Mask Set Errata for Mask 1N15P

This report applies to mask 1N15P for these products:

- MPC574xP

Mask Specific Information

Major mask revision number	1
Minor mask revision number	3
JTAG identifier	0x39B4501D

Table 1. Errata and Information Summary

Erratum ID	Erratum Title
ERR009061	ADC: ADC operations may not work when ADC_MCR[ADCLKSEL] = 0
ERR050079	CMU: Sudden loss of clock does not signal the Fault Collection and Control Unit
ERR009696	DEBUG: Nexus trace messages may be corrupt when transferred to the debugger via the Aurora interface
ERR009976	DSPI: Incorrect data received by master with Modified transfer format enabled when using Continuous serial communication clock mode
ERR010385	e200z4: Incorrect branch displacement at 16K memory boundaries
ERR006407	e200zx: Circular Addressing issue on LSP Load/Store instructions, and zcircinc instruction
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
ERR010715	eDMA: When master ID replication is enabled, the stored ID and privilege level will change if read by another master.
ERR006358	ENET: Write to Transmit Descriptor Active Register (ENET_TDAR) is ignored
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
ERR007227	FCCU: FCCU Output Supervision Unit (FOSU) will not monitor faults enabled while already pending
ERR010624	FCCU: FOSU may assert destructive reset when a hardware recoverable fault of width less than one safe clock period occurs
ERR007869	FCCU: FOSU monitoring of a fault is blocked for second or later occurrence of the same fault

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR009595	FlexCAN: Corrupted frame possible if Freeze Mode or Low Power Mode are entered during a Bus-Off state
ERR051036	FlexCAN: Dedicated Receive Message Buffers may have its Code Field corrupted if the Receive FIFO function is used
ERR009527	FlexCAN: The transmission abort mechanism may not work properly
ERR009928	FlexPWM: Half cycle automatic fault clearing does not work in PWM submodule 0 under some conditions
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
ERR007274	LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state
ERR008933	LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set
ERR008970	LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State
ERR010639	MC_CGM: Auxiliary clock dividers get stuck if programmed to divide by 2 and a reset occurs during operation
ERR011198	MC_CGM: HALFSYS_CLK clock divider can be stuck in divide by one when shutdown self-test is aborted by external reset which impact the LINFlexD, DMA, SIPI and ENET module functionality along with the Read and Writes to SRAM.
ERR010640	MC_CGM: The device can behave unpredictably, including possible overclocking on the PBRIDGE0/1_CLK domain if its divider is configured to divide by 2, when a reset occurs during a system clock switch to any PLL.
ERR011073	MC_CGM: The system clock divider can become stuck which leads to the unpredictable device behavior
ERR008228	MC_ME: Wakeup from STOP mode may lead to a system hang scenario
ERR008049	MPC574xP: Current injection causes leakage path across the LFAST LVDS pins
ERR010644	NAL: Trace connections to the device are lost if a device reset occurs
ERR010479	NPC: Nexus enable required for mode changes when a debugger is attached
ERR050130	PIT: Temporary incorrect value reported in LMTR64H register in lifetimer mode
ERR011116	PMC: A destructive or functional reset source during PMC Self-test may cause false LVD/HVD event
ERR011060	PMC: Destructive reset can be triggered by the false temperature sensor event caused by the Shutdown Self-test reset
ERR010657	PMC: Low Voltage Detect (LVD) self test may incorrectly indicate a self test fail if the supply is out with its operating range during power up
ERR010875	PMC: The temperature sensor flag can be set incorrectly.
ERR050129	PMC: VDD_LV_CORE is close to the Cold LVD threshold during the device startup
ERR050049	SAR ADC: Incorrect value of ADC power down exit delay evaluated by the formula given in PDEDR [PDED] field description
ERR007204	SENT: Number of Expected Edges Error status flag spuriously set when operating with Option 1 of the Successive Calibration Check method
ERR007425	SENT: Unexpected NUM_EDGES_ERR error in certain conditions when message has a pause pulse
ERR009658	SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event
ERR011027	STCU2: Startup/Shutdown Self-Test can fail because the reset reaction on LVD/HVD is masked during LBIST self-test execution

Table continues on the next page...

Table 1. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR011049	STCU2: STCU watchdog timer timeouts due to supply excursions
ERR010530	STCU2: The self-test watchdog timer will time out if startup self-test is performed after an shutdown self-test followed by long external reset
ERR010531	STCU2: Unexpected STCU self-test timeout can occur when a short sequence for external reset is triggered during execution of shutdown self-test
ERR008967	TSENS: (MPC5744P) Temperature sensor flag glitch during power up
ERR008683	TSENS: Temperature sensor status output bits in PMC_ESR_TD register shows indeterminate behavior
ERR007236	XBIC: XBIC may trigger false FCCU alarm
ERR008730	XBIC: XBIC may store incorrect fault information when a fault occurs
ERR010436	ZipWire: SIPI can have only one initiator with one outstanding write frame at time

Table 2. Revision History

Revision	Changes
SEP 2015	Initial revision
JAN2017	The following errata were added. • e10531 • e9976 • e10676 • e9527 • e10657 • e10385 • e10624 • e9658 • e9696 • e9595 • e10639 • e10640 • e10479 • e9928 • e10715 The following errata were revised. • e7305 • e8933
APR2018	The following errata were removed. • e10676 The following errata were added. • e11073 • e11198 • e11049 • e11027 • e10436

Table continues on the next page...

Table 2. Revision History (continued)

Revision	Changes
	• e10875 • e10900 • e10530 • e10644 The following errata were revised. • e8967 • e10639
JUL2019	The following errata were added. • e50049 • e11116 • e11060 • e50079 • e50119 • e50129 The following errata were revised. • e10639 • e6407 • e8967 • e7227 • e7869
AUG2021	The following errata were added. • ERR009061 • ERR050130 • ERR051036 The following errata were revised. • ERR010479 • ERR010639 • ERR010657

ERR009061: ADC: ADC operations may not work when ADC_MCR[ADCLKSEL] = 0

Description: Any Successive Approximation Register (SAR) Analog to Digital Converter (ADC) operations including calibration, conversions or self test with the Module Configuration Register Analog Clock frequency Selector field (ADC_MCR[ADCLKSEL]) = 0 may never complete or lead to incorrect results.

Workaround: Use the Clock Generation Module (CGM) auxiliary clock divider for ADC_CLK to achieve desired ADC clock frequency for all operations including calibration. See the Reference Manual for the exact CGM registers to perform the configuration. This will lead to all ADC instances to run at the same clock frequency as configured in CGM auxiliary divider. Ensure in any write access to the ADC_MCR register the ADCLKSEL bit is always set to 1.

ERR050079: CMU: Sudden loss of clock 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 is derived from one of the system clock sources: IRCOSC, XOSC, PLL0, PLL1. Loss of a CMU monitored clock can occur if the source clock signal is lost; this can be detected and reported to FCCU using the following mechanisms:

• XOSC loss can be detected by CMU_0 using CLKMN0_RMT supervisor (this particular CMU_0 monitor is not affected by the erratum.) Loss of XOSC will assert the OLR signal which is routed to FCCU fault input NCF[26].

• PLL0 loss can be detected by the PLL Digital Interface (PLLDIG) through PLL0 Status Register (PLLDIG_PLL0SR) asserting the Loss-of-lock flag (LOLF). This flag is routed to the FCCU through fault input NCF[24].

• PLL1 loss can be detected by the PLL Digital Interface (PLLDIG) through PLL1 Status Register (PLLDIG_PLL1SR) asserting the Loss-of-lock flag (LOLF). This flag is routed to the FCCU through fault input NCF[25].

In the event that a fault in the internal clock signal propagation path (from the clock source through Clock Generation Module) causes a monitored clock to stop then CMU Interrupt Status Register bit index 28 (CMU_x_ISR[28]) will be asserted. This bit is not routed to FCCU or an interrupt request so it must be checked by software in the event that an interruption in the monitored clock causes the consuming module to stop functioning.

ERR009696: DEBUG: Nexus trace messages may be corrupt when transferred to the debugger via the Aurora interface

Description: When Nexus tracing is enabled and the Aurora interface is used to transfer those Nexus trace messages to the debugger, it may happen that sometime Aurora frames that contain the trace messages are corrupt and the Aurora frames can not be correctly decoded by the debugger. Consequently the trace messages and their content are lost.

Most likely this situation occurs when the Aurora interface is heavily loaded, i.e. many Nexus trace messages are generated by the Nexus clients within the device and need to be transferred via the Aurora interface.

Workaround: Limit the number of Nexus trace messages to be transferred via the Aurora interface by configuration of the Nexus clients within the device.

ERR009976: DSPI: Incorrect data received by master with Modified transfer format enabled when using Continuous serial communication clock mode

Description: When the Deserial Serial Peripheral Interface (DSPI) module is configured as follows:

1. Master mode is enabled (Master/Slave Mode Select bit in Module Configuration Register is set (DSPI_MCR [MSTR] = 0b1))

2. Modified transfer format is enabled (Modified Transfer Format Enable bit in Module Configuration Register is set (DSPI_MCR [MTFE] = 0b1))

3. Continuous serial communication clock mode is enabled (Continuous SCK Enable bit in Module Configuration Register is set (DSPI_MCR [CONT_SCKE] = 0b1))

In this configuration if the frame size of the current frame is greater than the frame size of the next received frame, corrupt frames are received in two scenarios:

a) Continuous Peripheral Chip Select Enable bit in PUSH TX FIFO Register is set (DSPI_PUSHR [CONT] = 0b1)

b) DSPI_PUSHR [CONT] = 0b0 and lower significant bit of the frame is transferred first (LSB first bit in Clock and Transfer Attributes Register is set (DSPI_CTAR [LSBFE] =0b1))

Workaround: To receive correct frames:

a) When DSPI_PUSHR [CONT] = 0b1, configure the frame size of the current frame less than or equal to the frame size of the next frame (for all frames).

b) When DSPI_PUSHR [CONT] = 0b0, configure DSPI_CTAR [LSBFE] = 0b0. Alternatively, configure the frame size of the current frame less than or equal to the frame size of the next frame (for all frames).

Make sure that for all received frames, the bits are read equal to their respective frame sizes and any extra bits during POP operation are masked.

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.

ERR006407: e200zx: Circular Addressing issue on LSP Load/Store instructions, and zcircinc instruction

Description: The circular addressing mode of the e200zx Lightweight Signal Processing (LSP) Auxiliary Processing Unit for the circular increment instruction (zcircinc) and Load/Store instructions that use the modify form addressing mode (zl*mx, zs*mx) do not wrap properly in some cases when using positive offset.

Workaround: Use one of the following options to workaround the issue.

1. Always use negative offset with these instructions;

or

2. For a small buffer size of 1, 2, 3, or 4 double-words (8,16,24, or 32 bytes), a positive offset can be emulated by using a negative offset value equal to the “desired_positive_offset - buffer length in bytes”. An example for a buffer length of 2 double-words with a desired offset of 2 bytes, an offset of 2-16=-14 can be used;

or

3. Use ODD index and EVEN positive offset greater than 1.

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.

ERR010715: 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 to access the DMA_TCDn_CSR[DONE:START] byte

ERR006358: ENET: Write to Transmit Descriptor Active Register (ENET_TDAR) is ignored

Description: If the ready bit in the transmit buffer descriptor (TxBD[R]) is previously detected as not set during a prior frame transmission, then the ENET_TDAR[TDAR] bit is cleared at a later time, even if additional TxBDs were added to the ring and the ENET_TDAR[TDAR] bit is set. This results in frames not being transmitted until there is a 0-to-1 transition on ENET_TDAR[TDAR].

Workaround: Code can use the transmit frame interrupt flag (ENET_EIR[TXF]) as a method to detect whether the ENET has completed transmission and the ENET_TDAR[TDAR] has been cleared. If ENET_TDAR[TDAR] is detected as cleared when packets are queued and waiting for transmit, then a write to the TDAR bit will restart TxBD processing.

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.

ERR007227: FCCU: FCCU Output Supervision Unit (FOSU) will not monitor faults enabled while already pending

Description: The Fault Collection and Control Unit (FCCU) Output Supervision Unit (FOSU) will not monitor the FCCU reaction to fault inputs that are enabled with an already pending notification. The FOSU monitoring is triggered by an edge from a fault input. The edge detection will be blocked in following cases:

1) When a fault input occurs before it is enabled in the FCCU.

2) When a fault input is enabled in the FCCU and a fault occurs in the CONFIG state.

FOSU edge detection remains blocked until it gets initialized by a FCCU reaction or a destructive reset.

Workaround: Case 1: Before enabling a fault, check if any fault is pending in the corresponding Noncritical Fault Status Register (FCCU_NCF_Sx). If it is any pending, implement one of the workarounds below. Regardless of whether or not the faults are pending and after implementing the workaround (if necessary) subsequently enabling the desired fault will require entering CONFIG mode where it is possible to have Case 2 occur. So proceed to Case 2 handling next.

Case 2: Any time FCCU CONFIG mode is entered for any reason, check for pending faults immediately after exiting CONFIG mode. If any fault is pending, implement one of the workarounds below.

Workaround 1: Generate interrupt FCCU reaction by any fault to recover FOSU monitoring of the pending faults using the Noncritical Fake Fault register (FCCU_NCFF)

Workaround 2: Generate a destructive reset.

Caveat: If the fault in question is found to be pending immediately after reset, then workaround 2 is ineffective and workaround 1 must be employed.

ERR010624: FCCU: 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_NCF_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_NCF_CFGn. Set the Non-critical Fault Configuration bit to a ‘1’, this is the default condition for implemented faults after reset.

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.

ERR009595: FlexCAN: Corrupted frame possible if Freeze Mode or Low Power Mode are entered during a Bus-Off state

Description: In the Flexible Controller Area Network (FlexCAN) module, if the Freeze Enable bit (FRZ) of the Module Configuration Register (MCR) is asserted and the Freeze Mode is requested by asserting the Halt bit (HALT) of the MCR register during the Bus Off state, the transmission

after exiting the Bus-Off condition will be corrupted. The issue occurs only if a transmission is pending before the freeze mode request. In addition, the same issue can happen if Low-Power Mode is requested instead of Freeze Mode.

Workaround: The workaround depends on whether the bus-off condition occurs prior to requesting Freeze mode or low power mode.

A) Procedure to enter Freeze Mode:

1. Set the Freeze Enable bit (FRZ) in the Module Control Register (MCR).

2. Check if the Module Disable bit (MDIS) in MCR register is set. If yes, clear the MDIS bit.

3. Poll the MCR register until the Low-Power Mode Acknowledge (LPMACK) bit in MCR is cleared (timeout for software implementation is 2 CAN Bits length).

4. Read the Fault Confinement State (FLTCONF) field in the Error and Status 1 Register (ESR1) to check if FlexCAN is in bus off state. If yes, go to step 5A. Otherwise, go to step 5B.

5A. Set the Soft Reset bit (SOFTRST) in MCR.

6A. Poll the MCR register until the Soft Reset (SOFTRST) bit is cleared (timeout for software implementation is 2 CAN Bits length).

7A. Poll the MCR register until the Freeze Acknowledge (FRZACK) bit is set (timeout for software implementation is 2 CAN Bits length).

8A. Reconfigure the Module Control Register (MCR).

9A. Reconfigure all the Interrupt Mask Registers (IMASKn).

5B. Set the Halt FlexCAN (HALT) bit in MCR.

6B. Poll the MCR register until the Freeze Acknowledge (FRZACK) bit is set (timeout for software implementation is 178 CAN Bits length).

NOTE: The time between step 4 and step 5B must be less than 1353 CAN bit periods.

B) Procedure to enter in Low-Power Mode:

1. Enter in Freeze Mode (execute the procedure A).

2. Request the Low-Power Mode.

3. Poll the MCR register until the Low-Power Mode Acknowledge (LPMACK) bit in MCR is set (timeout for software implementation is 2 CAN Bits length).

ERR051036: FlexCAN: Dedicated Receive Message Buffers may have its Code Field corrupted if the Receive FIFO function is used

Description: When Dedicated Receive Message buffers are used together with Receive FIFO (Legacy Receive FIFO, (MCR[RFEN] = 0x1), the Dedicated Receive MB Code field can be corrupted if it is locked by the application longer than 20 x CAN bit time. It may turn a Dedicated Receive Message Buffer into any type/status of Message Buffer as defined in the Message buffer structure section in the device documentation.

Workaround: 1) Don’t use Legacy Receive FIFO (MCR[RFEN] = 0x0) together with Dedicated Receive Message buffers.

2) If available on the device, use the enhanced Rx FIFO feature instead of the legacy Rx FIFO together with the Dedicated Receive Message buffers. The Enhanced Rx FIFO is enabled by the ERFEN bit in the Enhanced Rx FIFO Control Register (ERFCR).

3) The defect does not occur if the Receive Message Buffer lock time is less than or equal to the time equivalent to 20 x CAN bit time.

The recommended way for the CPU to service (read) the frame received in a mailbox is by the following procedure:

a. Read the Control and Status word of that mailbox.

b. Check if the BUSY bit is deasserted, indicating that the mailbox is not locked. Repeat step 1) while it is asserted.

c. Read the contents of the mailbox.

d. Clear the proper flag in the IFLAG register.

e. Read the Free Running Timer register (TIMER) to unlock the mailbox

In order to guarantee that this procedure occurs in less than 20 CAN bit times the MB receive handling process in software (step a to step e above) should be performed as a 'critical code section' (interrupts disabled before execution) and should ensure that the MB receive handling occurs in a deterministic number of cycles.

ERR009527: FlexCAN: The transmission abort mechanism may not work properly

Description: The Flexible Controller Area NEtwork (FlexCAN) is not able to abort a transmission frame and the abort process may remain pending in the following cases:

a) If a pending abort request occurs while the FlexCAN is receiving a remote frame.

b) When a frame is aborted during an overload frame after a frame reception.

c) When an abort is requested while the FlexCAN has just started a transmission.

d) When Freeze Mode request occurs and the FlexCAN has just started a transmission.

Workaround: Use the Mailbox Inactivation mechanism instead of the transmission abort mechanism. The Abort Enable bit (AEN) of the Module Configuration Register should be kept cleared and the abort code value "0b1001" should not be written into the CODE field of the Message Buffer Control and Status word.

ERR009928: FlexPWM: Half cycle automatic fault clearing does not work in PWM submodule 0 under some conditions

Description: When

a) the EXT_SYNC signal is selected to cause initialization by setting the Submodule 0 Control 2 Register FlexPWM_SUB0_CTRL2[INIT_SEL] = 11 and

b) a specific FAULTx input is associated with the submodule 0 outputs using the Submodule 0 Fault Disable Mapping Register (FlexPWM_SUB0_DISMAP) and

c) the respective bit for that FAULTx is 0 in the FFULL bitfield of the Fault Status Register FlexPWM_FSTS and

d) the respective bit for that FAULTx is 1 in the FAUTO bitfield of the Fault Control Register FlexPWM_FCTRL,

then the PWM outputs of submodule 0 will only be re-enabled at the cycle boundary (full cycle) and will not be re-enabled at the cycle midpoint (half cycle).

Workaround: When the EXT_SYNC signal is used to cause initialization in submodule 0 and the submodule 0 PWM outputs are disabled by a specific FAULTx input, use full cycle automatic fault clearing for the specific FAULTx input by setting the corresponding bit of the Fault Status Register FlexPWM_FSTS[FFULL] to 1.

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.

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.

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.

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)

ERR010639: MC_CGM: Auxiliary clock dividers get stuck if programmed to divide by 2 and a reset occurs during operation

Description: When any functional reset or destructive reset (besides EXT_POR_B and power on/off) occurs during operation, any auxiliary clock divider in the Clock Generation Module (CGM) that is programmed to divide by 2 and is not sourced by the Internal RC Oscillator (IRCOSC) may get stuck and cannot subsequently be reprogrammed. AUX0_DIV2, AUX1_DIV0, AUX1_DIV1 and AUX11_DIV0 dividers can also stuck during Startup self-test where they are programmed to divide by 2 by internal DCF record which cause LBIST2/3 fail.

A stuck auxiliary clock divider output can be detected by the corresponding Clock Monitor Unit (CMU).

Workaround: The impact on the divider depends on their internal structure. AC0_DC0, AC0_DC1, AC1_DC0, AC1_DC1, AC2_DC0, AC11_DC0 divider can be overclocked (the output clock could be as high as the case of divide-by-1 leading to overclocking of logic) and AC0_DC2, AC5_DC0, AC6_DC0, AC10_DC0 will not have a clock on the output.

A) FlexRay clock divider (AC1_DC0) workaround:

Avoid a peripheral overclocking condition, use one of the following configuration:

1) Use the 40 MHz external crystal oscillator as a clock source for FlexRay.

2) Use 80 MHz PLL clock with divider value 0x0 (divide by 1). This means configure PLL0 to 80 MHz.

B) FlexCAN clock divider (AC2_DC0) workaround:

Avoid a peripheral overclocking condition, use one of the following configurations:

1) Use the external crystal oscillator as a clock source for FlexCAN.

2) Use PLL clock with divider value different than 0x1 (divide by 2).

3) For maximum clock use 80 MHz PLL clock with divider value 0x0 (divide by 1). It means configure PLL0 to 80 MHz.

C) SENT_TIME_CLK clock divider (AC1_DC1) workaround:

Avoid a peripheral overclocking condition, use one of the following configuration:

1) Use divider value different from 0x1 (divide by 2).

2) For maximum clock use 80 MHz PLL clock with divider value 0x0 (divide by 1). This means configure PLL0 to 80 MHz.

3) The divider can be recovered by POR. Use CMU4 to detect that the SENT_TIME_CLK divider gets stuck.

D) Other Dividers:

Changing an auxiliary clock source selection value via software resets all its corresponding dividers and recovers them.

Apply the following sequence after each reset for enabled auxiliary clock dividers that are to be configured to divide by 2 for the application.

1. Disable all CMUs which can be impacted by the steps 3 – 6. For example when the ADC clock divider stuck there are impacted CMU0 and CMU3.

2. Clear the fault flags in the impacted CMUs.

3 Clear the faults caused by the CMU0 (NCF[27]), CMU3 (NCF[30]), CMU4 (NCF[31]) in the FCCU which can be caused by the divider stuck.

4. Disable the corresponding auxiliary clock divider by writing to the Aux Clock Divider Configuration (MC_CGM_ACn_DCx[DE] = 0).

5. Change the auxiliary clock source selection to IRCOSC (MC_CGM_ACn_SC[SELCTL] = 0b0000). For the AUX clock selector 5 divider select different clock source than one used in the application.

6. Select the desired clock source as the auxiliary clock source (e.g. for PLL0 PHI: MC_CGM_AC0_SC[SELCTL] = 0b010).

7. Configure and enable the corresponding auxiliary clock divider by writing to the Aux Clock Divider Configuration (MC_CGM_ACn_DCx[DIV] = 1 and MC_CGM_ACn_DCx[DE] = 1).

8. Wait for get stable clock on the output of the divider. The minimum time is (2*Divider_current_value)/finput. For example when then divider input clock is 40 MHz and it is configured to divide by 2 the minimum delay is (2*2)/40MHz = 100ns.

9. Enabled the CMUs disabled in the step 1.

Note: It is assumed that no safety related tasks are running during the switching of clocks and hence disabling of CMUs don't not have any impact on safety function of the device.

E) Dividers during Start-up SelfTest:

Program following DCF records to change the dividers values from Div by 2 to Div by 3 during startup selftest when there is programmed value 0xFFFF_FFFF at address 0x0040_00F0 in utest flash. When there is different value than 0xFFFF_FFFF at address 0x0040_00F0 the workaround is covered by the internal DCF records and nothing needs to be programmed.

DCF records:

0x13E1A4B8, 0x000401C4

0xEC1E5B47, 0x000401C8

0x09F0D25C, 0x000401D0

ERR011198: MC_CGM: HALFSYS_CLK clock divider can be stuck in divide by one when shutdown self-test is aborted by external reset which impact the LINFlexD, DMA, SIPI and ENET module functionality along with the Read and Writes to SRAM.

Description: When the external reset RESET_B occur during the Shutdown self-test execution the HALFSYS_CLK divider can stuck in divide by 1 and NCF[13] will be set. The impact of the modules which are using the HALFSYS_CLK clock are follow:

ENET: The "hclk" is affected by this behavior and this may cause the ENET to read/write wrong values from/into the SRAM.

SRAM: Read and Writes to the SRAM would be corrupted. Reads may indicate error in ECC.

eDMA:. The register reads and writes to this block causes the core to hang and NCF[13] is set.

SIPI: The "hclk" is affected by this behavior and this may cause the SIPI to read/write wrong values from/into the SRAM and peripherals.

LINFlexD: The Baud clock at the LINFlexD modules doubles. If the frequency at the HALFSYS_CLK is less than or equal to 50MHz, then this can be perceived as a doubling of baud rate at the LINFlexD lines. Anyway the baud clock is overclocked and the output of LINFlexD is corrupted.

Workaround: 1) Before the start of the shutdown self-test the functional event status register (FES) in the reset generation module (MC_RGM) needs to be clear together with the FCCU NCF status registers (FFCU.NCF_Sx.R).

2) When external reset is indicated by the MC_RGM.FES.B.F_EXT bit and the NCF[13] in the FCCU.NCF_ S0.R register is set the HALSYS_CLK clock divider is stuck and the system must be recovered by issuing a POR to the system by either power cycling the device or through the EXT_POR pin.

ERR010640: MC_CGM: The device can behave unpredictably, including possible overclocking on the PBRIDGE0/1_CLK domain if its divider is configured to divide by 2, when a reset occurs during a system clock switch to any PLL.

Description: When a reset occurs during a system clock switch to any Phase Locked Loop (PLL), the device may behave in an unexpected manner due toa glitch on the system clockandpotentialoverclocking of theperipheral clockdomain logic working on the clock from MC_CGM_SC_DC0 (PBRIDGE0/1_CLK).

PBRIDGE0/1_CLK overclocking can only occur if the MC_CGM_SC_DC0 divider is configured to divide by 2 and can be detected by Clock Monitor Unit 2 (CMU_2).

The PBRIDGE0/1_CLK overclocking can be corrected only by a power on reset (POR).

Workaround: To properly clear any logic that may have been affected by the system clock glitch and avoid a peripheral overclocking condition, perform the following steps.

1. Enable ‘functional’ reset escalation: escalate the first ‘functional’ reset to ‘destructive’ reset (MC_RGM_FRET[FRET] = 1).

2. Configure the Dual PLL Digital Interface (PLLDIG) to desired frequencies as per the application requirements.

3. Configure the Clock Generation Module (MC_CGM):

- When selecting the desired PLL as the system clock source, set the divider values as follows:

MC_CGM_SC_DC0 = 0x80030000 (divide by 4)

4. Change the system clock source to the desired PLL:

- Select the desired PLL in the target mode’s configuration register (MC_ME_ **mode** _MC[SYSCLK] = 2 for PLL0 or 4 for PLL1)
- Initiate a mode change via the Mode Control (MC_ME_MCTL) register.
- Wait until the mode transition has completed.

(**mode** is DRUN, RUN0, RUN1, RUN2, or RUN3,)

5. Configure the MC_CGM a second time:

- Set the system clock dividers a second time as the application need (e.g. MC_CGM_SC_DC0 = 0x80010000 (divide by 2))

6. Disable ‘functional” reset escalation (MC_RGM_FRET[FRET] = 0)

7. Continue other configurations normally:

   - Configure the auxiliary clocks as needed (MC_CGM_ACn_SC and MC_CGM_ACn_DCm registers)
   - Enable/disable peripherals as needed (MC_ME_RUN_PCn, MC_ME_LP_PCn, and MC_ME_PCTLn registers).
   - Initiate a mode change via the Mode Control (MC_ME_MCTL) register.
   - Wait until the mode transition has completed.

ERR011073: MC_CGM: The system clock divider can become stuck which leads to the unpredictable device behavior

Description: A clock glitch may occur when the under voltage detects (LVD) or over voltage detects (HVD), which occur during the startup/shutdown of logic self-test (LBIST) or after the startup/shutdown LBIST is finished before the self-test reset occur, may cause the system clock divider configured to divide by 2 to become stuck in divide by 1 mode. If the system clock is greater than 50 MHz, the PBRIDGE0/1_CLK will be overclocked which can lead to unpredictable device behavioral.

Workaround: Program the Clock Monitoring Unit (CMU_2) High Frequency Reference Register (CMU_HFREFR) value to detect that the PBRIDGE0/1_CLK clock is overclocked and if CMU_2 detects that the PBRIDGE0/1_CLK clock is overclocked issue a power on reset by asserting the EXT_POR_B signal

ERR008228: MC_ME: Wakeup from STOP mode may lead to a system hang scenario

Description: If a wakeup is given to the microcontroller (MCU) within 10us during transition into STOP mode the system may hang. If the transition into STOP Mode is not complete and an abort command is issued waking up the MCU within 10us the phase lock loop (PLL) specification if violated leading to an unknown output from the PLL.

Workaround: While transitioning to STOP mode, do not generate a wake-up within 10us of the execution of STOP mode transition

ERR008049: MPC574xP: Current injection causes leakage path across the 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 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 LVDS receive and transmit GPIO pairs on the MPC574xP 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 MPC574xP are C[12] to/from I[5], and G[7] to/from I[6].

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 the input voltage cannot be above the supply, below ground, and no current injection is allowed. 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.

ERR010644: NAL: Trace connections to the device are lost if a device reset occurs

Description: During reset, the Nexus Aurora transmit pins (TXxN/TXxP, where x is 0 through 1) are put into a high impedance state until either self-test completes or the Self-Test Control Unit (STCU) determines that self-test is disabled. In addition, the system clock frequency is reset and the system clock is set to the Internal RC oscillator. Therefore, if the device is reset, the system clock is reset to a frequency that is less than 1/10 of the Nexus trace clock, the connection to a tool will be disconnected. The tool will have to establish a new connection with the device.

Workaround: The user should expect the trace connection to be lost through a reset. Tools will have to be reconnected following the reset and the pins re-enabled.

ERR010479: NPC: 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 Port Controller (NPC) tracing must be enabled to allow mode changes via the Mode Entry module if debug mode is enabled (debugger connected to the MCU) 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: NPC tracing must be enabled by enabling the Message Clock Output (MCKO) in the NPC Port Configuration Register (NPC_PCR) when a debugger is connected to allow messaged to exit the core Nexus module. In addition, the Full Port Mode bit should also be set.

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.

ERR011116: PMC: A destructive or functional reset source during PMC Self-test may cause false LVD/HVD event

Description: The LVDs/HVD events are masking during the PMC self-test while the PMC analog is tested. When a reset occurs during the PMC self-test, the masking of the LVDs/HVDs is removed, but the PMC analog block is still recovering from the PMC self-test.

The LVD/HVD signal that was being tested may keep asserting during this time. This can cause false LVD/HVD event, and the LVD/HVD flag may be set for it. Based on the device configuration the destructive reset can occur.

Workaround: When the destructive reset is selected for LVDs/HVD event do the following step for executing SW triggered PMC selftest:

1. Select functional reset reaction for the LVDs/HVD by configuring PMC_RES_0 register.

2. Run PMC Self-tests.

3. Return the configuration of the LVDs/HVD reset reaction as application required by configuring PMC_RES_0 register.

When the startup LVDs/HVD PMC selftest is interrupted by destructive reset the F_VOR_DEST flag can be set in the MC_RGM_DES register besides the source of the reset.

ERR011060: PMC: Destructive reset can be triggered by the false temperature sensor event caused by the Shutdown Self-test reset

Description: When the shutdown self-test trigger the reset, it can cause the false temperature sensor event at hot and cold temperature which trigger the destructive reset regardless the destructive reset reaction is enabled or nor by the DCF for the temperature sensor event.

Workaround: Run the Normal shutdown selftest with the configuration mentioned in the RM with the following changes. As a consequence there would not be at-speed transition coverage.

1) Clock and system configuration

Set PLL0 to 50 MHz from IRC

dcl_ips_0 set to 0x03008212

MC_CGM_SC_DC0 = 0x80030000

MC_CGM_AC0_SC = 0x02000000

MC_CGM_AC0_DC0 = 0x0x000000

MC_CGM_AC0_DC1 = 0x00070000

MC_CGM_AC0_DC2 = 0x00010000

MC_CGM_AC1_DC0 = 0x80010000

MC_CGM_AC1_DC1 = 0x00010000

MC_CGM_AC2_DC0 = 0x80030000

MC_CGM_AC3_SC = 0x00000000

MC_CGM_AC4_SC = 0x03000000

MC_CGM_AC5_SC = 0x02000000

MC_CGM_AC5_DC0 = 0x00000000

MC_CGM_AC6_SC = 0x02000000

MC_CGM_AC6_DC0 = 0x00070000

MC_CGM_AC10_SC = 0x02000000

MC_CGM_AC10_DC0 = 0x80030000

MC_CGM_AC11_SC = 0x02000000

MC_CGM_AC11_DC0 = 0x80010000

2) STCU configuration changes:

STCU_CFG 0x12100008

STCU_WDG 0x00020000

STCU_LB0_CTRL 0x83071107

STCU_LB0_PCS 0x00000A5A

STCU_LB0_MISRELSW 0x8CBF311B

STCU_LB0_MISREHSW 0xCD9077D8

STCU_LB1_PCS 0x00000540

STCU_LB1_MISRELSW 0xAC435093

STCU_LB1_MISREHSW 0x01599F6C

STCU_LB2_PCS 0x00000b54

STCU_LB2_MISRELSW 0x37732C00

STCU_LB2_MISREHSW 0x23EA1647

STCU_LB2_PCS 0x0000076C

STCU_LB2_MISRELSW 0xB4B9D509

STCU_LB2_MISREHSW 0x 0xF3A1B551

ERR010657: PMC: Low Voltage Detect (LVD) self test may incorrectly indicate a self test fail if the supply is out with its operating range during power up

Description: The Power Management Controller (PMC) executes a self test of the Low Voltage Detect (LVD) and High Voltage Detect (HVD) mechanisms during device start up (at Phase 3 of the reset sequence) after a power on reset or any destructive reset. If a scenario occurs during this self test where a ramping-up supply is out of the operating range when that supply's monitor is being tested, the PMC self test would indicate a failure incorrectly once self test execution has completed (indicated by PMC_VD_UTST [ST_RESULT]=0b0 when PMC_VD_UTST [ST_DONE]=0b1). This means that if a failure is evident after testing it is not certain that a genuine failure has occurred or whether the supply may have been out with the operating ranges whilst the test was executing.

Workaround: In the event of a PMC LVD/HVD self test failure (PMC_VD_UTST[ST_RESULT]=0b0), the user should wait until all supplies return to the correct operating range, and then perform a software initiated PMC self test through PMC_VD_UTST[ST_MODE]=0b01. Once the software initiated self test has completed it will now indicate the correct status via the self-test result bit (PMC_VD_UTST [ST_RESULT]).

ERR010875: PMC: The temperature sensor flag can be set incorrectly.

Description: The Power Management Controller block temperature sensor flags in PMC_ESR_TD register may get set when ADC supply is varied below LVD level and its LVD destructive reset is masked.

Workaround: 1) Enable the LVD destructive reset for the ADC power supply.

2) When the LVD destructive reset is disabled for the ADC power supply:

a) Disable the temperature sensor reset reaction.

b) Check if there is set LVD flag for the ADC power supply when the temperature sensor flag is set.

If the ADC LVD flag is set the following needs to be done:

a) Clear the temperature sensor flag by writing 1 into the PMC_ESR_TD register bitfield.

b) Check if the temperature sensor flag was cleared in the PMC_ESR_TD register.

c) When the temperature flag is still set the temperature sensor detection is correct but when the flag was cleared the flag was set by the ADC power supply variation below LVD.

This is possible workaround because a true temperature sensor event is active much longer than it is necessary for the above workaround sequence.

ERR050129: PMC: VDD_LV_CORE is close to the Cold LVD threshold during the device startup

Description: When the internal regulator is used the VDD_LV_CORE may be up to 50 mV lower thannominal valuewhen device leave the reset. This is caused by the oscillation on the reference voltage of the Power Management Controller (PMC) which is originated by internal

coupling. When there is high current transient on IDD_LV during application start, VDD_LV_CORE drop can occur. Together with lower value of VDD_LV_CORE the Cold Low voltage detector (LVD) can be triggered and generates Destructive reset.

Workaround: When the Cold LVD reset reaction is enabled (PMC_REE_0[VD3RE_C] = 0x1) during the device start up it needs to be disabled by programming PMC_REE DCF record into the utest flash.

ERR050049: SAR ADC: Incorrect value of ADC power down exit delay evaluated by the formula given in PDEDR [PDED] field description

Description: The formula in the register field ADC_PDEDR [PDED] provides the delay between the power down bit reset and start of conversion value in number of clock cycles of the ADC module clock, however the given formula of PDED x 1/[ADC_clock_frequency] is incorrect. This gives a calculated value that is short by 1 cycle of ADC Bus clock and 1 cycle of ADC clock (ADC_CLK).

Workaround: The correct formula that should be used to calculate the value for the ADC_PDEDR[PDED] register is -

(1/ADC Bus clock) + ((PDED+1) x 1/[ADC_clock_frequency])

Where:

ADC_clock_frequency = Frequency of ADC clock (ADC_CLK)

ADC Bus clock= Module interface clock for register access (PBRIDGEx_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.

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 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.	Ignore both NUM_EDGES_ERR error
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	Ignore NUM_EDGES_ERR error

		message (which has CAL_LEN_ERR); hence the NUM_EDGES_ERR gets asserted.	
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

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.

ERR011027: STCU2: Startup/Shutdown Self-Test can fail because the reset reaction on LVD/HVD is masked during LBIST self-test execution

Description: LVDs/HVD reset reaction are masked during LBIST for the SOC and will not result in a reset event during the sequence of LBIST. The PMC_ESR status bits will be set to indicate that such an event occurred during LBIST. FCCU_NCF[13] (RCCU_1 failure) may also be set due to an RCCU mismatch resulting from this event.

Workaround: If the startup/shutdown self-test fails and PMC_ESR bits indicate an LVDs/HVD event, it is advisable to rerun the self-test.

For shutdown self-test, this can be achieved by rerunning the same through software.

For startup self-test, this can be achieved by initiating a software destructive or long functional reset through MC_RGM.

ERR011049: STCU2: STCU watchdog timer timeouts due to supply excursions

Description: A Self-Test Control Unit (STCU) watchdog timeout, as indicated by the watchdog 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 startup self-test operation thus skipping the execution of the startup self-test.

The watchdog time out can be caused by a slow power on ramp of the High Voltage (HV) power supply, ramp rate slower than 2.9V/s, while all other supplies are already powered on or voltage excursions that trigger one of the Low Voltage Detects (LVDs) or High voltage detect (HVD) with a duration that exceeds the default time of the STCU watchdog timeout which equates to about 2.6 ms as per startup self-test configuration.

Workaround: After reset, for applications configured to run a STCU startup self-test sequence, if any of the self-test fails check the STCU2_ERR_STAT[WDTO] flag and if set, perform a reset (either EXT_POR_B or any destructive reset) which will re-run the start-up self-test sequence.

ERR010530: STCU2: The self-test watchdog timer will time out if startup self-test is performed after an shutdown self-test followed by long external reset

Description: The System Status and Control Unit (STCU) shutdown Built-in Self-test (BIST) can be run using 200 MHz PLL as the clock source by programming the DCF clients dcl_ips_0 in software through access over the peripheral bridge (PBRIDGE). If this is done and startup BIST is enabled, after a long external reset is triggered the startup BIST that follows during device start-up will time out as the self-test watchdog awaits flags that are never asserted. This is because these DCF clients are only updated on destructive reset and will remain configured for 200 MHz PLL, which is an invalid configuration for startup BIST. The startup BIST can only be run with 50 MHz PLL setting.

Workaround: Run the Normal shutdown selftest with the configuration mentioned in the RM with the following changes. As a consequence there would not be at-speed transition coverage.

1) Clock and system configuration

Set PLL0 to 50 MHz from IRC

dcl_ips_0 set to 0x03008212

MC_CGM_SC_DC0 = 0x80030000

MC_CGM_AC0_SC = 0x02000000

MC_CGM_AC0_DC0 = 0x0x000000

MC_CGM_AC0_DC1 = 0x00070000

MC_CGM_AC0_DC2 = 0x00010000

MC_CGM_AC1_DC0 = 0x80010000

MC_CGM_AC1_DC1 = 0x00010000

MC_CGM_AC2_DC0 = 0x80030000

MC_CGM_AC3_SC = 0x00000000

MC_CGM_AC4_SC = 0x03000000

MC_CGM_AC5_SC = 0x02000000

MC_CGM_AC5_DC0 = 0x00000000

MC_CGM_AC6_SC = 0x02000000

MC_CGM_AC6_DC0 = 0x00070000

MC_CGM_AC10_SC = 0x02000000

MC_CGM_AC10_DC0 = 0x80030000

MC_CGM_AC11_SC = 0x02000000

MC_CGM_AC11_DC0 = 0x80010000

2) STCU configuration changes:

STCU_CFG 0x12100008

STCU_WDG 0x00020000

STCU_LB0_CTRL 0x83071107

STCU_LB0_PCS 0x00000A5A

STCU_LB0_MISRELSW 0x8CBF311B

STCU_LB0_MISREHSW 0xCD9077D8

STCU_LB1_PCS 0x00000540

STCU_LB1_MISRELSW 0xAC435093

STCU_LB1_MISREHSW 0x01599F6C

STCU_LB2_PCS 0x00000b54

STCU_LB2_MISRELSW 0x37732C00

STCU_LB2_MISREHSW 0x23EA1647

STCU_LB2_PCS 0x0000076C

STCU_LB2_MISRELSW 0xB4B9D509

STCU_LB2_MISREHSW 0x 0xF3A1B551

.

ERR010531: STCU2: Unexpected STCU self-test timeout can occur when a short sequence for external reset is triggered during execution of shutdown self-test

Description: While an shutdown self-test is in progress there is a finite window during the self-test execution during which if an external reset is asserted (RESET_B pulled low) and this reset is configured as short sequence for external reset by setting the Short Sequence for External Reset bit in the Reset Generation Module Functional Event Short Sequence Register (RGM_FESS[SS_EXR] = 1b1), or if another functional reset source is triggered during this window, the time after which the part waits for self-test to complete is longer than expected. This time-out value is governed by the watchdog time-out value set in the STCU Watchdog Register Granularity register (STCU_WDG). Further, the self-test issues signal a time-out (STCU_ERR_STAT [WDTOSW] = 1b1). If the shutdown self-test is being run with PLL enabled then an unexpected PLL unlock event is also observed (STCU_ERR_STAT[LOCKESW] = 1b1).

Workaround: To avoid the longer than expected duration for self-test completion, allow the shutdown self-test to complete without applying external reset when the external reset is configured as a short sequence for external reset. The other functional resets must also not be triggered during the shutdown self-test execution.

ERR008967: TSENS: (MPC5744P) Temperature sensor flag glitch during power up

Description: On a destructive reset generated by a low voltage detection (LVD), high temperature detection, or software there is a point where the Temperature Sensor (TSENS) loads trim values from memory. While this is happening there is a chance of a glitch on the TSENS output status even if the current temperature is within the normal temp range of the device. As a result of this glitch the corresponding TSENS status flag (TEMPx_y) in Power Management Controller's Temperature Event Status registers (PMC_ESR_TD) will get set.

Workaround: After coming out of reset read the PMC_ESR_TD register and check the values of TSENS status flags TEMPx_y. When any of the status flag is set do the following steps:

1) Disable the TSENS module: To disable the TSENS module it is necessary to clear both the digital output enable bit (TSx_DOUT_EN) and the analog output enable bit (TSx_AOUT_EN) in the Temperature Detector Configuration Register (PMC_CTL_TD).

2) Wait at least 10uS.

3) Enable the TSENS module: To enable the TSENS module it is necessary to set both the digital output enable bit (TSx_DOUT_EN) and the analog output enable bit (TSx_AOUT_EN) in the Temperature Detector Configuration Register (PMC_CTL_TD).

4) It has been seen that upon enabling the TSENS that all flag bits (TEMPx_y) may be set and require clearing. This can be done based on of the status of the status bits (TEMPx_y_OP) to determine whether a valid over/under temperature event is there

ERR008683: TSENS: Temperature sensor status output bits in PMC_ESR_TD register shows indeterminate behavior

Description: There are 3 real time status bits as part of the Temperature Sensor (TSENS) that are associated with the -40C trip point, 150C trip point, and 165C trip point. These are the TEMPx_y_OP bits in the Power Management Controller's Temperature Event Status register (PMC_ESR_TD) where x = 0 or 1 (two TSENS instances) and y = 0 ,2 or 3 (-40C, 150C, and 165C flags). These bits reflect the current status of the TSENS output for their respective trip points. There are also 3 status flag bits for each trip point and TSENS instance, TEMPx_y. These bits get set when the temperature exceeds the corresponding threshold and clears when the temperature falls below its corresponding threshold and a one is written to it.

When the temperature crosses the 150C trip point setting the TEMPx_2 flag bit noting the 150C over temp condition the corresponding status bit, TEMP_x_2_OP, may be unstable and oscillate between a high and low state. This status bit should remain high as long as the over temp condition remains, but in this error condition it will toggle. The TEMPx_2_OP status bit is intended to allow the customer to monitor the over temp condition and know when to clear the TEMPx_2 flag bit. In the error condition this status bit could give a false low reading when the temperature is still above 150C.

When the unstable condition is occurring the 165C flag, TEMPx_3, may not set even if the temperature does exceed the 165C trip point. The toggling of the 150C over temp status bit will continue until the temperature is below the hysteresis window for the 150C trip point. After the temperature drops below that hysteresis window the status will correctly reflect a cleared condition.

The probability that the instability will occur will vary with the DCF trim settings (and customer trim settings) for the hot and cold flag trims. The least probable is when the cold flag trim is set to all 0's and the most probable is when the cold flag trim is set to all 1's. Also, while the status bit for the 150C over temp condition is toggling there is some loss of accuracy in the linear temperature sensor converted by the ADC.

Workaround: To get around this issue after the part heats to 150C and the user detects PMC_ESR_TD[TEMPx_2] = 1 disable the TSENS module. To disable the TSENS module it is necessary to clear both the digital output enable bit (TSx_DOUT_EN) and the analog output enable bit (TSx_AOUT_EN) in the Temperature Detector Configuration Register (PMC_CTL_TD). Then enable the temperature sensor by setting both bits back to 1. The status bits (TEMPx_y_OP) will now be operating correctly. It has been seen that upon enabling the TSENS that all flag bits (TEMPx_y) may be set and require clearing. This can be done based on of the status of the status bits (TEMPx_y_OP) to determine whether a valid over/under temperature event is still occurring.

If the TSENS's 150C detection is used to generate system resets through setting of bits in the Temp Reset Event Enable Register (PMC_REE_TD) register either through flash programming in the DCF records or software configuration it is important to note that the TEMPx_y_OP bit is the bit that signals detection to the PMC_REE_TD. A user could decide to not implement the workaround if their desired result of 150C detection was to generate a system reset and continue to do so until temperature dropped below 150C.

Customers who are only concerned about the 165C detection also must enable the 150C flag, TEMPx_2, and implement the workaround. When the unstable condition is occurring the 165C flag, TEMPx_3, may not set even if the temperature does exceed the 165C trip point.

ERR007236: 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 corresponding to the XBIC will be signaled.

Workaround: Software should handle faults on FCCU alarm corresponding to the XBIC in case of a system bus error.

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.

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.