (temperature over 150C), and TEMP_3 (temperature over 165C).

Workaround: All resets, interrupts, and Fault Collection and Control Unit (FCCU) events should be disabled in the respective PMC registers. This also includes all temperature sensor status flags in the Hardware Security Module (HSM), if it is in use.

Register	Bit	Value
Temperature Sensor Reset Event Enable register (PMCDIG_REE_TD)	TEMP_0	0b0
Temperature Sensor Reset Event Enable register (PMCDIG_REE_TD)	TEMP_2	0b0
Temperature Sensor Reset Event Enable register (PMCDIG_REE_TD)	TEMP_3	0b0
Temperature Sensor configuration register (PMCDIG_CTL_TD)	TS0SIE	0b0
Temperature Sensor configuration register (PMCDIG_CTL_TD)	TS2SIE	0b0
Temperature Sensor configuration register (PMCDIG_CTL_TD)	TS3SIE	0b0
Temperature Sensor FCCU Event Enable register (PMCDIG_FEE_TD)	FEE_TS0	0b0
Temperature Sensor FCCU Event Enable register (PMCDIG_FEE_TD)	FEE_TS2	0b0
Temperature Sensor FCCU Event Enable register (PMCDIG_FEE_TD)	FEE_TS3	0b0
HSM External Sensor Triggers Status register (HSM_ESIE)	ESIE[0]	0b0
HSM External Sensor Triggers Status register (HSM_ESIE)	ESIE[1]	0b0
HSM External Sensor Triggers Status register (HSM_ESIE)	ESIE[2]	0b0
HSM External Sensor Triggers Status register (HSM_ESIE)	ESIE[3]	0b0

Software should always monitor the temperature of the device and if the temperature is over 130C, measures should be taken to reduce the temperature until the temperature is below 130 C (hysteresis window plus additional inaccuracy factor).

ERR007801: WKPU: functional NMI filter enable trigger FCCU fault monitor channel #47

Description: The Wake Up Unit (WKPU) supports a glitch filter for the Non-maskable Interrupt (NMI) pin. This filter can be enabled through the NMI Filter Enable bit in the NMI Configuration Register (WKPU.NCR[NFE0]) and is disabled by default.

When this glitch filter is enabled, the Fault Collection and Control Unit (FCCU) channel #47 will be permanently asserted. This prevents monitoring of the Test Circuitry Group 2 (DFT2) fault reporting associated with channel #47, indicating that circuitry has been put into a non-functional (test) mode.

Workaround: Do not enable the NMI glitch filter (set WKPU.NCR[NFE0]=0).

ERR008738: XBAR: Masters on peripheral shell bus concentrator may stall or fetch data incorrectly

Description: The default programming of bus traffic optimization for a subset of masters on the peripheral shell crossbar (SXBAR/XBAR_1) can cause those masters to stall, receive wrong read data, or get a spurious read access when uncorrectable Error Correction Code (ECC) errors are received from slave modules. For a read transaction following an uncorrectable ECC error from a slave on the computational shell crossbar (FXBAR/XBAR_0), a subset of masters can stall or receive the wrong data. For high-latency situations, a spurious read could occur after the mentioned read transaction. If a master of a bus concentrator on the SXBAR receives an uncorrectable ECC error from a slave on the computational shell crossbar, any master on that concentrator may stall, receive wrong data, or have a spurious read access. The masters on concentrator 0 are the Hardware Security Module (HSM), and the DMA controller(s). The masters on concentrator 1 are the Ethernet controller, Flexray controller(s), SIPI_0 (Zipwire/ Interprocessor Interface), and SIPI_1 (debug Zipwire).

Workaround: Software should disable the Pending Read Enable for Slave 0 of the peripheral shell crossbar in the platform configuration module by clearing the Pending Read Enable S0 bit (PCM.IAHB_BE1.PRE_S0). This bit is initialized to 1 during reset. The PRE_S0 bit should be cleared prior to enabling additional masters, and prior to performing accesses by any peripheral shell crossbar master (Peripheral core 2 load/store, HSM, DMAx, Ethernet, FlexRayx, Zipwire, debug Zipwire) which might encounter an uncorrectable ECC error. After PRE_S0 is disabled, pipe-lined accesses from any peripheral shell crossbar master to a slave on the computational FXBAR (system ram, overlay ram, flash controller, core 0/1 local memory [IMEM/DMEM]) may have increased latency of up to 3 FXBAR clocks.

ERR006994: XBIC: XBIC may trigger false FCCU alarm

Description: The Crossbar Integrity Checker (XBIC) will incorrectly signal a fault alarm when a system bus request results in a bus error termination from a crossbar client. The Fault Correction and Collection Unit (FCCU) alarm number 40 (for XBIC_0) or number 35 for (XBIC_1) will be signaled.

Workaround: Software should handle faults on FCCU alarm #35 and alarm #40 in case of a system bus error.

ERR008310: XBIC: Crossbar Integrity Checker may miss recording information from an initial fault event in the case of back-to-back faults

Description: When the Crossbar Integrity Checker (XBIC) detects back-to-back faults on a system bus path through the crossbar switch (AXBS), the fault information captured in the XBIC Error Status Register (XBIC_ESR) and the XBIC Error Address Register (XBIC_EAR) does not correspond to the initial fault event, but rather the subsequent fault event. While the fault event is properly detected, diagnostic status information in the XBIC_ESR and XBIC_EAR registers describing the initial fault event is lost. This defect can only occur in the event of a series of bus transactions targeting the same crossbar slave target, where the series of bus transactions are not separated by idle or stall cycles.

Workaround: Expect that the XBIC_EAR and XBIC_ESR registers may not contain the initial fault information, but will contain the latest fault information.

ERR008730: XBIC: XBIC may store incorrect fault information when a fault occurs

Description: The Crossbar Integrity Checker (XBIC) may incorrectly identify a fault's diagnostic information in the case when the slave response signals encounter an unexpected fault when crossing the crossbar switch (XBAR) during the data phase. While the fault event is detected, the diagnostic status information stored in the XBIC's Error Status Register (XBIC_ESR) and Error Address Register (XBIC_EAR) does not reflect the proper master and slave involved in the fault. Instead, the preceding master or slave ID may be recorded.

Workaround: Expect that when a fault is reported in the XBIC_EAR and XBIC_ESR registers the actual fault information may be from the preceding transition.

ERR004136: XOSC and IRCOSC: Bus access errors are generated in only half of non-implemented address space of XOSC and IRCOSC, and the other half of address space is mirrored

Description: Bus access errors are generated in only half of the non-implemented address space of Oscillator External Interface (40MHz XOSC) and IRCOSC Digital Interface (16MHz Internal RC oscillator [IRC]). In both cases, the other half of the address space is a mirrored version of the 1st half. Thus reads/writes to the 2nd half of address space will actually read/write the registers of corresponding offset in the 1st half of address space.

Workaround: Do not access unimplemented address space for XOSC and IRCOSC register areas OR write software that is not dependent on receiving an error when access to unimplemented XOSC and IRCOSC space occurs.

ERR007947: XOSC: Incorrect external oscillator status flag after CMU event clear

Description: If an external oscillator (XOSC) is enabled and it becomes unstable (or the crystal fails), the Oscillator Lost Reference status flag in the Clock Monitor Unit Interrupt Status register (CMU0.CMU_ISR[OLRI]) will be set. In addition, the Crystal Oscillator Status flag in the Mode Entry module Global Status Register (MC_ME_GS.S_XOSC) will be cleared (1 = stable clock, 0 = no valid clock). However, if the CMU_ISR[OLRI] is cleared while the oscillator is still in a failing condition, the MC_ME_GS.S_XOSC will incorrectly be set, indicating a valid crystal oscillator.

Workaround: Monitor the XOSC external oscillator status using the MC_ME_GS.S_XOSC before the CMU0.CMU_ISR.OLRI flag is set. After the CMU0.CMU_ISR.OLRI flag has been set, the MC_ME_GS.S_XOSC flag is valid only after a functional reset. Alternately, the response to the OLRI flag after loss of XOSC clock, can be set in the FCCU to cause a functional reset to clear the MC_ME_GS.S_XOSC flag.

ERR009709: XOSC: Oscillator jitter may be impacted by noise on VDD_HV_OSC and may cause PLL jitter

Description: Noise on the oscillator power supply (VDD_HV_OSC) can impact the jitter performance of the oscillator module. In extreme cases, a loss of clock may be signaled.

Workaround: Design the VDD_HV_OSC power network using good noise immunity techniques to reduce noise.

When using the external oscillator functionality, noise on the VDD_HV_OSC must be limited to a maximum amplitude of 500 mV and maximum slope of 150 V/millisecond (ms).

Falling slopes greater than 50 mV and faster than 80 V/ms should be avoided on VDD_HV_OSC to prevent the possibility of a loss of clock. This criteria also limits the long-term oscillator jitter to less than +7 ns and -7 ns.

When using the oscillator as the reference to the Phase Lock Loop 0 (PLL0), the PLL0 may be 10% worse than the specification for the PLL. The combined total jitter is less than 8 ns. This should be taken into account when calculating communication interface speeds.

However, if more accuracy is required in the PLL jitter to meet a specification of +/- 1350ps, the following guidelines can be used:

Noise frequency	**Noise frequency**	* *
Minimum	**Maximum**	**Maximum allowable VDD_HV_OSC variation (peak to peak voltage)**
-	100 KHz	Less than 150 mV
100 KHz	250 kHz	25 mV
250 KHz	5 MHz	4 mV
5 MHz	-	50 mV

ERR007862: ZIPWIRE: Incorrect Device ID returned from SIPI

Description: The Device Identification (ID) returned from a Serial Interprocessor Interface (SIPI) identification request is incorrect. It reads the value as 0x0AF0_E01D, instead of matching the JTAG ID of the device, which is 0x0AF0_F01D.

Workaround: When accessing the Device ID from the SIPI, software should expect the incorrect value of 0x0AF0_E01D instead of 0x0AF0_F01D.

ERR010436: ZipWire: SIPI can have only one initiator with one outstanding write frame at time

Description: The Serial Inter-processor Interface (SIPI) module of the Zipwire interface only supports one initiator and one outstanding write frame at a time.

If a new write is initiated (by setting SIPI_CCRn[WRT] = 0b1, where n is the respective channel number for the transmission), or a new streaming write is initiated (by setting SIPI_CCRn[ST] =0b1) with acknowledgement of a previous frame pending, then the initiator node may get a timeout error (indicated by SIPI_ERR[TOEn]=0b1). The previous write frame last byte may also be corrupted at the target node.

This also means that the target node cannot initiate a write transfer while the initiator node is in the process of a write transfer.

Workaround: The initiator should maintain only one outstanding write/streaming write frame to the target node at any one time.

The user must ensure that before initiating a new write request or initiating a new streaming write that it has received an acknowledgement for the previous write transaction (indicated by SIPI_CSRn[ACKR] =0b1). The write acknowledgement interrupt can be enabled by setting SIPI_CIRn[WAIE]=0b1.

Implement a protocol that ensures both sides of the link cannot initiate a transfer at the same time. For example, a token-passing protocol could be implemented using the SIPI trigger command feature. Send a trigger command to pass the token to the other end of the link. Upon receipt of the trigger command, either initiate a write transfer if one is pending, or pass the token back by sending a trigger command. If a write transfer is initiated, wait until ACK is received and then send a trigger command to pass the token back. In this manner, if each side agrees only to initiate a transfer when it obtains the token, there will be no simultaneous transfers that can cause the problem described.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.