Mask Set Errata for Mask 1N65H

This report applies to mask 1N65H for these products:
- MPC574xP

Mask Specific Information

<table>
<thead>
<tr>
<th>Major mask revision number</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minor mask revision number</td>
<td>1</td>
</tr>
<tr>
<td>JTAG identifier</td>
<td>0x19B4501D</td>
</tr>
</tbody>
</table>

Table 1. Errata and Information Summary

<table>
<thead>
<tr>
<th>Erratum ID</th>
<th>Erratum Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>e9061</td>
<td>ADC: ADC operations may not work when ADC_MCR[ADCLKSEL] = 0</td>
</tr>
<tr>
<td>e8714</td>
<td>ADC: Conversions on an open channel with the presampling feature enabled do not return the expected results</td>
</tr>
<tr>
<td>e7682</td>
<td>ADC: DNL performance is marginal around -1</td>
</tr>
<tr>
<td>e8135</td>
<td>ADC: Dynamic performance parameters ENOB and SINAD are below data sheet value for shared ADC channels in the MAPBGA package for Fin=125 KHz at 3.3V reference</td>
</tr>
<tr>
<td>e7965</td>
<td>ADC: Dynamic performance parameters ENOB and THD below data sheet value for Fin=125 KHz</td>
</tr>
<tr>
<td>e8804</td>
<td>ADC: Setting the DMA request to be cleared on read of data registers does not work</td>
</tr>
<tr>
<td>e7181</td>
<td>ADC: Threshold low violation is undetected in step 0 of self test algorithm C</td>
</tr>
<tr>
<td>e6407</td>
<td>e200zx: Circular Addressing issue on LSP Load/Store instructions, and zcircinc instruction</td>
</tr>
<tr>
<td>e7259</td>
<td>e200zx: ICNT and branch history information may be incorrect following a nexus overflow</td>
</tr>
<tr>
<td>e7305</td>
<td>e200zx: JTAG reads of the Performance Monitor Counter registers are not reliable</td>
</tr>
<tr>
<td>e6966</td>
<td>eDMA: Possible misbehavior of a preempted channel when using continuous link mode</td>
</tr>
<tr>
<td>e6358</td>
<td>ENET: Write to Transmit Descriptor Active Register (ENET_TDAR) is ignored</td>
</tr>
<tr>
<td>e6802</td>
<td>eTimer: Extra input capture events can set unwanted DMA requests</td>
</tr>
<tr>
<td>e8042</td>
<td>FCCU: EOUT signals are active, even when error out signaling is disabled</td>
</tr>
<tr>
<td>e7099</td>
<td>FCCU: Error pin signal length is not extended when the next enabled fault, with its alarm timeout disabled, occurs</td>
</tr>
<tr>
<td>e7227</td>
<td>FCCU: FCCU Output Supervision Unit (FOSU) will not monitor faults enabled while already pending</td>
</tr>
<tr>
<td>e7869</td>
<td>FCCU: FOSU monitoring of a fault is blocked for second or later occurrence of the same fault</td>
</tr>
</tbody>
</table>

Table continues on the next page...
Table 1. Errata and Information Summary (continued)

<table>
<thead>
<tr>
<th>Erratum ID</th>
<th>Erratum Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>e7638</td>
<td>FCCU: Unsuccessful decorated storage access may cause erroneous signaling of FCCU Channel NCF[39]</td>
</tr>
<tr>
<td>e8891</td>
<td>FLASH: (MPC574xP) Address Encode False Report (MCR[AEE] and possible FCCU channels)</td>
</tr>
<tr>
<td>e7990</td>
<td>FLASH: Do not use proprietary sequence addressing for margin reads</td>
</tr>
<tr>
<td>e7989</td>
<td>FLASH: Reading while Erasing, causing ECC Double Bit Error</td>
</tr>
<tr>
<td>e8770</td>
<td>FlexRAY: Missing TX frames on Channel B when in dual channel mode and Channel A is disabled</td>
</tr>
<tr>
<td>e8890</td>
<td>IAHBG: (MPC5474xP) Certain master accesses can stall when uncorrectable ECC errors are received from a slave</td>
</tr>
<tr>
<td>e9426</td>
<td>IRCOSC: MCU may not exit reset due to IRCOSC not starting</td>
</tr>
<tr>
<td>e8128</td>
<td>LBIST: LBIST of the flash may leave flash in an unknown state and stress flash bit cells</td>
</tr>
<tr>
<td>e7013</td>
<td>LINFlexD: Auto synchronization functionality does not work as intended</td>
</tr>
<tr>
<td>e7274</td>
<td>LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state</td>
</tr>
<tr>
<td>e8933</td>
<td>LINFlexD: Inconsistent sync field may cause an incorrect baud rate and Sync Field Error Flag may not be set</td>
</tr>
<tr>
<td>e8970</td>
<td>LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State</td>
</tr>
<tr>
<td>e8228</td>
<td>MC_ME: Wakeup from STOP mode may lead to a system hang scenario</td>
</tr>
<tr>
<td>e8049</td>
<td>MPC574xP: Current injection causes leakage path across the LFAST LVDS pins</td>
</tr>
<tr>
<td>e8013</td>
<td>MPC574xP: RMII_CLK can not be output to external pin</td>
</tr>
<tr>
<td>e8747</td>
<td>PAD_RING: Vih values do not match datasheet</td>
</tr>
<tr>
<td>e8614</td>
<td>PMC: In internal regulation mode device can get stuck in reset and drive 1.2V out of specification for certain 3.3V ramp rates or after a brownout on the 3.3V rail</td>
</tr>
<tr>
<td>e7591</td>
<td>RGM: Possible MC_RGM_FERD and MC_RGMDERD register corruption</td>
</tr>
<tr>
<td>e8634</td>
<td>SAR_ADC: Conversions may fail if Pre-Sampling is enabled</td>
</tr>
<tr>
<td>e7204</td>
<td>SENT: Number of Expected Edges Error status flag spuriously set when operating with Option 1 of the Successive Calibration Check method</td>
</tr>
<tr>
<td>e7202</td>
<td>SENT: Increased tolerance to noise for Nibble length measurement is not available</td>
</tr>
<tr>
<td>e7886</td>
<td>SENT: Jitter tolerance is limited to 1/8 of the utick time</td>
</tr>
<tr>
<td>e7404</td>
<td>SENT: Message overflow in SENT Receiver can lead to stall condition in the MCU</td>
</tr>
<tr>
<td>e7425</td>
<td>SENT: Unexpected NUM_EDGES_ERR error in certain conditions when message has a pause pulse</td>
</tr>
<tr>
<td>e7139</td>
<td>SSCM: The SSCM can incorrectly detect a configuration error</td>
</tr>
<tr>
<td>e8967</td>
<td>TSENS: (MPC5744P) Temperature sensor flag glitch during power up</td>
</tr>
<tr>
<td>e8683</td>
<td>TSENS: Temperature sensor status output bits in PMC_ESR_TD register shows indeterminate behavior</td>
</tr>
<tr>
<td>e7236</td>
<td>XBIC: XBIC may trigger false FCCU alarm</td>
</tr>
<tr>
<td>e8730</td>
<td>XBIC: XBIC may store incorrect fault information when a fault occurs</td>
</tr>
</tbody>
</table>

Table 2. Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>APR 2015</td>
<td>The following errata are now described in the device Reference Manual or Datasheet:</td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
Table 2. Revision History (continued)

<table>
<thead>
<tr>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7230, 7226, 8004, 7991, 7589, 7643, 7250, 7103, 7858, 6574, 6726, 7974, 8082, 7788, 8014, 7251, 7352, 6904, 8137, 7338</td>
</tr>
<tr>
<td>SEP 2015</td>
<td>The following errata were added: 7274, 8967, 8770, 8228, 8747, 8970, 8890, 9061, 8891, 8804, 8714, 8683, 8634, 8614, 8730, 8933</td>
</tr>
<tr>
<td>SEP 2015</td>
<td>The following errata were revised: 7227, 8042</td>
</tr>
<tr>
<td>SEP 2015</td>
<td>The following errata were added.</td>
</tr>
<tr>
<td>SEP 2015</td>
<td>• e9426</td>
</tr>
<tr>
<td>SEP 2015</td>
<td>• e7202</td>
</tr>
</tbody>
</table>

**e9061: 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.

**e8714: ADC: Conversions on an open channel with the presampling feature enabled do not return the expected results**

**Description:** The analog-to-digital converter (ADC) presampling feature will precharge an ADC channel sample capacitor to an internal voltage. The internal voltage is selected by the Internal Voltage Selection for Presampling bit field (PREVAL0) of the ADC’s Presampling Control Register (ADC_PSCR). The user has either the option of sampling an ADC reference voltage rail (VDD) or a ground (VSS). If the convert presampled value bit (PRECONV) of the ADC_PSCR register is cleared (ADC_PSCR[PRECONV]=0b0) then the presampling stage is followed by the sampling of the ADC channel input and then the conversion is performed. If the user has selected VSS as the internal sample voltage when this is done on an open or unconnected channel, the conversion result will be closer to 1000 when it is expected to be 0. If the user has selected VDD as the internal voltage the conversion result will be closer to 2000 rather then 4095. If ADC_PSCR[PRECONV]=0b1 then sampling of the ADC channel input is bypassed and the presampled voltage is converted directly. This conversion result is close to the expected value for the presampled voltage.

**Workaround:** Do not expect conversion result to be close to zero or full-scale on an open channel with presampling enabled and ADC_PSCR[PRECONV] = 0b0.
e7682: ADC: DNL performance is marginal around -1

Description: The 12 bit ADC shows marginal behavior with respect to the low end of its Differential Non-Linearity specification (DNL= -1).

Workaround: To improve the performance the ADC needs to be run in a different configuration. To change the configuration the user needs to change the operational mode bits (OPMODE) of the ADC’s Calibration, BIST control and status register (ADC_CALBISTREG). They default to 0b001 and they need to be change to 0b110, ADC_CALBISTREG[OPMODE] = 0x6. This configuration takes an extra 4-cycles during conversion. The user may counter act this by making the input sample time shorter by changing the value of the sample period (INPSAMP) in the ADC Conversion Timing Register 0 (ADC_CTR0[INPSAMP]) in case that the total conversion and sampling time needs to be kept within 1µs.

e8135: ADC: Dynamic performance parameters ENOB and SINAD are below data sheet value for shared ADC channels in the MAPBGA package for Fin=125 KHz at 3.3V reference

Description: The Analog to Digital Converter (ADC) dynamic performance parameter value for ENOB (Effective Number of Bits) and SINAD (Signal to Noise And Distortion Ratio) are not met at the input signal frequency of 125 KHz and VREF_AD0/1 of 3.3V for few shared channels of ADC1, ADC2, and ADC3 on the 257-ball Molded Array Process Ball Grid Array (MAPBGA) package.

ADC1 Channels: AN[11]:ADC0/ADC1, AN[12]:ADC0/ADC1, AN[13]:ADC0/ADC1, AN[14]:ADC0/ADC1
ADC2 Channels: AN[0]:ADC2/ADC3, AN[1]:ADC2/ADC3, AN[2]:ADC2/ADC3
ADC3 Channels: AN[4]/AN[3]:ADC1/ADC3, AN[5]/AN[4]:ADC1/ADC3, AN[6]/AN[5]:ADC1/ADC3, AN[7]/AN[6]:ADC1/ADC3, AN[8]/AN[7]:ADC1/ADC3

ENOB will be >10.2b and SINAD will be >63dB for these channels.

At VREF_AD0/1 of 5.0V, all ADC channels work as per data sheet. For lower input frequency (<=10KHz), all ADC channels work as per data sheet.

Workaround: User needs to expect a lower ENOB and SINAD value for shared ADC channels listed on the MAPBGA at input signal frequencies higher than 10 KHz while operating at VREF_AD0/1 of 3.3V.

e7965: ADC: Dynamic performance parameters ENOB and THD below data sheet value for Fin=125 KHz

Description: The Analog to Digital Converter (ADC) dynamic performance parameter values for ENOB (Effective Number of Bits) and THD (Total Harmonic Distortion) are not met at the input signal frequency of 125 KHz and VREF_AD0/1 of 3.3V.

For Fin<=50 KHz the value of ENOB=10.5 bits and THD=65 dB is met.
For 50KHz<=Fin<=125 KHz, ENOB is expected to be ~10.1 bits.
At VREF_AD0/1 of 5.0V, the ADC works as per data sheet.
**Workaround:** User needs to expect the lower ENOB and THD value at input signal frequencies higher than 50 KHz while operating at VREF_AD0/1 of 3.3V.

---

**e8804: ADC: Setting the DMA request to be be cleared on read of data registers does not work**

**Description:** The Successive Approximation Register (SAR) Analog-to-Digital Converter (ADC) can generate DMA requests to the DMA controller. If the DMA Clear sequence enable bit in the ADC DMA Enable register (ADC_DMAE[DCLR]) is set to 0b1 the DMA request should be cleared upon a read of the data registers but it is cleared automatically without a read.

**Workaround:** Do not use the ADC module with the DMA request cleared on read of data registers enabled, ADC_DMAE[DCLR]=0b1. Instead configure ADC_DMAE[DCLR]=0b0. With DMA enabled, the request will only be cleared once the DMA controller acknowledges it and accesses the conversion data register.

---

**e7181: ADC: Threshold low violation is undetected in step 0 of self test algorithm C**

**Description:** If self-test is enabled (in scan or one-shot mode) the SAR ADC runs additional conversions for self-test at the end of a normal conversion chain. Various events like end of conversion, end of the self test algorithm, and violation of a threshold at any step of any algorithm can be captured during self-test execution. Step 0 of the algorithm C self-test does not generate an error when the conversion result for the step goes below the low threshold value programmed in the Self-Test Analog Watchdog Register 4, ADC_STAW4R[THRL].

**Workaround:** The workaround is different depending on whether you are using one-shot mode or scan mode. If the user is using one-shot mode they can either avoid executing step 0 of algorithm C or execute it knowing they won’t be notified of a low threshold violation. If they need to execute step 0 and catch any low threshold violations then the below interrupt based workaround should be implemented. If the user is using scan mode and they need to catch any low threshold violations of step 0 of algorithm C then also the following interrupt based workaround should be implemented.

The following setup needs to be done during ADC initialization if using one-shot mode.

1. Enable analog watchdog for step 0 of algorithm C, STAW4R[AWDE] = 1b.
2. Enable end-of-conversion interrupt for self test, ADC_STCR2[MSKST_EOC] = 1b.

The following setup needs to be done during ADC initialization if using scan mode.

1. Enable analog watchdog for step 0 of Algorithm C, STAW4R[AWDE] = 1b.

The following processing is to be used for either one-shot or scan mode upon generation of an interrupt.

1. Check the interrupt type by reading the Self Test Status Register (ADC_STSR1).
2. If the interrupt is for end-of-algorithm C (ADC_STSR1[WDG_EOA_C]=1b) then do the following:
   a. Clear the interrupt by writing 1b to it, ADC_STSR1[WDG_EOA_C] = 1b
   b. Enable self-test end-of-conversion interrupt in the Self Test Configuration Register 2
      (ADC_STCR2[MSKST_EOC]=1b)
   c. Clear self-test data register (ADC_STDR1) by reading it; this will un-block the loading of
      data on it for subsequent self test conversion
3. If the interrupt is for self-test end-of-conversion (ADC_STSR1[ST_EOC]) then do following:
   a. Clear the interrupt by writing 1b to it, ADC_STSR1[ST_EOC]=1b
   b. If scan mode, disable end-of-conversion interrupt for self test
      (ADC_STCR2[MSKST_EOC]=0b)
   c. Read conversion data from data register ADC_STDR1[TCDATA] and low threshold of step 0
      of algorithm C from ADC_STAW4R[THRL]
   d. Check if the data is negative and violates the threshold value. The MSB of
      ADC_STDR1[TCDATA] is the sign bit for the data value.
      If it is 1b then the data is negative. The threshold is always negative and if the data falls below
      it then there is a violation.
      ADC_STDR1[20]=1b && (ADC_STDR1[TCDATA] < ADC_STAW4R[THRL])
   e. If above condition is true then flag an error to the respective process to handle error
      scenarios.

There are some limitations to this workaround. This works with 4 or more channels in a chain.
It is observed that with minimal load the processor can handle the ISR in ~2.8 uS of generation
of interrupt. The user must observe for sufficient gap between assertion of EOA_C (End of C-
Algorithm interrupt) followed by first EOC (End of conversion) for self test conversion. This gap
is defined by the chain length (1 channel = ~1 uS) or the baud rate of self test. The gap must
be 4 uS or more depending on load on processor and ISR. If the step is missed it may produce
false alarm for any next step(s).

**e6407: 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 (zcircinc) and Load/Store (zl*, zs*) instructions 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 instructionss;
   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 “buffer length in bytes -
desired_positive_offset”. 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.
**e7259: 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.

**e7305: e200zx: JTAG reads of the Performance Monitor Counter registers are not reliable**

**Description:** Reads of the Performance Monitor Counter (PMC0, PMC1, PMC2, and PMC4) 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.

**e6966: 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 preemption channel will experience the normal startup latency (one read/write sequence + startup) instead of the shortened latency (startup only) provided by continuous link mode.

### e6358: 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.

### e6802: eTimer: Extra input capture events can set unwanted DMA requests

**Description:** When using the DMA to read the eTimer channel capture registers (ETIMER_CHn_CAPTn) and the DMA has completed its programmed number of transfers an extra input capture event will set the eTimers input capture flag bit in the status register (ETIMER_CHn_STS[ICFn]) and also set the internal DMA request signal. While the input capture flag status bits (ICFn) can be cleared by writing a 1 to their bit positions the DMA request can only be cleared by the DMA done signal. This means that when a new DMA transfer is programmed the eTimer will request a DMA read with possibly unwanted data.

This behavior occurs once the DMA requests are disabled on the side of eDMA (DMA_ERC[ERQn] = 0), but are still enabled in eTimer (ETIMER_CHn_INTDMA[ICFnDE] = 1), and the active edge is detected

**Workaround:** In cases where extra eTimer input capture events might occur the following procedure can be used to prevent unwanted DMA read requests:

1. Upon completion of the DMA transfer, disable the DMA requests by clearing the ETIMER_CHn_INTDMA[ICFnDE] bits.
2. If ETIMER_CHn_STS[ICFn] bits are clear then there are no extra input capture events and the eTimer is ready for further operation.
3. If the ICFn bits are set then read the ETIMER_CHn_CAPTn registers until the ETIMER_CHn_CTRL3[CnFCNT] fields are both 0 indicating the the capture FIFO’s are empty. Then write a 1 to the ICFn bits to clear them. Next, create a dummy DMA read transfer to read the CAPTn registers. The DMA done signal will clear any pending DMA request.

### e8042: 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.

**e7099:** **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 arises 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 $T_{\text{min}}$, 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].

**e7227: 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 is disabled in the FCCU and a fault occurs,
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:** Apply the following procedure when enabling fault inputs in the FCCU in order to ensure correct monitoring by the FOSU:

1. Check for FCCU pending faults and clear them.
2. Configure the FCCU as desired. In addition enable fault input for interrupt reaction (software recovery mode) to an injected error on this input.
3. Immediately on exiting the CONFIG state, check for FCCU pending faults. If there is a fault status set then initiate a destructive reset.
4. Clear the FOSU status by injecting a fault on the FCCU fault input configured for software recovery mode. This will generate a FCCU reaction that will clear the FOSU edge detection logic.

Apply the following procedure when exiting FCCU CONFIG state in order to ensure correct monitoring by the FOSU:

- Check for FCCU pending faults. If there is a fault status set then initiate a destructive reset.

**e7869: 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:** There are two possible workarounds. Either one can be used with same effectiveness.

1. In addition to the reset reaction, enable either the interrupt (IRQ) or Non-maskable Interrupt (NMI) or error out signaling reaction for the faults that have a reset reaction enabled.
2. Apply the following procedure during the FCCU configuration after a reset and in the fault service routine while clearing the fault status inside the FCCU.

i. Check for FCCU pending faults and clear them.

ii. Configure the FCCU as desired.

iii. Enable a fault as software recoverable by setting its corresponding bit in the NCF Configuration Register (FCCU_NCF_CFGn)

iv. Inject a fake fault to the fault set up in step “iii” by writing the corresponding code into the NCF Fake Register (FCCU_NCFF)

v. Check that there are no pending faults else clear the pending faults and repeat steps “iv” and “v”

vi. Reconfigure the fault that was configured for software recovery mode.

e7638: 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.

e8891: FLASH: (MPC574xP) 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 occur if both the read and write operations to flash occur in even numbered RWW partitions, or if both the read and write occur in odd numbered RWW partitions.

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 false AEE event issue.

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

Workaround: Option 1:

Disable Non-Critical Fault 33 (NCF[33]) and Non-Critical Fault 36 (NCF[36]) in the Fault Collection and Control Unit (FCCU) by clearing the corresponding bits in Non Critical Fault Enable Register 1 (FCCU_NCF_E1). NCF[33] is the AEE event indication and is controlled by
bit 30 in FCCU_NCF_E1. NCF[33] is also used for voltage and read reference errors of the flash memory array. Disabling NCF[33] removes visibility to these errors. NCF[36] is the flash controller mismatch event indication and is controlled by bit 27 in FCCU_NCF_E1.

Option 2:

After receiving both NCF[33] and NCF[36], check the Flash’s Module Configuration Register (C55FMC_MCR) for possible causes of these faults. If the AEE bit is set, clear it and treat NCF[33] and NCF[36] as invalid. If the Read Voltage Error (RVE) bit is set or the Read Reference Error (RRE) bit is set, treat NCF[33] and NCF[36] as valid and respond appropriately.

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, faults that would be detected by AEE are still detected by other mechanisms as part of the MPC574xP 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.

e7990: FLASH: Do not use proprietary sequence addressing for margin reads

Description: To issue a Margin Read operation to the Flash the Margin Read Enable bit (MRE) of the Flash’s UTest 0 register (C55FMC_UT0) is set. Before executing a Margin Read operation the user needs to select which address sequence it uses. The user can select either Sequential Sequence or a Proprietary Sequence that is meant to replicate the sequence normal user code follows. The sequence choice is controlled by the Array Integrity Sequence (AIS) bit in the C55FMC_UT0 register. For Margin Reads it is recommended to use Sequential Sequence (C55FMC_UT0[AIS]=1).

However, if the Proprietary Sequence is selected (C55FMC_UT0[AIS]=0) and the user performs multiple consecutive Margin Read operations, if an Array Integrity Suspend is issued, by setting the Array Integrity Suspend bit (C55FMC_UT0[AISUS]=1), immediately after the second Margin Read operation is started, the Margin Read may not run correctly and the Multiple Input Signature Registers (MISR) may not update after a resume from the suspend.

Workaround: Do not use the Proprietary Sequence addressing Margin Read, and instead use the Sequential Sequence (C55FMC_UT0[AIS]=1) for Margin Read operations.

If the Proprietary Sequence is required, on back to back Margin Read requests, ensure that on the second Margin Read operation that a suspend request only occurs after the MISR registers are updated.

e7989: FLASH: Reading while Erasing, causing ECC Double Bit Error

Description: Reads to flash memory locations available for read-while-write (RWW) performed while erasing 256kB code flash blocks may result in an Error Correction Code (ECC) double bit detection event. The possibility to observe this behavior is increased as the operating temperature raises. Programmed bits (0’s) are falsely read as erased (1’s), even though the actual flash bit cell state does not change.

Workaround: Do not performs reads on 256kB blocks that are associated with a RWW partition that has erase operation ongoing.
If reads must be performed on 256kB block in an associated RWW partition while another block is being erased, the exception handler for ECC double bit detection events should check if an erase is being performed, and if it is, re-read the locations after the erase operation is completed or suspend the erase operation to read the flash locations.

For cases where executing code from flash to erase and reprogram 256kB code blocks, first copy the flash loader to SRAM and execute from SRAM to avoid reading flash available for read-while-write while erasing the 256kB code flash blocks.

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

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

**Workaround:**
1. Configure the FlexRay module in Single Channel mode (FR_MCR[SCM]=1) and enable Channel B (FR_MCR[CHB]=1) and disable Channel A (FR_MCR[CHA]=0). In this mode the internal Channel A behaves as FlexRay Channel B. Note that in this mode only the internal channel A and the FlexRay Port A is used. So externally you must connect to FlexRay Port A.
2. Enable both Channel A and Channel B when in Dual Channel mode (FR_MCR[CHA]=1 and FR_MCR[CHB]=1). This will allow all configured TX frames to be transmitted correctly on Channel B.

e8890: **IAHBF: (MPC5474xP) Certain master accesses can stall when uncorrectable ECC errors are received from a slave**

**Description:** The default configuration of bus traffic optimization for the Ethernet, LFAST/SIPI, and eDMA masters on the crossbar (XBAR) can cause those masters to stall, receive wrong read data, or get a spurious read access when uncorrectable ECC errors are received from slaves.

**Workaround:** Software should disable the Pending Read Enable feature by clearing the Pending Read Enable bit (PRE) in the Platform Configuration Module Bus Bridge Configuration Register 1 (PCM_IAHB_BE1). From power on reset the PCM_IAHB_BE1[PRE] bit is set.

e9426: **IRCOSC: MCU may not exit reset due to IRCOSC not starting**

**Description:** The Internal RC Oscillator (IRCOSC) is held in reset whenever the VDD_HV or VDD_LV supplies are below their power-on-reset (POR) level. Once all supplies are above their POR level, the IRCOSC is released from reset and the IRCOSC should begin generating a clock...
signal. In rare instances the IRCOSC will fail to generate a clock signal. When this occurs the MCU is unable to exit PHASE0 of the reset process, as defined by the Reset Generation Module (MC_RGM).

**Workaround:** A power cycle is required to restart the IRCOSC. An external watchdog should be present that is capable of cycling the MCU power supply. The MCU should provide an activity indicator to this watchdog immediately after exiting reset. If the watchdog does not receive the activity indicator then the watchdog should cycle the power supply.

**e8128: LBIST: LBIST of the flash may leave flash in an unknown state and stress flash bit cells**

**Description:** Performing a Logical Built-In Self-test (LBIST) on the flash leaves the flash in an unknown state prior to reset. During this time (after performing the LBIST and the microcontroller [MCU] being reset), the flash array may be disturbed, which could potentially lead to data corruption.

**Workaround:** To avoid the possibility that the flash array is disturbed, do not perform LBIST on the partition which contains the flash. The customer needs to assess the safety impact to the overall system level safety concept of no LBIST fault coverage on the partition which contains the flash.

LBIST is a safety mechanism used to detect potential permanent faults in the MCU digital logic. When LBIST is disabled, any faults which would have been detected by LBIST during start-up (in other words, at time 0, before the application runs) will no longer be detected at that point in time. Nevertheless, a subset of these faults will still be detected by other safety mechanisms during application runtime, likely as soon as the application starts.

The LBIST coverage for all modules in the LBIST partition which is disabled will be 0% (reduced from ~90% stuck-at fault coverage of digital logic). This represents ~10% of the MCU. This impacts the Crossbar Switch (XBAR), Decorated Storage Memory Controller (DSMC), Direct Memory Access Controller (DMA_0), Embedded Flash Memory (C55FMC), Error Injection Module (EIM), Flash Memory Controller (PFLASH), Interrupt Controller (INTC), Logic Built In Self Test (LBIST), Main Core_0, Memory Built In Self Test (MBIST), Memory Error Management Unit (MEMU), Nexus debug modules, RAM Controller (PRAMC), Software Watchdog Timer (SWT), System Memory Protection Unit (SMPU), System Timer Module (STM), and Register Protection. For a detailed list of how each module is impacted please contact your local Freescale representative.

When assessing the impact of disabling the LBIST of the partition that contains the flash for the Failure Modes, Effects, and Diagnostic Analysis (FMEDA), the LBIST safety mechanism should be disabled in the SRAM FMEDA, Flash FMEDA and Core FMEDA. However, despite this, the overall ISO 26262 Latent Fault Metric target is still achieved since the failure rate of permanent die faults is very low (~1 FIT for the MCU according to IEC TR 62380) and only a very small portion of the MCU faults are considered Latent Faults (the majority are considered to be Single-Point Faults and are covered by other safety mechanisms, and some Safe Faults).

**e7013: LINFlexD: Auto synchronization functionality does not work as intended**

**Description:** When the Local Interconnect Network module (LINFlexD) is configured in LIN slave mode with the LIN Auto-synchronization Enable bit in LIN Control Register 1 (LINCR1[LASE]) set, once the auto-synchronization is complete during the header reception, the ‘autosync_comp’ bit of
the LIN Status Register (LINSR[autosync_comp]) register is cleared in the subsequent clock cycle after being asserted. User software can not poll the autosync_comp bit to detect whether auto synchronization is complete.

Workaround: During reception of header, check the completion of Auto-synchronization by reading a value of ‘6’ in the LIN state bits of LINSR (LINSR[LINS]).

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

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

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

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

1) Configure slave to Set the MODE bit in the LIN Time-Out Control Status Register (LINTCSR[MODE]) to ‘0’.
2) Configure slave to Set Idle on Timeout in the LINTCSR[IOT] register to ‘1’. This causes the LIN Slave to go to an IDLE state before the next header arrives, which will be accepted without any framing error.
3) Configure master to wait for Frame maximum time (T Frame_Maximum as per LIN specifications) before sending the next header.

**Note:**

\[
\text{THeader}\_\text{Nominal} = 34 \times \text{TBit}
\]

\[
\text{TResponse}\_\text{Nominal} = 10 \times (\text{NData} + 1) \times \text{TBit}
\]

\[
\text{THeader}\_\text{Maximum} = 1.4 \times \text{THeader}\_\text{Nominal}
\]

\[
\text{TResponse}\_\text{Maximum} = 1.4 \times \text{TResponse}\_\text{Nominal}
\]

\[
\text{TFrame}\_\text{Maximum} = \text{THeader}\_\text{Maximum} + \text{TResponse}\_\text{Maximum}
\]

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

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

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

- LIN (Local interconnect network) slave mode is enabled by clearing the Master Mode Enable (MME) bit in the LIN Control Register 1 (LINCR1)
- Auto synchronization is enabled by setting the LIN Auto Synchronization Enable bit (LASE) in the LINCR1 register
- Sync Field value is not equal to 0x55

the LINFlexD module may automatically synchronize to an incorrect baud rate without setting the Sync Field Error Flag (SFEF) in the LIN Error Status register (LINESR).
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) [in other words, LINTCSR[MODE]= 0x0]:

1. Set the LIN state Interrupt enable bit (LSIE) in the LIN Interrupt Enable register (LINIER) [LINIER[LSIE] = 0x1]
2. When the Data Reception Completed Flag (DRF) get set in the LIN Status Register (LINSR), read the LIN State field (LINS) in LINSR
3. If LINSR[LINS]= 0b0101, read the Counter Value field (CNT) of the LINTCSR register, otherwise repeat step 2
4. If LINTCSR[CNT] 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 LSIE bit in the LINIER register
2. When the LINSR[DRF] bit get set in the LIN Status Register (LINSR), 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 get set 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 (by clearing LINCR1[LASE]=0) in LIN slave mode.

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

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

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

As consequence, the state machine may go to the Idle State when the LINFlexD module tries the transmission of the next eight bytes, after the first ones have been successfully transmitted and Data Buffer Empty Flag (DBEF) was set in the LIN Status Register (LINSR).
Workaround: Do not use the extended frame mode by configuring Data Field Length (DFL) bit-field with a value less than eight in the Buffer Identifier Register (BIDR) (BIDR[DFL] < 8)

e8228:  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

e8049:  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.

e8013:  MPC574xP: RMII_CLK can not be output to external pin

Description: The Ethernet (ENET) clock for Reduced Media Independent Interface (RMII) mode is not connected to an output pin.

Workaround: Customer has to use the external PHY clock in RMII mode.
e8747: PAD_RING: Vih values do not match datasheet

**Description:** Some of the Input High level voltages (Vih) do not match the Datasheet. The following specifications are affected:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Minimum value (Data Sheet)</th>
<th>Minimum value (Erratum)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vih</td>
<td>CMOS Input Buffer High Voltage (with hysteresis disabled)</td>
<td>0.55*VDD_HV_IO</td>
<td>0.56*VDD_HV_IO</td>
</tr>
<tr>
<td>Vih_hys</td>
<td>CMOS Input Buffer High Voltage (with hysteresis enabled)</td>
<td>0.65*VDD_HV_IO</td>
<td>0.66*VDD_HV_IO</td>
</tr>
</tbody>
</table>

Where VDD_HV_IO is the High Voltage Input / Output power supply voltage.

**Workaround:** Use the Input High level voltages specified in this erratum description.

e8614: PMC: In internal regulation mode device can get stuck in reset and drive 1.2V out of specification for certain 3.3V ramp rates or after a brownout on the 3.3V rail

**Description:** When the Power Management Controller (PMC) is configured for internal regulation and experiences a slow ramp rate of the 3.3V supply, usually but not limited to < 300V/s, it is possible that the internal regulator will loose regulation of the 1.2V and it will drive out of absolute maximum specification of 1.5V. This can also occur during a brown out of the 3.3V rail where the voltage supply momentarily dips far enough below 3.3V to cause a reset. This can only be remedied by completing removing the 3.3V from the part. Over time this will cause reliability issues with the device.

**Workaround:** Only use external regulation mode of the PMC.

e7591: RGM: Possible MC_RGM_FERD and MC_RGM_DERD register corruption

**Description:** It is possible that when writing either the Reset Generation Module’s (RGM) Functional Event Reset Disable Register (MC_RGM_FERD) or Destructive Event Reset Disable Register (MC_RGM_DERD) and a functional reset occurs during the same clock cycle as the write the contents of these registers may be corrupted.

**Workaround:** Every write to MC_RGM_FERD and MC_RGM_DERD should be followed by a read-back of the contents of these registers and checked against the expected value. In case of a mismatch a software destructive reset should be issued.
e8634: SAR_ADC: Conversions may fail if Pre-Sampling is enabled

Description: Successive Approximation Register (SAR) Analog to Digital (ADC) conversions may fail if Pre-Sampling is enabled. In this case, the ADC output may be unreliable. The failure occurs at minimum sampling time and when the internal voltage sample selection (PREVALn = 0, 1 or 2) is configured for VSSA, VDDA or VREFL as pre-sample voltage.

Workaround: Either do not use pre-sampling, or use PREVALn = 3 (VREFH as pre-sample voltage) or increase the sampling time to at least 375ns if pre-sampling is enabled.

e7204: 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.

e7202: SENT: Increased tolerance to noise for Nibble length measurement is not available

Description: The Single Edge Nibble Transmission (SENT) Receiver (SRX) supports the SAE J2716 (January 2010) standard that supports a maximum of 12.5% tolerance for the nibble length adjustment for noise and clock jitter. It does not support up to 50% that may be required for some applications.
**Workaround:** Include extra noise filtering on the target board or use the per channel noise filter to increase tolerance to noise by programming the appropriate value in Input Filter Sample Count field of the Channel Configuration Register (CHn_CONFIG[FIL_CNT]). In addition, writing a 1 to the Nibble Length Variation Limit bit (GBL_CTRL[NIB_LEN_VAR_LIMIT]) of the SRX Global Control Register will not enable the optional 50% variation support.

**e7886: SENT: Jitter tolerance is limited to 1/8 of the utick time**

**Description:** The Single Edge Nibble Transmitter (SENT) Receiver does not properly round off incoming data to nearest nibble. The SAE J2716 (SENT) specification dated January 2010 (revision 3) jitter specification is not met which leads to an incorrect rounding of nibble measurement. As a result, the Channel n Fast Message Data Read Register (CHn_FMSG_DATA) or Channel n Fast Message Cyclic Redundancy Check register (CHn_FMSG_CRC) values may be incorrect for the Status, Data, and Cyclic Redundancy Check values or the Message is not received at all because of the CRC mismatch.

**Workaround:** When the total accumulated jitter added by the SENT transmitter is less than or equal to 10% of the total utick duration, message reception at the SENT receiver would be correct. Use SENT transmitter devices that have total accumulated errors that results in 10% of the uTick period or less.

**e7404: SENT: Message overflow in SENT Receiver can lead to stall condition in the MCU**

**Description:** Under certain conditions, the Single Edge Nibble Transmission (SENT) Receiver (SRX) stalls and the Fast Message Data Ready bit for the SENT channel (FMSG_RDY[F_RDYn]) will no longer get set to indicate that a fast message is available. Reads of any of the fast message registers by the MCU core will stall and not complete. The registers affected are:

<table>
<thead>
<tr>
<th>Register</th>
<th>Register Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA_FMSG_DATA</td>
<td>Direct Memory Access (DMA) Fast Message Data Read Register</td>
</tr>
<tr>
<td>DMA_FMSG_CRC</td>
<td>DMA Fast Message Cyclic Redundancy Check Register</td>
</tr>
<tr>
<td>DMA_FMSG_TS</td>
<td>DMA Fast Message Time-stamp Register</td>
</tr>
<tr>
<td>CHn_FMSG_DATA</td>
<td>Channel Fast Message Data Read Register</td>
</tr>
<tr>
<td>CHn_FMSG_CRC</td>
<td>Channel Cyclic Redundancy Check Register</td>
</tr>
<tr>
<td>CHn_FMSG_TS</td>
<td>Channel Fast Message Time-stamp Register</td>
</tr>
</tbody>
</table>

A stall may occur if an overflow status condition is detected in the SENT Receiver Channel Status register (CHn_STATUS[FMSG_OFLW] = 1).

The overflow occurs when two messages are allowed to queue in the internal buffers of the SENT Receiver.

**Workaround:** 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 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 eDMA access 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 SENT 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).

If the stall occurs, a reset will be required to clear the stall condition. A Software Watchdog Timer (SWT) should be enabled to force a reset of the MCU if the device becomes stalled.

e7425: 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

<table>
<thead>
<tr>
<th>First Error Detected</th>
<th>Other error bits asserted</th>
<th>Cause for extra error bits getting asserted</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIB_VAL_ERR</td>
<td>NUM_EDGES_ERR asserted twice</td>
<td>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 Pulse and the Calibration pulse are seen as back to back calibration pulses and no edges in between.</td>
<td>Ignore both NUM_EDGES_ERR errors</td>
</tr>
<tr>
<td>FMSG_CRC_ERR</td>
<td>NUM_EDGES_ERR asserted twice</td>
<td>Same as NIB_VAL_ERR.</td>
<td>Ignore both NUM_EDGES_ERR errors</td>
</tr>
</tbody>
</table>
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.

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.

<table>
<thead>
<tr>
<th>First Error Detected</th>
<th>Other error bits asserted</th>
<th>Cause for extra error bits getting asserted</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM_EDGES_ERR (when edges are less than expected)</td>
<td>NIB_VAL_ERR is asserted</td>
<td>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.</td>
<td>Ignore the NIB_VAL_ERR</td>
</tr>
<tr>
<td>NUM_EDGES_ERR (when edges are more than expected)</td>
<td>NIB_VAL_ERR and PP_DIAG_ERR are asserted</td>
<td>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</td>
<td>Ignore NIB_VAL_ERR and PP_DIAG_ERR</td>
</tr>
</tbody>
</table>

Table 2. Expected behavior, clarification of NUM_EDGES_ERR cases

Mask Set Errata for Mask 1N65H, Rev. SEP 2015
e7139: SSCM: The SSCM can incorrectly detect a configuration error

**Description:** The System Status and Configuration Module’s System Status register’s Configuration Error bit (SSCM_STATUS.CER) indicates that the SSCM has detected an error during reset while loading Device Configuration Format (DCF) records from flash. This bit can get incorrectly set in a rare event when SSCM is scanning flash for DCF records and finds a stop record and a functional reset occurs at the same time. This will cause the SSCM to initiate a DCF transfer to a non-existent DCF client resulting in a DCF transfer error which sets the SSCM_STATUS.CER bit.

**Workaround:** When coming out of reset if the SSCM_STATUS.CER bit is set and the Functional Event Status Register in the Reset Generation Module (MC_RGM_FES) is non zero then it could be a false indication that a configuration error has occurred. Clear the SSCM_STATUS.CER bit and the MC_RGM_FES register and issue software destructive reset to reload the DCF records. To execute a software destructive reset initiate 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.TARGET_MODE). This is a two step process as this register is protected and requires a special sequence to modify the Target Mode bits.

```
MC_MC.MCTL.R = 0xF0005AF0
MC_MC.MCTL.R = 0xF000A50F
```

e8967: 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 output bits (TEMPx_y_OP) and status flags TEMPx_y. Refer to ERR008683. There are chances that the temp exceeds the overtemp limit (eg 150C) and TEMPx_y_OP flag is toggling at 3 MHz. Hence through software the TEMPx_y_OP status flag have to be sampled multiple times at > 6 MHz in order to prevent a wrong flag status detection. If the TEMPx_y_OP output bit is found to be zero after multiple sampling indicating that current temperature is okay then clear the corresponding TEMPx_y status flag if set.

e8683: 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.

---

**e7236: 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.
e8730: 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.
Information in this document is provided solely to enable system and software implementers to use Freescale 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.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 Freescale 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. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, AltiVec, C-5, CodeTest, CodeWarrior, ColdFire, ColdFire+, C-Ware, Energy Efficient Solutions logo, Kinetis, mobileGT, PowerQUICC, Processor Expert, QorIQ, Qorivva, StarCore, Symphony, and VortiQa are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Airfast, BeeKit, BeeStack, CoreNet, Flexis, Layerscape, MagniV, MXC, Platform in a Package, QorIQ Qonverge, QUICC Engine, Ready Play, SafeAssure, SafeAssure logo, SMARTMOS, Tower, TurboLink, Vybrid, and Xtrinsic are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© 2015 Freescale Semiconductor, Inc.