Customer Information Notification

Issue Date: 21-Apr-2018
Effective Date: 22-Apr-2018

Dear Valerie Kirkland,

Here's your personalized quality information concerning products Mouser Electronics purchased from NXP. For detailed information we invite you to view this notification online.

Management Summary
Update to the LS1012A and LS1020A family of devices errata document.

Release of New Erratum A-011218 Impacting LS1012A & LS1020/21/22A Family of Devices

Description
The following changes were made to the errata documents for LS1012A and LS1020A/21A/22A:
1- Updated (LS1012A) example code in the workaround section for eDMA A-011218.
2- Added (LS1020A/21A/22A) eDMA erratum A-011218.
3- Removed QSPI erratum A-009282 since it is not applicable. The QSPI DDR feature is not supported.
4- Moved SFP erratum A-006879 to Trust Architecture document.

Reason
Notify customers of erratum release.

Anticipated Impact on Form, Fit, Function, Reliability or Quality
No impact on form, fit, function, reliability or quality.

Additional information
Additional documents: view online

Contact and Support
For all inquiries regarding the ePCN tool application or access issues, please contact NXP "Global Quality Support Team".
For all Quality Notification content inquiries, please contact your local NXP Sales Support team.
For specific questions on this notice or the products affected please contact our specialist directly:
Name: Safwan Raad  
Position: Apps Mgr  
e-mail address: safwan.raad@nxp.com

At NXP Semiconductors we are constantly striving to improve our product and processes to ensure they reach the highest possible Quality Standards.  
Customer Focus, Passion to Win.

NXP Quality Management Team.

About NXP Semiconductors

NXP Semiconductors N.V. (NASDAQ: NXPI) provides High Performance Mixed Signal and Standard Product solutions that leverage its leading RF, Analog, Power Management, Interface, Security and Digital Processing expertise. These innovations are used in a wide range of automotive, identification, wireless infrastructure, lighting, industrial, mobile, consumer and computing applications.

NXP Semiconductors  
High Tech Campus, 5656 AG Eindhoven, The Netherlands  
© 2006-2010 NXP Semiconductors. All rights reserved.
LS1012A Chip Errata

This document details all known silicon errata for LS1012ACE. The following table provides a revision history for this document.

Table 1. Document revision history

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Significant Changes</th>
</tr>
</thead>
</table>
| 3        | 04/2018 | Updated the following erratum:  
• eDMA A-011218  
Removed the following errata:  
• QSPI A-009282  
• SFP A-006879  
The SFP erratum A-006879 has been moved to the Trust Architecture document, contact NXP sales representative for further details. |
| 2        | 02/2018 | Added the following erratum:  
• eDMA A-011218  
Updated the following errata:  
• Ethernet A-010336, A-010897, and A-010904 |
| 1        | 08/2017 | Added the following errata:  
• Ethernet erratum A-010897  
• Ethernet erratum A-010904  
• Arm erratum A-011000  
• Arm erratum A-011001  
• Arm erratum A-011002  
• Arm erratum A-011003  
• Arm erratum A-011004  
• Arm erratum A-011005  
• Arm erratum A-011006  
• Arm erratum A-011007  
• Arm erratum A-011008  
• Arm erratum A-011009  
• Arm erratum A-011010  
• Arm erratum A-011011  
Updated the following errata:  

Table continues on the next page...
Table 1. Document revision history (continued)

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Significant Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01/2017</td>
<td>Initial public release</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Ethernet erratum A-010336</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Workaround updated with a note about link partners that use recovered clock as reference clock and may fail to comply the common clock requirement as per this erratum</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Fix plan updated to &quot;Plan to fix on silicon Rev 2.0&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• eSDHC erratum A-008171 has an updated workaround. Few steps were missing in the last revision of the document</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• QSPI erratum A-009282 description updated to state DDR mode is the only affected mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Removed the following errata:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• I2C erratum A-007812</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• I2C erratum A-007810</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replaced Arm erratum A-008896 with Arm errata summarized in new table &quot;Summary of Arm Software Developers Errata Notice&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added SVR values for silicon revision 2.0 in Table 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added a column for silicon revision 2.0 in Table 4</td>
</tr>
</tbody>
</table>

The following table provides a cross-reference to match the revision code in the processor version register to the revision level marked on the device.

Table 2. Revision Level to Part Marking Cross-Reference

<table>
<thead>
<tr>
<th>Part</th>
<th>Device Revision</th>
<th>Arm® Cortex®-A53 MPCore processor Revision</th>
<th>Arm Core Main ID Register</th>
<th>System Version Register Value</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1012A</td>
<td>1.0</td>
<td>r0p4-50rel0</td>
<td>0x410F_D034h</td>
<td>0x8704_0110</td>
<td>Without Encryption</td>
</tr>
<tr>
<td>LS1012AE</td>
<td>1.0</td>
<td>r0p4-50rel0</td>
<td>0x410F_D034h</td>
<td>0x8704_0010</td>
<td>With Encryption</td>
</tr>
<tr>
<td>LS1012A</td>
<td>2.0</td>
<td>r0p4-50rel0</td>
<td>0x410F_D034h</td>
<td>0x8704_0120</td>
<td>Without Encryption</td>
</tr>
<tr>
<td>LS1012AE</td>
<td>2.0</td>
<td>r0p4-50rel0</td>
<td>0x410F_D034h</td>
<td>0x8704_0020</td>
<td>With Encryption</td>
</tr>
</tbody>
</table>
Table 3 and Table 4 summarizes all known errata and lists the corresponding silicon revision level to which they apply. A ‘Yes’ entry indicates the erratum applies to a particular revision level, and an ‘No’ entry means it does not apply.

Table 3. Summary of Arm Software Developers Errata Notice

<table>
<thead>
<tr>
<th>SDEN ID</th>
<th>Category</th>
<th>NXP Errata Number</th>
<th>Summary of Erratum</th>
<th>Silicon Rev 1.0</th>
<th>Silicon Rev 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>859081</td>
<td>CatB</td>
<td>A-011000</td>
<td>Some AT instructions executed from EL3 might incorrectly report a domain fault</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>855873</td>
<td>CatB</td>
<td>A-010966</td>
<td>An eviction might overtake a cache clean operation</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>855871</td>
<td>CatB</td>
<td>A-010967</td>
<td>ETM does not report IDLE state when disabled using OSLOCK</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>853175</td>
<td>CatB</td>
<td>A-010968</td>
<td>APB access to ETM space while core is in Retention will never complete</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>845719</td>
<td>CatB</td>
<td>A-010969</td>
<td>A load might read incorrect data</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>843819</td>
<td>CatB</td>
<td>A-010970</td>
<td>Memory locations might be accessed speculatively due to instruction fetches when HCR.VM is set</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859372</td>
<td>CatB Rare</td>
<td>A-011001</td>
<td>Advanced SIMD integer multiply instructions in IT block might cause data corruption</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>855872</td>
<td>CatB Rare</td>
<td>A-010971</td>
<td>A Store-Exclusive instruction might pass when it should fail</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859374</td>
<td>CatC</td>
<td>A-011002</td>
<td>AArch64 memory streaming performance might be lower than expected when using a 64KB translation granule</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859373</td>
<td>CatC</td>
<td>A-011003</td>
<td>Mismatch between EDPRSR.SR and EDPRSR.R</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859083</td>
<td>CatC</td>
<td>A-011004</td>
<td>Debug not entering Memory Access mode without setting EDSCR.ERR</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859082</td>
<td>CatC</td>
<td>A-011005</td>
<td>ATS12NSOPR instruction might incorrectly translate when the HCR.TGE bit is set</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859080</td>
<td>CatC</td>
<td>A-011006</td>
<td>ETM trace reports incorrect branch target</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859077</td>
<td>CatC</td>
<td>A-011007</td>
<td>Incorrect fault status codes used for reporting synchronous uncorrectable ECC aborts in the IFSR</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859076</td>
<td>CatC</td>
<td>A-011008</td>
<td>Instruction cache parity error might cause an incorrect abort to be taken</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859075</td>
<td>CatC</td>
<td>A-011009</td>
<td>Accessing EDPCSR has side-effects when OS Lock is locked</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859074</td>
<td>CatC</td>
<td>A-011010</td>
<td>WFx can cause the PC to jump from lower VA subrange to upper VA subrange</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>859072</td>
<td>CatC</td>
<td>A-011011</td>
<td>ETM might output an incorrect exception return address</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>855830</td>
<td>CatC</td>
<td>A-010972</td>
<td>Loads of mismatched size might not be single-copy atomic</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>855829</td>
<td>CatC</td>
<td>A-010973</td>
<td>Reads of PMEVCNTR&lt;n&gt; are not masked by HDCR.HPMN</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Table 3. Summary of Arm Software Developers Errata Notice (continued)

<table>
<thead>
<tr>
<th>SDEN ID</th>
<th>Category</th>
<th>NXP Errata Number</th>
<th>Summary of Erratum</th>
<th>Silicon Rev 1.0</th>
<th>Silicon Rev 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>855827</td>
<td>CatC</td>
<td>A-010974</td>
<td>PMU counter values might be inaccurate when monitoring certain events</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>853172</td>
<td>CatC</td>
<td>A-010975</td>
<td>ETM might assert AFREADY before all data has been output</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>852521</td>
<td>CatC</td>
<td>A-010976</td>
<td>A64 unconditional branch might jump to incorrect address</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>852071</td>
<td>CatC</td>
<td>A-010977</td>
<td>Direct branch instructions executed before a trace flush might be output in an atom packet after flush acknowledgement</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>851871</td>
<td>CatC</td>
<td>A-010978</td>
<td>ETM might lose counter events while entering wfx mode</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>851672</td>
<td>CatC</td>
<td>A-010979</td>
<td>ETM might trace an incorrect exception address</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>845819</td>
<td>CatC</td>
<td>A-010980</td>
<td>Instruction sequences containing AES instructions might produce incorrect results</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>836919</td>
<td>CatC</td>
<td>A-010981</td>
<td>Write of JMCR in EL0 does not generate an UNDEFINED exception</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>836870</td>
<td>CatC</td>
<td>A-010982</td>
<td>Non-allocating reads might prevent a store exclusive from passing</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Notes:
1. For details, see the latest version of the Arm document, Cortex-A53 MPCore Software Developers Errata Notice. Applicable r0p4 errata are shown in the table above. This document is available on the Arm website.

Table 4. Summary of Silicon Errata and Applicable Revision

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>DCFG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-006748</td>
<td>DCFG_CCSR_CRSTSRn momentarily reports core ready status while the threads are still coming out of boot hold-off state</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>DDR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010441</td>
<td>Write-Leveling Hardware Calibration Error bits are unreliable</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>DUART</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-004737</td>
<td>BREAK detection triggered multiple times for a single break assertion</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>eDMA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-011218</td>
<td>eDMA may not work with SPI</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>eSDHC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008171</td>
<td>eSDHC transactions may fail in tuning modes of operation when the input data window is close to 1 UI during tuning</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table continues on the next page...
### Table 4. Summary of Silicon Errata and Applicable Revision (continued)

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>A-009620</td>
<td>Data timeout error not getting set in case of command with busy response (R1b) as well as for busy period after last write block transfer</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Ethernet</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010336</td>
<td>SGMII operation requires common clock between PHY/link partner and LS1012A</td>
<td>Fixed on silicon Rev 2.0</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010897</td>
<td>Jumbo frame feature is not supported</td>
<td>Fixed on silicon Rev 2.0</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010904</td>
<td>2.5G SGMII is not supported</td>
<td>Fixed on silicon Rev 2.0</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>FlexTimer</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007728</td>
<td>The process of clearing the FTMx_SC[TOF] bit does not work as expected under a certain condition when the FTM counter reaches FTM_MOD value</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009659</td>
<td>Incorrect match may be generated if intermediate load feature is used in toggle mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>I2C</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010124</td>
<td>Attempting a start cycle while the bus is busy may generate a short clock pulse</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010650</td>
<td>I2C controller is unable to generate clocks when SDA is low</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>PCle</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007189</td>
<td>PCI Express inbound error message handled incorrectly in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008432</td>
<td>Optional programmable PCI Express iATU CFG shift feature is not supported for outbound configuration transactions in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008822</td>
<td>Change the default AXI system error response behavior for PCI Express outbound non-posted requests</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009151</td>
<td>PCI Express controller does not correctly set the Link Autonomous Bandwidth Status and Link Bandwidth Management Status bits of the Link Status Register in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009410</td>
<td>PCI Express iATU logic does not retain configuration after a hot reset or link-down event in EP mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009518</td>
<td>PCI Express EP controller's non-D0 PowerState is overridden to D0 when any single or combination of the Bus Master Enable (BME) or Memory Space Enable (MSE) bits are enabled from disabled setting by host software</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009520</td>
<td>PCI Express completion transaction layer packets (TLP) of a decomposed outbound memory read request must be returned in order</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009531</td>
<td>IDO bit is set in completion packet header even when the IDO Completion Enable bit is cleared</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009700</td>
<td>Secondary PCI Express extended capability is included in the capability structure linked list in Gen1 and Gen2 modes</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
### Table 4. Summary of Silicon Errata and Applicable Revision (continued)

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>A-010053</td>
<td>PCI Express controller's transmitter does not enter ASPM L0s state when enabled</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010305</td>
<td>The Maximum Link Width and Supported Link Speed bit fields of the Link Capabilities register may reset to values higher than that configured by RCW in the EP mode after link-down or hot-reset event</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>QSPI</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008886</td>
<td>Sometimes unexpected data is written to the external flash memory even though the underrun bit (QuadSPI_FR[TBUF]) is not set</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009283</td>
<td>Illegal accesses to SPI flash memory can result in a system hang</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010284</td>
<td>Insufficient read data may be received in the Rx Data Buffer register</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>SATA</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008588</td>
<td>SATA controller may hang when processing multiple 16-byte non-aligned PRD entries for read operations</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009185</td>
<td>The default Rx watermark value may be insufficient for some hard drives</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010554</td>
<td>SATA controller might fail to detect some hard drives</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010635</td>
<td>Early termination of Read DMA Operation due to erroneous CRC Error</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>SEC</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-006385</td>
<td>SEC watchdog timer does not prevent all cases of illogical descriptors from hanging DECOs</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010541</td>
<td>Unexpected SEC behavior at soft reset when DECO is busy waiting on PKHA</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>SPI</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007731</td>
<td>Mixing 16-bit and 32-bit frame sizes in XSPI mode can cause incorrect data to be transmitted</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009776</td>
<td>Loading of shift register data into the receive FIFO following an overflow event</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009895</td>
<td>Inconsistent addition to Transmit or Command FIFO if an entry is added while FIFOs are full</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>USB</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-005697</td>
<td>Suspend bit asserted before the port is in Suspend state</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008118</td>
<td>USB initial sequence impact for external ULPI PHY</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008260</td>
<td>Device controller accepts new OUT packets even when TRBs are not available</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008428</td>
<td>Device initiates low power when ACK pending for data packet</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008459</td>
<td>Device in 2.0 mode handles DATA PID error on SETUP data incorrectly</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008997</td>
<td>USB3 LFPS peak-to-peak differential output voltage adjustment settings</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table continues on the next page...
<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>A-009007</td>
<td>USB3 PHY observes an intermittent failure in the receive compliance tests at higher jitter frequencies using the default register values</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009008</td>
<td>USB High Speed (HS) eye height adjustment</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009149</td>
<td>USB 2.0 protocol data pins USB_D_M and USB_D_P do not support IEEE 1149.1</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009163</td>
<td>Device prohibits low power after sending LPF = 1 for isochronous IN</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009377</td>
<td>Incorrect value for S and E fields for ISOC OUT in Start-Split token</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009668</td>
<td>Stop Endpoint command does not complete</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009798</td>
<td>USB high speed squelch threshold adjustment</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010127</td>
<td>Hot reset failure during U1/U2 entry for USB 3.0</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010129</td>
<td>USB 2.0 reset not driven while port is in the Resume state</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010131</td>
<td>U3 request gets dropped when controller tries U1-to-U2 entry</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010151</td>
<td>Unreliable receiver detection in low power P3 mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>
A-006748: DCFG_CCSR_CRSTSRn momentarily reports core ready status while the threads are still coming out of boot hold-off state

Affects: DCFG

Description: When the cores exit the boot hold-off state at the end of the reset sequence, the Core Reset Status Register n (DCFG_CCSR_CRSTSRn) momentarily reports an incorrect core ready status while the threads are still halted and are about to exit the halted state.

Impact: The DCFG_CCSR_CRSTSRn reports an incorrect core ready status from the time the core HRESET is negated to the time the core negates the halted signal.

Workaround: Software should only read DCFG_CCSR_CRSTSRn[READY] field 256 core clock cycles after the bit for the corresponding core is set in DCFG_BRRL to release the core from boot hold off.

Fix plan: No plans to fix
A-010441: Write-Leveling Hardware Calibration Error bits are unreliable

**Affects:** DDR

**Description:** When any error or issue occurs during auto write-leveling hardware calibration, the Write-Leveling Hardware Calibration Error bits (MPWLGC[WL_HW_ERR#] at address 0x01080808) are set and cleared automatically before they can be captured for use. Therefore, at the end of the auto write-leveling hardware calibration sequence, MPWLGC shows no error, even when an error has occurred.

**Impact:** Invalid error status could lead to invalid write-leveling calibration that could result in DDR failures.

**Workaround:** Auto write-leveling hardware calibration can be performed only when the following two conditions are true:
1. The DRAM prime DQ pin is connected to the DQ0 and DQ8 of LS1012A.
2. 0.25 inch < (clock trace length – strobe trace length) < 4 inches

If these two requirements are met on the board, the auto write-leveling hardware calibration will always complete successfully, and the Write-Leveling Hardware Calibration Error bits (MPWLGC[WL_HW_ERR#] at address 0x01080808) should be ignored.

**Fix plan:** No plans to fix
A-004737: BREAK detection triggered multiple times for a single break assertion

Description: A UART break signal is defined as a logic zero being present on the UART data pin for a time longer than (START bit + Data bits + Parity bit + Stop bits). The break signal persists until the data signal rises to a logic one.

A received break is detected by reading the ULSR and checking for BI = 1. This read to ULSR clears the BI bit. After the break is detected, the normal handling of the break condition is to read the URBR to clear the ULSR[DR] bit. The expected behavior is that the ULSR[BI] and ULSR[DR] bits do not get set again for the duration of the break signal assertion. However, the ULSR[BI] and ULSR[DR] bits continue to get set each character period after they are cleared. This continues for the entire duration of the break signal.

At the end of the break signal, a random character may be falsely detected and received in the URBR, with the ULSR[DR] being set.

Impact: The ULSR[BI] and ULSR[DR] bits get set multiple times, approximately once every character period, for a single break signal. A random character may be mistakenly received at the end of the break.

Workaround: The break is first detected when ULSR is read and ULSR[BI]=1. To prevent the problem from occurring, perform the following sequence when a break is detected:

1. Read URBR, which returns a value of zero, and clears the ULSR[DR] bit
2. Delay at least 1 character period
3. Read URBR again, which return a value of zero, and clears the ULSR[DR] bit

ULSR[BI] remains asserted for the duration of the break. The UART block does not trigger any additional interrupts for the duration of the break.

This workaround requires that the break signal be at least 2 character-lengths in duration.

This workaround applies to both polling and interrupt-driven implementations.

Fix plan: No plans to fix
A-011218: eDMA may not work with SPI

**Affects:** eDMA

**Description:** eDMA can be used to push/populate data to SPI Tx/Rx FIFO. Due to this erratum, the eDMA's interaction with SPI may fail with the following signatures:

- While transmitting, the SPI may fail to indicate that the Tx FIFO is full. Hence, the eDMA may keep pushing data to Tx FIFO resulting in a corruption in the Tx FIFO.
- While transmitting, the SPI may wrongly indicate that the Tx FIFO is full. The eDMA will keep waiting for Tx FIFO to have an empty space.
- While receiving, the SPI may not indicate that the Rx FIFO is empty. The eDMA will keep reading the data from an empty FIFO.
- While receiving, the SPI may wrongly indicate that the Rx FIFO is empty. The eDMA will keep waiting for Rx FIFO to become non-empty.

**Impact:** The eDMA may fail to properly work with SPI.

**Workaround:** An extra dummy channel must be created which can be triggered by the SPI. The dummy channel should be configured to transfer four bytes from a dummy source address to a dummy destination address in the DDR memory. The dummy channel should be linked to an actual data serving channel on a minor and major loop basis.

1. The dummy channel citer and biter should be equal to the citer and biter of the data servicing channel.
2. When using the fixed priority scheme, the linked eDMA channels that push/populate the SPI FIFOs must have a higher priority than the intermediate dummy eDMA channels to which the SPI requests are registered.
3. Rx side dummy-channel and linked Rx-FIFO servicing channels must have a higher priority than the Tx side counterparts. This is to avoid the Rx FIFO overflow.
4. However, restrictions in #2 and #3 are not applicable when implemented with the default round-robin priorities within groups and channels in groups (EDMA\_CR[ERCA] and EDMA\_CR[ERGA] both are set). Default round-robin priority scheme is recommended for this workaround implementation when there are channels from other peripherals are also present.

The below example shows an eDMA configuration in which eDMA is serving the SPI Tx FIFO and Rx FIFO simultaneously.

Where, rx\_ch=0, rx\_dummy\_ch=1, tx\_ch=2, and tx\_dummy\_ch=3 are Rx data servicing channel, Rx dummy channel, Tx data servicing, and Tx dummy channels, respectively.

Setting value requires byte-swap to write to eDMA register.

```c
/* **** eDMA Channel TCD for TX Data Servicing channel ****/
Address -> Value
(edMA\_BASE + 0x1000 + 0x20*tx\_ch) -> 0xA0000000 /*tx data buffer address
in DDR*/
(edMA\_BASE + 0x1004 + 0x20*tx\_ch) -> 0x02020004 /*SSIZE and DSIZE are for
32 bits, SOFF = 4*/
(edMA\_BASE + 0x1008 + 0x20*tx\_ch) -> 0x4 /*NBYTES = 4*/
(edMA\_BASE + 0x100C + 0x20*tx\_ch) -> 0x0 /*SLAST = 0*/
(edMA\_BASE + 0x1010 + 0x20*tx\_ch) -> 0x2100034 /*SPI PUSH register
address*/

(edMA\_BASE + 0x1014 + 0x20*tx\_ch) -> (TX\_DATA\_DDR\_QUEUE\_SIZE\_IN\_BYTES/4) <<
16 /*CITER = (ddr buffer data size/nbytes), DOFF = 0*/
(edMA\_BASE + 0x1018 + 0x20*tx\_ch) -> 0 /*DLAST\_SGA = 0*/
```
(edMA_BASE + 0x101C + 0x20*tx_ch) -> 0x0000000A /*D_REQ = 1, INT_MAJ = 1*/

/**** eDMA Channel TCD for RX Data Servicing channel ****/
Address -> Value
(edMA_BASE + 0x1000 + 0x20*rx_ch) -> 0x2100038 /*SPI POP register address*/
(edMA_BASE + 0x1004 + 0x20*rx_ch) -> 0x02020000 /*SSIZE and DSIZE are for 32 bits, SOFF = 0*/
(edMA_BASE + 0x1008 + 0x20*rx_ch) -> 0x4 /*NBYTES = 4*/
(edMA_BASE + 0x100C + 0x20*rx_ch) -> 0x0 /*SLAST = 0*/
(edMA_BASE + 0x1010 + 0x20*rx_ch) -> 0xB0000000 /*rx data buffer address in DDR*/
(edMA_BASE + 0x1014 + 0x20*rx_ch) -> ((RX_DATA_DDR_QUEUE_SIZE_IN_BYTES_IN_BYTES/4) << 16 | 0x4) /*CITER = ddr_buffer_data_size/nbytes, DOFF = 0*/
(edMA_BASE + 0x1018 + 0x20*rx_ch) -> 0 /*DLAST_SGA = 0*/
(edMA_BASE + 0x101C + 0x20*rx_ch) -> ((1<<31)|(tx_ch<<25) | (RX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16) /*D_REQ = 1, INT_MAJ = 1*/

/**** eDMA Channel TCD for TX Dummy channel ****/
Address -> Value
(edMA_BASE + 0x1000 + 0x20*tx_dummy_ch) -> 0xF0000000 /*Dummy source address in DDR for dummy transfer*/
(edMA_BASE + 0x1004 + 0x20*tx_dummy_ch) -> 0x02020000 /*SSIZE, DSIZE are for 32 bits, SOFF = 0*/
(edMA_BASE + 0x1008 + 0x20*tx_dummy_ch) -> 0x4 /*NBYTES = 4*/
(edMA_BASE + 0x100C + 0x20*tx_dummy_ch) -> 0x0 /*SLAST = 0*/
(edMA_BASE + 0x1010 + 0x20*tx_dummy_ch) -> 0xF0000004 /*Dummy destination address in DDR for dummy transfer*/
(edMA_BASE + 0x1014 + 0x20*tx_dummy_ch) -> ((1<<31)|(tx_ch<<25) | (TX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16) /*CITER = ddr_buffer_data_size/nbytes, link channel number tx_ch=2 to this channel, DOFF = 0*/
(edMA_BASE + 0x1018 + 0x20*tx_dummy_ch) -> 0 /*DLAST_SGA = 0*/
(edMA_BASE + 0x101C + 0x20*tx_dummy_ch) -> ((1<<31)|(tx_ch<<25) | (tx_ch << 8) | (1<<5) | 0x0000000A) /*D_REQ = 1, INT_MAJ = 1 link channel number tx_ch=2 to this channel on major as well as on minor loop basis*/

/**** eDMA Channel TCD for RX Dummy channel ****/
Address -> Value
(edMA_BASE + 0x1000 + 0x20*rx_dummy_ch) -> 0xF0000008 /*Dummy source address in DDR for dummy transfer*/
(edMA_BASE + 0x1004 + 0x20*rx_dummy_ch) -> 0x02020000 /*SSIZE and DSIZE are for 32 bits, SOFF = 0*/
(edMA_BASE + 0x1008 + 0x20*rx_dummy_ch) -> 0x4 /*NBYTES = 4*/
(edMA_BASE + 0x100C + 0x20*rx_dummy_ch) -> 0x0 /*SLAST = 0*/
(edMA_BASE + 0x1010 + 0x20*rx_dummy_ch) -> 0xF000000C /*Dummy destination address in DDR for dummy transfer*/
(edMA_BASE + 0x1014 + 0x20*rx_dummy_ch) -> ((1<<31)|(tx_ch<<25) | (RX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16) /*CITER = ddr_buffer_data_size/nbytes, link channel number rx_ch=0 to this channel DOFF = 0*/
(edMA_BASE + 0x1018 + 0x20*rx_dummy_ch) -> 0 /*DLAST_SGA = 0*/
(edMA_BASE + 0x101C + 0x20*rx_dummy_ch) -> ((1<<31)|(tx_ch<<25) | (tx_ch << 8) | (1<<5) | 0x0000000A) /*D_REQ = 1, INT_MAJ = 1, link channel number rx_ch=0 to this channel*/

/**** eDMA channel priority configuration ****/
if (priority_scheme == ROUND_ROBIN ) { /*Default round robin channels and groups priority scheme*/
EDMA_CR[ERCA] = 1 /*round robin arbitration for channels in each group*/
EDMA_CR[ERGA] = 1 /*round robin arbitration is used among the channel groups*/
} else { /*Fixed priority scheme*/
  /*Make sure data servicing channels has higher priority than their dummy channel counterpart*/
  EDMA_CR[ERCA] = 0 /* fixed priority arbitration for channels in each group*/
  EDMA_CR[ERGA] = 0 /* fixed priority arbitration is used among the channel groups*/

  **** Configuration for group priorities ****/
  if (data_servicing_channels_grp == GRP1) { /*Data servicing channels are in channel group-1 i.e. in the same group where dummy channels are present*/
    /*either group-1 has higher priority */
    EDMA_CR[GRP1PRI] = 0 /*Channel group-1 has higher priority*/
    EDMA_CR[GRP2PRI] = 1 /*Channel group-2 has lower priority*/
    /*Or group-2 has higher priority*/
    EDMA_CR[GRP1PRI] = 1 /*Channel group-1 has higher priority */
    EDMA_CR[GRP2PRI] = 0 /*Channel group-2 has lower priority*/
  } else { /*data servicing channels are in channel group group-2 i.e. in the different group where the dummy channels are present*/
    EDMA_CR[GRP1PRI] = 1 /*Channel group-1 has lower priority */
    EDMA_CR[GRP2PRI] = 0 /*Channel group-2 has higher priority */
  }

  **** Configuration for channel priorities in the group ****/
  EDMA_DCHPRI0 = 0x00; /*priority for channel number rx_ch is 0*/
  EDMA_DCHPRI1 = 0x01; /*priority for channel number tx_dummy_ch is 1*/
  EDMA_DCHPRI2 = 0x02; /*priority for channel number tx_ch is 2*/
  EDMA_DCHPRI3 = 0x03; /*priority for channel number tx_dummy_ch is 3*/
}

**** Enable eDMA channels ****/
DMAMUX1_CHCFG1 = (0x80 | 0x3C) ; /* Enable channel number, rx_dummy_ch=1 to triggered by SPI1 RFDF i.e by source number 60*/
DMAMUX1_CHCFG3 = (0x80 | 0x3E) ; /* Enable channel number, tx_dummy_ch=3 to triggered by SPI1 TF i.e by source number 62*/
Figure 1. Dummy channel based workaround

Fix plan: No plans to fix
A-008171: eSDHC transactions may fail in tuning modes of operation when the input data window is close to 1 UI during tuning

**Affects:** eSDHC

**Description:** In tuning mode of operation, when TBCTL[TB_EN] is set, eSDHC may report one of the following errors:

- Tuning error while running tuning operation where SYSCTL2[SAMPCLKSEL] will not get set even when SYSCTL2[EXTN] is reset.
- Data transaction error (for example, IRQSTAT[DCE], IRQSTAT[DEBE]) during data transaction errors.

This issue occurs when the data window sampled within eSDHC is full cycle. So, in that case eSDHC is not able to find out the start and end points of the data window and sets the sampling pointer at default location (which is middle of the internal SD clock). If this sampling point coincide with the data eye boundary then it can results into above mentioned errors.

**Impact:** Tuning mode of operation for SDR50, SDR104 or HS200 speed modes may not work properly.

**Workaround:** In case the eSDHC reports tuning error or data errors in the tuning mode of operation, then the software can use one of the below mentioned workarounds:

- Shift the sampling pointer by half SD_CLK cycle from its default location.
- Increase the sampling clock (per_clk) frequency so that data window is sampled less than full cycle.
- Change the SD_CLK frequency so that the sampling point does not coincide with the start or end of data eye.
- Use the Fixed sampling technique for the SDR50 mode.

**NOTE**

\[ \text{DIV\_RATIO} = \text{ESDHCCCTL}[SDCLKFS]*\text{ESDHCCCTL}[DVS] \text{ when} \]
\[ \text{CRS=0;}\ \text{DIV\_RATIO} = \text{ESDHCCCTL}[USDCLKFS[0:1],SDCLKFS[0:7]] \text{ when} \text{CRS=1}. \]

Procedure to shift the sampling pointer by half SD_CLK period from its default location:

   TBCTL[0:31] register address = ESDHC_BASE_ADDRESS+12'h120.
2. Read the TBCTL[0:31] register and rewrite again. Wait for 1 ms second.
3. Read the TBSTAT[0:31] register twice. TBSTAT[0:31] register address = ESDHC_BASE_ADDRESS+12'h124.
4. If TBSTAT[8:15]-TBSTAT[0:7] > 4 *DIV\_RATIO or TBSTAT[0:7] –TBSTAT[8:15] > 4 *DIV\_RATIO, then program TB PTR[TB\_WNDW\_END\_PTR] = 4*DIV\_RATIO (that is, four times the division ration of SD_CLK) and program TB PTR[TB\_WNDW\_START\_PTR] = 8*DIV\_RATIO.
5. Else program TB PTR[TB\_WNDW\_END\_PTR] = 3*DIV\_RATIO and program TB PTR[TB\_WNDW\_START\_PTR] = 5*DIV\_RATIO.
6. Program the software tuning mode by setting TBCTL[TB\_MODE] = 2'h3.
7. Set SYSCTL2[EXTN] and SYSCTL2[SAMPCLKSEL].
8. Issue SEND\_TUNING\_BLK command (CMD19 for SD, CMD21 for MMC).
9. Wait for IRQSTAT[BRR], the buffer read ready to be set.
10. Clear IRQSTAT[BRR].
11. Check SYSCTL2[EXTN] to be cleared.
12. Check SYSCTL2[SAMPCLKSEL], the sampling clock select. Its set value indicate tuning procedure success, and clear indicate failure. In case of tuning failure, fixed sampling scheme could be used by clearing TBCTL[TB_EN].

**Fix plan:** No plans to fix
A-009620: Data timeout error not getting set in case of command with busy response (R1b) as well as for busy period after last write block transfer

Affects: eSDHC

Description: In the event that a busy timeout occurs for a command with a busy response (for example, R1b response) as well as busy period after the last write block, the eSDHC does not set the IRQSTAT[DTOE] bit or the IRQSTAT[TC]. Therefore, the current command transfer is never completed.

Impact: Software has to track the busy timeout error in this case.

Workaround: Choose one of the following workarounds for commands with a busy response:

• Implement a software timer to track the busy timeout error. If this software timer expires before the IRQSTAT[TC] event then consider it as data timeout error event (same as if IRQSTAT[DTOE] gets set) and execute the error recovery sequence for data timeout error specified in section 3.10.1 "Error Interrupt Recovery" in SD Host specification 3.0.

OR

• Don't set the XFRTPY[RSP]=2'b11 for commands with a busy response. Rather, poll the busy status of the card from the PRSSTAT[DLSL].

The workaround sequence for a busy period after last write block is as follows:

1. After the command completion interrupt (IRQSTAT[CC]), wait for de-assertion of PRSTAT[WTA].
2. As soon as PRSTAT[WTA] is de-asserted, start the software timer and poll the busy signal (DAT0) using PRSTAT[DLSL[0]].
3. Wait for DAT0 signal to go high (which indicates that the transfer is complete) or for the software timer expiry (which indicates a data timeout error).
4. Issue a soft reset for data (SYSCTL[RSTD]), if PRSTAT[DLA] bit is set.
5. In the case of a data timeout error (detected in step 3), perform error recovery.

Fix plan: No plans to fix
A-010336: SGMII operation requires common clock between PHY/link partner and LS1012A

**Affects:** Ethernet

**Description:** Proper operation of SGMII (10/100/1000/2500 Mbps) requires that the receive and transmit clocks have the same clock source. Any ppm deviation between the receive clock (included in Rx data) and the transmit clock results in link down (MDIO SGMII Status Register[LINK_STAT] = 0) and packet CRC errors resulting in dropped packets.

**Impact:** Any difference between the receive clock and the transmit clock results in link down and packet CRC errors resulting in dropped packets.

**Workaround:** Ensure that the LS1012A SerDes and the Ethernet PHY (or link partner) it connects to use the same clock source.

**Option 1:** For RCW[SerDes_INT_REFCLK] = 1, crystal with internal oscillator cannot be used. Use multiple clock generator to feed clock to SGMII PHYs (or link partner) and feed 25 MHz single ended clock to LS1012A through EXTAL pin.

**Option 2:** For RCW[SerDes_INT_REFCLK] = 0, use multiple clock generator to generate 100/125 MHz differential clock for SD_REF_CLK and clocks for SGMII PHYs (or link partner).

**NOTE**
Some link partners use a recovered clock for its reference clock. This mode is NOT recommended. Please ensure the mode of operation allows the usage of a common clock source with the LS1012A.

**Fix plan:** Fixed on silicon Rev 2.0
A-010897: Jumbo frame feature is not supported

**Affects:** Ethernet

**Description:** For LSI012A rev 1.0, Jumbo frames are not supported. The following failure modes have been observed:
- Egress: When the first jumbo frame is being transmitted, internal transmit logic gets stuck once payload bytes are being sent and does not complete.
- Ingress: When the first jumbo frame is being received, internal receive logic reports an overflow error and stops processing received packets.

**Impact:** Jumbo frame causes the PFE controller to hang, a reset of the entire chip is required to resume normal operation.

**Workaround:** No workaround exits to allow the usage of Jumbo frames. Ensure only normal sized Ethernet packets (1518 bytes or 1522 bytes w/VLAN tagging) are transmitted or received.

**NOTE**
If a jumbo frame causes the controller to hang, a reset of the entire chip is required to resume normal operation.

**Fix plan:** Fixed on silicon Rev 2.0
A-010904: 2.5G SGMII is not supported

Affects: Ethernet
Description: The use of 2.5G SGMII may lead to a corrupted data packet on receiving or transmitting the data.
Impact: 2.5G SGMII is not supported on silicon Rev 1.0.
Workaround: Use only 1G/100M/10M SGMII.
Fix plan: Fixed on silicon Rev 2.0
A-007728: The process of clearing the FTMx_SC[TOF] bit does not work as expected under a certain condition when the FTM counter reaches FTM_MOD value

Affects: FlexTimer
Description: If the FTM counter reaches the FTM_MOD value between the reading of the TOF bit and the writing of 0 to the TOF bit, the process of clearing the TOF bit does not work as expected when FTMx_CONF[NUMTOF] ≠ 0 and the current TOF count is less than FTMx_CONF[NUMTOF]. If the above condition is met, the TOF bit remains set. If the TOF interrupt is enabled (FTMx_SC[TOIE] = 1), the TOF interrupt also remains asserted.

Impact: None

Workaround: Two possible workarounds exist for this erratum and the decision on which one to use is based on the requirements of the application that is used.

OPTION 1:
Repeat the clearing sequence mechanism until the TOF bit is cleared.
Below is a pseudo-code snippet that must be included in the TOF interrupt routine:

```c
while (FTM_SC[TOF] != 0)
{
    void FTM_SC() ;       // Read SC register
    FTM_SC[TOF]=0 ;      // Write 0 to TOF bit
}
```

OPTION 2:
With FTMx_CONF[TOFNUM] = 0 and a variable in the software, count the number of times that the TOF bit is set. In the TOF interrupt routine, clear the TOF bit and increment the variable that counts the number of times that the TOF bit is set.

Fix plan: No plans to fix
A-009659: Incorrect match may be generated if intermediate load feature is used in toggle mode

Affects: FlexTimer
Description: When a channel (n) match is used as an intermediate reload, an incorrect second match may occur immediately following the correct match. The issue is problematic only if channel (n) is configured for output compare with the output configured to toggle mode. In this scenario, channel (n) toggles on the correct match and again on the incorrect match. The issue may also occur if a certain channel has a match which is coincident with an intermediate reload point of any other channel.

Impact: Incorrect match may lead to undesired functionality.

Workaround: If any channel is configured for output compare mode with the output set for toggle mode, the intermediate reload feature must not be used.

Fix plan: No plans to fix
A-010124: Attempting a start cycle while the bus is busy may generate a short clock pulse

Affects: I2C

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

Impact: I2C controller may generate a glitch to the SCL in a multi-master network. It may cause the bus to hang and other protocol violations.

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

Fix plan: No plans to fix
A-010650: \(^2\text{C controller is unable to generate clocks when SDA is low}\)

**Affects:** \(^2\text{C}

**Description:** Based on the \(^2\text{C}\) specification, if the data line (SDA) is stuck low, the master should send nine clock pulses and the \(^2\text{C}\) slave device that holds the bus low should release it sometime within those nine clocks. But the \(^2\text{C}\) controller cannot generate nine clock pulses, as a result the slave device is unable to release the bus.

**Impact:** The \(^2\text{C}\) bus cannot be set to idle when a slave device holds the bus low.

**Workaround:** Use an open drain GPIO pin to connect to the IICx_SCL to drive nine clock pulses to unlock the \(^2\text{C}\) bus.

**For \(^2\text{C1 controller}**: 

The IIC2_SCL pin can be used as GPIO, if IIC2_SCL pin is configured for \(^2\text{C}\) or if no other open drain GPIO pin is available. No pull-up resistor is needed in this case and the system becomes a multi-master system. The following solution is based on using the IIC2_SCL pin as open drain GPIO. Connect the IIC2_SCL pin to IIC1_SCL pin. The software has to perform the following steps:

1. Check the I2C1_IBSR[IBB] register value before switching to master mode and attempting a START cycle.
   a. Set a software timer of 50 ms and check if I2C1_IBSR[IBB] == 1.
   b. Clear the timer if I2C1_IBSR[IBB] changes to 0 before the timer expires and skip to Step 7. Otherwise, when the timer times out continue to Step 2.
2. Save the SCFG register PMUXCR0 value and configure IIC2_SCL pin as GPIO (GPIO_1[13]) through PMUXCR0[QSPI_IIC2_OVRD] = 0b00 and PMUXCR0[QSPI_MUX_OV RD] = 0b1.
3. Set GPDIR_1 = 0h00001000 and GPODR_1 = 0h00001000 to configure the GPIO_1[13] pin as an output and open drain.
4. Write data to generate the pulse through GPIO_1[13].
   a. Set GPDAT_1 = 0h00001000; (HIGH)
   b. Set GPDAT_1 = 0h00000000; (LOW)
   c. Repeat the above Steps a and b nine times.
   d. Set GPDAT_1 = 0h00001000 to ensure that the last level sent is always high.
5. Set I2C1_IBCR = 0h00 to generate a STOP then set I2C1_IBCR = 0h80 to reset the \(^2\text{C1}\) controller.
6. Restore the saved value of the SCFG register PMUXCR0 as listed in Step 2.
7. Set I2C1_IBSR[IBAL] to clear the IBAL bit if I2C1_IBSR[IBAL] = 1.
8. Restore to normal operation.

**For \(^2\text{C2 controller}**: 

No hardware modification is needed. The software has to perform the following steps:

1. Check I2C2_IBSR[IBB] before switching to the master mode and attempting a START cycle.
   b. Clear the timer if I2C2_IBSR[IBB] changes to 0 before the timer expires and skip to Step 7. Otherwise, when the timer times out continue to Step 2.
2. Save the SCFG register PMUXCR0 value and configure IIC2_SCL pins as GPIO (GPIO_1[13]) through PMUXCR0[QSPI_IIC2_OVRD] = 0b00 and PMUXCR0[QSPI_MUX_OV RD] = 0b1.
3. Set GPDIR_1 = 0h00001000, GPODR_1 = 0h00001000 to configure the GPIO_1[13] pin as output and open drain pin.
4. Write data to generate the pulses through GPIO_1[13].
   a. Set GPDAT_1 = 0h00001000; (HIGH)
   b. Set GPDAT_1 = 0h00000000; (LOW)
   c. Repeat the above Steps a and b nine times.
   d. Set GPDAT_1 = 0h00001000 to ensure that the last level sent is always high.
5. Set I2C2_IBCR = 0h00 to generate a STOP then set I2C2_IBCR = 0h80 to reset the I²C2 controller.
6. Restore the saved value of the SCFG register PMUXCR0 as listed in Step 2.
7. Set I2C2_IBSR[IBAL] to clear the IBAL bit if I2C2_IBSR[IBAL] = 1
8. Restore to normal operation.

**NOTE**
When ITS=1, GPIO_1[13] is always available and the above mentioned override procedure is not required. Hence, skip Steps 2 and 6. In this case, either GPIO_1[13] or any other dedicated GPIO has to be connected to IIC1_SCL. No additional pull-up resistor besides that designed in pull-up for the I²C bus should be added. The I²C2 interface is not available when ITS=1.

**NOTE**
When GPIO is connected to the existing I²C net, it already has a pullup on the net as required by the I²C specification. The value is based on the speed desired versus capacitance on that net. No additional pull-up resistor besides that designed in pullup for the I²C bus should be added.

**Fix plan:** No plans to fix
A-007189: PCI Express inbound error message handled incorrectly in RC mode

Affects: PCIe
Description: According to Figure 6-3 of the PCI Express base specification REV 3.0, Inbound errors reported to PCI Express RC:
- In its Root Error Status Register can be gated by the “SERR# Enable” bit in the PCI Express Command Register and the “Error Reporting Enables” bit fields (the URR, FER, NFER, or CER bit) in the Device Control Register.
- In its PCI Express Status Register can be gated by the “SERR# Enable” bit in the PCI Express Command Register

However, due to this error, the PCI Express controller in RC mode does not allow the error status reporting in both the PCI Status Register and the Root Error Status Register to be controllable by the PCI Express Command Register [SERR# Enable] bit and the Device Control Register [URR, FER, NFER, CER] bits.

Impact: The inbound errors, including both detected by the PCI Express RC itself and from externally-received error messages, are directly reported to both the PCI Express Status Register [Signaled System Error] bit and the Root Error Status Register [FEMR, NFEMR, FUF, MEFNFR, EFNFR, MECR, ECR] bits, regardless of the setting of the PCI Express Command Register [SERR# Enable] bit or the Device Control Register [URR, FER, NFER, CER] bits.

However, for externally-received error messages, the error status reporting in both the PCI Express Status Register [Signaled System Error] bit and the Root Error Status Register [FEMR, NFEMR, FUF, MEFNFR, EFNFR, MECR, ECR] bits can still be controlled by the Bridge Control Register [SERR_EN] bit as normal, when propagating the error messages from the secondary to primary side of the root port.

Workaround: Although the PCI Express RC error status reporting in both the PCI Status Register and the Root Error Status Register cannot be controllable by the PCI Express Command Register [SERR# Enable] bit and the Device Control Register [URR, FER, NFER, CER] bits, the system software can still use the Root Control Register [SEFEE, SENFEE, SECEE] bits and Root Error Command Register [FERE, NFERE, CERE] bits to control the interrupt generation, for the inbound errors reported in the Root Error Status Register [FEMR, NFEMR, FUF, MEFNFR, EFNFR, MECR, ECR] bits.

Note that the inbound errors reported in the PCI Express Status Register [Signaled System Error] bit does not trigger an interrupt.

Fix plan: No plans to fix
**A-008432:** Optional programmable PCI Express iATU CFG shift feature is not supported for outbound configuration transactions in RC mode

**Affects:** PCIe

**Description:** The PCI Express RC controller’s implementation of the optional enhanced configuration access mechanism (ECAM) feature violates the PCI Express base specification requirement that all reserved fields should always be set to zero. The ECAM implementation requires the iATU to have the bus/device/function (BDF) address shifted 4 bits down from bits 31 to 27 so that the entire configuration space can be mapped into a 256 MB region, rather than requiring multiple address translation tables, or a 4 GB translation space. The BDF is then supposed to be shifted back up from bits 27:12 to 31:16 when forming the byte 8 and 9 of the outgoing configure TLP header.

Due to this erratum, the PCI Express RC controller’s iATU does not perform the EACM required address translation correctly when the CFG shift bit is set in an iATU entry.

**Impact:** The optional CFG shift feature for PCI Express ECAM cannot be used for outbound configuration transactions in RC mode.

**Workaround:** Software must set the CFG shift mode bit (bit 28) to zero in the Outbound iATU Region Control 2 Register. This register is located at offset 908h of the PCI Express RC controller address space. Additionally, use a regular address translation region for configure access.

**Fix plan:** No plans to fix
A-008822: Change the default AXI system error response behavior for PCI Express outbound non-posted requests

Affects: PCIe
Description: By default, when the PCI Express controller experiences an erroneous completion from an external completer for its outbound non-posted request, it always sends an OKAY response to the device’s internal AXI slave system interface. This is desirable for outbound configure transactions to prevent an unnecessary error response from propagating through higher-level system hierarchy, because erroneous completion is a commonly expected behavior during PCI Express bus scan.

However, such default system error response behavior cannot be used for other types of outbound non-posted requests. For example, the outbound memory read transaction requires an actual ERROR response when experiencing erroneous completion from an external completer, like UR completion or completion timeout.

Impact: The device's higher level system hierarchy cannot detect the error condition when the PCI Express controller experiences an erroneous completion from the external completer for its outbound non-posted request. This is not the case for configure transactions.

Workaround: Write to the PCI Express controller’s configure space offset 8D0h with 0000_9401h during the pre-boot initialization (PBI) process.

Fix plan: No plans to fix
A-009151: PCI Express controller does not correctly set the Link Autonomous Bandwidth Status and Link Bandwidth Management Status bits of the Link Status Register in RC mode

Affects: PCIe

Description: The Link Status Register Bit 15, Link Autonomous Bandwidth Status (LABS) and Bit 14, Link Bandwidth Management Status (LBMS) are only meaningful for PCI Express RC’s downstream port. The PCI Express base specification requires the RC downstream ports to:

- Set the LABS bit when the link width or speed is changed by downstream link partner for autonomous reasons only, not for reason of attempting to correct unreliable link operation
- Set the LBMS bit when the link width or speed is changed due to reliability issues

Due to this erratum, the PCI Express RC controller behaves incorrectly as below:

- Does not set the LABS bit when the link width or speed is changed by downstream link partner for autonomous reasons
- Does not set the LBMS bit when the link width or speed is changed due to reliability issues
- Set the LBMS bit when the link width or speed is changed by the downstream link partner for autonomous reasons, which should be the function of LABS bit

Impact: The system software cannot utilize the PCI Express RC controller's Link Status Register [LABS, LBMS] bits to determine the cause of the possible link width and/or speed change, especially for test and debug purpose.

Workaround: Software should not utilize the functionality of the PCI Express RC controller's LABS and LBMS bits of the Link Status Register.

Fix plan: No plans to fix
A-009410: **PCI Express iATU logic does not retain configuration after a hot reset or link-down event in EP mode**

**Affects:** PCIe

**Description:** The internal address translation unit (iATU) is used for remapping the address regions between system application and PCI Express address space.

When the PCI Express Controller is configured in EP mode, the iATU logic's configuration is reset after a hot reset or link-down event.

**Impact:** When a hot reset or link-down event occurs, PCI Express EP controller's existing iATU configuration is lost. The address translation required for all the inbound and outbound memory transactions does not work as expected. The PCI Express EP controller's local software has to reconfigure the iATU immediately after a hot reset or link-down event occurs.

**Workaround:** Upon detecting a hot reset or link-down event, EP’s iATU configuration must be re-configured immediately so that the remote PCI Express host can obtain the correct information from EP during bus re-scan. A PCI Express link-down-and-link-up interrupt handler has to be developed for both bootloader and operating system (OS) environment running locally in EP to cover the following possible scenarios:

- If the link goes down during the bootloader stage, the above interrupt handler has to be called and re-initialize the iATU with its original configuration.
- If the link goes down after the control is passed from the bootloader (like uBoot) to OS (like Linux kernel), a similar interrupt handler must be called to re-initialize the iATU with its original configuration.

System and software developers must take precaution to cover any uncertainty when the control is transitioning from bootloader to OS.

**Fix plan:** No plans to fix
A-009518: PCI Express EP controller's non-D0 PowerState is overridden to D0 when any single or combination of the Bus Master Enable (BME) or Memory Space Enable (MSE) bits are enabled from disabled setting by host software

Affects: PCIe
Description: When the PCI Express EP controller’s PMCSR [PowerState] is set to non-D0 (including D1, D2 or D3), host software’s update to any single or combination of the BME or MSE bits in EP’s Command Register should not cause the EP’s PowerState to settle at D0.

Due to this erratum, when the EP’s PowerState is at non-D0, if the host software enables BME and/or MSE bits from their disabled setting (writing an 1b to a bit when its previous value is 0b), the EP controller’s PowerState is overridden to D0.

Impact: Instead of remaining in Non-D0, PCI Express EP controller’s PMCSR [PowerState] bit field is overridden to D0 due to the host software’s action of enabling any single or combination of the BME or MSE bit fields in EP’s Command Register from their disabled setting. As the result, the link will exit from L1.

However the impact should be small, since host software should not update the EP controller’s Command Register when the EP is in non-D0 PowerState. Instead, it should update EP’s PMCSR [PowerState] first.

Workaround: When the PCI Express EP controller's PMCSR [PowerState] is in non-D0, host software should not issue configure cycles to write to EP Command Register's BME or MSE bit fields.

Fix plan: No plans to fix
A-009520: PCI Express completion transaction layer packets (TLP) of a decomposed outbound memory read request must be returned in order

Affects: PCIe

Description: The PCI Express protocol allows completion TLPs associated with different memory read (MRd) request TLPs to pass each other and be returned out-of-order. If these MRd TLPs are split from a single internal MRd request, the requester is required to re-order the completion TLPs returned out of order.

When the PCI Express controller’s Device Control Register [Max_Read_Size] is set to 000b (this means the maximum MRd request size is 128 bytes), any outbound MRd request from device’s internal bus master with request size larger than 128 bytes is split into multiple smaller MRd TLPs by the controller to be sent out on the PCI Express link. After receiving these decomposed MRd TLPs, the completer may return the completion TLPs out of order. Due to this erratum, the PCI Express controller as a requester is unable to re-order the completion TLPs back when re-assembling the completion data.

Impact: When the PCI Express controller’s Device Control Register [Max_Read_Size] is set to 128 bytes, if the completion TLPs returned are out of order for MRd TLPs split from a single internal MRd request with a size larger than 128 bytes, the controller is unable to re-order these completion TLPs when re-assembling the completion data. This results in data corruption.

There is no impact when the PCI Express controller’s Device Control Register [Max_Read_Size] is set to either 001b (256 bytes) or 010b (512 bytes), since the size of MRd requests received by the controller is always 256 bytes or less, which is limited by the device’s internal bus masters. Therefore, no decomposition occurs at the PCI Express controller level.

Workaround: Do not use 128 bytes for Max_Read_Size. Software should ensure that the PCI Express controller’s Device Control Register [Max_Read_Size] is set to either 001b (256 bytes) or 010b (512 bytes).

Fix plan: No plans to fix
A-009531: IDO bit is set in completion packet header even when the IDO Completion Enable bit is cleared

**Affects:** PCIe

**Description:** Section 2.2.6.4, "Relaxed Ordering and ID-Based Ordering (IDO) Attributes" of the PCI Express Base Specification Rev 3.1 defines:

"A Completer is permitted to set IDO only if the IDO Completion Enable bit in the Device Control 2 Register is set. It is not required to copy the value of IDO from the Request into the Completion(s) for that Request".

However, the PCI Express controller as the completer sets the IDO bit in the completion packet header. This is in response to non-posted requests (memory read) with the IDO bit set in the packet header, even if the IDO Completion Enable bit in the Device Control 2 Register is not set.

**Impact:** The PCI Express controller as the completer sends completion packets with the IDO bit set in packet header even when the IDO Completion Enable bit is cleared in the controller's Device Control 2 Register.

However, this impact is expected to be minimum as explained below.

The PCI Express IDO is an uncommon feature with some limited benefits in specific applications or usage models or both, such as without PCIe switch or Peer-to-Peer traffic within the PCI Express fabric. The Appendix E.5.2, "Software Control of Root Port IDO Use" of the PCI Express Base Specification Rev 3.1 defines:

"Since there are no envisioned high-benefit "simple" use models for Root Ports setting the IDO bit with TLPs they originate, and there are known communication failure cases if Root Ports set the IDO bit with all applicable TLPs they originate, it is anticipated that Root Ports will rarely be enabled to set IDO in TLPs they originate".

Therefore, when the PCI Express controller is configured in:

- **EP mode:** Since the remote Root Ports will rarely be enabled to set the IDO bit in TLPs from which they originate, the chance of the PCI Express EP controller to falsely return the completion packets with the IDO bit set in packet header is very slim.
- **RC mode:** The impact occurs if the RC controller receives any memory read request with the IDO bit set in header from downstream EP. Therefore, the EP must anticipate and be able to handle the completion returned with the IDO bit set when the software turns on the IDO Request Enable bit in the EP's Device Control 2 Register.

**Workaround:** It is recommended that the software keeps both the IDO Request Enable and IDO Completion Enable bits of Device Control 2 Register cleared within the entire PCI Express system fabric.

**Fix plan:** No plans to fix
A-009700: Secondary PCI Express extended capability is included in the capability structure linked list in Gen1 and Gen2 modes

**Affects:** PCIe

**Description:** The secondary PCI Express extended capability structure is only applicable for PCI Express controller operating at Gen3 and mainly used for the equalization process during link training at Gen3 speed.

Due to this erratum, the secondary PCI Express extended capability structure presents in the PCI Express capability structure linked list even when the controller operates at Gen1 or Gen2 speed.

**Impact:** The secondary PCI Express extended capability structure irrelevant to the current controller configuration speed may cause confusion to the PCI Express enumeration software, if the software cannot ignore this irrelevant capability structure presented in the linked list.

However, the impact should be negligible, since the Gen1 or Gen2 PCI Express controller should not initiate any link training process toward the Gen3 speed.

**Workaround:** Workaround procedure to overwrite the affected capabilities pointers is not feasible since the affected PCI Express capabilities pointer registers cannot be updated. The alternative approach described below can be adopted for the host system software running at RC side to ignore the irrelevant Secondary PCI Express Extended Capability structure presented in the linked list.

For the rest of the workaround section, the speed the controller configured to support refers to the maximum link speed that a PCI Express controller is configured to operate for a chosen SRDS_PRTCL_Sn setting within a RCW.

Regardless the PCI Express controller is configured as RC or EP, instead of directly using the information read from the configuration space registers, the host system software has to build its PCI Express capabilities linked list to ignore the irrelevant Secondary PCI Express Capability structure presented at the end of the linked list of this controller, located at configuration space offset 0x148.

The correct linked list should only contain the Advanced Error Reporting Capability structure for the non-Gen3-capable PCI Express controller.

**Fix plan:** No plans to fix
A-010053: PCI Express controller's transmitter does not enter ASPM L0s state when enabled

Affects: PCIe

Description: According to the PCI Express Base Specification Rev 3.0, a transmitter must implement L0s if its port advertises support for L0s, as indicated by the ASPM support bit field of the Link Capabilities register when the ASPM optionality compliance (AOC) bit is set. The ASPM control is independent on the transmit and receive path of a PCI Express link. For the transmitter, the ASPM L0s is enabled by setting the PCI Express Link Control register [ASPM_CTL] to 01b (L0s Entry Enabled). However, the receiver is required to enter L0s regardless of the setting of the Link Control register [ASPM_CTL] bit field as long as L0s is supported.

Although the Link Capabilities register of this device’s PCI Express controller advertises the support of ASPM L0s, its transmitter does not have the required implementation.

Impact: The PCI Express controller receiver’s ASPM function is not affected regardless of the setting of the Link Control register [ASPM_CTL] bit field. Only the transmit path of the link is impacted. When software enables the ASPM L0s by setting the PCI Express Link Control register [ASPM_CTL] to 01b (L0s Entry Enabled), it has no effect on the transmitter’s ASPM state. The transmitter remains in the fully active L0 state instead of entering L0s link ASPM state. Although this should not cause a functional failure, it is unable to deliver any power saving to the link for the ASPM feature, which is negligible.

Any compliance test expecting the transmitter of the PCI Express controller to enter L0s may fail.

Workaround: Although there is no workaround to recover the transmitter’s ASPM L0s functionality, the following procedures can be used to disable the ASPM functionality of the link that this PCI Express controller belongs to during normal operation:

- The PCI Express ASPM policy is normally controlled globally by the operating system. If it is feasible, turn off the ASPM support globally in the OS for the whole PCI Express fabric that the device belongs to.
- If ASPM cannot be turned off globally, the ASPM support of the affected device’s PCI Express link can be disabled by setting the PCI Express link control register [ASPM_CTL] = 00b for both the NXP PCI Express controller and its link partner.

Fix plan: No plans to fix
A-010305: The Maximum Link Width and Supported Link Speed bit fields of the Link Capabilities register may reset to values higher than that configured by RCW in the EP mode after link-down or hot-reset event

Affects: PCIe

Description: The device offers various SerDes protocol options in RCW. Some of these options configure the link width and speed to certain values smaller than what the PCI Express controller can natively support. For example, although a PCI Express controller can natively support up to Gen3 x4, some of the SerDes protocol options may only offer Gen2 x2, which are reflected in the Maximum Link Width and Supported Link Speed bit fields of the Link Capabilities Register.

When a link-down or hot-reset event occurs, the Link Capabilities Register should retain the values of the Maximum Link Width and Supported Link Speed bit fields configured by RCW. Due to this erratum, these values are reset to the controller's native Maximum Link Width and Supported Link Speed. For example, these values become Gen3 x4 even though RCW configuration is Gen2 x2.

Impact: The Maximum Link Width and Supported Link Speed bit field values of the Link Capabilities register are not maintained after link-down or hot-reset event when the PCI Express controller is configured in the EP mode. Only the Maximum Link Width and Supported Link Speed bit field values of the Link Capabilities register are affected. During link re-train after link down:

- The negotiated link speed is not affected, since the Target Link Speed bit field of the Link Control 2 Register is not affected and remains sticky, which is also used during link re-train.
- The negotiated link width should not be affected since the number of active lanes detected should remain the same during link re-train.

Workaround: Although the negotiated link width and speed are not impacted during link re-train after link-down or hot-reset event, PCI Express EP controller's local software should re-apply the correct values of these two bit fields set by RCW to the Link Capabilities Register. This allows the remote host to read the correct information from the EP controller. A PCI Express link-down and link-up interrupt handler has to be developed in both bootloader and operating system (OS) environments running locally at the PCI Express EP controller. This covers the following possible scenarios and avoids incorrect information being returned to remote host systems.

- If the link goes down or hot-reset event occurs during the bootloader stage, the above interrupt handler has to be called to re-apply the previously saved values of Maximum Link Width and Supported Link Speed bit fields to the read-only Link Capabilities Register.
- If the link goes down or hot-reset event occurs after the control is passed from the bootloader to OS, a similar interrupt handler must be called to re-apply the previously saved values of Maximum Link Width and Supported Link Speed bit fields to the read-only Link Capabilities Register.

System and software developers must take precaution to cover any uncertainty when the control is transitioning from bootloader to OS.

The following sequence represents guidelines for implementing the PCI Express link-down and link-up interrupt handler. It uses the bootloader stage as an example. The OS handler implementation should be the same.

1. Right after Power On Reset (POR), the software should read the EP controller's Link Capabilities Register to obtain and save the correct values of Maximum Link Width and Supported Link Speed bit fields configured by RCW.
2. After the boot loader determines the link has been trained successfully, it can proceed with the PCI Express initialization. Once done, it should perform a W1C to at least clear the Hot Reset Detected (HRD), Link Down Detected (LDD) and Link Up Detected (LUD) bits in the EP controller's PEX_PF0_PME_MES_DR Register located at controller's configuration space offset 0xC_0020. This allows the EP controller to detect future hot-reset, link-down and link-up events.

3. The Hot Reset Detect Disable (HRDD), Link Down Detect Disable (LDDD) and Link Up Detect Disable (LUDD) bits should be cleared in the EP controller's PEX_PF0_PME_MES_DISR Register located at controller's configuration space offset 0xC_0024.

4. The Hot Reset Detect Interrupt Enable (HRDIE), Link Down Detect Interrupt Enable (LDDIE) and Link Up Detect Interrupt Enable (LUDIE) bits should be set in the EP controller's PEX_PF0_PME_MES_IER Register located at controller's configuration space offset 0xC_0028.

5. The handler should be activated to function at this time. If any hot-reset or link-down event occurs, the handler should be called to service. Once the link-up event occurs, the handler can confirm whether the correct values has been re-applied to the Link Capabilities Register during the re-initialization process.

The handler has to ensure that the overwrite capability for read-only Link Capabilities Register is enabled locally by setting the DBI_RO_WR_EN bit (Bit 0) of the DBI Read-Only Write Enable Register at PCI Express configuration space offset 0x8BC, before writing to read-only registers in the configuration space. To prevent unintentional overwrite by software, the local handler should minimize the timing window of enabling the DBI_RO_WR_EN bit. In other words, the local handler should only set the DBI_RO_WR_EN bit when it is ready for the overwrite task and must clear the DBI_RO_WR_EN bit right after the overwrite task is finished.

**Fix plan:** No plans to fix
A-008886: Sometimes unexpected data is written to the external flash memory even though the underrun bit (QuadSPI_FR[TBUF]) is not set

**Affects:** QSPI

**Description:** While carrying out continuous writes from the Tx buffer to the flash memory, there may be scenarios when the buffer is empty for some duration and gets filled later. For example, QuadSPI_TBSR[TRBFL] changes from non-zero to zero and again to non-zero in the middle of a flash transaction. Such a case may trigger unexpected or wrong data to be written into the flash memory, even though the underrun bit (QuadSPI_FR[TBUF]) is not set.

**Impact:** A write to the flash memory may not work correctly.

**Workaround:** Break the flash page writes into smaller chunks of Tx FIFO size and fill the FIFO before the write is initiated. For example, for a flash page size of 128 bytes, two separate write transactions of 64 bytes each (Tx FIFO size) must be initiated. This ensures that, for a single continuous write, the Tx FIFO never becomes empty during the write transaction on the flash interface.

**Fix plan:** No plans to fix
A-009283: Illegal accesses to SPI flash memory can result in a system hang

Affects: QSPI

Description: Under normal circumstances you can program the QuadSPI_LUTn using a read instruction so that the software reads data correctly from flash memory through the AMBA-AHB system bus (AHB). However, when programming the QuadSPI_LUTn as a non-read sequence, the flash memory does not send back any data and the system may hang, which requires a reset to recover. This is considered illegal programming. There is no time-out mechanism to recover from this scenario.

Impact: The system must be reset for illegal programming.

Workaround: Use a watchdog timer of 1 second. The start of this timer can be triggered before a read through AHB. If it expires, the system needs to be reset (set DCFG_CCSR_RSTCR[RESET_REQ]). The timer must be cleared when Quads Pi_SR[BUSY]=0.

Fix plan: No plans to fix
A-010284: Insufficient read data may be received in the Rx Data Buffer register

Affects: QSPI

Description: Data read from flash through QuadSPI using the Internal Peripheral Bus (IPS) interface may return insufficient data in the Receiver (Rx) Buffer Data (QuadSPI_RBDRn) register when the read data size of a flash transaction is programmed to be greater than 32 bytes.

Impact: Read from a flash through QuadSPI using the IPS interface may cause buffer overflow.

Workaround: For data size greater than 32 bytes, program the IP data transfer size in the IP configuration register (QuadSPI_IPCR[IDATSZ]) to be in multiples of 8 bytes.

Fix plan: No plans to fix
A-008588: SATA controller may hang when processing multiple 16-byte non-aligned PRD entries for read operations

**Affects:** SATA

**Description:** SATA controller may enter a hang state when multiple PRD entries not 16-byte aligned are processed. The non-alignment from either the address or the byte count can lead to a hang. The hang results in the command time out.

**Impact:** If the workaround is not implemented, performance is impacted due to the command time out.

**Workaround:** For applications that use multiple PRDs, software must force the PRD alignment. Both address and byte count in each PRD should be 16-byte aligned.

**Fix plan:** No plans to fix
A-009185: The default Rx watermark value may be insufficient for some hard drives

**Affects:** SATA

**Description:** The PTC[RXWM] sets the watermark value for Rx FIFO. The default value 0x20 might be insufficient for some hard drives. If the watermark value is too small, a single-cycle overflow may occur and is reported as a CRC or internal error in the PxSERR register.

**Impact:** The default Rx watermark value might be insufficient and result in a false CRC or internal errors.

**Workaround:** Change PTC[RXWM] field at offset 0xC8 to 0x29. Do not change the other reserved fields of the register.

**Fix plan:** No plans to fix
A-010554: SATA controller might fail to detect some hard drives

**Affects:** SATA

**Description:** With the default SerDes register setting, the SATA controller might fail to detect some hard drives.

**Impact:** The SATA controller does not detect some hard drives reliably with the default SerDes register setting.

**Workaround:** Either during PBI phase, or in software before enabling the SATA controller, write value 0x80104e20 to address 0x1ea1300.

**Fix plan:** No plans to fix
A-010635: Early termination of Read DMA Operation due to erroneous CRC Error

**Affects:** SATA

**Description:** SATA controller implements a logic to check for ECC errors when accessing dual-port memory inside the controller. Due to this bug, the logic may mistakenly report ECC error and cause read DMA read operation to be terminated prematurely. The ECC error is reported as CRC error in the PxSERR status register.

**Impact:** Device may see false CRC errors causing unreliable SATA operation.

**Workaround:** Either in PBI or in software before enabling SATA, disable ECC check by writing the data 0x80000000 to the address 0x20140520.

**Fix plan:** No plans to fix
A-006385: SEC watchdog timer does not prevent all cases of illogical descriptors from hanging DECOs

Affects: SEC

Description: The SEC block contains a watchdog timer designed to clear errors that have hung a DECO. Although mostly effective, there are cases in which poorly constructed descriptors can hang a DECO from which a watchdog cannot recover.

The SEC's programming model is based on the construction of descriptors, which include commands and, optionally, embedded keys and context. Users are expected to use the SEC's driver (SEC Descriptor Runtime Assembler) to build descriptors; however, because there is considerable innate flexibility in descriptor construction — even with the Descriptor Runtime Assembler — it is possible to create descriptors that appear to follow the rules of descriptor construction but still lead to errors, hangs, or corrupted data.

A few examples of descriptor constructions that can lead to hangs not cleared by the watchdog are:

- Ending a descriptor with a LOAD IMM command.
- Creating descriptors that command the PKHA to unload more data than the RAMs actually store.

Impact: Certain non-standard descriptor constructions can lead to errors, hangs, or corrupted data.

Workaround: To avoid this erratum, it is highly recommended that users model their descriptors after the examples provided in the NXP reference software. If there is a need to create a descriptor for which an example does not exist, do so based on the SEC Programmer's reference manual.

Fix plan: No plans to fix
A-010541: Unexpected SEC behavior at soft reset when DECO is busy waiting on PKHA

Affects: SEC
Description: When DECO is processing a job and is waiting on the PKHA, and a SEC soft reset is requested by setting the SWRST bit in the Master Configuration register (MCFGR), DECO does not react to the soft reset. This causes unexpected behavior in SEC.

Impact: This could cause a hang or other errors if the PKHA job completes and is sent to a job ring, AI, or QI that is no longer expecting the result.

Workaround: Perform a DECO reset immediately before the CAAM soft reset. A DECO reset is performed by writing all 1s to the DECO Reset Register.

Fix plan: No plans to fix
A-007731: Mixing 16-bit and 32-bit frame sizes in XSPI mode can cause incorrect data to be transmitted

Affects: SPI

Description: The Serial Peripheral Interface (SPI) features an Extended SPI (XSPI) mode supporting frames of up to 32 bits. When the XSPI mode is enabled, transferring a mixture of frames having a size up to 16 bits and those having a size greater than 16 bits can cause an incorrect data transmission to occur. This happens when the First In/First Out (FIFO) queue read pointers roll over, and a frame needs to be extracted from both the bottom and top of the FIFO when the frame size is greater than 16 bits.

This erratum is applicable only on the Transmit (Tx) side. No problem has been found on the Receive (Rx) side.

Impact: Incorrect data transmission can occur if frames up to 16 bits are mixed with frames greater than 16 bits in XSPI mode.

Workaround: Do not mix Tx data frames that have data sizes less or equal to 16 bits with data frames greater than 16 bits.

When Tx FIFO depth is even (even number of TXFR registers): For frames less than or equal to 16 bits, each Tx FIFO entry constitutes one frame. For frames greater than 16 bits, two entries of FIFO constitute one frame. Therefore, it does not cause the Tx FIFO pointer to roll over.

When Tx FIFO depth is odd (odd number of TXFR registers): For frames greater than 16 bits, send one Tx FIFO entry as a dummy frame (that is, a frame with no chip select but containing data) because the FIFO depth is odd. After this dummy frame transfer, even entries are left in the FIFO. Now, two FIFO entries each constitute one frame until the whole depth of FIFO is covered. Therefore, a dummy frame is needed after sending (FIFO depth - 1)/2 frames (that is, after the whole FIFO depth has been covered).

Fix plan: No plans to fix
A-009776: Loading of shift register data into the receive FIFO following an overflow event

Affects: SPI
Description: 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 Receive FIFO bit in Module Configuration Register (MCR \( [CLR \ _RXF] \)) is asserted to clear the receive FIFO, the shift register data is wrongly loaded into the receive FIFO after the clear operation completes.

Impact: The receive FIFO content may be incorrect following an overflow event.

Workaround: Use one of the following workarounds:
- Avoid a receive FIFO overflow condition \( \([SR \ [RFOF] \text{ should never be } 0b1] \)\). To do this, monitor the receive FIFO counter field of the Status Register \( [SR \ [RXCTR]] \) which indicates the number of entries in receive FIFO and clear it before the counter equals the FIFO depth.
- 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.

Fix plan: No plans to fix
A-009895: Inconsistent addition to Transmit or Command FIFO if an entry is added while FIFOs are full

**Affects:** SPI

**Description:** In the Serial Peripheral Interface (SPI) module, the Transmit FIFO and Command FIFO are updated simultaneously when the PUSH TX FIFO Register (PUSHR) is written with a command and transmit data word. In the scenario where:
1. Frame size is configured to 16 bits or less by disabling any extended frame functionality;
2. Transmit FIFO is full (Transmit FIFO Fill Flag in Status Register (SR[TXFFF]) = 0b0); and
3. One entry each from both the Transmit FIFO and Command FIFO is read for next frame transfer.

If, at this time, a new entry is added to the FIFOs through PUSHR, only one of the two FIFOs is updated with the new data.

**Impact:** An addition to Transmit FIFO or Command FIFO may fail.

**Workaround:** Check the fill status of the transmit FIFO before attempting to add new entries. Write PUSHR only when transmit FIFO is not full (SR[TXFFF] = 0b1).

**Fix plan:** No plans to fix
A-005697: Suspend bit asserted before the port is in Suspend state

**Affects:** USB

**Description:** The EHCI specification states the following in the SUSP bit description:

> *In the Suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.*

In the USBDR controller, the PORTSCx[SUSP] bit changes immediately when the application sets it and not when the port is actually suspended.

**Impact:** Even though the behavior of the USBDR violates the EHCI specification statement, it does not cause any functional issue as, according to the EHCI specification, the application must wait for at least 10 milliseconds after a port indicates that it is suspended before initiating a port resume using the Force Port Resume bit.

**Workaround:** The software driver must follow the EHCI specification by waiting for at least 10 ms after setting the PORTSCx[SUSP] bit.

**Fix plan:** No plans to fix
A-008118: USB initial sequence impact for external ULPI PHY

Affects: USB

Description: USBDR controller soft reset has a dependency on PHY reset completion. In USBDR, the default value of the PORTSC[PTS] field is ULPI. Additionally, if the actual PHY connected is also ULPI, then the USBCMD[RST] bit should de-assert.

However, the USBDR controller requires a stable ULPI clock to detect ULPI PHY RESET completion. Due to an internal mux on ULPI and UTMI CLK, ULPI CLK may not reach USBDR in time. Hence, the reset might not de-assert. To enable this internal mux, write 1 to USB_CONTROL[USB_EN]. This results in ULPI CLK reaching the UDBDR core so the RESET bit de-asserts. This CLK dependency requires a change in the initialization sequence, that is, the soft reset can not be issued without setting USB_EN bit in the USB_CONTROL register.

Impact: Controllers having ULPI PHY may not come out of reset before USB_CONTROL[USB_EN] bit is selected.

Workaround: Do not check for USBCMD[RST] deassertion before writing 1 to USB_CONTROL[USB_EN].

Fix plan: No plans to fix
**A-008260:** Device controller accepts new OUT packets even when TRBs are not available

- **Affects:** USB
- **Description:** When the device application uses short packets on OUT endpoints, if more TRBs are present in a buffer descriptor chain, depending on certain conditions, the device might accept OUT data even though the device controller is not setup to accept OUT packets.

This issue occurs when all of the following conditions are met:
- TRBs are set up with CSP (Continue on Short Packet) = 1 for OUT endpoints
- An OUT short packet is received on USB
- There are more TRBs in the chain than the USB transfer size (fast forward TRBs)
- When the fast forward starts, only some TRBs in the chain are ready, but not all TRBs on the same TD (chain) are available in the controller
- TRB fetch happens close to the fast forward action and the internal cache RAM access for the OUT endpoint fetch logic gets delayed because of other traffic (probably on other endpoints)
- TRBs are not available for the new chain (that is, TRB cache is empty)
- USB packet arrives during the cache empty condition

When all the above conditions are met, the controller incorrectly accepts a new OUT packet, even though there is no TRB available to process it. When this happens, depending on the state of the previously cached TRB (described below), the controller might DMA the data or get hung.
- If the old TRB has a valid buffer size, then the controller performs DMA to the data address in the old TRB. This address is incorrect; however, if the system is not decoding the full address, then it might accept the packet.
- If the old TRB has zero buffer size, then the controller hangs trying to do a receive DMA of zero bytes

**Probability of Occurrence:** It is rare for all the above conditions to occur simultaneously, so this issue is not common. However, the condition could occur on a system with CSP mode and frequent fast-forwarding.

**Impact:** When this issue occurs and controller performs a receive DMA of zero bytes, then the device hangs, sending NAKs on the USB for the OUT endpoint. The host needs to re-enumerate and the device needs to be reset (software reset) to return to normal operation.

**Workaround:** Use one of the following workarounds:
- Not using CSP=1 mode. If CSP feature is not used, then the controller stops processing the TRB list after the current TRB is written back on a short packet, and waits for the SW to restart the list using the Start Transfer command. This can introduce minor performance impacts.
- When using CSP=1 mode, use single TRB per TD. If CSP feature is used, but only one TRB is prepared for a TD (Transfer Descriptor), then the controller on receiving a short packet, retires the current TRB/TD and goes to the next one in the list. There is no fast-forward (that is, skipping of TRBs) when using one TRB per TD.
- When using CSP=1 mode with multiple TRBs in a chain, do not allow the TRB cache to go empty. This can be ensured by having either a larger TRB cache or fewer TRBs in a TD so that the TRB cache can fit multiple TDs, prepare multiple TDs in advance, and issue update transfers upfront. If you make sure that the device does not fetch the TRBs...
of the same chain when doing the fast-forward operation, it does not encounter the incorrect TRB cache empty condition during fast-forward operation, which is one of the required conditions for this issue to occur.

**Fix plan:** No plans to fix
A-008428: Device initiates low power when ACK pending for data packet

**Affects:** USB

**Description:** When the USB controller is configured as a USB device mode, the device initiates low power when an ACK is pending for a data packet (DP). When operating in SuperSpeed mode and when the internal condition for low power (u1/u2) is satisfied, the device initiates u1/u2 even though it has just received a DPH of the DP header (DPH). This causes the link to enter and exit low power before the device sends an ACK for the DP. This behavior can cause a transaction timeout on the host for the DP.

**Impact:** Depending on the host transaction timeout value, the host may timeout on the transaction and the host retries the transfer. If the same issue happens again, this could result in the host resetting the device and re-enumerating.

**Workaround:** Disable USB_DCTL (InitU1Ena, InitU2Ena) bits. As a result, the device does not initiate low-power requests; however, it can still accept low-power requests from the host/hub and enter low power.

**Fix plan:** No plans to fix
### A-008459: Device in 2.0 mode handles DATA PID error on SETUP data incorrectly

**Affects:** USB  
**Description:** When the device operates in 2.0 mode and the SETUP data packet has incorrect DATA PID (! =DATA0), the device handles the data incorrectly and subsequent transactions are missed. This is a rare occurrence. When it does happen, the device eventually recovers.

The SETUP data packet always has the PID of DATA0. Due to an error on USB, if the DATA PID on the SETUP data arrives to the controller as non-DATA0, then the device times out (as expected). However, it handles this transaction incorrectly internal to the controller.

When this condition happens, the subsequent transaction is ignored and, if it is a good transaction, the data is missed/dropped for that transaction. The device retries the SETUP transfer and, this time, the device recovers and responds correctly.

**Impact:** It is rare for DATA PID to be corrupted for SETUP. Typically, when a control transfer is in progress and PID errors occur, the host retries the same setup until it is successful. This behavior eliminates any effects on device functionality because the device syncs after a successful setup.

**Workaround:** None  
**Fix plan:** No plans to fix
A-008997: USB3 LFPS peak-to-peak differential output voltage adjustment settings

**Affects:** USB

**Description:** Low Frequency Periodic Signaling (LFPS) peak-to-peak differential output voltage test compliance fails using default transmitter settings. Software is required to change the transmitter signal swings to pass compliance tests.

**Impact:** LFPS peak-to-peak differential output voltage compliance test fails.

**Workaround:** Set SCFG_USBxPRM2CR[16:22] (USB3 Parameter 2 Control Register SCFG_USBxPRM2CR[PCSTXSWINGFULL]) = 7'b 1000_111, where x is number x of the USB controller PHY.

**Fix plan:** No plans to fix
A-009007: USB3 PHY observes an intermittent failure in the receive compliance tests at higher jitter frequencies using the default register values

Affects: USB
Description: The receive compliance tests may fail intermittently at high jitter frequencies using the default register values.

Impact: The receive compliance test fails at the default register settings.

Workaround: Setting (RX_OVRD_IN_HI.RX_EQ) different (identified) value makes the Rx compliance test pass.

The following steps are needed to modify the RX_OVRD_IN_HI.RX_EQ value:
1. Write 1'b0 to RX_OVRD_IN_HI.RX_EQ_EN [offset 16'h200C: bit 6].
2. Write 1'b1 to RX_OVRD_IN_HI.RX_EQ_EN_OVRD [offset 16'h200C: bit 7].
3. Write a fixed value of 4 to RX_OVRD_IN_HI.RX_EQ [offset 16'h200C: bits 10–8]
4. Write 1'b1 to RX_OVRD_IN_HI.RX_EQ_OVRD [offset 16'h200C: bit 11].

Therefore, start with the default value.
1. Write RX_OVRD_IN_HI (0x084F_200C) 0x0000
2. Write RX_OVRD_IN_HI (0x084F_200C) 0x0080
3. Write RX_OVRD_IN_HI (0x084F_200C) 0x0480
4. Write RX_OVRD_IN_HI (0x084F_200C) 0x0C80

NOTE
Applying the transmit errata A-008997 is also required as receiver JTOL testing is done in loopback mode (Rx to Tx).

Fix plan: No plans to fix
**A-009008: USB High Speed (HS) eye height adjustment**

**Affects:** USB

**Description:** USB HS eye diagram fails with the default value at many corners, particularly at a high temperature (105°C).

**Impact:** USB HS eye diagram may fail using the default value.

**Workaround:** Set USB3PRM1CR[TXVREFTUNE](0x157_0070) = 4'b1001 for USB 1.

**Fix plan:** No plans to fix
**A-009149:** USB 2.0 protocol data pins USB\_D\_M and USB\_D\_P do not support IEEE 1149.1

**Affects:** USB

**Description:** For the USB 3.0 PHY with USB 2.0 protocol data pins, the USBBn\_D\_M and USBBn\_D\_P pins do not support IEEE 1149.1 and do not have boundary scan registers associated with the pins. Board-level interconnect for these pins cannot be done with IEEE 1149.1-2001 instructions, such as EXTEST.

**Impact:** Components with the USB 3.0 PHY are not compliant to IEEE 1149.1. The standard requires all digital data pins to have compliant 1149.1 boundary scan registers.

Unless noted elsewhere, the other components are compliant to the IEEE 1149.1-2001 standard.

**Workaround:** Test the USB 2.0 data pins board-level interconnects using other methods at the board level.

**Fix plan:** No plans to fix
A-009163: Device prohibits low power after sending LPF = 1 for isochronous IN

**Affects:** USB

**Description:** This issue happens in SS Device mode for isochronous IN endpoints. When the device does not have data to send during the initial isochronous IN request from the host, the device sends ZLP (zero length packet) with LPF = 1. Based on the reference manual, this causes the controller to allow low power (if all other conditions are met). However, the controller cannot accept low power requests from the host. When the host sends lgo_uX after ZLP (LPF = 1), the device responds with lXu. If there is no ping or no TP for isochronous IN for the next three uframes, the controller allows low power after three uframes of sending the ZLP. The controller does not allow low power only during the three uframes following the ZLP.

**Impact:** While software is preparing the data/TRB and the device controller is prefetching data, the device responds with ZLP DP for isochronous INs. During this time, the device link cannot go into low power mode, thereby, impeding power saving.

**Workaround:** Choose one of the following workarounds:

- Create dummy zero-length TRBs as soon as the host starts polling for isochronous INs, with the starting uframe to be the next or near-future interval/uframe and starting the transfer. This way the controller actually sends zero-byte packets on USB. (This is treated differently than the controller sending ZLP due to the packet not available.)
- After getting the XferNotREady event for the isochronous endpoint, send a DEPCFG command for the isochronous endpoint with "Modify" action with no change in parameters. This resets the internal u1_ep_timer to zero, which allows low power to succeed.

**Fix plan:** No plans to fix
A-009377: Incorrect value for S and E fields for ISOC OUT in Start-Split token

Affects: USB

Description: On the USB, a start-split token is used by the controller to send a high-speed (HS) data packet to a hub which in turn connects to a full-speed (FS) device. The HS payload is split into maximum chunks of 188 bytes in each microframe on the HS USB to be sent to the FS device through the hub. Each 188-byte payload is preceded by a Start-Split token with S and E field encoded to indicate the position of the 188-byte payload (start, middle, end, or only one payload).

On the xHCI, the driver prepares an isochronous (ISOC) TRB indicating the transfer length. The controller reads the TRB and decides on further processing before translating it to comply with the USB protocol. The issue manifests because the S and E encoding values are not properly decoded by the host controller when CH=1 (Chain) and ENT=1 (Evaluate Next TRB). The issue does not occur if the TRB has CH=0.

Impact: There may be a temporary glitch in the ISOC OUT application (for a FS device connected to a hub) because of the incorrect S and E field indication (for example, a glitch in the audio stream).

Workaround: None

Fix plan: No plans to fix
A-009668: Stop Endpoint command does not complete

Affects: USB

Description: This issue is observed in USB 2.0 mode when the USB 3.0 host controller is connected to a FS/LS device via a hub.

The host controller issues start-split (SSPLIT) and complete-split (CSPLIT) tokens to accomplish a split-transaction. A split-transaction consists of a SSPLIT token, token/data packets, CSPLIT token and token/data/handshake packets. A SSPLIT token is issued by the host controller to the hub followed by token/data/handshake packets. The hub then relays the token/data/handshake packets to the FS/LS device. Sometime later, the host controller issues a CSPLIT token followed by the same token/data/handshake packets to the hub to complete the split-transaction.

As an example scenario, when the xHCI driver issues an Address device command with BSR=0, the host controller sends SETUP(SET_ADDRESS) tokens on the USB as part of split-transactions. If the host controller receives a NYET response from the hub for the CSPLIT SETUP token, it means that the split-transaction has not yet been completed or the hub is not able to handle the split transaction. In such a case, the host controller keeps retrying the split-transactions until such time an ACK response is received from the hub for the CSPLIT SETUP token. If the split-transactions do not complete in a time bound manner, the xHCI driver may issue a Stop Endpoint command. The host controller does not service the Stop Endpoint command and eventually the xHCI driver times out waiting for the Stop Endpoint command to complete.

Impact: Stop Endpoint command does not complete.

Workaround: Instead of issuing a Stop Endpoint command, issue a Disable Slot Command with the corresponding Slot ID. Alternatively, you can issue an Address Device command with BSR=1.

Fix plan: No plans to fix
A-009798: USB high speed squelch threshold adjustment

**Affects:** USB

**Description:** The default setting for USB high speed squelch threshold results in a threshold close to or lower than 100mV. This leads to a receiver compliance test failure for a 100mV threshold.

**Impact:** If the errata is not applied, only the USB high speed receiver sensitivity compliance test fails, however USB data continues to transfer.

**Workaround:** Complete the following writes:

```
SCFG_USB3PRM1CR (0157_0070h) write bit[6:8]=3'b000
```

These register writes allow the USB high speed receiver sensitivity compliance test to pass.

**Fix plan:** No plans to fix
A-010127: Hot reset failure during U1/U2 entry for USB 3.0

**Affects:** USB

**Description:** When the xHCI driver issues Port Reset (PORTSC.PR = 1) after the link partners have already exchanged LGO_Ux, LAU, and LPMA link commands for U1/U2 entry, the USB 3.0 controller incorrectly enters the Recovery state to perform a Hot Reset (TS2 ordered sets and TS1/TS2 handshake) instead of entering into U1/U2.

The window for internally transitioning to U1/U2 after LAU and LPMA link commands is 16 mac3_clk (128 ns). This problem occurs only when the xHCI driver programs PORTSC.PR = 1 during the 128 ns window.

**Impact:** There is a delay for the USB 3.0 controller to receive Warm Reset from the xHCI driver.

**Workaround:** No workaround is required. If a Hot Reset fails because of a TS1/TS2 handshake timeout, a downstream port transitions to SS.Inactive, which generates a PORTSC.PLC interrupt to the xHCI driver. The xHCI driver then programs a Warm Reset to the controller.

**Fix plan:** No plans to fix
A-010129: USB 2.0 reset not driven while port is in the Resume state

Affects: USB

Description: This issue is applicable only to USB 2.0 ports in Host mode and assumes that PORTSC[PLS] = 1111b. If the xHCI driver resets the USB 2.0 port by programming PORTSC[PR] = 1, then the USB controller does not drive reset and does not generate an interrupt (PORTSC[PRC] = 1) while it is in the USB 2.0 Resume state.

Impact: The following are impacts for this erratum:
- The USB 2.0 port cannot be reset.
- The xHCI driver does not receive a port status change event interrupt (PORTSC[PRC] = 1).

Workaround: The xHCI driver should not program a USB 2.0 reset (PORTSC[PR] = 1) while in the Resume state. When the xHCI driver is ready to program a USB 2.0 reset, check the PORTSC[PLS] bit and only program PORTSC[PR] = 1 when PORTSC[PLS] is not set to 1111b.

Fix plan: No plans to fix
**A-010131: U3 request gets dropped when controller tries U1-to-U2 entry**

**Affects:** USB

**Description:** This erratum is applicable for the USB 3.0 Super Speed host mode operation. When the U2 timer expires while in U1 mode, the USB 3.0 controller completes a U1->U2 entry operation lasting three mac3_clk (24 ns). If the xHCI driver issues a U3 request during this operation, the controller drops this request.

**Impact:** The controller ignores the request when the xHCI driver programs the U3 entry (PORTSC.PLS = U3). The occurrence of this issue is rare because of the 24 ns window.

**Workaround:** The xHCI driver must include the following steps:
1. Before initiating U3 entry, save PORTPMSC.
2. Disable U2 entry by programming PORTPMSC[U2 Timeout] = h'FF.
3. After U3 entry, re-enable the U2 timer by programming PORTPMSC with the value saved in Step 1.

**Fix plan:** No plans to fix
A-010151: Unreliable receiver detection in low power P3 mode

Affects: USB
Description: The USB 3.0 controller enables the Receiver (Rx) Detection feature in low power mode 3 (P3 mode). However, USB 3.0 PHY does not reliably support receiver detection in P3 mode. Therefore, some USB 3.0 devices are not detected reliably in Super Speed mode.

This erratum does not cause a compliance issue, because the receiver detection in P3 mode is beyond the PHY Interface for PCI Express and USB 3.0 (PIPE) specification which only requires receiver detection in power mode 2 (P2 mode).

Impact: Some USB 3.0 pen devices cannot be detected reliably in Super Speed mode.

Workaround: Set GUSB3PIPECTL[DisRxDetP3]=1 to configure USB 3.0 in P2 mode for receiver detection.

Fix plan: No plans to fix
LS1021A, LS1020A, and LS1022A Chip Errata

This document details all known silicon errata for the LS1021A, LS1020A, and LS1022A. This table provides a revision history for this document.

### Table 1. Document revision history

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Significant changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>04/2018</td>
<td>Moved SFP erratum A-006879 to Trust Architecture document. Contact NXP sales representative for further details.</td>
</tr>
<tr>
<td>8</td>
<td>04/2018</td>
<td>Added eDMA erratum A-011218 &lt;br&gt;Removed QSPI erratum A-009282</td>
</tr>
<tr>
<td>7</td>
<td>09/2017</td>
<td>Added FlexTimer erratum A-011026 &lt;br&gt;Replaced ARM with Arm throughout the document as per new Arm branding guidelines</td>
</tr>
<tr>
<td>6</td>
<td>07/2017</td>
<td>Added the following errata: &lt;br&gt;• qDMA A-010812 and A-010840 &lt;br&gt;Removed the following errata: &lt;br&gt;• I2C A-007810 and A-007812</td>
</tr>
<tr>
<td>5</td>
<td>02/2017</td>
<td>Added the following errata: &lt;br&gt;• I2C A-010650 &lt;br&gt;• LPUART A-010727 &lt;br&gt;• SEC A-010442</td>
</tr>
<tr>
<td>4</td>
<td>11/2016</td>
<td>Added the following errata: &lt;br&gt;• Arm Cortex-A7 A-010476 &lt;br&gt;• GEN A-010539 &lt;br&gt;• LPUART A-010581 &lt;br&gt;• QSPI A-010284 &lt;br&gt;Modified the following erratum: &lt;br&gt;• DDR A-009942</td>
</tr>
<tr>
<td>3</td>
<td>05/2016</td>
<td>Added the following errata: &lt;br&gt;• GIC A-010111 &lt;br&gt;• PCIe A-010315 &lt;br&gt;• QSPI A-009283</td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
Table 1. Document revision history (continued)

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Significant changes</th>
</tr>
</thead>
</table>
|          |          | • SATA A-010240  
|          |          | • USB A-010127, A-010129, A-010131, and A-010151 |
|          |          | Modified the following errata:  
|          |          | • DDR A-009942  
|          |          | • eSDHC A-009620  
|          |          | • PCIe A-008822  
| 2        | 03/2016  | Added the following errata:  
|          |          | • DDR A-009942  
|          |          | • I2C A-010124  
|          |          | • IFC A-009241  
|          |          | Modified the following errata:  
|          |          | • PCIe A-008822  
|          |          | • USB A-009116  
|          |          | Removed the following errata:  
|          |          | • SPI A-009774  
| 1        | 01/2016  | Added the following errata:  
|          |          | • PCIe A-009518  
|          |          | • SPI A-009895  
|          |          | • USB 3.0 A-009008  
|          |          | Modified the following errata:  
|          |          | • DDR A-008514 and A-009663  
|          |          | • eSDHC A-009620  
|          |          | • QSPI A-009277  
|          |          | • SATA A-009185  
|          |          | • USB 3.0 A-008997 and A-009007  
|          |          | Re-ordered the USB 3.0 section to be in numerical order.  
| 0        | 10/2015  | As of this revision, the silicon has qualified. This document remains NDA.  

The following table provides a cross-reference to match the revision code in the processor version register to the revision level marked on the device.

Table 2. Revision level to part marking cross-reference

<table>
<thead>
<tr>
<th>Part</th>
<th>Arm® Cortex®-A7 MPCore processor revision</th>
<th>Arm core main ID register</th>
<th>System version register value (Rev 2.0 silicon)</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1020AE</td>
<td>r0p5</td>
<td>410FC075h</td>
<td>8708_1020h</td>
<td>With security</td>
</tr>
<tr>
<td>LS1020A</td>
<td>r0p5</td>
<td>410FC075h</td>
<td>8700_1020h</td>
<td>Without security</td>
</tr>
<tr>
<td>LS1021AE</td>
<td>r0p5</td>
<td>410FC075h</td>
<td>8708_1120h</td>
<td>With security</td>
</tr>
<tr>
<td>LS1021A</td>
<td>r0p5</td>
<td>410FC075h</td>
<td>8700_1120h</td>
<td>Without security</td>
</tr>
<tr>
<td>LS1022AE</td>
<td>r0p5</td>
<td>410FC075h</td>
<td>8708_1220h</td>
<td>With security</td>
</tr>
<tr>
<td>LS1022A</td>
<td>r0p5</td>
<td>410FC075h</td>
<td>8700_1220h</td>
<td>Without security</td>
</tr>
</tbody>
</table>
Table 3 summarizes all known errata and lists the corresponding silicon revision level to which they apply. A ‘Yes’ entry indicates the erratum applies to a particular revision level, and an ‘No’ entry means it does not apply.

### Table 3. Summary of Silicon Errata and Applicable Revision

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.0</td>
</tr>
<tr>
<td><strong>2D_ACE</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008849</td>
<td>Gamma registers need to be byte-swapped</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Arm_Cortex-A7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007702</td>
<td>Arm® Cortex®-A7 MPCore Errata 814220: Cache maintenance by set/way operations can execute out of order</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007703</td>
<td>Arm® Cortex®-A7 MPCore Errata 809719: PMU events 07h, 0Ch, and 0Eh do not increment correctly</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007704</td>
<td>Arm® Cortex®-A7 MPCore Errata 805420: PMU event counter 14h does not increment correctly</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007705</td>
<td>Arm® Cortex®-A7 MPCore Errata 804069: Exception mask bits are cleared when an exception is taken in Hyp mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007856</td>
<td>Arm® Cortex®-A7 MPCore Errata 823274: Load or store that fails condition code check might cause deadlock or data corruption</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009159</td>
<td>Arm® Cortex®-A7 MPCore Errata 844169: Memory locations might be accessed speculatively due to instruction fetches when HCR.VM is set</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010476</td>
<td>Arm Cortex-A7 MPCore Errata 856125: Stage 2 XN attribute is suppressed when stage 1 MMU is disabled</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>DCFG</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-006748</td>
<td>DCFG_CCSR_CRSTSRn momentarily reports core ready status while the threads are still coming out of boot hold-off state</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>DDR</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007864</td>
<td>DDR controller may fail to operate DDR4-1600 for bus widths for 32-bit or 16-bit or less</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008514</td>
<td>Memory controller requires a register write before being enabled</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008850</td>
<td>DDR controller should be configured before barrier transactions are issued</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008964</td>
<td>DDR PLL does not lock or is out of range for hard-coded RCW</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009002</td>
<td>RCW field definitions 186 and 187 in the reference manual are incorrect</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009663</td>
<td>DDR controller violates JEDEC specification during DRAM VRef training</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009942</td>
<td>DDR controller can train to non-optimal setting</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>DEBUG</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007980</td>
<td>RCW reconfiguration over JTAG cannot work when PBL is enabled</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>DMA</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008370</td>
<td>Scatter/gather table extension before a 4KB page crossing may cause QDMA to fetch incorrect descriptor address</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table continues on the next page...
Table 3. Summary of Silicon Errata and Applicable Revision (continued)

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>DUART</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-004737</td>
<td>BREAK detection triggered multiple times for a single break assertion</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008006</td>
<td>DUART may miss characters during transmission in FIFO mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>eDMA</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008797</td>
<td>eDMA ICID is bit flipped</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-011218</td>
<td>eDMA may not work with SPI</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>eSDHC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008171</td>
<td>eSDHC transactions may fail in tuning modes of operation when the</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>input data window is close to 1 UI during tuning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008703</td>
<td>SDHC DDR50 AC timing specification for (tshdivkh) setup time and</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>output delay are not met.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008705</td>
<td>SDHC eMMC DDR mode AC timing specification for setup time</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>(tshdivkh) is not met.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008708</td>
<td>Issue with input setup time tshivkh for eSDHC high speed mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009620</td>
<td>Data timeout error not getting set in case of command with busy</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>response (R1b) as well as for busy period after last write block</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>transfer</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>eTSEC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-006264</td>
<td>eTSEC may drop bytes in FIFO mode if frame matches a Control Frame</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>ethertype</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-006293</td>
<td>Mixing TOE = 0 and TOE = 1 frames may cause data corruption</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-006502</td>
<td>Incomplete GRS or invalid parser state after receiving a 1- or 2-byte</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>frame</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Ethernet</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007207</td>
<td>TBI link status bit may stay up after SGMII electrical idle is detected</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>eTSEC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008699</td>
<td>RGMII AC timing specifications for tSKRGT_TX max are not met</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008700</td>
<td>RGMII AC timing specifications for tSKRGT_TX min is not met</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>FlexCAN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-009729</td>
<td>Corrupted frame possible if Freeze Mode or Low Power Mode are entered</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>during a bus-off state</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-009732</td>
<td>The transmission abort mechanism may not work properly</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>FlexTimer</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007728</td>
<td>The process of clearing the FTMx_SC[TOF] bit does not work as</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>expected under a certain condition when the FTM counter reaches</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FTM_MOD value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-009659</td>
<td>Incorrect match may be generated if intermediate load feature is</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>used in toggle mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-011026</td>
<td>Safe state is not removed from the channel output after fault</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>condition ends if SWOCtrl register is used to control the pin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
Table 3. Summary of Silicon Errata and Applicable Revision (continued)

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>GEN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008213</td>
<td>LVCMOS pins VIH and VIL level do not meet JEDEC specification standard values for LVCMOS as specified on JESD8C.01</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010539</td>
<td>Some eSDHC and GPIO pins are not functional if the source of the RCW is selected as QSPI</td>
<td>No plans to fix.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>GIC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010111</td>
<td>Non-core master can only access Generic Interrupt Controller (GIC) banked registers of Core 0</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>I2C</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-009286</td>
<td>Missing glitch filter implementation</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010124</td>
<td>Attempting a start cycle while the bus is busy may generate a short clock pulse</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010650</td>
<td>I2C controller is unable to generate clocks when SDA is low</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>IEEE1588</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007734</td>
<td>Stale time stamps and overflow conditions can occur when using an external 1588 input clock at certain frequencies</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>IFC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008172</td>
<td>Issue with write protect toggling in NAND NVDDR mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008175</td>
<td>NV-DDR program operation with ECC encoder enabled does not work with DDR clock division ratio 2</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009241</td>
<td>Unaligned write transactions to IFC may result in corruption of data</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009652</td>
<td>t_{DOSQ} and t_{DVW} does not meet NAND industrial standard ONFI AC spec for 3.3 V NV-DDR mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>LPUART</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010581</td>
<td>LPUART may send two break characters during idle mode, where one is expected</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010727</td>
<td>LPUART_RX Pin Active edge flag cannot be set as expected</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td><strong>PCle</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007189</td>
<td>PCI Express inbound error message handled incorrectly in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007815</td>
<td>The read-only-write-enable bit must be cleared to prevent overwriting read-only registers</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007997</td>
<td>PCI Express hot-plug-related bits in the Slot Capabilities Register need to be cleared</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008236</td>
<td>PCI Express controller does not exit disabled state if the Link Control register [Link Disable] bit is cleared before lanes enter electrical idle in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008329</td>
<td>PCI Express controller forwards received message TLPs to system application address space by default in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008432</td>
<td>Optional programmable PCI Express iATU CFG shift feature is not supported for outbound configuration transactions in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008822</td>
<td>Change the default AXI system error response behavior for PCI Express outbound non-posted requests</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
# Table 3. Summary of Silicon Errata and Applicable Revision (continued)

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-008913</td>
<td>New A11 message request handling rules not supported</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009000</td>
<td>Accessing the PCI Express controller's reserved CCSR space offset FFF0h to FFFFh is prohibited</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009142</td>
<td>PCIe EP mode is not supported</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009151</td>
<td>PCI Express controller does not correctly set the Link Autonomous Bandwidth Status and Link Bandwidth Management Status bits of the Link Status Register in RC mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009518</td>
<td>PCI Express EP controller's non-D0 PowerState is overridden to D0 when any single or combination of the Bus Master Enable (BME) or Memory Space Enable (MSE) bits are enabled from disabled setting by host software</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009520</td>
<td>PCI Express completion transaction layer packets (TLP) of a decomposed outbound memory read request must be returned in order</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009700</td>
<td>Secondary PCI Express extended capability is included in the capability structure linked list in Gen1 and Gen2 modes</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009719</td>
<td>Excessive correctable errors observed when the link's Active State Power Management (ASPM) is enabled</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010315</td>
<td>Read access to an unselected PCI Express controller’s space causes core(s) and platform to hang</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>PM</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008646</td>
<td>Reading configuration register RCPM_IPPDEYPCR1 return zero</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>qDMA</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-007769</td>
<td>DDE error may be flagged incorrectly when qDMA address hold is used</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007771</td>
<td>Mixing command queue mode and legacy direct mode jobs may cause hang if an error is detected</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007773</td>
<td>qDMA does not function correctly for certain combinations of source stride size and source stride distance</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007774</td>
<td>RTE error may be flagged on incorrect qDMA job</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010812</td>
<td>The qDMA controller fails to capture the command descriptor in DECCDR[0-3] registers when the enqueue rejection error bit DEDR[ERE] is set to 1</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010840</td>
<td>Descriptors which are required to be enqueued in the status queue may be duplicated</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>QE</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008892</td>
<td>QUICC Engine's data and buffer descriptor cannot be stored in MURAM</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>QSPI</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-008886</td>
<td>Sometimes unexpected data is written to the external flash memory even though the underrun bit (QuadSPI_FR[TBUF]) is not set</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009277</td>
<td>Parallel mode cannot be used on full memory map of dual die packages</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>Errata</td>
<td>Name</td>
<td>Projected Solution</td>
<td>Silicon Rev.</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>--------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>A-009283</td>
<td>Illegal accesses to SPI flash memory can result in a system hang</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009284</td>
<td>Incorrect data is returned in multiple flash use case when the read</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>crosses the flash boundary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010284</td>
<td>Insufficient read data may be received in the Rx Data Buffer register</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**SATA**

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-008402</td>
<td>Maximum PRD length (4MB) issue</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008407</td>
<td>SATA transaction causes incorrect CRC error</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008588</td>
<td>SATA controller may hang when processing multiple 16-byte non-</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>aligned PRD entries for read operations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-009042</td>
<td>The device detection initialization sequence mistakenly resets some</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>registers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-009185</td>
<td>The default Rx watermark value may be insufficient for some hard</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>drives</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-010240</td>
<td>SATA interface in BIST-L mode fails</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**SerDes**

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-009102</td>
<td>Default SerDes setting may cause excess errors in SATA controller</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**SEC**

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-006385</td>
<td>SEC watchdog timer does not prevent all cases of illogical descriptors from hanging DECOs</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008857</td>
<td>Anti-replay early-rollover error</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010442</td>
<td>SEC WPA2 (WiFi) protocol produces incorrect CCM nonce for QoS Data frames</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**SPI**

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-007731</td>
<td>Mixing 16-bit and 32-bit frame sizes in XSPI mode can cause incorrect data to be transmitted</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009776</td>
<td>Loading of shift register data into the receive FIFO following an overflow event</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009895</td>
<td>Inconsistent addition to Transmit or Command FIFO if an entry is added while FIFOs are full</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**TMU**

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-008360</td>
<td>Reported temperature may indicate +4°C higher at range start</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**USB**

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-005697</td>
<td>Suspend bit asserted before the port is in Suspend state</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008118</td>
<td>USB initial sequence impact for external ULPI PHY</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007463</td>
<td>After transaction error, controller switches control transfer data stage from IN to OUT direction</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007467</td>
<td>When the LSP cache size is less than the number of TRBs needed for a microframe's worth of transfers, for an isochronous IN endpoint, the controller reads from invalid address</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007468</td>
<td>When the host extends an IN burst transaction, the device responds with both DP and NRDY on underrun</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
### Table 3. Summary of Silicon Errata and Applicable Revision (continued)

<table>
<thead>
<tr>
<th>Errata</th>
<th>Name</th>
<th>Projected Solution</th>
<th>Silicon Rev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-007469</td>
<td>Short packet with EOB = 1 can cause delay on transfer event until ERDY is sent by device</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007516</td>
<td>Device responds to ACK TP with bad device address</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007517</td>
<td>Debug target may request U1/U2 entry after light reset before SetFeature(U1_ENABLE/U2_ENABLE)</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007694</td>
<td>When software initiates a hardware LPM, do not use PORTHLPMLC[HIDM]</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007695</td>
<td>Device violates minimum USB turnaround time in FS mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007696</td>
<td>Controller generates transfer event with RING_UNDERRUN or RING_OVERRUN completion code, but ED = 1 and Data = 0h</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007697</td>
<td>xHCI debug: port status changes event not generated when USBCMD.Run/Stop = 0</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-007839</td>
<td>Host core may hang if xHCI driver wrongly sets DCS bit to 0 in the Endpoint Context for Address device command while host's parameter check is disabled</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008260</td>
<td>Device controller accepts new OUT packets even when TRBs are not available</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008428</td>
<td>Device initiates low power when ACK pending for data packet</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008459</td>
<td>Device in 2.0 mode handles DATA PID error on SETUP data incorrectly</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-008997</td>
<td>USB3 LFPS peak-to-peak differential output voltage adjustment settings</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009007</td>
<td>USB3PHY observing intermittent failure in receive compliance tests at higher jitter frequency using default register values</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009008</td>
<td>USB High Speed (HS) eye height adjustment</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009116</td>
<td>Frame length of USB3 controller for USB 2.0 and USB 3.0 operation is incorrect</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009149</td>
<td>USB 2.0 protocol data pins USB_D_M and USB_D_P do not support IEEE 1149.1</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009163</td>
<td>Device prohibits low power after sending LPF = 1 for isochronous IN</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009377</td>
<td>Incorrect value for S and E fields for ISOC OUT in Start-Split token</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009668</td>
<td>Stop Endpoint command does not complete</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-009798</td>
<td>USB high speed squelch threshold adjustment</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010127</td>
<td>Hot reset failure during U1/U2 entry for USB 3.0</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010129</td>
<td>USB 2.0 reset not driven while port is in the Resume state</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010131</td>
<td>U3 request gets dropped when controller tries U1-to-U2 entry</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
<tr>
<td>A-010151</td>
<td>Unreliable receiver detection in low power P3 mode</td>
<td>No plans to fix</td>
<td>Yes</td>
</tr>
</tbody>
</table>
A-008849: **Gamma registers need to be byte-swapped**

**Affects:** 2D-ACE

**Description:** Gamma_R, Gamma_G and Gamma_B registers are little-endian registers while the rest of the address-space in 2D-ACE is big-endian. 2D-ACE Gamma_R, Gamma_G and Gamma_B registers are 32 bit registers, where the first 24 bits are reserved and last 8 bits denote the gamma value. Because of a connection issue in the device, the first 8-bit [31:24] is connected and the rest of the 24-bits[23:0] are reserved.

**Impact:** Gamma registers are not configurable.

**Workaround:** Perform the byte_swapping for Gamma_[R/G/B]_registers.

For example: While writing 0000_00ABh to any of the gamma registers, byte swap the data so it results in AB00_0000h. Write this value to the gamma register.

**Fix plan:** No plans to fix
A-007702: Arm® Cortex®-A7 MPCore Errata 814220: Cache maintenance by set/way operations can execute out of order

**Affects:** Cortex-A7

**Description:** The Armv7-A architecture states that all cache and branch predictor maintenance operations that do not specify an address execute, relative to each other, in program order. However, because of this erratum, an L2 set/way cache maintenance operation can overtake an L1 set/way cache maintenance operation.

**Impact:** Code that intends to clean dirty data from L1 to L2 and then from L2 to L3 using set/way operations might not behave as expected. The L2 to L3 operation might happen first and result in dirty data remaining in L2 after the L1 to L2 operation has completed.

If dirty data remains in L2, then an external agent, such as a DMA agent, might observe stale data.

If the processor is reset or powered down while dirty data remains in L2, then the dirty data is lost.

**Workaround:** Force the correct ordering between set/way cache maintenance operations by executing a DSB before changing cache levels.

**Fix plan:** No plans to fix
A-007703: Arm® Cortex®-A7 MPCore Errata 809719: PMU events 07h, 0Ch, and 0Eh do not increment correctly

Affects: Cortex-A7
Description: The Cortex-A7 MPCore processor implements version 2 of the Performance Monitor Unit architecture (PMUv2). The PMU can gather statistics on the operation of the processor and memory system during runtime. This event information can be used when debugging or profiling code.

The PMU can be programmed to count architecturally executed stores (event 07h), software changes of the PC (event 0Ch), and procedure returns (event 0Eh). However, because of this erratum, these events do not fully adhere to the descriptions in the PMUv2 architecture.

Impact: The information returned by PMU counters that are programmed to count events 07h, 0Ch, or 0Eh might be misleading when debugging or profiling code executed on the processor.

Workaround: None
Fix plan: No plans to fix
A-007704: Arm® Cortex®-A7 MPCore Errata 805420: PMU event counter 14h does not increment correctly

Affects: Cortex-A7
Description: The Cortex-A7 MPCore processor implements version 2 of the Performance Monitor Unit architecture (PMUv2). The PMU can gather statistics on the operation of the processor and memory system during runtime. This event information can be used when debugging or profiling code. When a PMU counter is programmed to count L1 instruction cache accesses (event 14h), the counter should increment on all L1 instruction cache accesses. However, because of this erratum, the counter increments on cache hits but not on cache misses.

Impact: A PMU counter that is programmed to count L1 instruction cache accesses counts instruction cache hits but not instruction cache misses. Therefore, the information returned can be misleading when debugging or profiling code executed on the processor.

Cache-bound code execution is not affected by this erratum because of the absence of cache misses.

Workaround: To obtain a better approximation for the number of L1 instruction cache accesses, enable a second PMU counter and program it to count instruction fetches that cause linefills (event 01h). Add the value returned by this counter to the value returned by the L1 instruction access counter (event 14h). The result of the addition is a better indication of the number of L1 instruction cache accesses.

Fix plan: No plans to fix
A-007705: Arm® Cortex®-A7 MPCore Errata 804069: Exception mask bits are cleared when an exception is taken in Hyp mode

Affects: Cortex-A7
Description: The Cortex-A7 MPCore processor implements the Arm Virtualization Extensions and the Arm Security Extensions. Exceptions can be routed to Monitor mode by writing SCR.{EA, FIQ, IRQ} = 1. Exceptions can be masked by writing the corresponding bit CPSR.{A, I, F} to 1.

The Armv7-A architecture states that an exception taken in Hyp mode does not change the value of the mask bits for exceptions routed to Monitor mode. However, because of this erratum, the corresponding mask bits will be cleared to 0.

Impact: If SCR.{AW, FW} = 0, then writing the corresponding bit CPSR.{A,F} to 0 has no effect. The value of CPSR.{A, F} is ignored. Otherwise, when CPSR.{A, F, I} = 1, secure code cannot rely on CPSR.{A, F, I} remaining equal to 1. An exception that should be masked might be routed to Monitor mode.

This is Category C as it is expected that users will:
1. write SCR.{AW, FW} = 0 when SCR.{EA, FIQ} = 1.
2. write SCR.IRQ = 0.

Workaround: None

Fix plan: No plans to fix
A-007856: Arm® Cortex®-A7 MPCore Errata 823274: Load or store that fails condition code check might cause deadlock or data corruption

**Affects:** Cortex-A7

**Description:** Under rare circumstances, a conditional load instruction that fails its condition code check might cause deadlock or data corruption.

**Impact:** It is believed that the various conditions required to cause this erratum are very unlikely to occur together in practice. This erratum has never been observed in deployed products. As such, the practical implications for real systems should be minimal.

**Workaround:** There is no workaround for this erratum.

**Fix plan:** No plans to fix
A-009159: Arm® Cortex®-A7 MPCore Errata 844169: Memory locations might be accessed speculatively due to instruction fetches when HCR.VM is set

Affects: Cortex-A7

Description: The Armv7 architecture requires that when all associated stages of translation are disabled for the current privilege level, memory locations are only accessed due to instruction fetches within the same or next 4KB region as an instruction which has been or will be fetched due to sequential execution. In the conditions detailed below, the Cortex-A7 MPCore processor might access other locations speculatively due to instruction fetches.

Errata conditions:
1. The processor must be executing at PL2 or Secure PL1.
2. Address translation is disabled for the current exception level (by clearing the appropriate SCTLR.M or HSCTLR.M bit).
3. The HCR.VM bit is set.

Impact: If these conditions are met, then speculative instruction fetches might be made to memory locations not permitted by the architecture.

Workaround: Because the HCR.VM bit is reset low, this situation is most likely to arise in powerdown code, if PL2 or Secure PL1 software disables address translation before the core is powered down. To work around this erratum, software should ensure that HCR.VM is cleared before disabling address translation at PL2 or Secure PL1.

Fix plan: No plans to fix
A-010476: Arm Cortex-A7 MPCore Errata 856125: Stage 2 XN attribute is suppressed when stage 1 MMU is disabled

**Affects:** Arm Cortex-A7

**Description:** The Cortex-A7 MPCore processor implements the Arm virtualization extensions. The virtualization extensions provide independent translation regimes for memory accesses from different modes. In the non-secure PL1 and PL0 translation regimes, address translation occurs in two stages. Stage 1 maps the virtual address (VA) to an intermediate physical address (IPA). Stage 2 maps the IPA to the physical address (PA).

The Armv7-A Architecture states that if the stage 2 eXecute-never (XN) attribute is set to 1, execution from the region is not permitted, regardless of the value of the XN attribute in the stage 1 translation. If a permission fault is generated because the stage 2 XN bit is set to 1, this is reported as a stage 2 MMU fault.

Because of this erratum, the stage 2 XN attribute is suppressed when the stage 1 MMU is disabled.

**Impact:** There are two main implications of this erratum:

1. For code running at non-secure PL1 in a CPU for which the stage 1 MMU disabled, the stage 2 XN attribute does not give any protection from speculative instruction fetches occurring at read-sensitive locations. These locations might rarely be corrupted because of this erratum. This might occur during the PL1 initialization of a CPU under virtualization.
2. For code running at PL0 with HCR.TGE == 1, which permits running an application directly on a hypervisor, the stage 2 XN permission cannot be used to protect areas of memory from being executed. This means that enforcing a security policy is not possible for the writeable areas from being executed at EL0 with HCR.TGE == 1.

**Workaround:** The first implication can be worked around by ensuring that no read-sensitive locations are mapped into the stage 2 page tables or having read access in the stage 2 page tables when running at non-secure EL1 with the MMU disabled.

The second implication cannot be worked around when HCR.TGE == 1, but it does not lead to functional misbehavior of correctly constructed code.

**Fix plan:** No plans to fix
A-006748:  **DCFG_CCSR_CRSTSRn momentarily reports core ready status while the threads are still coming out of boot hold-off state**

**Affects:** DCFG

**Description:** When the cores exit the boot hold-off state at the end of the reset sequence, the Core Reset Status Register n (DCFG_CCSR_CRSTSRn) momentarily reports an incorrect core ready status while the threads are still halted and are about to exit the halted state.

**Impact:** The DCFG_CCSR_CRSTSRn reports an incorrect core ready status from the time the core HRESET is negated to the time the core negates the halted signal.

**Workaround:** Software should only read DCFG_CCSR_CRSTSRn[READY] field 256 core clock cycles after the bit for the corresponding core is set in DCFG_BRRL to release the core from boot hold off.

**Fix plan:** No plans to fix
A-007864: DDR controller may fail to operate DDR4-1600 for bus widths for 32-bit or 16-bit or less

**Affects:** DDR

**Description:** For Rev 1 silicon: If a DRAM data bus width of 32 bits or less is used with DDR4 memories running at 1600 MT/s data rate, then tCCD_L is violated by the controller. The JEDEC requires this to be 5 cycles for DDR4 running at 1600 MT/s data rate, but the controller only provides 4 cycles. DDR4 running at 1333 MT/s data rate only requires 4 cycles and is not affected.

For Rev 2 silicon: If a DRAM data bus width of 16 bits or less is used with DDR4 memories running at 1600 MT/s data rate, then tCCD_L is violated by the controller. The JEDEC requires this to be 5 cycles for DDR4 running at 1600 MT/s data rate, but the controller only provides 4 cycles. DDR4 running at 1333 MT/s data rate only requires 4 cycles and is not affected.

**Impact:** For Rev 1 silicon: DDR interface may fail to operate when DDR4 memories are configured for 32-bit or less data bus and run at 1600 MT/s data rate.

For Rev 2 silicon: DDR interface may fail to operate when DDR4 memories are configured for 16-bit or less data bus and run at 1600 MT/s data rate.

**Workaround:** For Rev 1 silicon: To ensure that tCCD_L is met by the DDR controller, do not operate DDR4 faster than 1333 MT/s data rate when the data bus is configured to 32-bit or less bus width. If DDR4 running at 1600 MT/s data rate is required for a 32-bit or less data bus interface, then check with the specific DRAM vendors to determine if this specification violation leads to DRAM data corruption.

For Rev 2 silicon: To ensure that tCCD_L is met by the DDR controller, do not operate DDR4 faster than 1333 MT/s data rate when the data bus is configured to 16-bit or less bus width. If DDR4 running at 1600 MT/s data rate is required for a 16-bit or less data bus interface, then check with the specific DRAM vendors to determine if this specification violation leads to DRAM data corruption.

**Fix plan:** No plans to fix
A-008514: Memory controller requires a register write before being enabled

Affects: DDR
Description: Memory controller performance is not optimal with default internal target queue register values.
Impact: Memory controller performance is not optimal.
Workaround: Write a value of 63b2_0042h to address 157_020Ch.
Fix plan: No plans to fix
A-008850: DDR controller should be configured before barrier transactions are issued

Affects: DDR
Description: Barrier transactions from CCI400 need to be disabled till the DDR is configured. CCI400 does the broadcasting of barrier transactions on the DDR controller. However, because the DDR controller is not programmed, it does not return the response for barrier transaction and this situation leads to system hang. The DDR controller needs to be configured before software can do any data synchronization barrier transactions.

Impact: System hangs if barrier transactions are received before the DDR controller initialization.

Workaround: Before issuing any data barrier instruction and while the DDR controller is not configured:
1. Set the bit 3 of the CCI400 Control Override Register (for address: 0118_0000h with data 0000_0008h). This disables the propagation of barrier transactions to DDRC from CCI400.

After DDR controller is configured:
3. Clear the bit 3 for CCI400 Control Override Register (for 0118_0000h with data 0000_0000h).
4. Enable the re-ordering in DDR controller using DDR Enhanced Optimization Register (DDR_EOR: for address=0108_0c00, DDR_EOR[RD_REOD_DIS 5:7]=3'b000, DDR_EOR[WR_REOD_DIS 11]=1'b

Fix plan: No plans to fix
A-008964: DDR PLL does not lock or is out of range for hard-coded RCW

Affects: DDR  
Description: Hard-coded RCW DDR PLL ratios are such that PLL may lose lock and behavior is unpredictable. Hard-coded configurations are out of specification.

Impact: Hard-coded RCW cannot be used when DDR memory interface is active.

Workaround: Soft RCW should be used to correctly configure the DDR PLL.

Fix plan: No plans to fix
A-009002: RCW field definitions 186 and 187 in the reference manual are incorrect

**Affects:** DDR

**Description:** RCW field definition register width [0-511] must be corrected on bits [186-187].

This table provides the original descriptions of bits [186-187] in the reference manual.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The DDRCLK pin provides the reference clock to the DDR PLL.</td>
</tr>
<tr>
<td>01</td>
<td>DIFF_SYSCLK/DIFF_SYSCLK_B provides the reference clock to the DDR PLL.</td>
</tr>
<tr>
<td>10</td>
<td>Reserved</td>
</tr>
<tr>
<td>11</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

This table provides the correct descriptions of bits [186-187].

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The DDRCLK pin provides the reference clock to the DDR PLL.</td>
</tr>
<tr>
<td>01</td>
<td>Reserved</td>
</tr>
<tr>
<td>10</td>
<td>DIFF_SYSCLK/DIFF_SYSCLK_B provides the reference clock to the DDR PLL.</td>
</tr>
<tr>
<td>11</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

**NOTE**
Fields 00 and 11 remained unchanged, while fields 01 and 10 are swapped in hardware.

**Impact:**
The input reference clock to PLL is unknown when programmed to 01. This may result in an unknown, high frequency output from DDR PLL.

**Workaround:** Software must take care of swapped bits RCW[186] and RCW[187]. Field 186 should be treated as 187, and vice versa.

**Fix plan:** No plans to fix
A-009663: DDR controller violates JEDEC specification during DRAM VRef training

Affects: DDR

Description: The JEDEC specification requires all pages in DRAM to be closed during an MRS command. The DDR controller violates this specification during DRAM VRef training which only applies to DDR4 DRAM.

Impact: Although no failures have been observed, this is a specification violation that could potentially lead to data corruption or calibration failures.

Workaround: When a memory controller is planned and configured to operate in auto precharge mode, then no further action is required. Otherwise, program DDR_INTERVAL[BSTOPRE] to 0h or 4h before setting DDR_SDRAM_CFG[MEM_EN]. After DDR initialization has completed (that is, after DDR_SDRAM_CFG_2[D_INIT] has cleared if D_INIT was set), program DDR_INTERVAL[BSTOPRE] to the desired value. This programming after DDR initialization can be run without halting the DDR controller.

Fix plan: No plans to fix
A-009942: DDR controller can train to non-optimal setting

**Affects:** DDR

**Description:** During the receive data training, the DDRC may complete on a non-optimal setting.

**Impact:** Memory controller fails to complete initialization. In the very unlikely event that the memory controller completes initialization with non-optimized training results, most of the subsequent read transactions fail.

**Workaround:** Before setting MEM_EN, ensure the following:

<table>
<thead>
<tr>
<th>If operating at...</th>
<th>Then logical OR the DEBUG_29 with a value of...</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR-1333</td>
<td>0080006ah</td>
</tr>
<tr>
<td>DDR-1600</td>
<td>0070006fh</td>
</tr>
</tbody>
</table>

First read the DEBUG_29 register and then perform the following steps:
1. Logical AND the read value with 0xFF0FFF00.
2. Logical OR the read value with the value provide in the table above.
3. Write the result back to DEBUG_29.

**NOTE**
After implementing the existing workaround listed above, read the DEBUG_10, 11, 12, 13 and 14 from a working unit. Each debug register holds the CPO value for each byte lane as shown below (big-endian convention is used for all the following debug registers):
- DEBUG_10 [0:7] and DEBUG_10 [16:23] hold the CPO value for byte lane 0 and 1, respectively.
- DEBUG_11 [0:7] and DEBUG_11 [16:23] hold the CPO value for byte lane 2 and 3, respectively.
- DEBUG_12 [0:7] and DEBUG_12 [16:23] hold the CPO value for byte lane 4 and 5, respectively.
- DEBUG_13 [0:7] and DEBUG_13 [16:23] hold the CPO value for byte lane 6 and 7, respectively.
- DEBUG_14 [0:7] hold the CPO value for ECC byte.

When ECC byte lane is not used, ignore the ECC CPO information. Similarly, ignore the CPO information for any unused byte lanes.

Determine the maximum and minimum CPO values among the CPO values that are read. Then check the condition in steps below to determine if the CPO value in the existing workaround for A009942 needs to be updated. The maximum and minimum CPO values will be used in the following steps:
1. In the case of DDR4, if the minimum CPO value read + 0x3B < the DEBUG_29 [24:31] listed in the table above, then go to step 3; otherwise CPO update in step 3 is not required.
2. In the case of DD3/3L, if the minimum CPO value read + 0x3F < the DEBUG_29 [24:31] listed in the table above, then go to step 3; otherwise CPO update in step 3 is not required.
3. Update DEBUG_29 [24:31] = (Maximum CPO value read + Minimum CPO value read)/2 + 0x27.

LS1021A, LS1020A, and LS1022A Chip Errata, Rev. 9, 04/2018
For this erratum, DEBUG_29 is at DDR_OFFSET + f70h. DEBUG_10, 11, 12, 13, and 14 are at DDR_OFFSET +0xF24, 0xF28, 0xF2C, 0xF30, and 0xF34 respectively, these are read-only registers. All the debug registers are 32 bit.

**Fix plan:** No plans to fix
A-007980: RCW reconfiguration over JTAG cannot work when PBL is enabled

**Affects:** DEBUG

**Description:** To overwrite RCW configuration via JTAG/TPR debugger, the debugger should stop the POR state machine. This causes the SM to enter a reset pause state (0x0B), which is a debug state for the SoC.

If PBL is enabled while the SM is in the reset pause state (0x0B), the PBL timeout counter is also enabled and increments every clock cycle, leading to a timeout counter expire 0x76 error.

In non-hardcoded RCW_SRC mode with PBL enabled, RESET_REQ is generated and PBL timeout counter expires (error code 0x76). In this case the POR_SM is stuck in a 0xE state after exiting from the reset pause state (0x0B).

**Impact:** Reconfiguration of RCW via JTAG/TPR debugger can't work in non-hard coded RCW when PBL is enabled.

**Workaround:** Reconfiguration of RCW via JTAG/TPR is functional only for RCW hard code mode or when PBL is disabled.

**Fix plan:** No plans to fix
A-008370: Scatter/gather table extension before a 4KB page crossing may cause QDMA to fetch incorrect descriptor address

Affects: DMA
Description: The QDMA read engine prefetches up to 4 scatter/gather table entries, each of which is 16 bytes in size, part of one memory request. If the 64 bytes memory read access crosses a 4K page boundary and the extension is used in the four entries, QDMA might fetch incorrect descriptor address.

Impact: If the scenario is encountered, data corruption is likely or an error may be flagged indicating the scatter/gather table is too short.

Workaround: This errata is only applicable if scatter/gather format are used for the source read. This is not applicable to write. One of the following can be used as a workaround:
  • Avoid scatter/gather extension when generating the QDMA source descriptors.
  • Assuming scatter/gather extension cannot be avoided, start of the scatter/gather table should be 64 byte aligned.

Fix plan: No plans to fix
A-004737: BREAK detection triggered multiple times for a single break assertion

**Affects:** DUART

**Description:** A UART break signal is defined as a logic zero being present on the UART data pin for a time longer than (START bit + Data bits + Parity bit + Stop bits). The break signal persists until the data signal rises to a logic one.

A received break is detected by reading the ULSR and checking for BI = 1. This read to ULSR clears the BI bit. After the break is detected, the normal handling of the break condition is to read the URBR to clear the ULSR[DR] bit. The expected behavior is that the ULSR[BI] and ULSR[DR] bits do not get set again for the duration of the break signal assertion. However, the ULSR[BI] and ULSR[DR] bits continue to get set each character period after they are cleared. This continues for the entire duration of the break signal.

At the end of the break signal, a random character may be falsely detected and received in the URBR, with the ULSR[DR] being set.

**Impact:** The ULSR[BI] and ULSR[DR] bits get set multiple times, approximately once every character period, for a single break signal. A random character may be mistakenly received at the end of the break.

**Workaround:** The break is first detected when ULSR is read and ULSR[BI]=1. To prevent the problem from occurring, perform the following sequence when a break is detected:

1. Read URBR, which returns a value of zero, and clears the ULSR[DR] bit
2. Delay at least 1 character period
3. Read URBR again, which return a value of zero, and clears the ULSR[DR] bit

ULSR[BI] remains asserted for the duration of the break. The UART block does not trigger any additional interrupts for the duration of the break.

This workaround requires that the break signal be at least 2 character-lengths in duration.

This workaround applies to both polling and interrupt-driven implementations.

**Fix plan:** No plans to fix
A-008006: DUART may miss characters during transmission in FIFO mode

Affects: DUART

Description: If configured in 16-byte FIFO mode, the DUART controller transmits only 15 characters. Similarly, it transmits 63 characters in 64-byte FIFO mode.

Impact: The DUART may lose the 16th character in 16-byte FIFO mode and the 64th character in 64-byte FIFO mode.

Workaround: In 16-byte FIFO mode, software should write 15 bytes in the FIFO.
In 64-byte FIFO mode, software should write 63 bytes in the FIFO.

Fix plan: No plans to fix
A-008797: eDMA ICID is bit flipped

Affects: eDMA

Description: The eDMA Stream ID Register in SCFG Register Space(SCFG_eDMA_ICID--157_0128h) is mentioned as 0:7 field with bit 0 as MSB and bit 7 as LSB but the ICID input to SMMU is connected in reverse manner.

- 7th Bit is connected to 0th Bit
- 6th Bit is connected to 1st Bit
- 5th Bit is connected to 2nd Bit
- 4th Bit is connected to 3rd Bit
- 3rd Bit is connected to 4th Bit
- 2nd Bit is connected to 5th Bit
- 1st Bit is connected to 6th Bit
- 0th Bit is connected to 7th Bit

Hence the Stream ID matching programming in SMMU needs to be different than SCFG_eDMA_ICID programming for eDMA. For example, if the software programs 1fh in SCFG_eDMA_ICID Register, the actual stream ID for eDMA is f8h due to connection mismatch.

Impact: Incorrect eDMA ICID is propagated to SMMU.

Workaround: To program the SCFG_eDMA_ICID Register, write the desired Stream ID in a bit-flipped manner. For example, if the software wants to allocate Stream ID value of 23h to eDMA ICID it should program the SCFG_eDMA_ICID field with C4h.

Fix plan: No plans to fix
A-011218: eDMA may not work with SPI

Affects: eDMA

Description: eDMA can be used to push/populate data to SPI Tx/Rx FIFO. Due to this erratum, the eDMA’s interaction with SPI may fail with the following signatures:

- While transmitting, the SPI may fail to indicate that the Tx FIFO is full. Hence, the eDMA may keep pushing data to Tx FIFO resulting in a corruption in the Tx FIFO.
- While transmitting, the SPI may wrongly indicate that the Tx FIFO is full. The eDMA will keep waiting for Tx FIFO to have an empty space.
- While receiving, the SPI may not indicate that the Rx FIFO is empty. The eDMA will keep reading the data from an empty FIFO.
- While receiving, the SPI may wrongly indicate that the Rx FIFO is empty. The eDMA will keep waiting for Rx FIFO to become non-empty.

Impact: The eDMA may fail to properly work with SPI.

Workaround: An extra dummy channel must be created which can be triggered by the SPI. The dummy channel should be configured to transfer four bytes from a dummy source address to a dummy destination address in the DDR memory. The dummy channel should be linked to an actual data serving channel on a minor and major loop basis.

1. The dummy channel citer and biter should be equal to the citer and biter of the data serving channel.
2. When using the fixed priority scheme, the linked eDMA channels that push/populate the SPI FIFOs must have a higher priority than the intermediate dummy eDMA channels to which the SPI requests are registered.
3. Rx side dummy-channel and linked Rx-FIFO servicing channels must have a higher priority than the Tx side counterparts. This is to avoid the Rx FIFO overflow.
4. However, restrictions in #2 and #3 are not applicable when implemented with the default round-robin priorities within groups and channels in groups (EDMA_CR[ERCA] and EDMA_CR[ERGA] both are set). Default round-robin priority scheme is recommended for this workaround implementation when there are channels from other peripherals are also present.

The below example shows an eDMA configuration in which eDMA is serving the SPI Tx FIFO and Rx FIFO simultaneously.

Where, rx_ch=0, rx_dummy_ch=1, tx_ch=2, and tx_dummy_ch=3 are Rx data servicing channel, Rx dummy channel, Tx data servicing, and Tx dummy channels, respectively.

Setting value requires byte-swap to write to eDMA register.

```c
/**** eDMA Channel TCD for TX Data Servicing channel ****/
Address -> Value
(eDMA_BASE + 0x1000 + 0x20*tx_ch) -> 0xA0000000 /*tx data buffer address in DDR*/
(eDMA_BASE + 0x1004 + 0x20*tx_ch) -> 0x02020004 /*SSIZE and DSIZE are for 32 bits, SOFF = 4*/
(eDMA_BASE + 0x1008 + 0x20*tx_ch) -> 0x4 /*NBYTES = 4*/
(eDMA_BASE + 0x100C + 0x20*tx_ch) -> 0x0 /*DLAST = 0*/
(eDMA_BASE + 0x1010 + 0x20*tx_ch) -> 0x2100034 /*SPI PUSH register address*/
(eDMA_BASE + 0x1014 + 0x20*tx_ch) -> (TX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16 /*CITER = (ddr_buffer_data_size/nbytes), DOFF = 0*/
(eDMA_BASE + 0x1018 + 0x20*tx_ch) -> 0 /*DLAST_SGA = 0*/
```
(eDMA_BASE + 0x101C + 0x20*tx_ch) -> 0x0000000A /*D_REQ = 1, INT_MAJ = 1*/

/**** eDMA Channel TCD for RX Data Servicing channel ****/
Address -> Value

(eDMA_BASE + 0x1000 + 0x20*rx_ch) -> 0x2100038 /*SPI POP register address*/
(eDMA_BASE + 0x1004 + 0x20*rx_ch) -> 0x02020000 /*SSIZE and DSIZE are for 32 bits, SOFF = 0*/
(eDMA_BASE + 0x1008 + 0x20*tx_ch) -> 0x4 /*NBYTES = 4*/
(eDMA_BASE + 0x100C + 0x20*tx_ch) -> 0x0 /*SLAST = 0*/
(eDMA_BASE + 0x1010 + 0x20*tx_ch) -> 0xB0000000 /*rx data buffer address in DDR*/
(eDMA_BASE + 0x1014 + 0x20*rx_ch) -> ((RX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16 | 0x4) /*CITER = ddr_buffer_data_size/nbytes, DOFF = 0*/
(eDMA_BASE + 0x1018 + 0x20*rx_ch) -> 0 /*DLAST_SGA = 0*/
(eDMA_BASE + 0x101C + 0x20*rx_ch) -> 0x0000000A /*D_REQ = 1, INT_MAJ = 1*/

/**** eDMA Channel TCD for TX Dummy channel ****/
Address -> Value

(eDMA_BASE + 0x1000 + 0x20*tx_dummy_ch) -> 0xF0000000 /*Dummy source address in DDR for dummy transfer*/
(eDMA_BASE + 0x1004 + 0x20*tx_dummy_ch) -> 0x02020000 /*SSIZE, DSIZE are for 32 bits, SOFF = 0*/
(eDMA_BASE + 0x1008 + 0x20*tx_dummy_ch) -> 0x4 /*NBYTES = 4*/
(eDMA_BASE + 0x100C + 0x20*tx_dummy_ch) -> 0x0 /*SLAST = 0*/
(eDMA_BASE + 0x1010 + 0x20*tx_dummy_ch) -> 0xF0000004 /*Dummy destination address in DDR for dummy transfer*/
(eDMA_BASE + 0x1014 + 0x20*tx_dummy_ch) -> ((1<<31)|(tx_ch<<25)| (TX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16) /*CITER = ddr_buffer_data_size/nbytes, link channel number tx_ch=2 to this channel, DOFF = 0*/
(eDMA_BASE + 0x1018 + 0x20*tx_dummy_ch) -> 0 /*DLAST_SGA = 0*/
(eDMA_BASE + 0x101C + 0x20*tx_dummy_ch) -> ((1<<31)|(tx_ch<<25)|(tx_ch << 8)| (1<<5)|0x00000000) /*D_REQ = 1, INT_MAJ = 1 link channel number tx_ch=2 to this channel on major as well as on minor loop basis*/

/**** eDMA Channel TCD for RX Dummy channel ****/
Address -> Value

(eDMA_BASE + 0x1000 + 0x20*rx_dummy_ch) -> 0xF0000008 /*Dummy source address in DDR for dummy transfer*/
(eDMA_BASE + 0x1004 + 0x20*rx_dummy_ch) -> 0x02020000 /*SSIZE and DSIZE are for 32 bits, SOFF = 4*/
(eDMA_BASE + 0x1008 + 0x20*rx_dummy_ch) -> 0x4 /*Nbytes = 4*/
(eDMA_BASE + 0x100C + 0x20*rx_dummy_ch) -> 0x0 /*SLAST = 0*/
(eDMA_BASE + 0x1010 + 0x20*rx_dummy_ch) -> 0xF000000C /*Dummy destination address in DDR for dummy transfer*/
(eDMA_BASE + 0x1014 + 0x20*rx_dummy_ch) -> ((1<<31)|(rx_ch<<25) | (RX_DATA_DDR_QUEUE_SIZE_IN_BYTES/4) << 16) /*CITER = ddr_buffer_data_size/nbytes, link channel number rx_ch=0 to this channel DOFF = 0*/
(eDMA_BASE + 0x1018 + 0x20*rx_dummy_ch) -> 0 /*DLAST_SGA = 0*/
(eDMA_BASE + 0x101C + 0x20*rx_dummy_ch) -> ((1<<31)|(rx_ch<<25)|(rx_ch << 8)| (1<<5)|0x00000000) /*D_REQ = 1, INT_MAJ = 1, link channel number rx_ch=0 to this channel*/

/**** eDMA channel priority configuration ****/
if ( priority_scheme == ROUND_ROBIN ) { /*Default round robin channels and groups priority scheme*/
EDMA_CR[ERCA] = 1 /*round robin arbitration for channels in each
  group*/
EDMA_CR[ERGA] = 1 /*round robin arbitration is used among the
  channel groups*/
} else { /*Fixed priority scheme*/
  /*Make sure data servicing channels has higher
  priority than their dummy channel counterpart*/
  EDMA_CR[ERCA] = 0 /* fixed priority arbitration for channels in
  each group*/
  EDMA_CR[ERGA] = 0 /* fixed priority arbitration is used among the
  channel groups*/

  /**** Configuration for group priorities ****/
  if (data_servicing_channels_grp == GRP1) { /*Data servicing
  channels are in channel group-1 i.e. in the same group where dummy channels
  are present*/
    /*either group-1 has higher priority */
    EDMA_CR[GRP1PRI] = 0 /*Channel group-1 has higher priority*/
    EDMA_CR[GRP2PRI] = 1 /*Channel group-2 has lower priority*/
    /*Or group-2 has higher priority*/
    EDMA_CR[GRP1PRI] = 1 /*Channel group-1 has higher priority */
    EDMA_CR[GRP2PRI] = 0 /*Channel group-2 has lower priority*/
  } else { /*data servicing channels are in channel group group-2
  i.e. in the different group where the dummy channels are present*/
    EDMA_CR[GRP1PRI] = 1 /*Channel group-1 has lower priority */
    EDMA_CR[GRP2PRI] = 0 /*Channel group-2 has higher priority
  */
  }

  /**** Configuration for channel priorities in the group ****/
  EDMA_DCHPRI0  = 0x00; /*priority for channel number rx_ch is 0*/
  EDMA_DCHPRI1  = 0x01; /*priority for channel number tx_dummy_ch is
  1*/
  EDMA_DCHPRI2  = 0x02; /*priority for channel number tx_ch is 2*/
  EDMA_DCHPRI3  = 0x03; /*priority for channel number tx_dummy_ch is
  3*/
}

/**** Enable eDMA channels ****/
DMAMUX1_CHCFG1 = (0x80 | 0x3C) ; /* Enable channel number, rx_dummy_ch=1 to
  triggered by SPI1 RFDF i.e by source number 60*/
DMAMUX1_CHCFG3 = (0x80 | 0x3E) ; /* Enable channel number, tx_dummy_ch=3 to
  triggered by SPI1 TF i.e by source number 62*/
Figure 1. Dummy channel based workaround

**Fix plan:** No plans to fix
A-008171: eSDHC transactions may fail in tuning modes of operation when the input data window is close to 1 UI during tuning

**Affects:** eSDHC

**Description:** In tuning mode of operation, when TBCTL[TB_EN] is set, eSDHC may report one of the following errors:
- Tuning error while running tuning operation where SYSCTL2[SAMPCLKSEL] will not get set even when SYSCTL2[EXTN] is reset.
- Data transaction error (for example, IRQSTAT[DCE], IRQSTAT[DEBE]) during data transaction errors.

This issue occurs when the data window sampled within eSDHC is full cycle. So, in that case eSDHC is not able to find out the start and end points of the data window and sets the sampling pointer at default location (which is middle of the internal SD clock). If this sampling point coincide with the data eye boundary then it can results into above mentioned errors.

**Impact:** Tuning mode of operation for SDR50, SDR104 or HS200 speed modes may not work properly.

**Workaround:** In case the eSDHC reports tuning error or data errors in the tuning mode of operation, then the software can use one of the below mentioned workarounds:
- Shift the sampling pointer by half SD_CLK cycle from its default location.
- Increase the sampling clock (per_clk) frequency so that data window is sampled less then full cycle.
- Change the SD_CLK frequency so that the sampling point does not coincide with the start or end of data eye.
- Use the Fixed sampling technique for the SDR50 mode.

**NOTE**

\[
\text{DIV\_RATIO} = \text{ESDHCCCTL[SDCLKFS]} \times \text{ESDHCCCTL[DVS]} \quad \text{when} \quad \text{CRS}=0
\]

\[
\text{DIV\_RATIO} = \text{ESDHCCCTL[USDCLKFS[0:1],SDCLKFS[0:7]}} \quad \text{when} \quad \text{CRS}=1.
\]

**Procedure to shift the sampling pointer by half SD_CLK period from its default location:**

1. Program TBPTR[TB_WNDW_END_PTR] = 3*DIV\_RATIO (that is, three times division ration of SD_CLK).
2. Program TBPTR[TB_WNDW_START_PTR] = 5*DIV\_RATIO.
3. Program the software tuning mode by setting TBCTL[TB_MODE] = 2'h3.
4. Set SYSCTL2[EXTN] and SYSCTL2[SAMPCLKSEL].
5. Issue the SEND_TUNING_BLK command (CMD19 for SD and CMD21 for MMC).
6. Wait for IRQSTAT[BRR], the buffer read ready to be set.
7. Clear IRQSTAT[BRR].
8. Check the SYSCTL2[EXTN] to be cleared.
9. Check SYSCTL2[SAMPCLKSEL], the sampling clock select. It's set value indicate tuning procedure success, and clear indicate failure. In case of tuning failure, the fixed sampling scheme could be used by clearing TBCTL[TB_EN].

**Fix plan:** No plans to fix
A-008703: SDHC DDR50 AC timing specification for (tshdivkh) setup time and output delay are not met.

Affects: eSDHC

Description: To work with an SD card in the DDR50 mode, the setup time (tshdivkh) requirement should be no less than 0.5 ns after considering the voltage translator skew, duty cycle margin, and board skews. Because of this erratum, the setup time may not meet that time requirement. Similar to output delay (tshdkhov), a value of 5.7 ns may not be met at times due to this erratum.

Impact: The part cannot run at the maximum speed of 50 MHz as specified in the SD 3.0 Physical specification. Out-of-specification operation could lead to undefined behavior.

Workaround: Reduce the maximum clock to no more than 44 MHz from 50 MHz.

Fix plan: No plans to fix
A-008705: SDHC eMMC DDR mode AC timing specification for input setup time (tshdivkh) issue

**Affects:** eSDHC

**Description:** In SDHC eMMC DDR AC timing, the tshdivkh parameter is defined as input setup times: SDHC_DATx to SDHC_CLK. To run at the maximum speed of 50 MHz in the DDR mode, the tshdivkh should be 0.5 ns considering duty cycle margin, board skews, and voltage translator skew. Because of this erratum, the tshdivkh value is 1.98 ns instead of 0.5 ns.

**Impact:** eSDHC cannot run at the maximum clock speed for the eMMC DDR mode if a voltage translator is needed.

**Workaround:** Reduce the maximum SDHC clock to 44 MHz.

**Fix plan:** No plans to fix
A-008708: Issue with input setup time tshivkh for eSDHC high speed mode

Affects: eSDHC

Description: In SDHC high speed AC timing, the tshivkh parameter is defined as input setup times: SDHC_CMD, SDHC_DATx, to SDHC_CLK. The value of the tshivkh should be 2.5 ns considering the round trip delay, board/data skew. However, because of this erratum, it needs at least 4.1 ns.

Impact: eSDHC cannot run at the maximum clock speed for the high speed mode, or there is a limit on the length of the trace on the board for data, command, and clock lines of the SDHC.

Workaround: Choose one of the following workarounds:
- Reduce the maximum speed to 46.5 MHz.
- In the high speed mode, no voltage translator is needed. Limit the one-way delay to 0.7 ns for a SD card, and 0.5 ns for a MMC card.

Fix plan: No plans to fix
A-009620: Data timeout error not getting set in case of command with busy response (R1b) as well as for busy period after last write block transfer

Affects: eSDHC

Description: In the event that a busy timeout occurs for a command with a busy response (for example, R1b response) as well as busy period after the last write block, the eSDHC does not set the IRQSTAT[DTOE] bit or the IRQSTAT[TC]. Therefore, the current command transfer is never completed.

Impact: Software has to track the busy timeout error in this case.

Workaround: Choose one of the following workarounds for commands with a busy response:

• Implement a software timer to track the busy timeout error. If this software timer expires before the IRQSTAT[TC] event then consider it as data timeout error event (same as if IRQSTAT[DTOE] gets set) and execute the error recovery sequence for data timeout error specified in section 3.10.1 "Error Interrupt Recovery" in SD Host specification 3.0.

OR

• Don't set the XFRTYP[RSP]=2'b11 for commands with a busy response. Rather, poll the busy status of the card from the PRSSTAT[DLSL].

The workaround sequence for a busy period after last write block is as follows:

1. After the command completion interrupt (IRQSTAT[CC]), wait for de-assertion of PRSTAT[WTA].
2. As soon as PRSTAT[WTA] is de-asserted, start the software timer and poll the busy signal (DAT0) using PRSTAT[DLSL[0]].
3. Wait for DAT0 signal to go high (which indicates that the transfer is complete) or for the software timer expiry (which indicates a data timeout error).
4. Issue a soft reset for data (SYSCTL[RSTD]), if PRSTAT[DLA] bit is set.
5. In the case of a data timeout error (detected in step 3), perform error recovery.

Fix plan: No plans to fix
A-006264: eTSEC may drop bytes in FIFO mode if frame matches a Control Frame ethertype

**Affects:** eTSEC

**Description:** When operating in FIFO mode, receive frames are processed as if they do not contain an L2 header unless RCTRL[bit 26] = 1. When RCTRL[bit 26] = 0, there should be no check for an ethertype field, but if parsing is disabled (RCTRL[PRSDEP] = 00) the frame is erroneously checked for an ethertype field. If the 13th and 14th bytes of the frame match the Control Frame ethertype (0x8808), then the 13th-16th bytes of the frame may be dropped.

**Impact:** The controller may silently drop four bytes starting at the 13th byte of a received frame if the 13th and 14th bytes are 0x88 and 0x08, respectively. This could cause unwanted behavior with upper-layer applications.

**Workaround:** Set RCTRL[bit 16] = 1 when operating in FIFO mode with RCTRL[bit 26] = 0. RCTRL[bit 16] is Control Frame Accept (CFA). RCTRL[bit 26] is FIFO Mode Parsing (PRSFM).

**NOTE**
For products that support L2 headers in FIFO mode, eTSEC does not support control frames in FIFO mode. So, if a user attempts to send a control frame to eTSEC, the same failure may occur – 4 bytes are deleted, but the frame is not dropped.

If PRSFM = 1 and the Rx packet stream includes frames with ethertype = 0x8808, then RCTRL[CFA] must be 1.

**Fix plan:** No plans to fix
A-006293: Mixing TOE = 0 and TOE = 1 frames may cause data corruption

Affects: eTSEC

Description: eTSEC supports several TCP offload functions on transmitted frames, including IP checksum generation, TCP checksum generation, and VLAN tag insertion. If the controller is processing a mixture of frames with TxBD[TOE] = 1 and TxBD[TOE] = 0, it may use an incorrect Transmit Frame Control Block (TxFCB) for a frame and, therefore, incorrectly process the packet data.

The effects of incorrect TxFCB processing are as follows:
- TOE function executed on frame with TOE = 0
- TOE function not executed on frame with TOE = 1
- Different TOE function executed than intended (one or more fields of TxFCB perform an unintended action)

In most cases, this corrupts the frame in a way that is detected by the receiver (for example, checksum error). However, in some cases, the frame corruption may not be detected by the receiver.

Impact: Mixing TOE = 1 and TOE = 0 frames may cause corrupted packets.

Whether a particular frame is affected is dependent on the TxBD[TOE] settings and sizes of the frame(s) preceding the frame in question, as well as the transient state of the internal Tx FIFO SRAM.

Workaround: Do not mix TCP offload usage for frames. Write either TxBD[TOE] = 1 on all frames or TOE = 0 on all frames.

Fix plan: No plans to fix
A-006502: Incomplete GRS or invalid parser state after receiving a 1- or 2-byte frame

**Affects:** eTSEC

**Description:** Ethernet standards define the minimum frame size as 64 bytes. The eTSEC controller also supports receiving short frames less than 64 bytes, and can accept frames more than 16 bytes and less than 64 bytes if RCTRL[RSF] = 1. Frames shorter than 17 bytes are supposed to be silently dropped with no side-effects. There are, however, two scenarios in which receiving frames <= 2 bytes cause erroneous behavior in the controller.

In the first scenario, if the last frame (such as an illegal runt packet or a packet with RX_ER asserted) received prior to asserting graceful receive stop (DMACTRL[GRS]=1) is <= 2 bytes, then the controller fails to signal graceful receive stop complete (IEVENT[GRSC]) even though the GRS has successfully executed and the receive logic is completely idle. Any subsequent receive frame that is larger than 2 bytes resets the state so the graceful stop can complete (IEVENT[GRSC] = 1). A MAC Rx reset also resets the state.

In the second scenario, the parser and filer are enabled (RCTRL[PRSDEP] = 01,10,11). If a 1- or 1.5-byte frame is received, the controller carries over some state from that frame to the next, causing the next frame to be parsed incorrectly. This, in turn, may cause incorrect parser results in RxFCB and incorrect filing (accept versus reject, or accept to wrong queue) for that following frame. The parser state recovers itself after receiving any frame >= 2 bytes in length.

**Impact:** If software initiates a graceful receive stop after a 1- or 2-byte frame is received, the stop may not complete until another frame has been received.

A frame following a 1 or 1.5B frame may be parsed and filed incorrectly.

**Workaround:** For GRS scenario:

After asserting graceful receive stop (DMACTRL[GRS] = 1), initiate a timeout counter. The wait time is system and memory dependent, but a reasonable worst-case time is the receive time for a 9.6 Kbyte frame at 10/100/1000 Mbps. If IEVENT[GRSC] is still not set after the timeout, read the eTSEC register at offset 0xD1C. If bits 7-14 are the same as bits 23-30, the eTSEC Rx is assumed to be idle and the Rx can be safely reset. If the register fields are not equal, wait for another timeout period and check again.

MAX Rx reset procedure:
1) Clear MACCFG[RX_EN].
2) Wait three Rx clocks.
3) Set MACCFG2[RX_EN].

**Fix plan:** No plans to fix
A-007207:  TBI link status bit may stay up after SGMII electrical idle is detected

Affects: Ethernet
Description: The TBI Status register (SR) contains a Link Status bit (TBI SR [Link Status]) that represents the current state of the SGMII link. If auto-negotiation (AN) is disabled, the TBI Link Status bit should be 1 (indicating the link is up) after recognizing IDLE sequences, and stay at 1 as long as valid data is received and the TBI is not reset. The TBI Link Status bit should be 0 (indicating the link is down) after several invalid characters are received or the TBI is reset. If AN is enabled, the TBI Link Status bit is not set to 1 until auto-negotiation is complete (TBI CR [AN DONE] = 1), but the same conditions as AN disabled then apply for the TBI Link Status bit to be cleared to 0.

An electrical idle (common mode) condition on the SGMII link results in the reception of invalid data and should cause the TBI Link Status bit to get cleared. If the transition from active to common mode takes enough time that the Rx is able to recognize at least 4 more K28.5 characters (for IDLE sequences, 70-80 UI), the portion of the design intended to detect the link down condition may shut off before the link down condition is actually reflected in the TBI.

This premature shutdown may cause the TBI Link Status to remain set to 1, indicating the link is up. This 'stuck at 1' condition persists until valid K28.5 characters are received again.

Impact: If the system never enters SGMII electrical idle, or if the transition from active to common mode takes less than 40 UI (~32 ns), then there is no impact and the false link up scenario does not occur.

If the system can generate an SGMII electrical idle condition as described above, then the TBI status may stay stuck at 1 while the link is down and does not transition to 0 until valid K28.5 characters are received again.

Workaround: If TBI SR[Link Status] = 0, the link is down.

For affected systems, in addition to examining the TBI link status, examine the SerDes electrical idle state. The link is actually down if either the TBI link status is cleared or the SerDes lane receive electrical idle is detected.

SerDes electrical idle detected is SRDSxLNmGCR1[REIDL] for lane m.

Example pseudo-code:

```python
If (SRDSxLNmGCR1[REIDL] == b'1)
OR
(TBI SR[LINK STATUS] == b'0)
{ LINK is DOWN }
```

Fix plan: No plans to fix
A-008699: RGMII AC timing specifications for tSKRGT_TX max are not met

**Affects:** eTSEC

**Description:** In RGMII AC timing, the tSKRGT_TX parameter is defined as data-to-clock output skew (at transmitter). tSKRGT_TX max (at 1.8v) is +0.7ns instead of +0.5ns. The tSKRGT_TX timing of ±0.5ns typically requires 1.5ns to 2.1ns of introduced fixed delay in PCB, and/or PHY programmable delay, to the transmit clock signal relative to transmitted data to ensure that the receiving RGMII data to clock skew is between 1ns to 2.6ns. This erratum alters the range of typical delay.

**Impact:** Out-of-specification operation could lead to undefined behavior, which may include false errors, false passes, or CRC errors.

**Workaround:** None

**Fix plan:** No plans to fix
A-008700: RGMII AC timing specifications for tSKRGT_TX min is not met

**Affects:** eTSEC

**Description:** In RGMII AC timing, the tSKRGT_TX parameter is defined as data-to-clock output skew (at transmitter).

tSKRGT_TX_min (at 2.5v) is -0.77ns instead of -0.5ns.

The tSKRGT_TX timing of -0.5ns typically requires 1.5ns to 2.1ns of introduced fixed delay in PCB, and/or PHY programmable delay, to the transmit clock signal relative to transmitted data to ensure that the receiving RGMII data to clock skew is between 1ns to 2.6ns. This erratum alters the range of typical delay.

**Impact:** Out-of-specification operation could lead to undefined behavior, which may include false errors, false passes, or CRC errors.

**Workaround:** None

**Fix plan:** No plans to fix
**A-009729: Corrupted frame possible if Freeze Mode or Low Power Mode are entered during a bus-off state**

**Affects:** FlexCAN

**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 is 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.

**Impact:** Potential for corrupted transmit frame during the Freeze Mode or Low Power Mode.

**Workaround:** To request a Freeze or Low Power Mode during the Bus-Off state, the following procedures must be followed:

A) Procedure to enter in 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. Set the Soft Reset bit (SOFTRST) in MCR.
4. Poll the MCR register until the Soft Reset (SOFTRST) bit is cleared (timeout for software implementation is 2 CAN Bits length).
5. Poll the MCR register until the Freeze Acknowledge (FRZACK) bit is set (timeout for software implementation is 2 CAN Bits length).
6. Reconfigure the Module Control Register (MCR)
7. Reconfigure all the Interrupt Mask Registers (IMASKn).

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).

**Fix plan:** No plans to fix
A-009732: The transmission abort mechanism may not work properly

**Affects:** FlexCAN

**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:

- If a pending abort request occurs while the FlexCAN is receiving a remote frame.
- When a frame is aborted during an overload frame after a frame reception.
- When an abort is requested while the FlexCAN has just started a transmission.
- When Freeze Mode request occurs and the FlexCAN has just started a transmission.

**Impact:** Failure to abort transmit frame results in the transmit process remaining in pending state.

**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.

**Fix plan:** No plans to fix
A-007728: The process of clearing the FTMx_SC[TOF] bit does not work as expected under a certain condition when the FTM counter reaches FTM_MOD value

Affects: FlexTimer

Description: If the FTM counter reaches the FTM_MOD value between the reading of the TOF bit and the writing of 0 to the TOF bit, the process of clearing the TOF bit does not work as expected when FTMx_CONF[NUMTOF] != 0 and the current TOF count is less than FTMx_CONF[NUMTOF]. If the above condition is met, the TOF bit remains set. If the TOF interrupt is enabled (FTMx_SC[TOIE] = 1), the TOF interrupt also remains asserted.

Impact: None

Workaround: Two possible workarounds exist for this erratum and the decision on which one to use is based on the requirements of the application that is used.

OPTION 1:
Repeat the clearing sequence mechanism until the TOF bit is cleared.

Below is a pseudo-code snippet that must be included in the TOF interrupt routine:

```c
while (FTM_SC[TOF] != 0)
{
    void FTM_SC() ;         // Read SC register
    FTM_SC[TOF] = 0 ;    // Write 0 to TOF bit
}
```

OPTION 2:
With FTMx_CONF[TOFNUM] = 0 and a variable in the software, count the number of times that the TOF bit is set. In the TOF interrupt routine, clear the TOF bit and increment the variable that counts the number of times that the TOF bit is set.

Fix plan: No plans to fix
A-009659: Incorrect match may be generated if intermediate load feature is used in toggle mode

**Affects:** FlexTimer

**Description:** When a channel \((n)\) match is used as an intermediate reload, an incorrect second match may occur immediately following the correct match. The issue is problematic only if channel \((n)\) is configured for output compare with the output configured to toggle mode. In this scenario, channel \((n)\) toggles on the correct match and again on the incorrect match. The issue may also occur if a certain channel has a match which is coincident with an intermediate reload point of any other channel.

**Impact:** Incorrect match may lead to undesired functionality.

**Workaround:** If any channel is configured for output compare mode with the output set for toggle mode, the intermediate reload feature must not be used.

**Fix plan:** No plans to fix
A-011026: Safe state is not removed from the channel output after fault condition ends if SWOCTRL register is used to control the pin

**Affects:** FlexTimer

**Description:** If an FTM channel output is controlled using the software output control register (FTM_SWOCTRL) and the fault detection is also enabled for the channel, then when a fault is detected, the output is forced to its safe value. However, when the fault condition has been cleared, the channel output will stay in the safe state instead of reverting to the value programmed by the FTM_SWOCTRL register.

**Impact:** When the FTM_SWOCTRL register is used to control the FTM channel, the safe state is not removed from the FTM channel output when the fault condition ends.

**Workaround:** If the fault control is enabled while the software output control register is also being used (FTM_SWOCTRL), then the FTM should be configured as follows:

- FTM_MODE[FAULTM] configured for manual fault clearing (0b10)
- The FTM_CONF[NUMTOF] field must be cleared to 0b00000 (TOF set for each counter overflow).

The procedure below must be used in the TOF interrupt handler when a fault is detected to ensure that the output return to the value configured by the FTM_SWOCTRL register.

1. Check the value of the FTM_FMS[FAULTF] register.
   - If FTM_FMS[FAULTF] = 1 (fault occurred or is occurring), then set a variable to indicate that a fault was detected and continue to step 2.
   - If FTM_FMS[FAULTF] = 0 but the fault variable is set (fault is not active, but was previously detected), skip to step 6.
2. Write the FTM_OUTMASK register to set the bit(s) corresponding to any channels that are controlled by the FTM_SWOCTRL register to temporarily inactivate the channel output.
3. Clear the fault conditions by reading the FTM_FMS register and then writing FTM_FMS with all zeroes.
4. Clear the FTM_SC[TOF] bit by reading the FTM_SC register, then writing 0 to FTM_SC[TOF].
5. Exit the interrupt handler to skip following steps (they will execute next time the TOF handler is called).
6. Clear the FTM_SWOCTRL register by writing all zeroes to it.
7. Write the FTM_SWOCTRL register with the desired value again.
8. Clear the FTM_OUTMASK bits that were set in step 2.
9. Clear the fault variable that was set in step 1 when the fault condition was originally detected.
10. Clear the FTM_SC[TOF] bit by reading the FTM_SC register, then writing 0 to FTM_SC[TOF].

**Fix plan:** No plans to fix
A-008213: LVCMOS pins VIH and VIL level do not meet JEDEC specification standard values for LVCMOS as specified on JESD8C.01

**Affects:** GEN

**Description:** The VIH min and VIL max values are slightly higher/lower than specified on JEDEC specification for interface standard for nominal 3.3 V supply digital integrated circuit. For specific VIH/VIL values please see the DC specific section of the datasheet.

**Impact:** Device may not recognize proper logic level and thus may not operate properly if required VIH/VIL values are not satisfied.

**Workaround:** Ensure VOH min and VOL max values plus board margin exceed the VIH/VIL requirement.

**Fix plan:** No plans to fix
A-010539: Some eSDHC and GPIO pins are not functional if the source of the RCW is selected as QSPI

Affects: GEN

Description: If the source of RCW is selected as QSPI, the following alternate functionality will not be available in SPI_* pins:

- SDHC_CLK_SYNC_OUT
- SDHC_CLK_SYNC_IN
- SDHC_VS
- SDHC_DAT[4:7]
- SDHC_DAT0_DIR
- SDHC_DAT123_DIR
- SDHC_CMD_DIR
- GPIO2_[0:3]

Impact: The eMMC 8-bit mode is nonfunctional. Standard-speed and high-speed modes are supported when an SD card is connected to the eSDHC interface. All SDR modes that require 1.8 V and GPIO pins related to RCW[SPI_BASE] and RCW[SPI_EXT] pin mux do not work.

Workaround: NOTE

The HRESET pin will no longer function once the workaround sequence is executed.

Do the following when the system comes out of (POR) reset and is in the System Ready state:

1. Read DCFG_CCSR_PORSR1, dat = DCFG_CCSR_PORSR1
2. Clear RCW_SRC bits (bit 0:8): dat[RCW_SRC] = 0
3. Write the updated data value, dat, to address: 0x2014_0000
4. Write 0xFFFF_FFFF to address: 0x0157_0000 + 0x1A8

Fix plan: No plans to fix.
A-010111: Non-core master can only access Generic Interrupt Controller (GIC) banked registers of Core 0

**Affects:** GIC

**Description:** An Arm-GIC-per-core-banked register has the same address but one copy for each individual core. During normal operation, a core-specific register is accessed from each core by driving its unique core-ID attribute in addition to the address.

When accessing these registers from non-core masters like the debugger interface, the unique core-ID attribute is not implemented correctly. Therefore, only Core0-specific registers are accessible from non-core masters. Non-Core0-Arm-GIC-per-core-banked registers cannot be retrieved.

**Impact:** The non-Core0-Arm-GIC-per-core-banked registers cannot be retrieved by non-core masters. For example, the debugger cannot access per-core-banked-GIC registers belonging to cores other than Core 0.

**Workaround:** None. Use core software to access non-Core0-Arm-GIC-per-core-banked registers.

**Fix plan:** No plans to fix
A-009286: Missing glitch filter implementation

Affects: I2C

Description: The I2C controller specification specifies the filtering out of glitches spanning a maximum of 50 ns on the SDA and SCL lines in the fast mode of operation. Due to this erratum, the I2C controller does not implement the circuitry to filter these glitches.

Impact:
- A glitch on the SDA or SCL line can cause a momentary false trigger on the signal line.
- A glitch on SDA could result in the incorrect recognition of a START or STOP condition, thus breaking the bus protocol.
- A glitch on SCL could result in incorrect data transfer, also breaking bus protocol.
- In all cases, data transfer is corrupted and the bus could hang.

Workaround: In order to avoid this situation, the user needs to implement circuitry to filter out glitches from SDA and SCL lines.

Add an external low pass filter to filter out glitches spanning a maximum of 50 ns.

Fix plan: No plans to fix
A-010124: Attempting a start cycle while the bus is busy may generate a short clock pulse

*Affects:* I2C

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

*Impact:* I2C controller may generate a glitch to the SCL in a multi-master network. It may cause the bus to hang and other protocol violations.

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

*Fix plan:* No plans to fix
A-010650: I²C controller is unable to generate clocks when SDA is low

**Affects:** I²C

**Description:** Based on the I²C specification, if the data line (SDA) is stuck low, the master should send nine clock pulses and the I²C slave device that holds the bus low should release it sometime within those nine clocks. But the I²C controller cannot generate nine clock pulses, as a result the slave device is unable to release the bus.

**Impact:** The I²C bus cannot be set to idle when a slave device holds the bus low.

**Workaround: For I²C1 - I²C4 controllers:**

Use an open drain GPIO pin to connect to the IICx_SCL to drive nine clock pulses to unlock the I²C bus.

**NOTE**

When GPIO is connected to the existing I²C net, it already has a pullup on the net as required by the I²C specification. The value is based on the speed desired versus capacitance on that net. No additional pull-up resistor besides that designed in pullup for the I²C bus should be added.

**Fix plan:** No plans to fix
A-007734: Stale time stamps and overflow conditions can occur when using an external 1588 input clock at certain frequencies

Affects: IEEE 1588

Description: When the time stamp logic is enabled using an external 1588 input clock at certain frequencies, the eTSEC receiver can fail to update with the latest time stamp. This results in a stale time stamp in the Rx buffer padding even though eTSEC_TMR_RXTS is updated correctly.

In addition, the eTSEC receiver can hang because of overflow conditions. The setting of bit[6] of eTSEC_RSTAT indicates the overflow condition has occurred.

Impact: User cannot use the full frequency ranges of the 1588 input clock as specified in the data sheet:
- 17.85 MHz to 200 MHz for 1000 Mbps mode
- 3.57 MHz to 200 MHz for 100 Mbps mode
- 0.36 MHz to 200 MHz for 10 Mbps mode

Workaround: Limit the 1588 input clock frequency to:
- 66 MHz to 200 MHz for 1000 Mbps mode
- 13.2 MHz to 200 MHz for 100 Mbps mode
- 1.32 MHz to 200 MHz for 10 Mbps mode

Limit the maximum frequency of the 1588 input clock to the CCB frequency if it is lower than 200 MHz.

Fix plan: No plans to fix
A-008172: Issue with write protect toggling in NAND NVDDR mode

Affects: IFC
Description: IFC does not work properly in the following scenarios:
- CSPRn[WP] is changed immediately after switching from asynchronous SDR mode to
  synchronous NV-DDR mode.
- NAND device is in NVDDR mode and write protect value in CSPRn[WP] is changed.

Impact: Write protect WP# can’t be toggled in NVDDR mode.

Workaround: WP# (CSPRn[WP]) value can only be changed when NAND device is in asynchronous SDR
mode. To toggle the value of WP# when device is in synchronous NVDDR mode, use the
following steps:
1. Transition the NAND device to the asynchronous mode by issuing reset command (Refer
to IFC RM section "Switching to the asynchronous interface").
2. Change the value of WP# in corresponding CSPRn[WP] register per the requirement.
3. Transition the NAND device back to NVDDR mode using set feature command (Refer to
IFC RM section "Activating the NVDDR Interface").

Fix plan: No plans to fix
A-008175: NV-DDR program operation with ECC encoder enabled does not work with DDR clock division ratio 2

**Affects:** IFC

**Description:** In NV-DDR mode, NAND program operation does not work when DDR interface clock is set to “divide by 2” (DDR_CCR_LOW[DDR_LOW_CLKDIV] = 0) and ECC encoder is enabled (CSORn[ECC_ENC_EN=1]).

**Impact:** NV-DDR NAND flash does not support the divide-by-2 ratio for program operation.

**Workaround:** Higher division ratios can be programmed in the DDR_CCR_LOW[DDR_LOW_CLKDIV] register. The minimum ratio supported is the divide-by-4 ratio.

**Fix plan:** No plans to fix
A-009241:  Unaligned write transactions to IFC may result in corruption of data

Affects:  IFC

Description:  16 byte unaligned write from system bus to IFC may result in extra unintended writes on external IFC interface that can corrupt data on external flash.

Impact:  Data corruption on external flash may happen in case of unaligned writes to IFC memory space.

Workaround:  Following are the workarounds:
  • For write transactions from core, IFC interface memories (including IFC SRAM) should be configured as “device type” memory in MMU.
  • For write transactions from non-core masters (like system DMA), the address should be 16 byte aligned and the data size should be multiple of 16 bytes.

Fix plan:  No plans to fix
A-009652:  \( t_{DQSQ} \) and \( t_{DVW} \) does not meet NAND industrial standard ONFI AC spec for 3.3 V NV-DDR mode

Affects:  IFC
Description:  For 3.3 V source synchronous NAND (NV-DDR mode), ONFI standard for \( t_{DQSQ} \) and \( t_{DVW} \) for 83 MHz (mode 4) are:
• \( t_{DQSQ} \): 1 ns
• \( t_{DVW} \): 2.1 ns

The specification from the measurement is:
• \( t_{DQSQ} \): 0.57 ns
• \( t_{DVW} \): 2.95 ns

Impact:  For 3.3 V source synchronous NAND (NV-DDR mode), the board design must meet the tightened specification for DQ and DQS skew.

Workaround:  Use the 1.8 V NAND device instead. For 3.3 V NAND, minimize the DQS and DQ skew to meet the tightened specification.

Fix plan:  No plans to fix
A-010581: LPUART may send two break characters during idle mode, where one is expected

Affects: LPUART

Description: Two break characters may be sent when the LPUART_CTRL[SBK] bit is set and cleared immediately. This occurs when the transmitter is already in the idle mode, indicated by LPUART_STAT[TC] bit being high.

Impact: An unwanted character may be received after the second break character.

Workaround: Queue a single-break character through the transmit FIFO by writing to the following register bits:

Set LPUART_DATA[FRETSC]=1 with data bits LPUART_DATA[T9:T0]=0.

Fix plan: No plans to fix.
A-010727: LPUART_RX Pin Active edge flag cannot be set as expected

Affects: LPUART

Description: The LPUART_RX Pin Active Edge Interrupt flag (LPUART_STAT[RXEDGIF]) in the LPUART Status Register cannot be set when an active edge is detected and the Receiver Enable (LPUART_CTRL[RE]) bit in the LPUART Control Register is set. The LPUART_STAT[RXEDGIF] bit can only be set when in addition to the Receiver Enable (LPUART_CTRL[RE]) bit, the LPUART_RX Input Active Edge Interrupt Enable (LPUART_BAUD[RXEDGIE]) bit is also set in the LPUART Baud Rate register.

NOTE
The LPUART_BAUD[RXEDGIE] bit enables the interrupt for the LPUART_STAT[RXEDGIF] register bit.

Impact: The flag bit used by the software to detect an active edge on LPUART_RX cannot be set unless an interrupt is enabled to detect LPUART_RX active edge. Therefore, it is not possible for the software to poll for the LPUART_RX active edge. If detection of LPUART_RX active edge is required, it must be done by the use of interrupt.

Workaround: Set the LPUART_BAUD[RXEDGIE] register bit in addition to the LPUART_CTRL[RE] bit, when a software detection is needed to detect an active edge on the LPUART receiver.

Fix plan: No plans to fix
A-007189: PCI Express inbound error message handled incorrectly in RC mode

Affects: PCIe

Description: According to Figure 6-3 of the PCI Express base specification REV 3.0, Inbound errors reported to PCI Express RC:
• In its Root Error Status Register can be gated by the "SERR# Enable" bit in the PCI Express Command Register and the "Error Reporting Enables" bit fields (the URR, FER, NFER, or CER bit) in the Device Control Register.
• In its PCI Express Status Register can be gated by the “SERR# Enable" bit in the PCI Express Command Register

However, due to this error, the PCI Express controller in RC mode does not allow the error status reporting in both the PCI Status Register and the Root Error Status Register to be controllable by the PCI Express Command Register [SERR# Enable] bit and the Device Control Register [URR, FER, NFER, CER] bits.

Impact: The inbound errors, including both detected by the PCI Express RC itself and from externally-received error messages, are directly reported to both the PCI Express Status Register [Signaled System Error] bit and the Root Error Status Register [FEMR, NFEMR, FUF, MEFNFR, EFNFR, MECR, ECR] bits, regardless of the setting of the PCI Express Command Register [SERR# Enable] bit or the Device Control Register [URR, FER, NFER, CER] bits.

However, for externally-received error messages, the error status reporting in both the PCI Express Status Register [Signaled System Error] bit and the Root Error Status Register [FEMR, NFEMR, FUF, MEFNFR, EFNFR, MECR, ECR] bits can still be controlled by the Bridge Control Register [SERR_EN] bit as normal, when propagating the error messages from the secondary to primary side of the root port.

Workaround: Although the PCI Express RC error status reporting in both the PCI Status Register and the Root Error Status Register cannot be controllable by the PCI Express Command Register [SERR# Enable] bit and the Device Control Register [URR, FER, NFER, CER] bits, the system software can still use the Root Control Register [SEFEE, SENFEE, SECEE] bits and Root Error Command Register [FERE, NFERE, CERE] bits to control the interrupt generation, for the inbound errors reported in the Root Error Status Register [FEMR, NFEMR, FUF, MEFNFR, EFNFR, MECR, ECR] bits.

Note that the inbound errors reported in the PCI Express Status Register [Signaled System Error] bit does not trigger an interrupt.

Fix plan: No plans to fix
A-007815: The read-only-write-enable bit must be cleared to prevent overwriting read-only registers

Affects: PCIe

Description: The PCI Express controller features overwrite capability for read-only registers within its configuration space. This capability is turned on by default, which might cause read-only registers in the PCI Express configuration space to be overwritten unintentionally.

Impact: The read-only registers in the PCI Express configuration space may be inadvertently overwritten, which might cause system to malfunction.

Workaround: To prevent unintentionally overwriting to read-only registers in the PCI Express configuration space, the DBI_RO_WR_EN bit (bit 0) of the DBI Read-Only Write Enable Register at PCI Express configuration space offset 8BCh must be cleared.

Since system software might have a need to perform overwrite to certain PCI Express configuration space read-only registers during initialization, it's recommended to perform this clear action to the DBI_RO_WR_EN bit at the end of the Pre-Boot Initialization (PBI) process.

If the local software (for example, boot loader) has a need to utilize this overwrite capability after PBI, the software must clear the DBI_RO_WR_EN bit after finishing the overwrite task.

Fix plan: No plans to fix
A-007997: PCI Express hot-plug-related bits in the Slot Capabilities Register need to be cleared

Affects: PCIe

Description: The device does not support PCI Express hot-plug functionality. However, the value of the Slot Capabilities Register of the PCI Express controllers indicates that various items related to PCI Express hot-plug are supported.

Impact: Without clearing the bit value of all the PCI Express hot-plug-related bits in the Slot Capabilities Register, it appears that the device supports PCI Express hot-plug while in fact it doesn't.

Workaround: In order to avoid indicating that the device supports PCI Express hot-plug, the following bits in the Slot Capabilities Register need to be cleared during Pre-boot Initialization (PBI):
- Bit 18, No Command Completed Support
- Bit 17, Electromechanical Interlock Present
- Bit 6, Hot-Plug Capable
- Bit 5, Hot-Plug Surprise
- Bit 4, Power Indicator Present
- Bit 3, Attention Indicator Present
- Bit 2, MRL Sensor Present
- Bit 1, Power Controller Present
- Bit 0, Attention Button Present

The above register is defined by the PCI Express base specification as read-only register. The overwrite capability should be turned on by setting the DBI_RO_WR_EN bit (Bit 0) of the DBI Read-Only Write Enable Register at the PCI Express configuration space offset 0x8BC, before writing to any Read-Only register within the controller's configuration space. Refer to the A-007815 for further detail regarding the overwrite capability feature.

Fix plan: No plans to fix
A-008236: PCI Express controller does not exit disabled state if the Link Control register [Link Disable] bit is cleared before lanes enter electrical idle in RC mode

Affects: PCIe

Description: When the PCI Express controller is configured in RC mode, software can disable the link by setting the configuration space Link Control register [Link Disable] bit to direct the LTSSM (Link Training and Status State Machine) to the disabled state, which is reflected with the memory mapped SCFG_PEXnMSCPORTSR register [XMLHLTSSMSTATE] = 0x19, depending on the PCI Express controller in use.

Once the LTSSM is in the disabled state, the lanes enter electrical idle.

Because of this erratum, if the software immediately clears the Link Disable bit, the PCI Express RC controller may fail to detect this action from software.

Impact: The PCI Express RC controller may remain in the disabled LTSSM state and does not move to the detect state required for link retrain later.

Workaround: If software needs to disable and re-enable the link, the following procedure must be followed:

1. After software sets the PCI Express RC controller’s Link Control register [Link Disable] bit, it should poll the SCFG_PEXnMSCPORTSR register to confirm that the LTSSM is indeed in the disabled state, with read return value of “XMLHLTSSMSTATE = 0x19” for three times consecutively

2. Once the LTSSM is confirmed in the disabled state, software may proceed to clear the Link Control register [Link Disable] bit to bring the link out of the disabled state if it needs to.

Fix plan: No plans to fix
A-008329: PCI Express controller forwards received message TLPs to system application address space by default in RC mode

**Affects:** PCIe

**Description:** By default, the PCI Express RC controller forwards received message TLPs directly to the system application address space. This behavior may not always be desirable.

**Impact:** Directly forwarding the received message TLPs by the PCI Express RC controller to the system application address space could corrupt system memory or lead to a system hang.

**Workaround:** To avoid potential impact to the system due to undesirable direct forwarding of received message TLPs, you must clear bit 29 of the PCI Express RC controller’s configuration space offset 71Ch. The software must perform the following sequence in the order listed:

1. Perform a read to the PCI Express controller’s configuration space offset 71Ch. Save the read return value to a temporary location.
2. AND the temporary value obtained in Step 1 with DFFF_FFFFh and write back the result to the same configuration space offset 71Ch.

**Fix plan:** No plans to fix
A-008432: Optional programmable PCI Express iATU CFG shift feature is not supported for outbound configuration transactions in RC mode

Affects: PCIe

Description: The PCI Express RC controller’s implementation of the optional enhanced configuration access mechanism (ECAM) feature violates the PCI Express base specification requirement that all reserved fields should always be set to zero. The ECAM implementation requires the iATU to have the bus/device/function (BDF) address shifted 4 bits down from bits 31 to 27 so that the entire configuration space can be mapped into a 256 MB region, rather than requiring multiple address translation tables, or a 4 GB translation space. The BDF is then supposed to be shifted back up from bits 27:12 to 31:16 when forming the byte 8 and 9 of the outgoing configure TLP header.

Due to this erratum, the PCI Express RC controller’s iATU does not perform the EACM required address translation correctly when the CFG shift bit is set in an iATU entry.

Impact: The optional CFG shift feature for PCI Express ECAM cannot be used for outbound configuration transactions in RC mode.

Workaround: Software must set the CFG shift mode bit (bit 28) to zero in the Outbound iATU Region Control 2 Register. This register is located at offset 908h of the PCI Express RC controller address space. Additionally, use a regular address translation region for configure access.

Fix plan: No plans to fix
A-008822: Change the default AXI system error response behavior for PCI Express outbound non-posted requests

Affects: PCIe

Description: By default, when the PCI Express controller experiences an erroneous completion from an external completer for its outbound non-posted request, it always sends an OKAY response to the device’s internal AXI slave system interface. This is desirable for outbound configure transactions to prevent an unnecessary error response from propagating through higher-level system hierarchy, because erroneous completion is a commonly expected behavior during PCI Express bus scan.

However, such default system error response behavior cannot be used for other types of outbound non-posted requests. For example, the outbound memory read transaction requires an actual ERROR response when experiencing erroneous completion from an external completer, like UR completion or completion timeout.

Impact: The device’s higher level system hierarchy cannot detect the error condition when the PCI Express controller experiences an erroneous completion from the external completer for its outbound non-posted request. This is not the case for configure transactions.

Workaround: Write to the PCI Express controller’s configure space offset 8D0h with 0000_0001h during the pre-boot initialization (PBI) process.

Fix plan: No plans to fix
**A-008913: New A11 message request handling rules not supported**

**Affects:** PCIe

**Description:** As defined in the section 2.3.1, Request Handling Rules of the PCI Express Base Specification Rev 3.0, a received Message TLP can be treated as a valid request as long as the Message Code field of the TLP header is valid.

The Errata for the PCI Express Base Specification Rev 3.0 released on March 31, 2013 revises the above message request handling rules. Before a received message can be considered as a valid request, the new A11 request handling rules require checking the valid Message Routing subfield (Type [2:0] bits) and Msg/MsgD indication (Fmt [1:0] bits) within the Message TLP header in addition to checking the previous Message Code field.

The errata described above for the PCI Express Base Specification Rev 3.0 has been integrated into the PCI Express Base Specification Rev 3.1, released on October 8, 2014.

Nevertheless, since the PCI Express controller core follows the requirement of the PCI Express Base Specification Rev 3.0, it does not support the new A11 request handling rules defined in PCI Express Base Specification Rev 3.0 errata.

**Impact:** This might lead to inappropriate handling for some received Message TLP generated by some third-party link partner with invalid Message Routing subfield (type [2:0] bits) and Msg/MsgD indication (Fmt [1:0] bits) within the Message TLP header. For example:

- Inbound Message TLP with invalid Message Routing subfield (type [2:0] bits) within the TLP header is still treated as a valid message that is processed normally.
- Inbound Message TLP with invalid Msg/MsgD indication (Fmt [1:0] bits) within the Message TLP header is detected as malformed TLP and discarded. Since Message TLPs are posted TLPs, there is no UR completion TLP returned to link partner for such invalid message requests.

**Workaround:** None. The link partner should constrain itself to ensure that the Message TLPs sent out contain valid encoding for both the Message Routing subfield (type [2:0] bits) and Msg/MsgD indication (Fmt [1:0] bits), in addition to the Message Code field, within the Message TLP header.

**Fix plan:** No plans to fix
A-009000: Accessing the PCI Express controller's reserved CCSR space offset FFF0h to FFFFh is prohibited

Affects: PCIe

Description: According to PCI Express Base Specification Rev 3.0, accessing any reserved configuration space of a PCI Express controller should be completed with a return value of zero. Due to this erratum, accessing the PCI Express controller's reserved CCSR space offset FFF0h to FFFFh results in a system hang.

Impact: Accessing the PCI Express controller's reserved CCSR space offset FFF0h to FFFFh results in a system hang.

Workaround: None

Fix plan: No plans to fix
A-009142: PCIe EP mode is not supported

Affects: PCIe

Description: Documentation for this device states that both PCIe RC and EP modes are supported, but that is incorrect. PCIe EP mode is not supported.

Impact: PCIe EP mode is not supported.

Workaround: None

Fix plan: No plans to fix
A-009151: PCI Express controller does not correctly set the Link Autonomous Bandwidth Status and Link Bandwidth Management Status bits of the Link Status Register in RC mode

Affects: PCIe

Description: The Link Status Register Bit 15, Link Autonomous Bandwidth Status (LABS) and Bit 14, Link Bandwidth Management Status (LBMS) are only meaningful for PCI Express RC’s downstream port. The PCI Express base specification requires the RC downstream ports to:
- Set the LABS bit when the link width or speed is changed by downstream link partner for autonomous reasons only, not for reason of attempting to correct unreliable link operation
- Set the LBMS bit when the link width or speed is changed due to reliability issues

Due to this erratum, the PCI Express RC controller behaves incorrectly as below:
- Does not set the LABS bit when the link width or speed is changed by downstream link partner for autonomous reasons
- Does not set the LBMS bit when the link width or speed is changed due to reliability issues
- Set the LBMS bit when the link width or speed is changed by the downstream link partner for autonomous reasons, which should be the function of LABS bit

Impact: The system software cannot utilize the PCI Express RC controller's Link Status Register [LABS, LBMS] bits to determine the cause of the possible link width and/or speed change, especially for test and debug purpose.

Workaround: Software should not utilize the functionality of the PCI Express RC controller's LABS and LBMS bits of the Link Status Register.

Fix plan: No plans to fix
A-009518: PCI Express EP controller's non-D0 PowerState is overridden to D0 when any single or combination of the Bus Master Enable (BME) or Memory Space Enable (MSE) bits are enabled from disabled setting by host software

Affects: PCIe
Description: When the PCI Express EP controller's PMCSR [PowerState] is set to non-D0 (including D1, D2 or D3), host software’s update to any single or combination of the BME or MSE bits in EP’s Command Register should not cause the EP’s PowerState to settle at D0.

Due to this erratum, when the EP’s PowerState is at non-D0, if the host software enables BME and/or MSE bits from their disabled setting (writing an 1b to a bit when its previous value is 0b), the EP controller’s PowerState is overridden to D0.

Impact: Instead of remaining in Non-D0, PCI Express EP controller’s PMCSR [PowerState] bit field is overridden to D0 due to the host software’s action of enabling any single or combination of the BME or MSE bit fields in EP’s Command Register from their disabled setting. As the result, the link will exit from L1.

However the impact should be small, since host software should not update the EP controller’s Command Register when the EP is in non-D0 PowerState. Instead, it should update EP’s PMCSR [PowerState] first.

Workaround: When the PCI Express EP controller's PMCSR [PowerState] is in non-D0, host software should not issue configure cycles to write to EP Command Register's BME or MSE bit fields.

Fix plan: No plans to fix
A-009520:  PCI Express completion transaction layer packets (TLP) of a decomposed outbound memory read request must be returned in order

Affects: PCIe

Description: The PCI Express protocol allows completion TLPs associated with different memory read (MRd) request TLPs to pass each other and be returned out-of-order. If these MRd TLPs are split from a single internal MRd request, the requester is required to re-order the completion TLPs returned out of order.

When the PCI Express controller's Device Control Register [Max_Read_Size] is set to 000b (this means the maximum MRd request size is 128 bytes), any outbound MRd request from device's internal bus master with request size larger than 128 bytes is split into multiple smaller MRd TLPs by the controller to be sent out on the PCI Express link. After receiving these decomposed MRd TLPs, the completer may return the completion TLPs out of order. Due to this erratum, the PCI Express controller as a requester is unable to re-order the completion TLPs back when re-assembling the completion data.

Impact: When the PCI Express controller's Device Control Register [Max_Read_Size] is set to 128 bytes, if the completion TLPs returned are out of order for MRd TLPs split from a single internal MRd request with a size larger than 128 bytes, the controller is unable to re-order these completion TLPs when re-assembling the completion data. This results in data corruption.

There is no impact when the PCI Express controller's Device Control Register [Max_Read_Size] is set to either 001b (256 bytes) or 010b (512 bytes), since the size of MRd requests received by the controller is always 256 bytes or less, which is limited by the device's internal bus masters. Therefore, no decomposition occurs at the PCI Express controller level.

Workaround: Do not use 128 bytes for Max_Read_Size. Software should ensure that the PCI Express controller's Device Control Register [Max_Read_Size] is set to either 001b (256 bytes) or 010b (512 bytes).

Fix plan: No plans to fix
Secondary PCI Express extended capability is included in the capability structure linked list in Gen1 and Gen2 modes

Affects: PCIe
Description: The secondary PCI Express extended capability structure is only applicable for PCI Express controller operating at Gen3 and mainly used for the equalization process during link training at Gen3 speed.

Due to this erratum, the secondary PCI Express extended capability structure presents in the PCI Express capability structure linked list even when the controller operates at Gen1 or Gen2 speed.

Impact: The secondary PCI Express extended capability structure irrelevant to the current controller configuration speed may cause confusion to the PCI Express enumeration software, if the software cannot ignore this irrelevant capability structure presented in the linked list.

However, the impact should be negligible, since the Gen1 or Gen2 PCI Express controller should not initiate any link training process toward the Gen3 speed.

Workaround: Workaround procedure to overwrite the affected capabilities pointers is not feasible since the affected PCI Express capabilities pointer registers cannot be updated. The alternative approach described below can be adopted for the host system software running at RC side to ignore the irrelevant Secondary PCI Express Extended Capability structure presented in the linked list.

For the rest of the workaround section, the speed the controller configured to support refers to the maximum link speed that a PCI Express controller is configured to operate for a chosen SRDS_PRTCL_Sn setting within a RCW.

Regardless the PCI Express controller is configured as RC or EP, instead of directly using the information read from the configuration space registers, the host system software has to build its PCI Express capabilities linked list to ignore the irrelevant Secondary PCI Express Capability structure presented at the end of the linked list of this controller, located at configuration space offset 0x148.

The correct linked list should only contain the Advanced Error Reporting Capability structure for the non-Gen3-capable PCI Express controller.

Fix plan: No plans to fix
A-009719: Excessive correctable errors observed when the link's Active State Power Management (ASPM) is enabled

Affects: PCIe

Description: When the ASPM is enabled for the link, the PCI Express controller observes excessive correctable errors. The following registers may have some related correctable error status bits set as described below:
- Configure space offset 110h, Correctable Error Status Register [Replay timer timeout status, Bad DLLP status, Receiver error status]
- Configure space offset 7Ah, Device Status Register [Correctable Error Detected]

Impact: When the PCI Express controller has its link's ASPM enabled at any speed, there are excessive correctable errors reported that can impact performance due to the extra delay when going from L0s through recovery. The correctable errors may also generate interrupts if enabled.

Workaround: If PCI Express ASPM is really desired for the link that this PCI Express controller belongs to, interrupt generation from correctable errors should be disabled, given the system is able to tolerate the above mentioned impact.

If a system cannot tolerate the above mentioned impact from the excessive PCI Express correctable errors, the PCI Express ASPM should be disabled for the link that this PCI Express controller belongs to using one of the following options:
- The PCI Express ASPM policy is normally controlled globally by the operating system. If it's feasible, turn off the ASPM support globally in the OS for the whole PCI Express fabric that the device belongs to.
- If ASPM cannot be turned off globally, the ASPM support of the affected device's PCI Express link can be disabled by setting the PCI Express link Control register [ASPM_CTL] = 00b for both the NXP PCI Express controller and its link partner.

Fix plan: No plans to fix
**A-010315: Read access to an unselected PCI Express controller's space causes core(s) and platform to hang**

**Affects:** PCIe

**Description:** Read access to an unselected PCI Express controller's space causes core(s) and platform to hang. An unselected PCI Express controller is any PCI Express controller that is not selected by a specific SRDS_PRTCL_Sn option in RCW.

**Impact:** The core(s) and platform hang when reading an unselected PCI Express controller's space.

**Workaround:** First, prevent access to any unselected PCI Express controller by implementing the workaround procedure below such that any future unanticipated access will result in an exception. Then, configure the exception handler to identify and block such accesses to prevent the core(s) and platform from hanging.

To prevent access to any unselected PCI Express controller, write all zeros to the corresponding PCI Express controller bit fields in the related Configuration Security Level (CSL) register of Central Security Unit (CSU).

For example, if PCI Express controller 1 is unselected, perform the following:

1. While preserving other bit fields, write all zeros to CSU_CSL3[24:16]. This prevents access to the register space of PCI Express controller 1.
2. While preserving other bit fields, write all zeros to the CSU_CSL0[8:0]. This prevents the access to the I/O and memory space of PCI Express controller 1.

**For normal boot:** The workaround can be implemented in either PBI code or U-Boot. For U-Boot, the workaround should be implemented at the early stage of the boot code to avoid potential impact.

**For secure boot:** Because the CSU access is prohibited during PBI phase, implement the workaround in external secure boot code (ESBC) immediately after the PBI stage is complete (similar to the U-Boot implementation of normal boot process).

**Fix plan:** No plans to fix
A-008646: Reading configuration register RCPM_IPPDEXPCR1 return zero

Affects: PM
Description: RCPM_IPPDEXPCR1 register is used to disable the clock gating of the IPs, which can act as a wake-up source. This register always reads back as zero irrespective of what value is written into it.

Impact: Register value of RCPM_IPPDEXPCR1 cannot be confirmed.

Workaround: Do not read the register.

Fix plan: No plans to fix
A-007769: DDE error may be flagged incorrectly when qDMA address hold is used

**Affects:** qDMA

**Description:** When a qDMA destination address hold is used (Destination Descriptor CMD[DSEN] = 1 and DD[DSD] = 0), the Destination Descriptor Error (DDE) could be set incorrectly or not detected at all.

The address hold feature is mainly used for the FIFO interface.

**Impact:** For a qDMA job with address hold enabled, certain limitations and procedures must be followed.

**Workaround:** To use a qDMA job with address hold feature, the S/G table must not be used, and the job must be assigned to command queue 0 with strict priority. In addition, the job must be enqueued with a dummy DMA job immediately before it. The dummy job should be programmed to have the Compound S/G Format destination[LENGTH] set to 0 and source[LENGTH] set to an arbitrary non-zero value.

**Fix plan:** No plans to fix
Mixing command queue mode and legacy direct mode jobs may cause hang if an error is detected

**Affects:** qDMA

**Description:** qDMA allows jobs to be dispatched from the command queues and also by writing to legacy direct mode registers. If a legacy direct mode job is started at the same time a command queue job is dispatched with an error detected, the error may be reported incorrectly to the command queue dispatch unit, which causes the command queue dispatcher to hang.

Legacy direct mode is not affected by the hang.

**Impact:** The DMA may hang if command queue and legacy modes are used simultaneously.

**Workaround:** Do not operate command queue mode and legacy direct mode simultaneously.

Wait for qDMA to be idle before switching modes by reading command queue mode DSR[DB] or legacy direct mode DSLR[CB].

**Fix plan:** No plans to fix
A-007773: qDMA does not function correctly for certain combinations of source stride size and source stride distance

**Affects:** qDMA

**Description:** The qDMA read data engine allocates read data buffers in sizes of 256 bytes for store and forward between source and target. When source striding is enabled (source descriptor CMD[SSEN] = 1), the read data engine may fail to allocate buffers correctly because of a source stride size and source stride distance combination as follows:

1. Source stride size SD[SSS] modulo 256 is 1 byte to 32 bytes
2. Compound S/G[LENGTH] modulo SD[SSD] is greater than 32 bytes

When the read data engine fails to allocate sufficient read data buffers, it may cause data corruption for a subsequent qDMA job.

**Impact:** qDMA must avoid certain combinations of source stride size and source stride distance.

**Workaround:** Avoid the combinations of source stride size and source stride distance stated in this erratum description.

**Fix plan:** No plans to fix
A-007774: RTE error may be flagged on incorrect qDMA job

**Affects:** qDMA

**Description:** The qDMA read data engine may encounter a read transaction error (RTE) when reading from a source. If this occurs after the write data engine encounters an error, such as destination descriptor error (DDE) or write transaction error (WTE) for the same job, the RTE error may be flagged incorrectly for the 16th job following the one detecting the error. The 16th job does not complete as the result of this error.

**Impact:** An RTE error and WTE or DDE error together may corrupt a subsequent qDMA job by flagging an RTE error incorrectly.

**Workaround:** Check for an RTE error for the 16th job completed after the WTE/DDE error was detected, and resubmit the job to the command queue.

**NOTE**
Transaction errors are rare and normally happen because of a programming error or if an external source/target device becomes unavailable.

**Fix plan:** No plans to fix
A-010812: The qDMA controller fails to capture the command descriptor in DECCDR[0-3] registers when the enqueue rejection error bit DEDR[ERE] is set to 1

Affects: qDMA

Description: The qDMA controller has the option to add the completed transfer of a command descriptor (CD) to the status queue when CD[SER]=1. This status queue has a limited size and becomes full, if not processed by the software consumer of the status queue. When it becomes full, any attempt for further addition to the queue gets rejected. This causes an overflow and the last CD is returned to the qDMA controller. No additional enqueue attempts can be performed by the qDMA controller until the error has been cleared, indicating that the CD captured is handled by software.

The enqueue rejections occur because of the lack of processing by the consumer of the command descriptors in the status queue. This may be due to the size of the status queue, which is too small to be accounted for the delay in reaction to an exceeded queue threshold, or other means of determining a non-empty status queue. While increasing the status queue size may alleviate the occurrence of the enqueue rejections, it is not a complete solution.

A failure to handle the status enqueue rejection error in a timely manner may stall the qDMA controller from processing further transfers. The qDMA controller should respond by setting the enqueue rejection error bit of the DMA error detect register, DEDR[ERE]=1, and capturing the CD in error capture registers DECCDR[0-3]. However, due to this erratum, when an enqueue rejection occurs, the command descriptors are not captured in the error capture registers, DECCDR[0-3] and cannot be recovered.

Impact: The command descriptor lost during an enqueue rejection event cannot be recovered.

Workaround: The command descriptors (CD) cannot be captured in the event of an enqueue rejection, therefore the workaround is to ensure that the qDMA controller enqueue rejections do not occur.

The qDMA controller supports the flow control (XOFF) flowing from the status queue to the command queue(s) producing traffic. This flow control is initiated when an "enter XOFF watermark" is triggered as defined by the register, SQCCMR. Setting the watermark level in the register, SQCCMR[ENTER_WM], to a recommended value of 32 guarantees that no enqueue rejections occur.

NOTE
If the software operates within the remaining status queue entries, the application of the workaround has no impact on the qDMA controller performance.

Fix plan: No plans to fix
A-010840: Descriptors which are required to be enqueued in the status queue may be duplicated

Affects: qDMA

Description: When CD[SER] bit is set, the qDMA enqueues command descriptors to the status queue after it completes a job. Under the below conditions, there may be multiple command descriptors (CD) waiting to be written to the status queue in the memory:

- Memory congestions - Back pressure from writing the descriptors to the status queue in the memory.
- Last response return back-to-back - DMA completing the execution of two jobs back-to-back.

If these enqueue operations occur back-to-back, some descriptors will be written twice to the status queue in the memory, resulting in an error. The DMA transfer itself is not affected and gets completed only once.

Impact: Some descriptors will be written twice to the status queue in the memory, resulting in an error.

Workaround: For the applications with a single producer of the DMA jobs and a required status queue, the duplication can be avoided by:

- Setting the register bit DMR[SO].
- Using a single command queue for all transfers.

For the applications with multiple producers of the DMA jobs using multiple command queues and a required status queue, the software may capture the last processed descriptor from the status queue and compare it against the next completed descriptor. A match would indicate a duplication and the latter descriptor can be ignored. The 2-bit DD bit field of the CD can be used as an enumerator to differentiate between the CDs that are otherwise identical. The application of work around has no impact on the qDMA performance.

Fix plan: No plans to fix
A-008892: QUICC Engine's data and buffer descriptor cannot be stored in MURAM

Affects: QE

Description: MURAM access from QUICC Engine DMA has the following limitations on data structures or buffer descriptors:

- Must be multiples of 16 bytes
- Must start at a 16-byte aligned address

The data and buffer descriptors that are not aligned to 16 bytes cannot be read or written properly from MURAM.

Impact: QUICC Engine's data and buffer descriptors cannot be stored in MURAM; otherwise, the system might hang.

Workaround: Use other memories, such as DDR or OCRAM, to store QUICC Engine data and buffer descriptors.

Fix plan: No plans to fix
**A-008886:** Sometimes unexpected data is written to the external flash memory even though the underrun bit (QuadSPI_FR[TBUF]) is not set

**Affects:** QSPI

**Description:** While carrying out continuous writes from the Tx buffer to the flash memory, there may be scenarios when the buffer is empty for some duration and gets filled later. For example, QuadSPI_TBSR[TRBFL] changes from non-zero to zero and again to non-zero in the middle of a flash transaction. Such a case may trigger unexpected or wrong data to be written into the flash memory, even though the underrun bit (QuadSPI_FR[TBUF]) is not set.

**Impact:** A write to the flash memory may not work correctly.

**Workaround:** Break the flash page writes into smaller chunks of Tx FIFO size and fill the FIFO before the write is initiated. For example, for a flash page size of 128 bytes, two separate write transactions of 64 bytes each (Tx FIFO size) must be initiated. This ensures that, for a single continuous write, the Tx FIFO never becomes empty during the write transaction on the flash interface.

**Fix plan:** No plans to fix
A-009277: Parallel mode cannot be used on full memory map of dual die packages

Affects: QSPI

Description: When a QSPI works with dual die flashes, it is not possible to operate Flash A2 and B2 in parallel mode. This does not affect A1 and B1, which work correctly in parallel mode.

Impact: Parallel mode cannot be used on full memory map of dual die packages.

Workaround: If using two dual die packages, use parallel mode only on an A1 and B1 combination; read A2 and B2 in serial mode.

Fix plan: No plans to fix
A-009283:  Illegal accesses to SPI flash memory can result in a system hang

Affects:  QSPI

Description:  Under normal circumstances you can program the QuadSPI_LUTn using a read instruction so that the software reads data correctly from flash memory through the AMBA-AHB system bus (AHB). However, when programming the QuadSPI_LUTn as a non-read sequence, the flash memory does not send back any data and the system may hang, which requires a reset to recover. This is considered illegal programming. There is no time-out mechanism to recover from this scenario.

Impact:  The system must be reset for illegal programming.

Workaround:  Use a watchdog timer of 1 second. The start of this timer can be triggered before a read through AHB. If it expires, the system needs to be reset (set DCFG_CCSR_RSTCR[RESET_REQ]). The timer must be cleared when Quads Pi_SR[BUSY]=0.

Fix plan:  No plans to fix
A-009284: Incorrect data is returned in multiple flash use case when the read crosses the flash boundary

**Affects:** QSPI

**Description:** In a multiple flash use case, QuadSPI returns incorrect data when a single read transaction crosses the boundary between the flashes.

In serial mode, a flash boundary can occur between flash A1 and A2, A2 and B1 and B1 and B2.

In parallel mode this occurs if a single read begins in A1-B1 pair and continues into the A2-B2 pair.

**Impact:** Read could return wrong data if a read transaction crosses the boundary between the flashes.

**Workaround:** Software should ensure that no read transaction cross a flash boundary. This can be managed by using the following methods:

- Reduce the data fetch amount of the AHB buffers to 64 bit so that no prefetch occurs. This prevents any legal access from crossing the boundary.
- If prefetch is enabled, software should reserve a memory region with the size of the prefetch prior to the boundary.
- Ensure that the core's cache is not prefetching across the boundary.

**Fix plan:** No plans to fix
A-010284: Insufficient read data may be received in the Rx Data Buffer register

Affects: QSPI

Description: Data read from flash through QuadSPI using the Internal Peripheral Bus (IPS) interface may return insufficient data in the Receiver (Rx) Buffer Data (QuadSPI_RBDRn) register when the read data size of a flash transaction is programmed to be greater than 32 bytes.

Impact: Read from a flash through QuadSPI using the IPS interface may cause buffer overflow.

Workaround: For data size greater than 32 bytes, program the IP data transfer size in the IP configuration register (QuadSPI_IPCR[IDATSZ]) to be in multiples of 8 bytes.

Fix plan: No plans to fix
A-008402: Maximum PRD length (4MB) issue

**Affects:** SATA

**Description:** For a PRD entry, the maximum data length is 4 MB. AHCI standard specifies that for 4 MB size, PRD[DBC] must be set to 3F_FFFFh. Due to a logic error, 3F_FFFFh is misinterpreted by the device as zero length.

**Impact:** Commands with 4 MB PRD length entries fail if PRD[DBC] is set to the value according to AHCI standard spec.

**Workaround:** Set PRD[DBC] to 0 when creating a PRD entry for a maximum data transfer size of 4 MB.

**Fix plan:** No plans to fix
A-008407: SATA transaction causes incorrect CRC error

**Affects:** SATA  
**Description:** Normally, a small on-chip SRAM inside SATA controller has an ECC protection circuit. However, because of a logic problem, the ECC circuit generates an incorrect ECC error and is reported as a CRC error.

**Impact:** The small on-chip SRAM has no ECC protection.

**Workaround:** Disable ECC checking for the SRAM by writing to address 2022_0520h with the value 0002_0000h. The CRC checking for the frame works normally after ECC disabling.

**Fix plan:** No plans to fix
A-008588: SATA controller may hang when processing multiple 16-byte non-aligned PRD entries for read operations

**Affects:** SATA

**Description:** SATA controller may enter a hang state when multiple PRD entries not 16-byte aligned are processed. The non-alignment from either the address or the byte count can lead to a hang. The hang results in the command time out.

**Impact:** If the workaround is not implemented, performance is impacted due to the command time out.

**Workaround:** For applications that use multiple PRDs, software must force the PRD alignment. Both address and byte count in each PRD should be 16-byte aligned.

**Fix plan:** No plans to fix
A-009042: The device detection initialization sequence mistakenly resets some registers

**Affects:** SATA

**Description:** When setting the PxsCTL[DET] to 'b0001 to perform a device detection initialization sequence, the SATA controller mistakenly resets the parts of the PxCMD (PxCMD[27:26], PxCMD[1:0]) and PxIS registers.

**Impact:** Interrupt might stop working after the device detection initialization sequence.

**Workaround:** The software should read and store PxCMD and PxIS values before issuing the device detection initialization sequence. After the sequence is complete, software should restore the PxCMD and PxIS with the stored values.

**Fix plan:** No plans to fix
A-009185: The default Rx watermark value may be insufficient for some hard drives

**Affects:** SATA

**Description:** The PTC[RXWM] sets the watermark value for Rx FIFO. The default value 0x20 might be insufficient for some hard drives. If the watermark value is too small, a single-cycle overflow may occur and is reported as a CRC or internal error in the PxSERR register.

**Impact:** The default Rx watermark value might be insufficient and result in a false CRC or internal errors.

**Workaround:** Change PTC[RXWM] field at offset 0xC8 to 0x29. Do not change the other reserved fields of the register.

**Fix plan:** No plans to fix
A-010240: SATA interface in BIST-L mode fails

**Affects:** SATA

**Description:** The SATA interface fails in BIST-L mode (high CRC errors) during the receiver jitter tolerance test (RSG).

**Impact:** SATA interface fails in BIST-L mode operation, which results in SATA compliance test failure.

**Workaround:** None

**Fix plan:** No plans to fix
A-009102: Default SerDes setting may cause excess errors in SATA controller

Affects: SerDes
Description: The device has incorrect settings for receive electrical idle controls for SATA 6 Gbps operation. These are:

- LNmSSCR1[REIDL_TH_1] =101b
- LNmSSCR1[REIDL_EX_SEL_1] =001b
- LNmSSCR1[REIDL_ET_SEL_1] =110b

These settings may cause unreliable operation at 6 Gbps, and should be corrected.

The recommended settings are:

- LNmSSCR1[REIDL_TH_1] =000b
- LNmSSCR1[REIDL_EX_SEL_1] =000b
- LNmSSCR1[REIDL_ET_SEL_1] =000b

Impact: Incorrect setting of LNmSSCR1 may result in unreliable gen3 operation (character, disparity, CRC errors).

Workaround: Before enabling SATA operation, write 0050_2880h to LNmSSCR1 register for every lane using SATA protocol. The preferred mechanism for this is using PBI.

Fix plan: No plans to fix
A-006385: SEC watchdog timer does not prevent all cases of illogical descriptors from hanging DECOs

Affects: SEC
Description: The SEC block contains a watchdog timer designed to clear errors that have hung a DECO. Although mostly effective, there are cases in which poorly constructed descriptors can hang a DECO from which a watchdog cannot recover.

The SEC's programming model is based on the construction of descriptors, which include commands and, optionally, embedded keys and context. Users are expected to use the SEC's driver (SEC Descriptor Runtime Assembler) to build descriptors; however, because there is considerable innate flexibility in descriptor construction — even with the Descriptor Runtime Assembler — it is possible to create descriptors that appear to follow the rules of descriptor construction but still lead to errors, hangs, or corrupted data.

A few examples of descriptor constructions that can lead to hangs not cleared by the watchdog are:

- Ending a descriptor with a LOAD IMM command.
- Creating descriptors that command the PKHA to unload more data than the RAMs actually store.

Impact: Certain non-standard descriptor constructions can lead to errors, hangs, or corrupted data.

Workaround: To avoid this erratum, it is highly recommended that users model their descriptors after the examples provided in the NXP reference software. If there is a need to create a descriptor for which an example does not exist, do so based on the SEC Programmer's reference manual.

Fix plan: No plans to fix
A-008857: Anti-replay early-rollover error

Affects: SEC

Description: The anti-replay functionality is implemented as a protocol. It's designed to be used by other protocols, and is used by IPsec, DTLS, SRTP, MacSEC and WiMAX. However, this bug only affects the SRTP protocol. The functionality includes optional rollover checking of the sequence number. When rollover checking is enabled, it will incorrectly detect a rollover halfway… that is if it was supposed to detect a rollover at $2^{48}$ in SRTP, it detects, instead, at $2^{(48-1)}$. If the count is started at or near zero it is highly unlikely that a count of $2^{(48-1)}$ will ever be reached in practice.

Impact: The impact is minor. Our research indicates that even if SRTP were to run at 10x the fastest rates, it would take 47 years for the rollover error to issue prematurely when counts are started at or near zero. This defect is in all implementations of SEC back to ERA4, when the separate Anti-Replay functionality was first introduced as a stand-alone protocol.

Workaround: The only workaround for SRTP is to turn off anti-replay checking if a count of $2^{47}$ is ever expected. For the stand-alone protocol, the rollover check can be turned off and implemented more correctly by separate DECO commands.

Fix plan: No plans to fix
A-010442: SEC WPA2 (WiFi) protocol produces incorrect CCM nonce for QoS Data frames

Affects: SEC

Description: The QoS Control subfield TID is required to be included in creation of the CCMP Nonce. This subfield is used on a MPDU basis to mark the quality of service required for a given PDU. For example, a video webinar requires high quality of service, so associated MPDUs should be given higher priority than the MPDUs associated with a periodic check of the email server.

The support for the QoS Control field was added to WPA2 (Wifi) protocol in SEC era 6 with the assumed specification as a static TID, programmed into the Shared Descriptor PDB[Pri] field. This implementation makes it difficult for a single connection to support multiple QoS levels.

Impact: The QoS Data PDU calculation will not follow 802.11 standard. The SEC will follow SECRM. Any encapsulated QoS Data PDU will become encrypted and compute the authentication word incorrectly. No visible error within the SEC, but the decapsulator of the PDU will generate an ICV check error.

Any decapsulated QoS Data PDU will decrypt incorrectly, and the authentication word computed will not match the authentication word transmitted. As a result, SEC will generate an ICV check error. If QoS Data PDUs are not used, then this erratum does not apply.

If QoS Data PDUs are not used, then this erratum does not apply.

Workaround: A QoS Data PDU is marked by a QoS Control field in the PDU header. That QoS Control field contains a subfield called Traffic Identifier (TID). If the TID employed in the QoS Control field matches the value programmed to the PRI byte in the PDB, then the SEC will compute correctly. (This programmability of PDB[Pri] is the source of the workaround.)

This erratum can be worked around by any of the following methods:

- Software, prior to releasing the MPDU to SEC, creates a special byte that consists of the TID subfield from the PDU's MAC Header QoS Control Field. This byte contains TID in the upper 4 bits, and 0s in the lower 4 bits. This byte is then prepended to the frame presented to SEC. The Shared Descriptor contains a command, prior to the Protocol Operation command, to MOV the byte from the Data FIFO to the descriptor buffer offset 10 (0x79830A01).

- The Shared Descriptor PDB is prepared with the Pri field to contain the most common TID value. When the MAC Header contains a TID field that deviates from the programmed value, software prepares a 4-byte word that consists of MSbit set, and TID in bits 7-4 of the LSB. All other bits are zero. Software presents this word through Oman or AAP such that it is written into the DECO Datapath Override (DPOVRD) register. (The method here depends upon the device). The instructions are added to the descriptor to test the MSB of DPOVRD, and if set, then MOVE the LSB of DPOVRD to Descriptor Buffer offset 10 -- overwriting the Shared Descriptor PDB[Pri] field.

- This method requires an estimated 21 words of extra descriptor material, but no special per-PDU work on the part of software, and causes the DECO to:
  a. Test the MAC Header Frame Control field to see if there is a QoS Control Field present (if first byte matches 1xxx10xx).
  b. Test the MAC Header Frame Control field to see if there is an Address 4 field in the MAC Header and determines where in the MAC Header the TID is found.
  c. Mask the byte containing TID so that non-TID bits are zeroed, and move that value to PDB[Pri].
d. Prepare the input FIFO to contain the input frame as required by the WPA2 (WiFi) protocol operation Command.

**Fix plan:** No plans to fix
A-007731: Mixing 16-bit and 32-bit frame sizes in XSPI mode can cause incorrect data to be transmitted

Affects: SPI

Description: The Serial Peripheral Interface (SPI) features an Extended SPI (XSPI) mode supporting frames of up to 32 bits. When the XSPI mode is enabled, transferring a mixture of frames having a size up to 16 bits and those having a size greater than 16 bits can cause an incorrect data transmission to occur. This happens when the First In/First Out (FIFO) queue read pointers roll over, and a frame needs to be extracted from both the bottom and top of the FIFO when the frame size is greater than 16 bits.

This erratum is applicable only on the Transmit (Tx) side. No problem has been found on the Receive (Rx) side.

Impact: Incorrect data transmission can occur if frames up to 16 bits are mixed with frames greater than 16 bits in XSPI mode.

Workaround: Do not mix Tx data frames that have data sizes less or equal to 16 bits with data frames greater than 16 bits.

When Tx FIFO depth is even (even number of TXFR registers): For frames less than or equal to 16 bits, each Tx FIFO entry constitutes one frame. For frames greater than 16 bits, two entries of FIFO constitute one frame. Therefore, it does not cause the Tx FIFO pointer to roll over.

When Tx FIFO depth is odd (odd number of TXFR registers): For frames greater than 16 bits, send one Tx FIFO entry as a dummy frame (that is, a frame with no chip select but containing data) because the FIFO depth is odd. After this dummy frame transfer, even entries are left in the FIFO. Now, two FIFO entries each constitute one frame until the whole depth of FIFO is covered. Therefore, a dummy frame is needed after sending 
\[(\text{FIFO depth} - 1)/2\] frames (that is, after the whole FIFO depth has been covered).

Fix plan: No plans to fix
A-009776: Loading of shift register data into the receive FIFO following an overflow event

Affects: SPI

Description: 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 Receive FIFO bit in Module Configuration Register (MCR[CLR_RXF]) is asserted to clear the receive FIFO, the shift register data is wrongly loaded into the receive FIFO after the clear operation completes.

Impact: The receive FIFO content may be incorrect following an overflow event.

Workaround: Use one of the following workarounds:

• Avoid a receive FIFO overflow condition (SR[RFOF] should never be 0b1). To do this, monitor the receive FIFO counter field of the Status Register (SR[RXCTR]) which indicates the number of entries in receive FIFO and clear it before the counter equals the FIFO depth.
• 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.

Fix plan: No plans to fix
A-009895: Inconsistent addition to Transmit or Command FIFO if an entry is added while FIFOs are full

Affects: SPI

Description: In the Serial Peripheral Interface (SPI) module, the Transmit FIFO and Command FIFO are updated simultaneously when the PUSH TX FIFO Register (PUSHR) is written with a command and transmit data word. In the scenario where:

1. Frame size is configured to 16 bits or less by disabling any extended frame functionality;
2. Transmit FIFO is full (Transmit FIFO Fill Flag in Status Register (SR[TXFFF]) = 0b0); and
3. One entry each from both the Transmit FIFO and Command FIFO is read for next frame transfer.

If, at this time, a new entry is added to the FIFOs through PUSHR, only one of the two FIFOs is updated with the new data.

Impact: An addition to Transmit FIFO or Command FIFO may fail.

Workaround: Check the fill status of the transmit FIFO before attempting to add new entries. Write PUSHR only when transmit FIFO is not full (SR[TXFFF] = 0b1).

Fix plan: No plans to fix
A-008360: Reported temperature may indicate +4°C higher at range start

**Affects:** TMU

**Description:** The TMU uses four internal temperature ranges to cover the specified range 0°C to 125°C. The starting temperature of these ranges are 0°C, 38°C, 72°C, 97°C. If a temperature sensor reading from a remote site exactly matches the configured reading for one of these temperatures, TMU will report +4°C higher than actually measured. Note that it is still permissible to see a reported temperature of 0°C, 38°C, 72°C and 97°C, due to rounding of a measured remote sensor value to nearest temperature.

**Impact:** TMU reported temperature may indicate +4°C higher at select temperatures of 0°C, 38°C, 72°C and 97°C. This means a 4°C reported temperature may in reality be 0°C.

**Workaround:** The exact temperatures where a higher reported temperature may occur is known. Therefore any thresholds set should account for this discrepancy. For example a threshold of 101°C (97°C + 4°C) may be triggered when the temperature reaches 97°C. Thus is would be advisable to avoid a threshold setting between 98°C and 101°C. Same apply for the other affected temperatures.

**Fix plan:** No plans to fix
A-005697: Suspend bit asserted before the port is in Suspend state

Affects: USB

Description: The EHCI specification states the following in the SUSP bit description:

In the Suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.

In the USBDR controller, the PORTSCx[SUSP] bit changes immediately when the application sets it and not when the port is actually suspended.

Impact: Even though the behavior of the USBDR violates the EHCI specification statement, it does not cause any functional issue as, according to the EHCI specification, the application must wait for at least 10 milliseconds after a port indicates that it is suspended before initiating a port resume using the Force Port Resume bit.

Workaround: The software driver must follow the EHCI specification by waiting for at least 10 ms after setting the PORTSCx[SUSP] bit.

Fix plan: No plans to fix
A-008118: USB initial sequence impact for external ULPI PHY

**Affects:** USB

**Description:** USBDR controller soft reset has a dependency on PHY reset completion. In USBDR, the default value of the PORTSC[PTS] field is ULPI. Additionally, if the actual PHY connected is also ULPI, then the USBCMD[RST] bit should de-assert.

However, the USBDR controller requires a stable ULPI clock to detect ULPI PHY RESET completion. Due to an internal mux on ULPI and UTMI CLK, ULPI CLK may not reach USBDR in time. Hence, the reset might not de-assert. To enable this internal mux, write 1 to USB_CONTROL[USB_EN]. This results in ULPI CLK reaching the UDBDR core so the RESET bit de-asserts. This CLK dependency requires a change in the initialization sequence, that is, the soft reset can not be issued without setting USB_EN bit in the USB_CONTROL register.

**Impact:** Controllers having ULPI PHY may not come out of reset before USB_CONTROL[USB_EN] bit is selected.

**Workaround:** Do not check for USBCMD[RST] deassertion before writing 1 to USB_CONTROL[USB_EN].

**Fix plan:** No plans to fix
A-007463: After transaction error, controller switches control transfer data stage from IN to OUT direction

**Affects:** USB

**Description:** When a transaction error (defined in Section 4.10.2.3, "USB Transaction Error" of the xHCI Specification) occurs on the USB, the host controller reports this through a transfer event with the completion code “USB Transaction Error”. When this happens, the endpoint is placed in the Halted state. In response, software must issue a Reset Endpoint command to transition the endpoint to the Stopped state. In order to restart the transfer, the driver can perform either of the following:

- Ring the doorbell again, which restarts the transfer from where it stopped, or
- Issue a Set TR (Transfer Ring) Dequeue Pointer command for the endpoint to start the transfer from a different Transfer Ring pointer

Consider the following scenario:
1. The xHCI driver prepares a control transfer read to one of the device's control endpoints;
2. During the IN data stage, a transaction error occurs on the USB, causing a transfer event with the completion code "USB Transaction Error";
3. The driver issues a Reset Endpoint command;
4. The driver rings the doorbell of the control endpoint to resume the transfer.

In this scenario, the controller may reverse the direction of the data stage from IN to OUT. Instead of sending an ACK to the endpoint to poll for read data, it sends a Data Packet (DP) to the endpoint. It fetches the data from the data stage Transfer Request Block (TRB) that is being resumed, even though the data buffer is setup to receive data and not transmit it.

**NOTE**
This issue occurs only if the transaction error happens during an IN data stage. There is no issue if the transaction error happens during an OUT data stage.

**Impact:** When this issue occurs, the device likely responds in one of the following ways:

- The device responds with a STALL because the data stage has unexpectedly changed directions. The controller then generates a Stall Error transfer event, to which software must issue a Reset Endpoint command followed by a Set TR Dequeue Pointer command pointing to a new Setup TRB to clear the STALL condition.
- The device does not respond to the inverted data stage and the transaction times out. The controller generates another USB Transaction Error transfer event, to which software likely performs a USB Reset to the device because it is unresponsive.

It is not expected that any of these recovery steps will cause instability in the system because this recovery is part of a standard xHCI driver and could happen regardless of the defect.

Another possible system-level impact is that the controller attempts to read from the memory location pointed at by the Data Stage TRB or a Normal TRB chained to it. The buffer associated with this TRB is intended to be written by the controller, but the controller reads from it instead. Normally, this does not cause a problem. However, if the system has some type of memory protection where this unexpected read is treated as a bus error, it may cause the system to become unstable or to crash.
Workaround: If a USB Transaction Error occurs during the IN data phase of a control transfer, the driver must use the Set TR Dequeue Pointer command to either restart the data phase or restart the entire control transfer from the Setup phase.

Fix plan: No plans to fix
A-007467: When the LSP cache size is less than the number of TRBs needed for a microframe's worth of transfers, for an isochronous IN endpoint, the controller reads from invalid address

Affects: USB

Description: The DWC_USB3_CACHE_TRBS_PER_TRANSFER configuration parameter determines the number of TRBs the controller can cache per endpoint. When implementing an isochronous IN endpoint, the optimal programming model must ensure that all the packets to be transmitted in an interval fit into this many number of TRBs or less so that the controller can respond quickly to a burst of IN transfers from the host.

However, if the total number of packets to be sent in an interval does not fit into the DWC_USB3_CACHE_TRBS_PER_TRANSFER TRBs that the controller can cache, there may be a problem when the controller cannot respond to a multiple packet burst request from the host because the TRBs are not available. If this happens, the controller may read from an invalid system memory address because it is fetching a future microframe’s TRB into the same internal cache location that it was expecting to place the TRB for a packet that was supposed to be sent in the current microframe.

The consequence of this issue is that the unexpected read may cause a bus error if the memory is not allocated. If a bus error does not occur, the incorrect data is not transmitted and no failure is observed.

Impact: System-level impact: If the controller causes a bus error, the system may require unloading/reloading the device driver. Other systems may require a reboot of the system itself. If the unexpected read does not cause a bus error, there is no system-level impact.

Workaround: Use the following workarounds:
1. For each interval of isochronous IN data, use DWC_USB3_CACHE_TRBS_PER_TRANSFER or fewer number of TRBs to describe the packets to be transmitted during the interval.
2. For each data buffer that a TRB points to, allocate the same amount of memory. For example, in a system with three TRBs with sizes 10 bytes, 1 K, and 5 bytes, allocate 1 K for each of the three buffers that the TRBs point to. The BUFSIZ field can be left unchanged.

Fix plan: No plans to fix
A-007468: When the host extends an IN burst transaction, the device responds with both DP and NRDY on underrun

Affects: USB

Description: When all of the following conditions are met, the device controller may generate an unprompted NRDY response:
1. The host initiates an IN burst (Bulk or Interrupt) transaction with NumP greater than 1.
2. The device sets up the IN transfer.
3. The host also issues a ClearFeature(ENDPOINT_HALT) control transfer to the same IN endpoint.
4. On seeing the ClearFeature(ENDPOINT_HALT), the device application issues an EndTransfer command to the endpoint at the same time when the IN transfer is active on the USB. Internally, the EndTransfer causes an underrun on the packet being transmitted resulting in a Data Packet that fails CRC.
5. The host extends the original burst length using an ACK with NumP greater than the original NumP.
6. The ACK is delayed beyond the time that the underrun condition has been realized.

If all these conditions are met, the device may incorrectly respond with two packets: a DP with CRC error and an NRDY packet. Although an unwanted NRDY is sent, there is no system-level dead lock because the device correctly sends an ERDY for the next transfer.

Impact: This is a robustness fix. This condition is not expected in a system because the host issues a Stop Endpoint command to the IN endpoint to stop the transfer before it issues a ClearFeature(ENDPOINT_HALT) control transfer for the same IN endpoint. Even if this issue occurs, the system will recover.

Workaround: None

Fix plan: No plans to fix
A-007469: Short packet with EOB = 1 can cause delay on transfer event until ERDY is sent by device

Affects: USB

Description: In the following scenario:
- The host controller communicates with an IN endpoint on a SuperSpeed device.
- The host driver prepares a multi-TRB Transfer Descriptor (TD) for a transfer but only validates some of the TRBs by inverting their Cycle bit and does not validate the remaining TRBs.
- The host controller initiates an ACK to the endpoint.
- The device returns a short packet with the EOB flag in the Data Packets (DP) set to one, which results in the endpoint entering a flow-controlled state.
- The short packet is received in a TRB that is not the last TRB in the TD.
- The device does not transmit an ERDY.

After the short packet is received, if the host driver validates the rest of the TRBs in the multi-TRB TD by inverting their Cycle bit and ringing the doorbell, the controller might not generate transfer events for the newly validated TRBs. Depending on the ISP and IOC settings — and the usage of Event Data TRBs — this might result in the controller not generating any event for the short packet received.

Impact: The host driver times out and resets the endpoint/device in order to make it transition to a known state.

Workaround: Choose one of the following three workarounds.

Option 1:
This issue does not occur if a device writes EOB = 0 in short packets transmitted for IN endpoints. This is the normal behavior observed for all USB 3.0 devices.

Option 2:
When preparing multi-TRB TDs, the host driver must guarantee that the controller sees all of the TRBs as valid at the same time by validating them in reverse order (that is, first flip the Cycle bit in the Chain =0 TRB, then the previous one, and so on).

Option 3:
In the host driver, write ISP = 1 in all IN endpoint TRBs so that the driver is guaranteed to receive a transfer event when the device sends a short packet. In this situation, when the driver receives the SHORT_PACKET transfer event, it can assume the entire TD is complete. The rest of the TRBs generate transfer events when the device sends ERDY, potentially for a future transaction prepared through a different endpoint.

Fix plan: No plans to fix
A-007516: Device responds to ACK TP with bad device address

Affects: USB
Description: If:
- a USB 3.0 SuperSpeed IN (Control read data phase, Bulk IN, or Interrupt IN) transfer is in progress;
- the device receives an ACK Transaction Packet (TP) that is not an initiating ACK with incorrect device address as a response; and
- all other fields in the ACK TP are correct;

then the device responds to the ACK TP as if it is a valid packet. This issue does not happen in an ISOC IN transfer because ISOC transfers do not have response ACKs.

This scenario is highly unlikely because the USB 3.0 specification defines all SuperSpeed connections to be point-to-point. The hub (or the root hub) that forwards the packet from the host to the device must be malfunctioning in order to deliver the packet with an incorrect device address to the device. In addition, the TP packet has a CRC check that ensures data integrity. For this issue to occur, the device address field must be corrupted and the CRC for the packet must be recalculated with this corrupted field.

The consequence of this issue is that the device responds to the ACK TP as if it is a valid packet. This makes the device out-of-sync with the host (because of sequence numbers), and the device does not respond to any further operations on the affected endpoint. None of the other endpoints are affected.

Impact: System-level impact: Although this is a violation of the USB 3.0 specification, this condition is not expected in a USB system because the SuperSpeed packets are routed using the route string.

Workaround: No workaround from the device side under this situation. However, the system software at the USB host side can implement a timer to determine that some issue happened, and it can reset the endpoint after the timer times out.

Fix plan: No plans to fix
A-007517: Debug target may request U1/U2 entry after light reset before
SetFeature(U1_ENABLE/U2_ENABLE)

**Affects:** USB

**Description:** Consider that the Debug Host is connected to the controller, the controller is operating as the Debug Target, and the following sequence of events occur:
1. The Debug Host enables U1/U2 on the Debug Target by sending a SetFeature(U1_ENABLE) or SetFeature(U2_ENABLE) control transfer.
2. The driver on the Debug Target issues a light reset by writing USBCMD.LHCRST = 1.
3. The Debug Target disconnects from the Debug Host.
4. The Debug Host connects to the Debug Target again.

In this scenario, without the Debug Host issuing another SetFeature(U1_ENABLE) control transfer, the Debug Target may generate LGO_U1 LMPs.

The consequence of this problem is that the link may enter U1 or U2 unexpectedly. This is a violation of specification: the Debug Target should not request U1/U2 entry until the Debug Host enables U1 or U2 entry.

**Impact:** System-level impact: If the Debug Host rejects the U1/U2 requests from the Debug Target, there is no system-level impact because the link remains in U0. If the Debug Host accepts the U1/U2 request, there is still no system-level impact provided the Debug Host exits U1/U2 during the transfers (which is the expected behavior).

**Workaround:** No workaround needed because standard drivers, such as MS Windows, MCCI, and Linux, do not use light reset.

After issuing light reset, the driver on the Debug Target should write DCTL[12:9] (Init/Accept U1/U2 Enables) to 0000b. This disables U1/U2 entry regardless of the SetFeature requests from the Debug Host. In order to allow the Debug Host to control U1/U2 entry, DCTL[12:9] may be written to 1111b after the Debug Capability is in the Running state (DCR = 1).

**Fix plan:** No plans to fix
**A-007694: When software initiates a hardware LPM, do not use PORTHLPMC[HIRDM]**

**Affects:** USB

**Description:** Use the PORTHLPMC register for hardware LPM only. When the software initiates LPM by writing to the PORTSC[PLS] register, do not use the PORTHLPMC register values.

Currently, the issue has been observed under the following scenario:

- **PORTHLPMC[HIRDM] = 2'b01**
- Software initiates LPM
- The LPM token is expected to use the PORTPMSC[BESL] value, but it uses PORTHLPMC[BESLD] because PORTHLPMC[HIRDM] = 2'b01.
- If the LPM gets a NYET response, the LPM must not be retried. Instead, set the PORTSC[PLC] and set the PORTPMSC[L1S] to NYET. Due to this erratum, the host retries the LPM with the PORTPMSC[BESL] value because the PORTHLPMC[HIRDM] = 2'b01.

**Impact:** The intended BESL value from the software may be different than the actual BESL used in the LPM token. Due to this, the following cases may have issues.

**Case 1:**

The device gets and accepts a larger BESL value (from BESLD) than programmed. A good device is not expected to accept this value because the host usually programs the largest BESL value that the device can handle. If the actual BESL used is larger than this, the device must reject it. If a faulty device accepts it, then there could be data buffer overflow from the device side because the host has scheduled the transfers according to the BESL value. However the actual resume takes place for a longer duration.

**Case 2a:**

The device gets and accepts a smaller BESL value (from BESLD) than programmed. In this case, the device can set up the periodic buffers as soon as the device is up. But the host may not request the packets early enough because the expected BESL value is larger. The device eventually flushes the buffers, which causes data loss. However, the host driver usually programs the BESL.

**Case 2b:**

This is another case where the actual BESL can be smaller than the intended BESL value. The host does not reset the PORTHLPMC after a 2.0 device is disconnected. When a new device gets connected, all the port registers, including PORTHLPMC, are expected to be initialized properly. In this case, BESL is less than BESLD. However, if the driver doesn not initialize all the port registers, then Case 2a could occur.

**Workaround:** When the software initiates the LPM, ensure PORTHLPMC[HIRDM] = 0.

**Fix plan:** No plans to fix
A-007695: Device violates minimum USB turnaround time in FS mode

**Affects:** USB

**Description:** According to the UTMI specification, the controller must provide at least two CLks of delay after the utmi_linestate indicates idle (J) when transmitting a packet after receiving the previous one. However, in full-speed mode, when the device controller returns NAK/STALL for an OUT transaction, it does not meet the minimum turnaround time requirement. The device controller does not wait for the linestate to indicate idle and provides only one CLK delay after utmi_rxactive is de-asserted.

The USB 2.0 specification requires a turnaround delay of 2-bit time on DP/DM. The host drives/forces J on the line for 1-bit time after SE0. Therefore, the device controller and the PHY must contribute to the remaining 1-bit time (five CLks) after J.

According to the UTMI specification, the following must occur:
- Delay after 1-bit time idle (J) on DP/DM = 2 CLks
  - This is because the best-case PHY Rx End Delay is 17 CLks, which is comprised of 10 CLks of SE0, 5 CLks of J, and 2 CLks following J.
- Controller delay = 1 CLk
- PHY Tx Start Delay = 1 CLk

The delays contributed by the controller and the PHY add up to only 4 CLks (133.33 ns) instead of 5 CLks (166.66 ns) after idle (J) as required by the specification. Typically, in FS mode, the UTMI PHY takes about 4–9 CLks of Tx Start Delay. Therefore, no issue is observed in real systems.

**Impact:** There is no system-level impact because the delay contributed by the controller and PHY together exceeds the 1-bit time, which is enough to avoid the clash on DP/DM.

**Workaround:** No workaround is needed if the PHY’s Tx Start Delay is more than one CLk to meet USB 2.0 requirements. Synopsys PHY takes a minimum of four CLks for the Tx Start Delay in FS mode. Therefore, there is no clash on DP/DM even with this bug.

**Fix plan:** No plans to fix
A-007696: Controller generates transfer event with RING_UNDERRUN or RING_OVERRUN completion code, but ED = 1 and Data = 0h

Affects: USB
Description: When the host controller is connected to a device that has an isochronous endpoint, there is an expectation that the xHCI driver is able to provide valid TRBs quickly enough to maintain the traffic to the isochronous endpoint. (Section 4.14.2 in the xHCI specification describes this requirement.) The controller reports how early it needs the TRBs to be valid before the microframe in which the transfer takes place via the Isochronous Scheduling Threshold (IST) field of the HCSPARAMS2 register.

If the software is unable to provide TRBs quickly enough to the controller, it causes the traffic to the isochronous endpoint to either underrun (isochronous OUT endpoints, meaning that the controller cannot send valid data to the device in a given microframe) or overrun (isochronous IN endpoints, meaning that the controller cannot store the data that the device is sending in a given microframe). The controller reports this condition by generating a transfer event with the completion code set to "Ring Underrun" or "Ring Overrun." These transfer events are special because the TRB Pointer field is specified to be invalid and are ignored by the software. However, in some circumstances, if the software is using Event Data TRBs to report the status of TDs on an isochronous endpoint and a ring underrun or overrun occurs, the controller may generate a transfer event with the following fields:

- The Completion Code set to "Ring Underrun" or "Ring Overrun"
- The TRB Pointer field set to 0 (because it is invalid)
- The Event Data (ED) field set to 1
- When ED = 1, the xHCI specification states that the event should be generated by an Event Data TRB and the TRB Pointer field should contain a 64-bit value provided by the Event Data TRB.

Impact: System-Level Impact: The consequence of this problem depends on the host driver implementation.

If the driver interprets the completion code first and determines that it is a Ring Underrun or Overrun event, it ignores the TRB Pointer field and there is no consequence.

If the driver interprets the ED field first and uses the TRB Pointer field to look up an Event Data TRB based on the 64-bit value, it is possible that a memory access to location 0h occurs, which could cause a bus error and system failure.

The MCCI xHCI driver does not encounter this condition because it does not use Event Data TRBs for isochronous endpoints.

The Linux xHCI driver does not encounter this condition because it does not look at the ED bit within transfer events.

The Windows xHCI driver can encounter this condition because it uses Event Data TRBs for isochronous transfers and interprets the ED field before the completion code field.

Workaround: If the completion code is Ring Underrun or Ring Overrun, ignore the value in the TRB Pointer field.

Fix plan: No plans to fix
A-007697: xHCI debug: port status changes event not generated when USBCMD.Run/Stop = 0

Affects: USB

Description: When the xHCI Debug Capability operation is enabled (DCCTRL.DCE = 1), the DBC controller cannot generate a port status change event if the value of the USBCMD.run/stop bit of the host controller is 0. This is because the port status change event generator for DBC controller is controlled by an enable signal, which is connected to US BCMD.run/stop instead of DCCTRL.DCE = 1.

Impact: System level impact: When this happens, the port status change event for debug port is not generated in debug mode, and Debug Capability does not function.

Workaround: Write USBCMD.run/stop = 1 even if xHC is operating as a debug target.

Fix plan: No plans to fix
A-007839: Host core may hang if xHCI driver wrongly sets DCS bit to 0 in the Endpoint Context for Address device command while host's parameter check is disabled

Affects: USB

Description: This erratum occurs only if the parameter check is disabled. The parameter check is controlled by GUCTL1[2] (HParChkDisable) and is enabled by default. Unless it is disabled explicitly, this issue does not occur.

If the parameter check is disabled and the driver sets up an input context with the DCS bit in the Endpoint context as 0 for an Address Device command, then the host controller hangs while executing the command.

As per the xHCI specification, the DCS bit should be equal to 1 for an Endpoint context for an Address Device command.

Impact: If the parameter check is disabled and the issue occurs, the xHCI host hangs. The only solution is to reset the xHCI.

Workaround: Ensure the xHCI driver sets DCS = 1 and use the default behavior of the host core (do not disable the parameter check).

Fix plan: No plans to fix
A-008260:  Device controller accepts new OUT packets even when TRBs are not available

**Affects:** USB

**Description:** When the device application uses short packets on OUT endpoints, if more TRBs are present in a buffer descriptor chain, depending on certain conditions, the device might accept OUT data even though the device controller is not setup to accept OUT packets.

This issue occurs when all of the following conditions are met:

- TRBs are set up with CSP (Continue on Short Packet) = 1 for OUT endpoints
- An OUT short packet is received on USB
- There are more TRBs in the chain than the USB transfer size (fast forward TRBs)
- When the fast forward starts, only some TRBs in the chain are ready, but not all TRBs on the same TD (chain) are available in the controller
- TRB fetch happens close to the fast forward action and the internal cache RAM access for the OUT endpoint fetch logic gets delayed because of other traffic (probably on other endpoints)
- TRBs are not available for the new chain (that is, TRB cache is empty)
- USB packet arrives during the cache empty condition

When all the above conditions are met, the controller incorrectly accepts a new OUT packet, even though there is no TRB available to process it. When this happens, depending on the state of the previously cached TRB (described below), the controller might DMA the data or get hung.

- If the old TRB has a valid buffer size, then the controller performs DMA to the data address in the old TRB. This address is incorrect; however, if the system is not decoding the full address, then it might accept the packet.
- If the old TRB has zero buffer size, then the controller hangs trying to do a receive DMA of zero bytes

**Probability of Occurrence:** It is rare for all the above conditions to occur simultaneously, so this issue is not common. However, the condition could occur on a system with CSP mode and frequent fast-forwarding.

**Impact:** When this issue occurs and controller performs a receive DMA of zero bytes, then the device hangs, sending NAKs on the USB for the OUT endpoint. The host needs to re-enumerate and the device needs to be reset (software reset) to return to normal operation.

**Workaround:** Use one of the following workarounds:

- Not using CSP=1 mode. If CSP feature is not used, then the controller stops processing the TRB list after the current TRB is written back on a short packet, and waits for the SW to restart the list using the Start Transfer command. This can introduce minor performance impacts.
- When using CSP=1 mode, use single TRB per TD. If CSP feature is used, but only one TRB is prepared for a TD (Transfer Descriptor), then the controller on receiving a short packet, retires the current TRB/TD and goes to the next one in the list. There is no fast-forward (that is, skipping of TRBs) when using one TRB per TD.
- When using CSP=1 mode with multiple TRBs in a chain, do not allow the TRB cache to go empty. This can be ensured by having either a larger TRB cache or fewer TRBs in a TD so that the TRB cache can fit multiple TDs, prepare multiple TDs in advance, and issue update transfers upfront. If you make sure that the device does not fetch the TRBs
of the same chain when doing the fast-forward operation, it does not encounter the incorrect TRB cache empty condition during fast-forward operation, which is one of the required conditions for this issue to occur.

**Fix plan:** No plans to fix
A-008428: Device initiates low power when ACK pending for data packet

**Affects:** USB

**Description:** When the USB controller is configured as a USB device mode, the device initiates low power when an ACK is pending for a data packet (DP). When operating in SuperSpeed mode and when the internal condition for low power (u1/u2) is satisfied, the device initiates u1/u2 even though it has just received a DPH of the DP header (DPH). This causes the link to enter and exit low power before the device sends an ACK for the DP. This behavior can cause a transaction timeout on the host for the DP.

**Impact:** Depending on the host transaction timeout value, the host may timeout on the transaction and the host retries the transfer. If the same issue happens again, this could result in the host resetting the device and re-Enumerating.

**Workaround:** Disable USB_DCTL (InitU1Ena, InitU2Ena) bits. As a result, the device does not initiate low-power requests; however, it can still accept low-power requests from the host/hub and enter low power.

**Fix plan:** No plans to fix
A-008459: Device in 2.0 mode handles DATA PID error on SETUP data incorrectly

**Affects:** USB

**Description:** When the device operates in 2.0 mode and the SETUP data packet has incorrect DATA PID (! =DATA0), the device handles the data incorrectly and subsequent transactions are missed. This is a rare occurrence. When it does happen, the device eventually recovers.

The SETUP data packet always has the PID of DATA0. Due to an error on USB, if the DATA PID on the SETUP data arrives to the controller as non-DATA0, then the device times out (as expected). However, it handles this transaction incorrectly internal to the controller.

When this condition happens, the subsequent transaction is ignored and, if it is a good transaction, the data is missed/dropped for that transaction. The device retries the SETUP transfer and, this time, the device recovers and responds correctly.

**Impact:** It is rare for DATA PID to be corrupted for SETUP. Typically, when a control transfer is in progress and PID errors occur, the host retries the same setup until it is successful. This behavior eliminates any effects on device functionality because the device syncs after a successful setup.

**Workaround:** None

**Fix plan:** No plans to fix
A-008997: USB3 LFPS peak-to-peak differential output voltage adjustment settings

**Affects:** USB

**Description:** Low Frequency Periodic Signaling (LFPS) peak-to-peak differential output voltage test compliance fails using default transmitter settings. Software is required to change the transmitter signal swings to pass compliance tests.

**Impact:** LFPS peak-to-peak differential output voltage compliance test fails.

**Workaround:** Set SCFG_USBxPRM2CR[16:22] (USB3 Parameter 2 Control Register SCFG_USBxPRM2CR[PCSTXSWINGFULL]) = 7'b 1000_111, where x is number x of the USB controller/PHY.

**Fix plan:** No plans to fix
A-009007: USB3 PHY observes an intermittent failure in the receive compliance tests at higher jitter frequencies using the default register values

Affects: USB
Description: The receive compliance tests may fail intermittently at high jitter frequencies using the default register values.

Impact: The receive compliance test fails at the default register settings.

Workaround: Setting (RX_OVRD_IN_HI.RX_EQ) different (identified) value makes the Rx compliance test pass.

The following steps are needed to modify the RX_OVRD_IN_HI.RX_EQ value:
1. Write 1'b0 to RX_OVRD_IN_HI.RX_EQ_EN [offset 16'h200C: bit 6].
2. Write 1'b1 to RX_OVRD_IN_HI.RX_EQ_EN_OVRD [offset 16'h200C: bit 7].
3. Write a fixed value of 4 to RX_OVRD_IN_HI.RX_EQ [offset 16'h200C: bits 10–8]
4. Write 1'b1 to RX_OVRD_IN_HI.RX_EQ_OVRD [offset 16'h200C: bit 11].

Therefore, start with the default value.
1. Write RX_OVRD_IN_HI (0x0461200C) 0x0000
2. Write RX_OVRD_IN_HI (0x0461200C) 0x0080
3. Write RX_OVRD_IN_HI (0x0461200C) 0x0480
4. Write RX_OVRD_IN_HI (0x0461200C) 0x0C80

NOTE
Applying the transmit errata A-008997 is also required as receiver JTOL testing is done in loopback mode (Rx to Tx).

Fix plan: No plans to fix
A-009008: USB High Speed (HS) eye height adjustment

Affects: USB
Description: USB HS eye diagram fails with the default value at many corners, particularly at a high temperature (105°C).
Impact: USB HS eye diagram may fail using the default value.
Workaround: Set USB3PRM1CR[TXVREFTUNE](0x157_0070) = 4'b1001 for USB 1.
Fix plan: No plans to fix
A-009116: Frame length of USB3 controller for USB 2.0 and USB 3.0 operation is incorrect

Affects: USB

Description: Based on the USB specification, a (micro) frame length should be 125 u-sec. As per Section 5.2.4 in the xHCI specification, this requires the FLADJ register to have the value as 20h. Due to this erratum, the default value of this register in the USB 3.0 controller is 0h, which means the frame size is less than 125 u-sec. This causes the device to time out over a longer run.

The corresponding FLADJ register in xHCI is the GFLADJ (USB_BASE_ADDRESS + C630h) register.

Impact: There can be enumeration failure or read/write operation fails, which result in device timeout.

Workaround: Software must program the USB 3.0 register as part of the initialization sequence.


Fix plan: No plans to fix
A-009149: USB 2.0 protocol data pins USB_D_M and USB_D_P do not support IEEE 1149.1

Affects: USB

Description: For the USB 3.0 PHY with USB 2.0 protocol data pins, the USBN_D_M and USBN_D_P pins do not support IEEE 1149.1 and do not have boundary scan registers associated with the pins. Board-level interconnect for these pins cannot be done with IEEE 1149.1-2001 instructions, such as EXTEST.

Impact: Components with the USB 3.0 PHY are not compliant to IEEE 1149.1. The standard requires all digital data pins to have compliant 1149.1 boundary scan registers.

Unless noted elsewhere, the other components are compliant to the IEEE 1149.1-2001 standard.

Workaround: Test the USB 2.0 data pins board-level interconnects using other methods at the board level.

Fix plan: No plans to fix
A-009163: Device prohibits low power after sending LPF = 1 for isochronous IN

**Affects:** USB

**Description:** This issue happens in SS Device mode for isochronous IN endpoints. When the device does not have data to send during the initial isochronous IN request from the host, the device sends ZLP (zero length packet) with LPF = 1. Based on the reference manual, this causes the controller to allow low power (if all other conditions are met). However, the controller cannot accept low power requests from the host. When the host sends lgo_ux after ZLP (LPF = 1), the device responds with lxe. If there is no ping or no TP for isochronous IN for the next three uframes, the controller allows low power after three uframes of sending the ZLP. The controller does not allow low power only during the three uframes following the ZLP.

**Impact:** While software is preparing the data/TRB and the device controller is prefetching data, the device responds with ZLP DP for isochronous INs. During this time, the device link cannot go into low power mode, thereby, impeding power saving.

**Workaround:** Choose one of the following workarounds:

- Create dummy zero-length TRBs as soon as the host starts polling for isochronous INs, with the starting uframe to be the next or near-future interval/uframe and starting the transfer. This way the controller actually sends zero-byte packets on USB. (This is treated differently than the controller sending ZLP due to the packet not available.)
- After getting the XferNotREady event for the isochronous endpoint, send a DEPCFG command for the isochronous endpoint with "Modify" action with no change in parameters. This resets the internal u1_ep_timer to zero, which allows low power to succeed.

**Fix plan:** No plans to fix
A-009377: Incorrect value for S and E fields for ISOC OUT in Start-Split token

**Affects:** USB

**Description:** On the USB, a start-split token is used by the controller to send a high-speed (HS) data packet to a hub which in turn connects to a full-speed (FS) device. The HS payload is split into maximum chunks of 188 bytes in each microframe on the HS USB to be sent to the FS device through the hub. Each 188-byte payload is preceded by a Start-Split token with S and E field encoded to indicate the position of the 188-byte payload (start, middle, end, or only one payload).

On the xHCI, the driver prepares an isochronous (ISOC) TRB indicating the transfer length. The controller reads the TRB and decides on further processing before translating it to comply with the USB protocol. The issue manifests because the S and E encoding values are not properly decoded by the host controller when CH=1 (Chain) and ENT=1 (Evaluate Next TRB). The issue does not occur if the TRB has CH=0.

**Impact:** There may be a temporary glitch in the ISOC OUT application (for a FS device connected to a hub) because of the incorrect S and E field indication (for example, a glitch in the audio stream).

**Workaround:** None

**Fix plan:** No plans to fix
A-009668: Stop Endpoint command does not complete

**Affects:** USB

**Description:** This issue is observed in USB 2.0 mode when the USB 3.0 host controller is connected to a FS/LS device via a hub.

The host controller issues start-split (SSPLIT) and complete-split (CSPLIT) tokens to accomplish a split-transaction. A split-transaction consists of a SSPLIT token, token/data packets, CSPLIT token and token/data/handshake packets. A SSPLIT token is issued by the host controller to the hub followed by token/data/handshake packets. The hub then relays the token/data/handshake packets to the FS/LS device. Sometime later, the host controller issues a CSPLIT token followed by the same token/data/handshake packets to the hub to complete the split-transaction.

As an example scenario, when the xHCI driver issues an Address device command with BSR=0, the host controller sends SETUP(SET_ADDRESS) tokens on the USB as part of split-transactions. If the host controller receives a NYET response from the hub for the CSPLIT SETUP token, it means that the split-transaction has not yet been completed or the hub is not able to handle the split transaction. In such a case, the host controller keeps retrying the split-transactions until such time an ACK response is received from the hub for the CSPLIT SETUP token. If the split-transactions do not complete in a time bound manner, the xHCI driver may issue a Stop Endpoint command. The host controller does not service the Stop Endpoint command and eventually the xHCI driver times out waiting for the Stop Endpoint command to complete.

**Impact:** Stop Endpoint command does not complete.

**Workaround:** Instead of issuing a Stop Endpoint command, issue a Disable Slot Command with the corresponding Slot ID. Alternatively, you can issue an Address Device command with BSR=1.

**Fix plan:** No plans to fix
A-009798:  **USB high speed squelch threshold adjustment**

**Affects:**  USB

**Description:**  The default setting for USB high speed squelch threshold results in a threshold close to or lower than 100mV. This leads to a receiver compliance test failure for a 100mV threshold.

**Impact:**  If the errata is not applied, only the USB high speed receiver sensitivity compliance test fails, however USB data continues to transfer.

**Workaround:**  Complete the following writes:

```
SCFG_USB3PRM1CR (0157_0070h) write bit[6:8]=3'b000
```

These register writes allow the USB high speed receiver sensitivity compliance test to pass.

**Fix plan:**  No plans to fix
A-010127: Hot reset failure during U1/U2 entry for USB 3.0

Affects: USB
Description: When the xHCI driver issues Port Reset (PORTSC.PR = 1) after the link partners have already exchanged LGO_Ux, LAU, and LPMA link commands for U1/U2 entry, the USB 3.0 controller incorrectly enters the Recovery state to perform a Hot Reset (TS2 ordered sets and TS1/TS2 handshake) instead of entering into U1/U2.

The window for internally transitioning to U1/U2 after LAU and LPMA link commands is 16 mac3_clk (128 ns). This problem occurs only when the xHCI driver programs PORTSC.PR = 1 during the 128 ns window.

Impact: There is a delay for the USB 3.0 controller to receive Warm Reset from the xHCI driver.

Workaround: No workaround is required. If a Hot Reset fails because of a TS1/TS2 handshake timeout, a downstream port transitions to SS.Inactive, which generates a PORTSC.PLL interrupt to the xHCI driver. The xHCI driver then programs a Warm Reset to the controller.

Fix plan: No plans to fix
A-010129: USB 2.0 reset not driven while port is in the Resume state

**Affects:** USB

**Description:** This issue is applicable only to USB 2.0 ports in Host mode and assumes that PORTSC[PLS] = 1111b. If the xHCI driver resets the USB 2.0 port by programming PORTSC[PR] = 1, then the USB controller does not drive reset and does not generate an interrupt (PORTSC[PRC] = 1) while it is in the USB 2.0 Resume state.

**Impact:** The following are impacts for this erratum:
- The USB 2.0 port cannot be reset.
- The xHCI driver does not receive a port status change event interrupt (PORTSC[PRC] = 1).

**Workaround:** The xHCI driver should not program a USB 2.0 reset (PORTSC[PR] = 1) while in the Resume state. When the xHCI driver is ready to program a USB 2.0 reset, check the PORTSC[PLS] bit and only program PORTSC[PR] = 1 when PORTSC[PLS] is not set to 1111b.

**Fix plan:** No plans to fix
A-010131: U3 request gets dropped when controller tries U1-to-U2 entry

**Affects:** USB

**Description:** This erratum is applicable for the USB 3.0 Super Speed host mode operation. When the U2 timer expires while in U1 mode, the USB 3.0 controller completes a U1->U2 entry operation lasting three mac3_clk (24 ns). If the xHCI driver issues a U3 request during this operation, the controller drops this request.

**Impact:** The controller ignores the request when the xHCI driver programs the U3 entry (PORTSC.PLL = U3). The occurrence of this issue is rare because of the 24 ns window.

**Workaround:** The xHCI driver must include the following steps:
1. Before initiating U3 entry, save PORTPMSC.
2. Disable U2 entry by programming PORTPMSC[U2 Timeout] = h'FF.
3. After U3 entry, re-enable the U2 timer by programming PORTPMSC with the value saved in Step 1.

**Fix plan:** No plans to fix
A-010151: Unreliable receiver detection in low power P3 mode

Affects: USB

Description: The USB 3.0 controller enables the Receiver (Rx) Detection feature in low power mode 3 (P3 mode). However, USB 3.0 PHY does not reliably support receiver detection in P3 mode. Therefore, some USB 3.0 devices are not detected reliably in Super Speed mode.

This erratum does not cause a compliance issue, because the receiver detection in P3 mode is beyond the PHY Interface for PCI Express and USB 3.0 (PIPE) specification which only requires receiver detection in power mode 2 (P2 mode).

Impact: Some USB 3.0 pen devices cannot be detected reliably in Super Speed mode.

Workaround: Set GUSB3PIPECTL[DisRxDetP3]\=1 to configure USB 3.0 in P2 mode for receiver detection.

Fix plan: No plans to fix
<table>
<thead>
<tr>
<th>Changed Orderable Part#</th>
<th>Customer Part#</th>
<th>Changed Part #</th>
<th>Changed Part Description</th>
<th>Package Outline</th>
<th>Package Name</th>
<th>Status</th>
<th>New Orderable Part#</th>
<th>New part #</th>
<th>New part description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1021AXE7KQB</td>
<td>LS1022AXE7KQB</td>
<td>935314612557</td>
<td>LS1022AXE7KQB</td>
<td>LS1022A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1022AXE7KQB</td>
<td>LS1022AXE7KQB</td>
<td>935324055557</td>
<td>LS1022AXE7KQB</td>
<td>LS1022A-XT WE 600/1000</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1021AXE7HNB</td>
<td>LS1021AXE7HNB</td>
<td>935347136557</td>
<td>LS1021AXE7HNB</td>
<td>LS1021A-XT 1000/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1022AXE7HNB</td>
<td>LS1022AXE7HNB</td>
<td>935347135557</td>
<td>LS1022AXE7HNB</td>
<td>LS1022A-XT 1000/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313495557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313494557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313493557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313492557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313491557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313490557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313489557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313488557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313487557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313486557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313485557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313484557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313483557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313482557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LS1020AXE7KQB</td>
<td>LS1020AXE7KQB</td>
<td>935313481557</td>
<td>LS1020AXE7KQB</td>
<td>LS1020A-XTWE 1G/1600/300</td>
<td>FBA3525</td>
<td>FS</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>