*Errata*

# *AM64x/AM243x Processor Silicon Revision 1.0, 2.0*

**ABSTRACT**

This document describes the known exceptions to the functional specifications (advisories). This document may also contain usage notes. Usage notes describe situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness.

## Table of Contents

**1 Usage Notes and Advisories Matrices**.....2
**2 Silicon Usage Notes and Advisories**.....5
**Revision History**.....43

# 1 Usage Notes and Advisories Matrices

Table 1-1 lists all usage notes and the applicable silicon revision(s). Table 1-2 lists all advisories, modules affected, and the applicable silicon revision(s).

**Table 1-1. Usage Notes Matrix**

| ID | DESCRIPTION | SILICON REVISIONS AFFECTED | |
|---|---|---|---|
| | | AM64x/AM243x 1.0 | AM64x/AM243x 2.0 |
| Package | i2287 — Package Pin Assignment Difference between SR1.0 and SR2.0 | YES | YES |
| DDR | i2330 — DDRSS Register Configuration Tool Updates | YES | YES |
| OSPI | i2351 — OSPI: Controller does not support Continuous Read mode with NAND Flash | YES | YES |
| PLL | i2424 — PLL: PLL Programming Sequence May Introduce PLL Instability | YES | YES |

**Table 1-2. Advisories Matrix**

| MODULE | DESCRIPTION | SILICON REVISIONS AFFECTED | |
|---|---|---|---|
| | | AM64x/AM243x 1.0 | AM64x/AM243x 2.0 |
| BCDMA | i2431 — BCDMA: RX Channel can lockup in certain scenarios | YES | YES |
| BCDMA | i2436 — BCDMA: BCDMA RX_IGNORE_LONG setting in RX CHAN CFG register doesn't work | YES | YES |
| Boot | i2328 — Boot: USB MSC boots intermittently | YES | YES |
| Boot | i2257 — xSPI boot mode redundant image boot failure | YES | NO |
| Boot | i2307 — Boot: ROM does not properly select OSPI clocking modes based on BOOTMODE | YES | YES |
| Boot | i2306 — Boot: ROM does not turn off internal termination resistors in SERDES | YES | NO |
| Boot | i2363 — Boot: Peak voltage transitions may exceed PCIe spec during PCIe boot | YES | YES |
| Boot | i2366 — Boot: ROM does not comprehend specific JEDEC SFDP features for 8D-8D-8D operation | YES | YES |
| Boot | i2371 — Boot: ROM code may hang in UART boot mode during data transfer | YES | YES |
| Boot | i2413 — Boot: HS-FS ROM boots corrupted ROM boot image | YES | YES |
| Boot | i2414 — Boot: Ethernet PHY Scan and Bring-Up Flow doesn't work with PHYs that don't support Auto Negotiation | YES | YES |
| Boot | i2415 — Boot: UART Backup Boot Authentication Failure w/ xSPI Primary Boot Mode | YES | YES |
| Boot | i2417 — Boot: GPMC NAND configured to slower clock speed | YES | YES |
| Boot | i2418 — Boot: Secure ROM Panic due to Certificate Info not present | YES | YES |
| Boot | i2419 — Boot: When disabling deskew calibration, ROM does not check if deskew calibration was enabled | YES | YES |
| Boot | i2420 — Boot: XSPI Boot time is not consistent in SFDP mode | YES | YES |
| Boot | i2422 — Boot: ROM timeout for MMCSD filesystem boot too long | YES | YES |
| Boot | i2423 — Boot: HS-FS ROM applies debug access restrictions to all address space covered by the efuse controller firewall | YES | YES |
| Boot | i2435 — Boot: ROM timeout for eMMC boot too long | YES | YES |
| Boot | i2482 — Boot: ROM does not provide enough clocks during SD card initialization | YES | YES |
| CBASS | i2207 — CBASS: Command Arbitration Blocking | YES | YES |
| CBASS | i2235 — CBASS Null Error Interrupt Not Masked By Enable Register | YES | YES |
| CPSW | i2184 — CPSW: IET express traffic policing issue | YES | YES |
| CPSW | i2185 — CPSW: Policer color marking issue | YES | YES |
| CPSW | i2208 — CPSW: ALE IET Express Packet Drops | YES | YES |
| CPSW | i2331 — CPSW: Device lockup when reading CPSW registers | YES | YES |
| CPSW | i2401 — CPSW: Host Timestamps Cause CPSW Port to Lock up | YES | YES |
| DebugSS | i2317 — Debug is not available after TRST pin de-assertion when device pin EMU1 is repurposed as MCU_OBSCLK0 | YES | NO |
| DEBUG | i2283 — Restrictions on how CP Tracer Debug Probes can be used | YES | YES |
| DEBUG | i2291 — Debug: ATB Hang during Cortex-A53 trace | YES | YES |

**Table 1-2. Advisories Matrix (continued)**

| MODULE | DESCRIPTION | SILICON REVISIONS AFFECTED | |
|---|---|---|---|
| | | AM64x/AM243x 1.0 | AM64x/AM243x 2.0 |
| DDR | i2232 — DDR: Controller postpones more than allowed refreshes after frequency change | YES | YES |
| DDR | i2244 — DDR: Valid stop value must be defined for write DQ VREF training | YES | YES |
| DDR | i2259 — DDR: Possibility of starvation of low priority commands | YES | YES |
| DDR | i2274 — DDR: Including DDR in BSCAN causes current alarm on the DDR supply | YES | NO |
| DDR | i2160 — DDR: Valid VRef range must be defined during LPDDR4 Command Bus Training | YES | YES |
| DMA | i2285 — BCDMA: Blockcopy Gets Corrupted if TR Read Responses Interleave with Source Data Fetch | YES | NO |
| DMA | i2320 — BCDMA and PKTDMA: Descriptors and TRs required to be returned unfragmented | YES | NO |
| DMSC | i2245 — DMSC: Firewall Region requires specific configuration | YES | YES |
| ECC_AGGR | i2049 — ECC_AGGR: Potential IP Clockstop/Reset Sequence Hang due to Pending ECC Aggregator Interrupts | YES | YES |
| GPMC | i2313 — GPMC: Sub-32-bit read issue with NAND and FPGA/FIFO | YES | NO |
| Internal Diagnostic Modules | i2103 — Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors | YES | YES |
| Interrupt Aggregator | i2196 — IA: Potential deadlock scenarios in IA | YES | YES |
| ICSSG | i2305 — ICSSG: PRU RAM WRT during active FDB lookup write data corruption | YES | YES |
| ICSSG | i2368 — ICSSG: Device lockup when reading registers in ICSSG module | YES | YES |
| ICSS | i2433 — ICSS: Reading the 64-bit IEP timer does not have a lock MSW logic when LSW is read | YES | YES |
| ICSS | i2434 — ICSS: IEP RX SOF and TX SOF capture register issue in 32-bit shadow mode when nano seconds part is zero | YES | YES |
| JTAG | i2228 — JTAG: TAP used by Debuggers may be inaccessible if TRSTn device pin is never asserted | YES | NO |
| MCAN | i2279 — MCAN: Specification Update for dedicated Tx Buffers and Tx Queues configured with same Message ID | YES | YES |
| MCAN | i2278 — MCAN: Message Transmit order is not ensured from dedicated Tx Buffers configured with same Message ID | YES | YES |
| MDIO | i2329 — MDIO: MDIO interface corruption (CPSW and PRU-ICSS) | YES | YES |
| MMCSD | i2312 — MMCSD: HS200 and SDR104 Command Timeout Window Too Small | YES | YES |
| OSPI | i2189 — OSPI: Controller PHY Tuning Algorithm | YES | YES |
| OSPI | i2303 — OSPI: OSPI0_LBCLK0 pin has input floating by default | YES | NO |
| OSPI | i2249 — OSPI: Internal PHY Loopback and Internal Pad Loopback clocking modes with DDR timing inoperable | YES | YES |
| OSPI | i2383 — OSPI: 2-byte address is not supported in PHY DDR mode | YES | YES |
| PCIe | i2236 — PCIe: SERDES output reference clock cannot be used | YES | NO |
| PCIe | i2241 — PCIe: The SerDes PCIe Reference Clock Output can exceed the 5.0 GT/s Data Rate RMS jitter limit | YES | NO |
| PCIe | i2256 — PCIe: MSI or MSI-X does not trigger interrupt if address is not aligned to 8-bytes | YES | NO |
| PCIe | i2243 — PCIe: Timing requirement for disabling output refclk during L1.2 substate is not met | YES | YES |
| PCIe | i2326 — PCIe: MAIN_PLLx operating in fractional mode, which is required for enabling SSC, is not compliant with PCIe Refclk jitter limits | YES | YES |
| POK | i2277 — POK: De-Glitch (filter) is based upon only two samples | YES | NO |
| PRG | i2253 — PRG: CTRL_MMR STAT registers are unreliable indicators of POK threshold failure | YES | YES |
| PSIL | i2138 — PSIL: Configuration accesses and source thread teardowns may cause data corruption | YES | YES |
| RAT | i2062 — RAT: Error Interrupt Triggered Even When Error Logging Disable Is Set | YES | YES |
| USART | i2310 — USART: Erroneous clear/trigger of timeout interrupt | YES | YES |
| USART | i2311 — USART: Spurious DMA Interrupts | YES | YES |

**Table 1-2. Advisories Matrix (continued)**

| MODULE | DESCRIPTION | SILICON REVISIONS AFFECTED | |
|---|---|---|---|
| | | AM64x/AM243x 1.0 | AM64x/AM243x 2.0 |
| USB | i2091 — USB: USB 2.0 PHY hangs if received signal amplitude crosses squelch threshold multiple times within the same packet | YES | NO |
| USB | i2134 — USB: USB 2.0 Compliance Receive Sensitivity Test Limitation | YES | NO |
| USB | i2409 — USB: USB2 PHY locks up due to short suspend | YES | YES |

## 1.1 Devices Supported

This document supports the following devices:

- AM64x
- AM243x

Reference documents for the supported devices are:

- AM64x/AM243x Processors Technical Reference Manual (SPRUIM2)
- AM64x Processors Datasheet (SPRSP56)
- AM243x Processors Datasheet (SPRSP65)

## 2 Silicon Usage Notes and Advisories

This section lists the usage notes and advisories for this silicon revision.

### 2.1 Silicon Usage Notes

#### *i2287* *Package Pin Assignment Difference between SR1.0 and SR2.0*

**Details**

Two package terminals will be assigned new signal functions as the device transitions from Silicon Revision 1.0 (SR1.0) to Silicon Revision 2.0 (SR2.0).

- J15
  - SR1.0: Assigned to VDDS_MMC0 (MMC0 PHY IO supply). ADC0_REFP is internally connected to VDDA_ADC0
  - SR2.0: Assigned to ADC0_REFP (ADC0 positive reference).
- J16
  - SR1.0: Assigned to VSS (Ground). ADC0_REFN is internally connected to VSS
  - SR2.0: Assigned to ADC0_REFN (ADC Negative Reference)

Here is the guidance on using SR2.0 devices on existing boards, and recommendations for new board designs.

- When installing SR2.0 device on existing SR1.0 designed boards
  - For boards with VDDA_ADC0 and VDDS_MMC0 connected to the same power supply
    - An SR2.0 device may experience reduced ADC0 performance
      - This can occur because ADC0_REFP will be connected to the VDDS_MMC0 digital supply which will couple noise directly into ADC0 reference.
  - For boards with VDDA_ADC0 and VDDS_MMC0 connected to different power supplies
    - Potential applied to VDDS_MMC0 must never exceed the potential applied to VDDA_ADC0.
      - This condition of operation is required because the new SR2.0 pin assignment connects ADC0_REFP to the VDDS_MMC0 digital power supply and the internal ESD protection circuit associated with ADC0_REFP is referenced to VDDA_ADC0, which may be sourced from a separate analog power supply. This requires the potential applied to ADC0_REFP to be less than or equal to potential applied to VDDA_ADC0. This power sequencing requirement for ADC0_REFP relative to VDDA_ADC0 is passed along to VDDS_MMC0 when a SR2.0 device is installed on a PCB previously designed for a SR1.0 device. Therefore, it is important to confirm this condition of operation is not violated during power-up and power-down when VDDS_MMC0 and VDDA_ADC0 are sourced from different power supplies.
    - If ADC0 is not being used, where VDDA_ADC0 is connected to VSS as described in the Connections for Unused Pins section of the datasheet, a SR2.0 device cannot be installed
      - This is not allowed because the potential applied to ADC0_REFP can never exceed the potential applied to VDDA_ADC0, and ADC0_REFP will be connected to VDDS_MMC0 when a SR2.0 device is installed on a PCB previously designed for a SR1.0 device.
- New board design which will support both SR1.0 and SR2.0 devices
  - Place a zero ohm resistor on the back-side of the PCB, under the BGA array, connected between J13 and J15.
    - When SR1.0 is installed, do not populate this resistor
    - When SR2.0 is installed, populate this resistor.

***i2287*** (continued) ***Package Pin Assignment Difference between SR1.0 and SR2.0***

  - An additional high-frequency de-coupling capacitor should be placed on the back-side of the PCB, under the BGA array, connected between J15 and J16.
  - Connect J16 directly to VSS power plane
- New board design which will only support SR2.0 devices
  - Connect J13 to J15
  - Connect J16 directly to VSS power plane
  - An additional high-frequency de-coupling capacitor should be placed on the back-side of the PCB, under the BGA array, connected between J15 and J16.
  - See Note

**Note**

A SR1.0 device should not be installed on a PCB *only* designed to support SR2.0 devices. This PCB would connect J13 and J15 to the ADC0 analog power supply, and installing a SR1.0 device that internally connects J15 to K14 would short the VDDA_ADC0 analog power supply to the VDDS_MMC0 digital power supply.

## *i2330* *DDRSS Register Configuration Tool Updates*

**Details:**

The DDR Register Configuration Tool provides custom register settings based on system level details such as the architecture (density, data width, ranks) of the DDR device, frequency of operation, and IO settings determined through board simulations. This tool may be updated over time to support new devices and/or features, fix issues identified with the tool, and most importantly, capture work-arounds of errata or recent updates identified to register calculations which improve performance, signal integrity, or timing relationships between signals.

**Workaround(s):**

In order to ensure that parameters are set appropriately based on lessons learned and reduce the risk of functional failure, the latest DDR register configuration tool should always be used to generate register values. As the DDR register configuration tool can periodically be updated, the revision history of the tool should be reviewed and evaluated whether tool changes apply to existing systems. When applicable, the configuration of an existing system should be updated appropriately. The latest version of the tool can be found at http://dev.ti.com/sysconfig, and choosing DDR Configuration under Software Product drop down for the applicable device that is being used.

## *i2351* *OSPI: Direct Access Controller (DAC) does not support Continuous Read mode with NAND Flash*

**Details:**

The OSPI Direct Access Controller (DAC) doesn’t support Continuous Read mode with NAND Flash since the OSPI controller can deassert the CSn signal (by design intent) to the Flash memory between internal DMA bus requests to the OSPI controller.

The issue occurs because “Continuous Read” mode offered by some OSPI/QSPI NAND Flash memories requires the Chip Select input to remain asserted for an entire burst transaction.

The SoC internal DMA controllers and other initiators are limited to 1023 B or smaller transactions, and arbitration/queuing can happen both inside of the various DMA controllers or in the interconnect between any DMA controller and the OSPI peripheral.

***i2351*** (continued) ***OSPI: Direct Access Controller (DAC) does not support Continuous Read mode with NAND Flash***

This results in delays in bus requests to the OSPI controller that result in the external CSn signal being deasserted.

NOR Flash memories are not affected by CSn de-assertion and Continuous Read mode works as expected.

**Workaround(s):**

Software can use page/buffered read modes to access NAND flash.

***i2424*** ***PLL: PLL Programming Sequence May Introduce PLL Instability***

**Details:**

PLL programming sequence has been changed to ensure that, if used, all calibration fields are configured prior to enabling the PLL calibration. In addition to the change to the control of the calibration logic, other changes are implemented so that PLL parameters are unchanged while the PLL is enabled.

When in integer mode, the software enables the PLL calibration feature on calibration-capable PLLs. The previous software adjusted calibration modes after CAL_LOCK was asserted. These writes have been observed to cause a loss of PLL lock on some devices. Additionally, even on susceptible devices, the loss of lock is intermittent, but when the loss occurs, dependent circuitry runs at an incorrect frequency; this wrong frequency can show up as slow algorithm execution or communication failures.

Limit on the impact: The calibration logic cannot be used when the PLL is in fractional mode. Therefore, PLLs that are programmed to use fractional mode should not see a failure related to the calibration programmation. Nevertheless, because of the change to the full PLL sequence, the new software is recommended for all users.

**Workaround(s):**

Do not use clk_pll_16fft_cal_option4() in SYSFW. Ensure to use updated PLL programming sequences in SDK v10.0 or later when performing any PLL configuration change.

## 2.2 Silicon Advisories

### *i2049* — *ECC_AGGR: Potential IP Clockstop/Reset Sequence Hang due to Pending ECC Aggregator Interrupts*

**Details:**

The ECC Aggregator module is used to aggregate safety error occurrences (which are rare) and generate interrupts to notify software. The ECC Aggregator provides software control over the enabling/disabling and clearing of safety errors interrupts.

When software is performing a clockstop/reset sequence on an IP, the sequence can potentially not complete because the IP's associated ECC Aggregator instance is not idle. The ECC Aggregator idle status is dependent upon any pending safety error interrupts either enabled or disabled, which have not been cleared by software. As a result, the IP's clockstop/reset sequence may never complete (hang) if there are any pending safety errors interrupts that remain uncleared.

The affected ECC_AGGRs can be determined by the value listed in the Technical Reference Manual (TRM) for their REV register at Register Offset 0h. The REV register encodes the ECC_AGGR version in its fields as follows:

v[REVMAJ].[REVMIN].[REVRTL]

ECC_AGGR versions before v2.1.1 are affected. ECC_AGGR versions v2.1.1 and later are not affected.

Affected Example:

REVMAJ = 2

REVMIN = 1

REVRTL = 0

The above values decode to ECC_AGGR Version v2.1.0, which is Affected.

Not Affected Example:

REVMAJ = 2

REVMIN = 1

REVRTL = 1

The above values decode ECC_AGGR Version v2.1.1, which is Not Affected.

**Workaround(s):**

General Note:

Clockstopping the ECC Aggregator is not supported in functional safety use-cases.

Software should use the following workaround for non-functional safety use-cases:

1. Enable all ECC Aggregator interrupts for the IP
2. Service and clear all Pending interrupts
3. Step 3:
   a. Disable all interrupt sources to the ECC Aggregator, followed by performing Clockstop/reset sequence.
   b. Perform Clockstop/reset sequence, while continuing to service/clear pending interrupts.

Due to interrupts being external stimuli, software has two options for step 3:

***i2049*** (continued) ***ECC_AGGR: Potential IP Clockstop/Reset Sequence Hang due to Pending ECC Aggregator Interrupts***

1. Disable all interrupt sources (EDC CTRL checkers) that can generate pending ECC_AGGR interrupts prior to performing the clockstop/reset sequence
2. Continue to service/clear pending interrupts that occur while performing the clkstop/reset sequence. The sequence would proceed when all interrupts are cleared.

Software in general may need to detect pending interrupts that continuously fire during this entire sequence (ex. in the case of a stuck-at fault scenario), and disable their associated EDC CTRL safety checkers to allow the clockstop/reset sequence to progress towards completion.

## *i2062* — *RAT: Error Interrupt Triggered Even When Error Logging Disable Is Set*

**Details:**

If the RAT error logging is programmed to disable logging and enable interrupts, then an error will incorrectly trigger an interrupt but the error log registers will correctly not be updated. The error interrupt should not have been generated.

**Workaround(s):**

If the RAT error logging is disabled, then the error interrupt should also be disabled by software.

## *i2103* — *Internal Diagnostics Modules: Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors*

**Details:**

For functional safety errors, the logged information - ECC_GRP, ECC_BIT, and ECC_TYPE in the Error Status Registers may be incorrect for certain safety checkers. This only applies to safety checkers that map to ECC_GRP = 0,15,31,47,63...(N*16-1). In the case for the DDR Bridge/Controller, the issue only applies to the safety checkers where ECC_GRP = 0,31,63...(N*32-1).

This issue affects all Internal Diagnostics Module instances and their sub-banks.

Note: The detection and interrupt signaling of these safety errors is unaffected. Only the logging of the aforementioned fields of the Error Status Registers are affected.

**Workaround(s):**

None. For these specific safety checkers, software is limited to knowing whether a correctable or uncorrectable error occurred and which Internal Diagnostics Module instance had the error (thus knowing the IP module), but not which exact safety checker encountered the error.

## *i2184* — *CPSW: IET express traffic policing issue*

**Details:**

This applies to 9-port CPSW, 5-port CPSW, 3-port CPSW, and 2-port CPSW IET traffic.

In IET (Interspersed Express traffic), if a preempted packet was interrupted by an express packet, two things can occur:

1. If the express traffic is policed, the frame size for the preempted packets is applied to the express traffic policer. Assuming a policer was set up to rate schedule an express traffic stream, it would take a hit by the preempted packet size it interrupted. The preempted packet also takes on the express traffic policer status. As a result,

***i2184*** (continued) ***CPSW: IET express traffic policing issue***

preempted packets could get dropped along with other express traffic due to the express traffic policer.
2. If the express traffic was not policed, the interrupted preempt packet would not get its packet size applied to the preempted policer.

**Workaround(s):**

Do not police IET express traffic.

## *i2189* *OSPI: Controller PHY Tuning Algorithm*

**Details:**

The OSPI controller uses a DQS signal to sample data when the PHY Module is enabled. However, there is an issue in the module which requires that this sample must occur within a window defined by the internal clock. Read operations are subject to external delays, which change with temperature. To ensure valid reads at any temperature, a special tuning algorithm must be implemented which selects the most robust TX, RX, and Read Delay values.

**Workaround(s):**

The workaround for this bug is described in detail in SPRACT2. To sample data under some PVT conditions, users are required to increment the Read Delay field to shift the internal clock sampling window. This allows sampling of the data anywhere within the data eye. However, this has these side effects:

1. PHY Pipeline mode must be enabled for all read operations. Because PHY Pipeline mode must be disabled for writes, reads and writes must be handled separately.
2. Hardware polling of the busy bit is broken when the workaround is in place, so SW polling must be used instead. Writes must occur through DMA accesses, within page boundaries, to prevent interruption from either the host or the flash device. Software must poll the busy bit between page writes. Alternatively, writes can be performed in non-PHY mode with hardware polling enabled.
3. STIG reads must be padded with extra bytes, and the received data must be right-shifted.

## *i2236* *PCIe: SERDES output reference clock cannot be used*

**Details:**

The PCIe Reference Clock Output of the SERDES does not comply with the certain PCI-SIG specifications, which may cause issues for external PCIe components receiving and using the reference clock. Therefore, SERDES reference clock (signals SERDES0_REFCLK0P and SERDES0_REFCLK0N) cannot be used to output a PCIe reference clock.

**Workaround(s):**

None. This issue is still under investigation.

## *i2185* *CPSW: Policer color marking issue*

**Details:**

Only applies to CPSW9G and CPSW5G.

***i2185*** (continued) ***CPSW: Policer color marking issue***

When packets from two different ports hit the same policer such that one port has a large packet and the other has a short packet and the short packet arrives just after the large packet starts, the short packet will stop the backlog counting, resulting in potentially flagging the next frame for this policer as yellow when it should have been green. Because the policer is normally set up to not drop yellow, it should not cause an issue. This is only true of packets that arrive on different ports that share the same policer index.

**Workaround(s):**

Ensure policers are unique to ports.

***i2196*** ***IA: Potential deadlock scenarios in IA***

**Details:**

The interrupt Aggregator (IA) has one main function, which is to convert events arriving on the Event Transport Lane (ETL) bus, can convert them to interrupt status bits which are used to generate level interrupts. The block that performed this function in IA version 1.0 was called the status event block.

In addition to the status event block, there are two other main processing blocks; the multicast event block, and the counted event block. The multicast block really functions as an event splitter. For every event it takes in, it can generate two output events. The counted event block is used to convert high frequency events into a readable count. It counts input events and generates output events on count transitions to/from 0 to/from non-zero count values. Unlike the status event block, the multicast and counted event blocks generate output ETL events that are then mapped to other processing blocks.

An issue was found after design that could cause the IA to deadlock. The issue occurs when event “loops” occur between these three processing blocks. It is possible to create a situation where a processing block can not output an event because the path is blocked, and since it can not output an event, it can not take any new input events. This inability to take input events prevents the output path from being able to unwind, and thus both paths remain blocked.

**Workaround(s):**

Figure 2-1 shows the conceptual block diagram of IA 1.0. Potential loops are avoided by adopting the policy of not allowing the counted event block to send events to the multicast block. This method was chosen because it is more common to split an event first, and then count one while sending the other elsewhere. With this path blocked by convention, it is not possible for a single event to visit any block more than once and thus not possible for paths to become blocked so long as the outputs remain unblocked.

***i2196*** (continued) ***IA: Potential deadlock scenarios in IA***

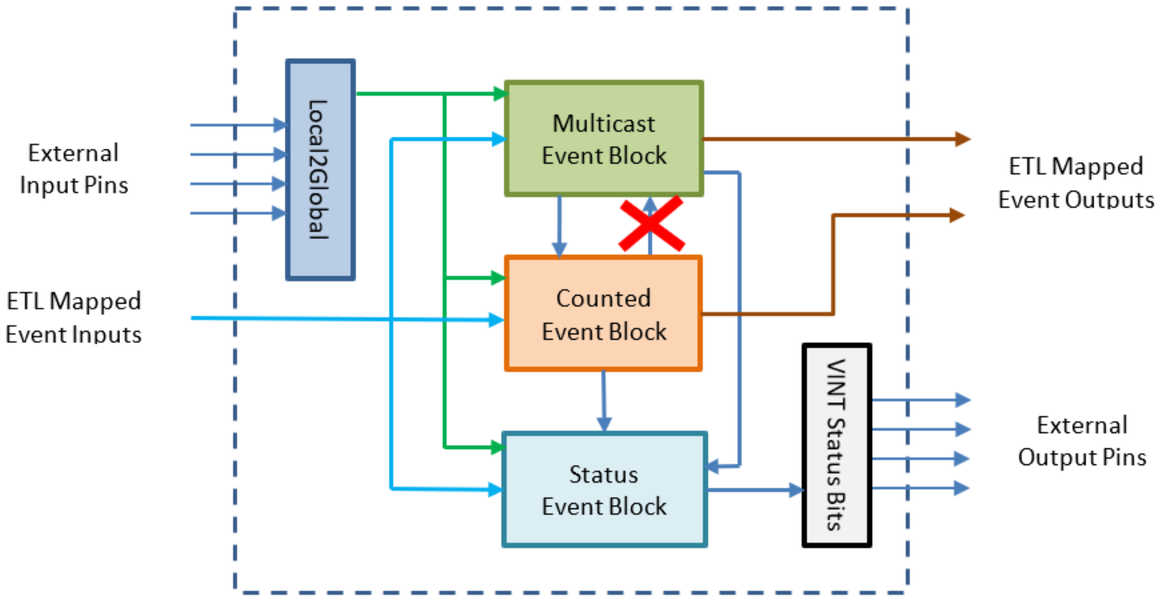

**Figure 2-1. Interrupt Aggregator Version 1.0**

IA version 1.1 introduced two additional changes to the architecture. First, there is a new processing block for “unmapped” events. These are events that are sent to a fixed destination (the IA) instead of being programmable at the source IP. Being fixed, the are called “unmapped”. Once in the IA, they are mapped to a traditional ETL event destination. The block that performs this mapping is called unmapped event block. The second change to the IA for 1.1 is that now, the multicast and counted event blocks (as well as the new unmapped event block) can directly set VINT status bits, and they do not need to forward output ETL events to the status event block.

Figure 2-2 shows the diagram of IA 1.1. Note that the same policy applies to avoid deadlock. In addition, the paths shown as dotted lines (sending ETL events from the multicast or counted blocks to the status event block) is now discouraged. These paths still function as before for backward compatibility, but are rendered obsolete as all blocks are now able to directly set status event bits.

***i2196*** (continued) ***IA: Potential deadlock scenarios in IA***

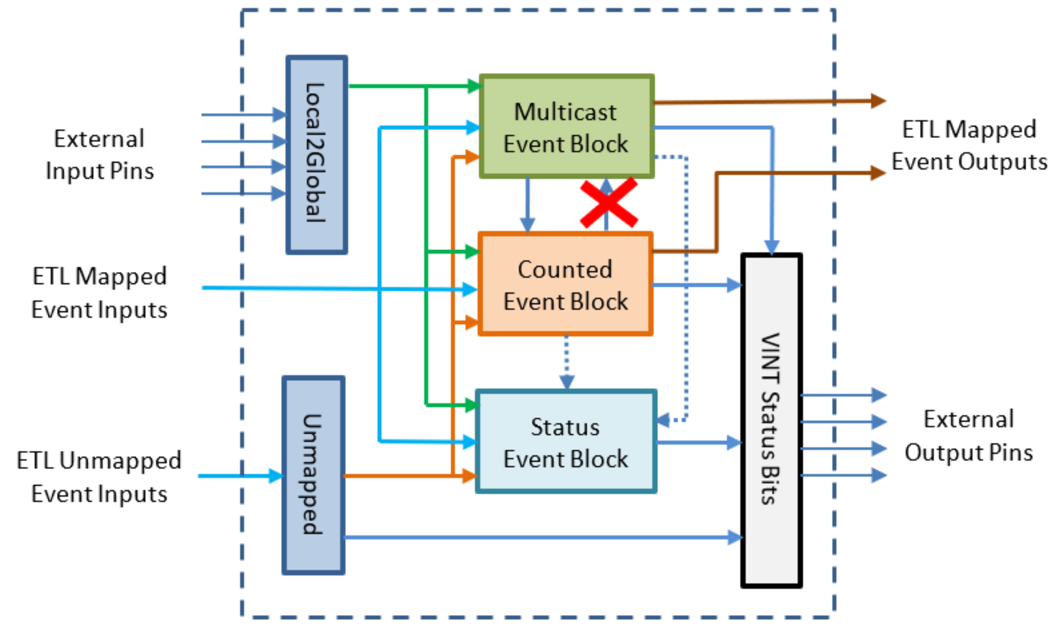

**Figure 2-2. Interrupt Aggregator Version 1.1**

By following the conventions outlined here, the system is safe from looping hazards that can create a deadlock scenario.

***i2207*** ***CBASS: Command Arbitration Blocking***

**Details:**

When the interconnect arbitrates commands from multiple sources, the higher priority request always takes precedence. Requests that are at the same priority level are arbitrated in round-robin fashion. The issue is after the higher priority request goes idle and there are two or more pending requests that are at the same priority level, the hardware selects one of them arbitrarily. A potential issue may arise when software polls from multiple sources to the same endpoint: after servicing the higher priority source, the hardware may repeatedly select the same lower priority source for access. That means other same-lower priority requests may be blocked for a long time, and in the worst case if there are dependencies between the polling sequences, the software may run into a livelock state.

This issue only affects certain interconnect where in one switch module there are at least three sources that can access the same target simultaneously. Also note that when all requests are at the same priority level, the issue does not apply.

**Workaround(s):**

When multiple sources are simultaneously polling from the same endpoint, and there is expected dependency based on the read data, ensure that all sources are sending the read commands at the same priority level. The source that breaks the dependency should be at equal or higher priority than other dependent sources.

***i2208*** ***CPSW: ALE IET Express Packet Drops***

**Details:**

This issue impacts the following Module:

**i2208** (continued) ***CPSW: ALE IET Express Packet Drops***

[AM64x] 3-port CPSW

The issue with ALE is due to CPSW frequency and IET operation with short express traffic and pre-empted packets that get pre-empted between 60-69 bytes on non-10G capable ports.

If an IET pre-emptible packet get interrupted at 60-69 bytes, the lookup will occur when the next chunk arrives. The CPSW only gives the ALE 64 bytes from the pre-emptible MAC.

As a result, a short express traffic lookup will start at the end of a 64 byte express traffic, but when the pre-empted queue continues, the pre-empted traffic will complete the 64 bytes and attempt a lookup for the pre-empt packet. But this lookup is less that 64 clocks from the express lookup start, so the express lookup will be aborted(express traffic dropped) and start the new lookup for the pre-empted traffic.

Rules to induce the issue:

1. You are in IET (Interspersed Express Traffic) mode on ports not capable of 5/10G operation
2. Remote express packets can be preempt packets as low as 60 bytes
3. Pre-empt packet traffic that is 128 bytes or more.
4. Express traffic that interrupts the pre-empt traffic between 60-69 bytes.
5. A short express traffic immediately followed by the continuation of the pre-empt traffic.
   a. Gap between express frame and pre-empt frame be its minimum.
6. The CPSW frequency is at its lowest capability for the speeds required.

**Workaround(s):**

During IET negotiation, tell the remote to fragment at 128 bytes.

**i2228** ***JTAG: TAP used by Debuggers may be inaccessible if TRSTn device pin is never asserted***

**Details**

If TRSTn is never observed LOW, access to the embedded Debugger scan chains might be blocked by uninitialized logic. JTAG bypass and Boundary Scan functionality is not affected.

**Workaround**

Prior to connecting a Debugger, ensure that the TRSTn pin is asserted LOW for 100ns and subsequently de-asserted HIGH at-least one time after device power on.

## *i2232* *DDR: Controller postpones more than allowed refreshes after frequency change*

**Details**

When dynamically switching from a higher to lower clock frequency, the rolling window counters that control the postponing of refresh commands are not loaded correctly to scale to the lower clock frequency. This will result in controller postponing more refresh commands than allowed by the DRAM specification, thus violating refresh requirement for the DRAM.

**Workaround**

Workaround 1:Disable dynamic frequency change by programing DFS_ENABLE = 0

Workaround 2:If switching frequency, program the register field values based on the pseudo code listed below.Note that the controller requires AREF_*_THRESHOLD values to be programmed before triggering initialization. Their values cannot be changed during mission mode after initialization . Therefore, the value of these parameters must be the lowest of all values needed for every frequency change transition planned to be used.

```
if (old_freq/new_freq >= 7){
    if (PBR_EN==1) { // Per-bank refresh is enabled
        AREF_HIGH_THRESHOLD = 19
        AREF_NORM_THRESHOLD = 18
        AREF_PBR_CONT_EN_THRESHOLD = 17
        AREF_CMD_MAX_PER_TREF = 8
    }
    else { // Per-bank refresh is disabled
        AREF_HIGH_THRESHOLD = 18
        AREF_NORM_THRESHOLD = 17
        // AREF_PBR_CONT_EN_THRESHOLD <=== don’t care, PBR not enabled
        AREF_CMD_MAX_PER_TREF = 8
    }
}
else {
    AREF_HIGH_THRESHOLD = 21
    AREF_NORM_THRESHOLD //<=== keep AREF_NORM_THRESHOLD < AREF_HIGH_THRESHOLD
    AREF_CMD_MAX_PER_TREF = 8
    if (PBR_EN==1) { // Per-bank refresh is enabled
    //keep AREF_PBR_CONT_EN_THRESHOLD<AREF_NORM_THRESHOLD<AREF_HIGH_THRESHOLD
        AREF_PBR_CONT_EN_THRESHOLD
    }
}
```

## i2244 — DDR: Valid stop value must be defined for write DQ VREF training

**Details**

The DDR PHY uses start, stop, and step-size values for write DQ VREF training. If the stop value is not equal to the start value + a multiple of the step-size, then the final VREF setting can go beyond the maximum VREF range, causing the training to hang.

**Workaround**

Program the stop value as follows:

PI_WDQLVL_VREF_INITIAL_STOP = (multiple of PI_WDQLVL_VREF_INITIAL_STEPSIZE) + PI_WDQLVL_VREF_INITIAL_START

This workaround is implemented in the DDR Subsystem Register Configuration Tool v0.03.00 or later. See https://dev.ti.com/sysconfig for more details.

## i2245 — DMSC: Firewall Region requires specific configuration

**Details**

The ECC Aggregator inside DMSC (DMSC0_ECC_AGGR) has an endpoint firewall which is used to protect this region. By default, this firewall blocks all the transactions except from the M3 core inside DMSC.

**Workaround**

If another processor or endpoint needs to access DMSC0_ECC_AGGR region, software shall configure the firewall region with starting address 0x0 and end address 0xFFFF_FFFF, using the CBASS_FW_REGION_i_START_ADDRESS and END_ADDRESS registers associated with the DMSC0_ECC_AGGR region. This is the only allowable address configuration for this region.

## i2091 — USB: 2.0 PHY Hangs if Received Signal Amplitude Crosses Squelch Threshold Mmultiple Times Within the Same Packet

**Details:**

USB 2.0 PHY implements a squelch detection circuit on the receiver to ensure noise is not interpreted as valid data when the bus is idle. The squelch circuit blocks invalid data by disabling the receiver output while the DP/DM differential signal amplitude is less than the squelch threshold.

The PHY may hang if the DP/DM differential signal amplitude drops below the squelch threshold for a brief period of time and increases back above the squelch threshold within the same packet. The issue does not occur if the DP/DM differential signal amplitude crosses the squelch threshold during the idle time between two packets.

**Workaround(s):**

The issue can be avoided by ensuring the DP/DM differential signal amplitude applied to the receiver input remains above the squelch threshold during valid data transfers.

### *i2235* *CBASS Null Error Interrupt Not Masked By Enable Register*

**Details**

There is optional feature in CBASS that adds the null error reporting MMR and interrupt source. When the feature is present and the interrupt is enabled, these two output ports: "err_intr_intr" (level interrupt source) and "err_intr_pls_intr" (pulse interrupt source) will be asserted when an access to a null region occurs. The enable for the interrupt is in the ERR_INTR_ENABLE_SET register (address offset 0x58).

The issue is CBASS ignores this enable bit, and as a result any null access always produces the interrupt sources/events.

**Workaround**

There is no spurious event due to this bug because of the default disable status of processor events. At system level, processors don't receive any event unless it's enabled in the associated GIC/VIM interrupt controller.

When the interrupt is enabled, and an interrupt does occur, write to the following registers at cbass level to clear it:

write 0x1 to the err_intr_enabled_stat register, then write 0x1 to the err_eoi register.

### *i2303* *OSPI: OSPI0_LBCLK0 pin has input floating by default*

**Details**

The IO associated with the OSPI0_LBCLK0 pin has its receiver enabled, transmitter disabled, and internal pulls disabled by default after MCU_PORz is de-asserted. This default configuration allows the input buffer to float when there is no external component sourcing a valid logic state to the pin. This condition could potentially create a reliability issue for the input buffer if left in this state for long periods of time. The normal recommendation would be to connect an external pull-up resistor to the pin. However, this is not an option if the application plans to use the internal pad lookback clocking option for OSPI0. The OSPI0_LBCLK0 pin must not have any signal trace connected if the application plans to use the internal pad lookback clocking option. This connectivity requirement prevents the looped clock signal from being distorted (glitched) when it is sent out through the output buffer to the pin and looped back into OSPI0 through the input buffer.

**Workaround**

Ensure the application software enables the internal pull via the PADCONFIG1 register as soon as possible to avoid a long term floating node if the application plans to use the internal pad lookback clocking option for OSPI0. Connect an external pull-up resistor to this pin if the application plans to use any other clocking option for OSPI0.

### *i2317* *DebugSS: Debug is not available after TRST pin de-assertion when device pin EMU1 is repurposed as MCU_OBSCLK0*

**Details**

The device JTAG port may unexpectedly be connected to the Boundary Scan TAP rather than the On-Chip Debug TAP (SWJ-DP) while using a JTAG debugger when the EMU1 pin is configured for the MCU_OBSCLK0 signal function. This will render the normal JTAG debugging mode inoperable

This occurs because the device selects the Boundary Scan TAP when a value of 01b is sampled on the EMU[1:0] inputs on the rising edge to TRSTn. This value will be driven into the SOC when MCU_PADCONFIG32 is configured for any mux mode value other than mode “0” and MCU_PADCONFIG31 is configured for a mux mode value of “0” with the EMU0 pin pulled high.

***i2317*** (continued)

### ***DebugSS: Debug is not available after TRST pin de-assertion when device pin EMU1 is repurposed as MCU_OBSCLK0***

- The EMU1 signal going into the SOC is driven low by the multiplexing logic when the EMU1 signal function is not selected.
- The EMU0 signal going into the SOC is driven high by internal and external pull-ups.
- The JTAG debugger de-asserts TRSTn when attempting to communicate with the On-Chip DebugTAP (SWJ-DP).

**Workaround**

This issue will only occur when the device samples a value of 01b on the SOC EMU[1:0] inputs on the rising edge of TRSTn. This condition can be avoided with any one or more of the following actions.

- Never connect a debugger to the JTAG port when the EMU1 pin is being used for the MCU_OBSCLK0 signal function.
- Configure the MCU_SPI0_CS1 pin for the MCU_OBSCLK0 signal function rather than the EMU1 pin, and leave the EMU1 pin in its default configuration.
- Configure the EMU0 pin to any mux mode other than “0” or “15” when configuring the EMU1 pin for the MCU_OBSCLK0 signal function. This causes the pin multiplexing logic to drive a EMU[1:0] value of 00b into the SOC, which will not select Boundary Scan mode.

***i2134***

### ***USB: 2.0 Compliance Receive Sensitivity Test Limitation***

**Details:**

Performing receive sensitivity tests (EL_16 and EL_17) as defined in the USB-IF USB 2.0 Electrical Compliance Test Specification may invoke the problem described in Advisory i2091.

The issue was originally found while performing these tests using automation software, which increased USB signal amplitude while sending packets. The software was sweeping the amplitude from a value less than 100 mV to a value greater than 150 mV while verifying the device under test (DUT) NAK’d all packets below 100 mV and NAK’d no packets above 150 mV. However, increasing the amplitude through the squelch threshold while sending valid packets may lock the PHY as described in Advisory i2091.

**Workaround(s):**

Enable both of the following hardware workarounds.

Set cdr_eb_wr_reset bit (bit 7) to 1’b1 in UTMI_REG28 register present in USB*_PHY2 region.

Set phyrst_a_enable bit (bit 0) to 1’b1 in PHYRST_CFG register present in USB*_MMR_MMRVBP_USBSS_CMN region. Please note that phyrst_a_value (bits 12:8) in PHYRST_CFG register should be retained at default value of 0xE.

***i2257***

### ***Boot: xSPI boot mode redundant image boot failure***

**Details**

xSPI boot is not able to boot from redundant image offset at 0x400000 when image at offset 0x0 is corrupted. xSPI boot failure API in the ROM does not handle the header check for xSPI properly.

**Workaround**

For xSPI 1S mode operation, enable SPI as backup boot mode. Note that this workaround does not apply to xSPI SFDP and 8D modes. No workaround exists for SFDP and 8D modes.

## *i2277* *POK: De-Glitch (filter) is based upon only two samples*

**Details**

The POK is sampled on an approximate period of 1.25us. The "near-by" sample history is saved in a circular buffer. The De-Glitch (filter) is designed to AND the last *n* entries from the sample history in order to generate the output (to the ESM).

The De-Glitch filter is programmable to {4, 8, 12, 16} samples. The De-Glitch output is based upon a check of only the last entry (0th) and programmed number of samples ago (i.e. 3rd, 7th, 11th, or 15th). The filter ANDs these two results (instead of 4, 8, 12, or 16) in order to to generate the FAIL output to the ESM.

Notice that when the POK is set to monitor a fixed threshold (UV or OV but not set to ping-pong), the un-checked samples will be used.

When the POKs are controlled in a ping-pong manner, the skipped samples will be discarded.

**Workaround**

There is no workaround.

However, the intent of the De-Glitch (filtering) is to insure that a discrete voltage dip or rise does not trigger FAIL. The sampling of two points significantly separated in time means that the voltage dip / rise was not a single isolated event.

Since the filter requires all N samples to fail before generating a FAIL signal to the ESM, the inclusion of 2 points instead of N makes this circuit more sensitive.

## *i2285* *BCDMA: Blockcopy Gets Corrupted if TR Read Responses Interleave with Source Data Fetch*

**Details**

BCDMA can get corrupted during a block copy operation if the 64B TR descriptor fetch goes to an endpoint that can fragment the 64B burst into smaller burst sizes. This includes DDR where the read transaction will be fragmented to 32B bursts. The only endpoint where fragmentation won't happen is the 2MB OCSRAM. Corruption occurs if the first 32B of the descriptor fetch is returned and the second 32B of the descriptor fetch is delayed such that the BCDMA starts fetching source data and starts receiving that data before the second 32B is received by the BCDMA.

**Workaround**

BCDMA block-copy TR descriptors have to be placed in the 2MB OCSRAM. Alternatively the block-copy TR descriptors may be placed in DDR with the requirement that the block copy source data must also be in DDR. Refer to the table below for details.

**Table 2-1.**

| Module Endpoint | Size | Could be used for pktDMA descriptors or TR | Could it be used for data buffer for pktDMA( source data or dest data) | Could be used for BCDMA descriptors or TR | BCDMA split TX/RX channel | any considerations on BCDMA source data | Can be used as BCDMA block copy dest data? | PG2.0 Considerations |
|---|---|---|---|---|---|---|---|---|
| DDR | up to 2GB | Yes | Yes | Yes | Yes. Limited to DDR size per TR. no additional TR storage restriction. | Limit to DDR size per TR, and TR for this channel should stored in DDR | Yes, no additional considerations | Yes. Limited to DDR size per TR. No additional TR storage restriction. |

***i2285*** (continued) ***BCDMA: Blockcopy Gets Corrupted if TR Read Responses Interleave with Source Data Fetch***

**Table 2-1. (continued)**

| Module Endpoint | Size | Could be used for pktDMA descriptors or TR | Could it be used for data buffer for pktDMA( source data or dest data) | Could be used for BCDMA descriptors or TR | BCDMA split TX/RX channel | any considerations on BCDMA source data | Can be used as BCDMA block copy dest data? | PG2.0 Considerations |
|---|---|---|---|---|---|---|---|---|
| main R5 TCM | single core mode 64KB ATCM+64KB BTCM. Dual core mode 32KB ATCM+32KB BTCM. | No | Yes | No | Yes, no additional TR storage restriction. | Not to be used as BCDMA's source data | Yes, no additional considerations | Yes, no additional TR storage restriction. |
| main domain on-chip SRAM | 8 banks of memory, each bank is 256KB | Yes | Yes | Yes | Limit to aligned 256KB per TR, can not cross 256KB boundary within the same TR. No additional TR storage considerations. | Limit to aligned 256KB per TR, can not cross 256KB boundary within the same TR. And TR and the source data needs to be in the same memory bank. | Yes, no additional considerations | Limit to aligned 256KB per TR. No additional TR storage requirement. |
| 1KB PSRAM in main_infra | 1KB, used for boot vector storage. Not for TR | No | Yes | No | Limit to aligned 64KB per TR. No additional TR storage considerations. | Not to be used as BCDMA's source data | Yes, no additional considerations | Yes, limited to DMSC_lite memory size (64KB_64KB). no additional TR storage restriction. |
| DMSC_lite SRAM | 64KB+64KB | Yes | Yes | Yes | Yes, no additional TR storage restriction. | Limit to the DMSC_lite memory size (64KB+64KB) per TR. And TR has to be stored in the same memory end point. | Yes, no additional considerations | Yes, limited to DMSC_lite memory size (64KB_64KB). No additional TR storage restriction. |
| OSPI flash | Up to 4GB | No | Yes | No | Yes, no additional TR storage restriction. | Not to be used as BCDMA's source data | Yes, no additional considerations | Yes. Limit to flash size. no additional TR storage restriction. |
| GPMC | up to 128MB | No | Yes | No | Yes, no additional TR storage restriction. | Not to be used as BCDMA's source data | Yes, no additional considerations | Yes. no additional TR storage restriction. |
| PCIe | Up to 4GB | Yes, but Linux does not support this use case | Yes | Yes, but Linux does not support this use case | Yes, no additional TR storage restriction. | The descriptor for Block copy has to be stored on the remote side. Not possible for Linux usecase. | Yes, no additional considerations | Yes. no additional TR storage restriction. |
| ROM | | No | Yes, only as source data | No | Only split RX channel. No additional TR storage restriction. | Not to be used as BCDMA's source data | No, ROM only supports read. | Yes. no additional TR storage restriction. |

## *i2310* *USART: Erroneous clear/trigger of timeout interrupt*

**Details:**

The USART may erroneously clear or trigger the timeout interrupt when RHR/MSR/LSR registers are read.

**Workaround(s):**

**For CPU use-case.**

- If the timeout interrupt is erroneously cleared:
  - This is Valid since the pending data inside the FIFO retriggers the timeout interrupt
- If timeout interrupt is erroneously set, and the FIFO is empty, use the following SW workaround to clear the interrupt:
  - Set a high value of timeout counter in TIMEOUTH and TIMEOUTL registers
  - Set EFR2 bit 6 to 1 to change timeout mode to periodic
  - Read the IIR register to clear the interrupt
  - Set EFR2 bit 6 back to 0 to change timeout mode back to the original mode

**For DMA use-case.**

- If timeout interrupt is erroneously cleared:
  - This is valid since the next periodic event retriggers the timeout interrupt
  - User must ensure that RX timeout behavior is in periodic mode by setting EFR2 bit6 to 1
- If timeout interrupt is erroneously set:
  - This causes DMA to be torn down by the SW driver
  - Valid since next incoming data causes SW to setup DMA again

## *i2311* *USART Spurious DMA Interrupts*

**Details:**

Spurious DMA interrupts may occur when DMA is used to access TX/RX FIFO with a non-power-of-2 trigger level in the TLR register.

**Workaround(s):**

Use power of 2 values for TX/RX FIFO trigger levels (1, 2, 4, 8, 16, and 32).

## *i2313* *GPMC: Sub-32-bit read issue with NAND and FPGA/FIFO*

**Details:**

Sub-32-bit reads on the GPMC interface will miss portions of the data, which will result in incorrect read data. This includes 8-bit or 16-bit reads from a NAND device or from an FPGA or FIFO interface. Note that 3-byte accesses are not allowed on the GPMC interface.

**Workaround(s):**

Read accesses on the GPMC interface must be performed as 32-bit reads. Writes are not affected by this erratum.

### *i2328* *Boot: USB MSC boots intermittently*

**Details:**

USB MSC Host boot may fail due to a protocol timing violation present in the ROM USB device enumeration process. USB DFU boot is unaffected.

**Workaround(s):**

No workaround is available. Some USB MSC devices may tolerate this protocol violation and function as expected. Due to the internal component variability of broad-market MSC devices, a list of tolerant devices cannot be provided.

### *i2241* — *PCIe: The SerDes PCIe Reference Clock Output can exceed the 5.0 GT/s Data Rate RMS jitter limit*

**Details**

When operating the SerDes PCIe Reference Clock in Output mode, the RMS jitter of the clock may exceed the PCIe specification limit for the 5.0 GT/s Data Rate.

**Workaround**

Option 1:

Configure the Reference Clock output in Derived Refclk mode (as opposed to Received Refclk mode) and program the PLL configuration registers as follows:

- Set CMN_PDIAG_PLL0_CP_PADJ_M0 = 0x0128 to enable lower jitter operation

(Note for Devices that support 8.0 GT/s operation: Derived Refclk mode has an associated errata i2242 related to temporary disabling of the Refclk while changing Data Rates to/from 8.0 GT/s in a Single PLL SerDes Configuration).

Option 2:

Do not operate the PCIe interface at the 5.0 GT/s Data Rate.

Option 3:

Use an external clock source to supply the PCIe Reference Clock to both the Root Complex and End Point Devices of the Link.

### *i2279* — *MCAN: Specification Update for dedicated Tx Buffers and Tx Queues configured with same Message ID*

**Details**

The erratum updates the descriptions in Section 3.5.2 Dedicated Tx Buffers and 3.5.4 Tx Queue of the M_CAN User’s Manual related to message transmission from multiple dedicated Tx Buffers configured with the same Message ID.

**Workaround**

Workaround #1:

After writing the Tx messages with same Message ID to the Message RAM, request transmission of all these message concurrently by single write access to TXBAR. Make sure none of these messages have a pending Tx request before making the concurrent request.

Workaround #2:

Use the Tx FIFO instead of dedicated Tx Buffers (set bit MCAN_TXBC[30] TFQM = 0 to use Tx FIFO) for the transmission of several messages with the same Message ID in a specific order.

### *i2307* — *Boot: ROM does not properly select OSPI clocking modes based on BOOTMODE*

**Details**

The ROM bootloader only selects an internal loopback mode for SPI/QSPI/OSPI/xSPI boot, regardless of the Iclk field value selected by the BOOTMODE pins (see the device specific TRM for BOOTMODE pin mappings), which is intended to allow the user to choose an internal or external clocking method. This results in less flexibility in board topology in customers designs. Customers intending to use the external board loopback mode could see timing issues in ROM boot because the external loopback clock is not being used.

## *i2307* (continued) *Boot: ROM does not properly select OSPI clocking modes based on BOOTMODE*

**Workaround**

The topology of the OSPI design must not use "External Board Loopback" if planning to use OSPI as a boot source. All other clocking topologies (including internal loopback or DQS) can be used. Refer to the device specific datasheet, section "Applications, Implementation, and Layout" for supported clocking topologies using OSPI.

## *i2320* *BCDMA and PKTDMA: Descriptors and TRs required to be returned unfragmented*

**Details**

The BCDMA and PKTDMA require that the descriptors and TRs are placed in a memory subsystem that returns the descriptor or TR without any fragmenting of the descriptors. However, there are some memories that contain a fragmentation bridge, which makes them not available for holding the descriptors and TRs.

For this device, the R5 TCM memory and PCIe cannot hold descriptors or TRs for PKTDMA or BCDMA

**Workaround**

None

## *i2329* *MDIO: MDIO interface corruption (CPSW and PRU-ICSS)*

**Details:**

It is possible that the MDIO interface of all instances of CPSW and PRU-ICSS peripherals (if present) returns corrupt read data on MDIO reads (e.g. returning stale or previous data), or sends incorrect data on MDIO writes. It is also possible that the MDIO interface becomes unavailable until the next peripheral reset (either by LPSC reset or global device reset with reset isolation disabled in case of CPSW).

Possible system level manifestations of this issue could be (1) erroneous ethernet PHY link down status (2) inability to properly configure an ethernet PHY over MDIO (3) incorrect PHY detection (e.g. wrong address) (4) read or write timeouts when attempting to configure PHY over MDIO.

For boot mode (only CPSW if supported), there is no workaround to guarantee the primary ethernet boot is successful. If this exception occurs during primary boot, the boot may possibly initiate retries which may or may not be successful. If the retries are unsuccessful, this would result in an eventual timeout and transition to the backup boot mode (if one is selected). If no backup boot mode is selected, then such failure will result in a timeout and force device reset via chip watchdog after which the complete boot process will restart again.

To select a backup boot option (if supported), populate the appropriate pull resistors on the boot mode pins. See boot documentation for each specific device options, but the typical timeout for primary boot attempts over ethernet is 60 seconds.

**Workaround(s):**

On affected devices, following workaround should be used:

**MDIO manual mode: applicable for PRU-ICSS and for CPSW.**

MDIO protocol can be emulated by reading and writing to the appropriate bits within the MDIO_MANUAL_IF_REG register of the MDIO peripheral to directly manipulate the MDIO clock and data pins. Refer to TRM for full details of manual mode register bits and their function.

In this case the device pin multiplexing should be configured to allow the IO to be controlled by the CPSW or PRU-ICSS peripherals (same as in normal

***i2329*** (continued) ***MDIO: MDIO interface corruption (CPSW and PRU-ICSS)***

intended operation), but the MDIO state machine must be disabled by ensuring MDIO_CONTROL_REG.ENABLE bit is 0 in the MDIO_CONTROL_REG and enable manual mode by setting MDIO_POLL_REG.MANUALMODE bit to 1.

*Contact TI regarding implementation of software workaround.*

**Note**

If using Ethernet DLR (Device Level Ring) (on CPSW or PRU-ICSS) or EtherCat protocol (on PRU-ICSS) there may be significant CPU or PRU loading impact to implement the run-time workaround 1 due to required polling interval for link status checks. Resulting system impact should be considered.

In case of PRU-ICSS, the loading of the software workaround may be reduced by using the MLINK feature of MDIO to do automatic polling of link status via the MIIx_RXLINK input pin to PRU-ICSS which must be connected to a status output from the external PHY which does not toggle while the link is active. Depending on the specified behavior of the external PHY device, this PHY status output may be LED_LINK or LED_SPEED or the logic OR of LED_LINK and LED SPEED. Refer to the MDIO section of TRM for details on using the MLINK feature of MDIO. This feature is not available on the CPSW peripheral.

For EtherCAT implementation on PRU-ICSS, the software workaround will be done in RTUx/ TX_PRUx Core. The core will have to be dedicated for workaround, which means this can’t be used for other purpose. The implementation will support two user access channels for MDIO access. This provides option for R5f core and PRU core to have independent access channel. The APIs will be similar to the ones we will have in RTOS Workaround implementation.

EtherCAT will continue to use PHY fast link detection via MDIO MLINK bypassing state m/c for link status (as this path is not affected by errata). This makes sure that cable redundancy related latency requirements are still met.

***i2329*** (continued) ***MDIO: MDIO interface corruption (CPSW and PRU-ICSS)***

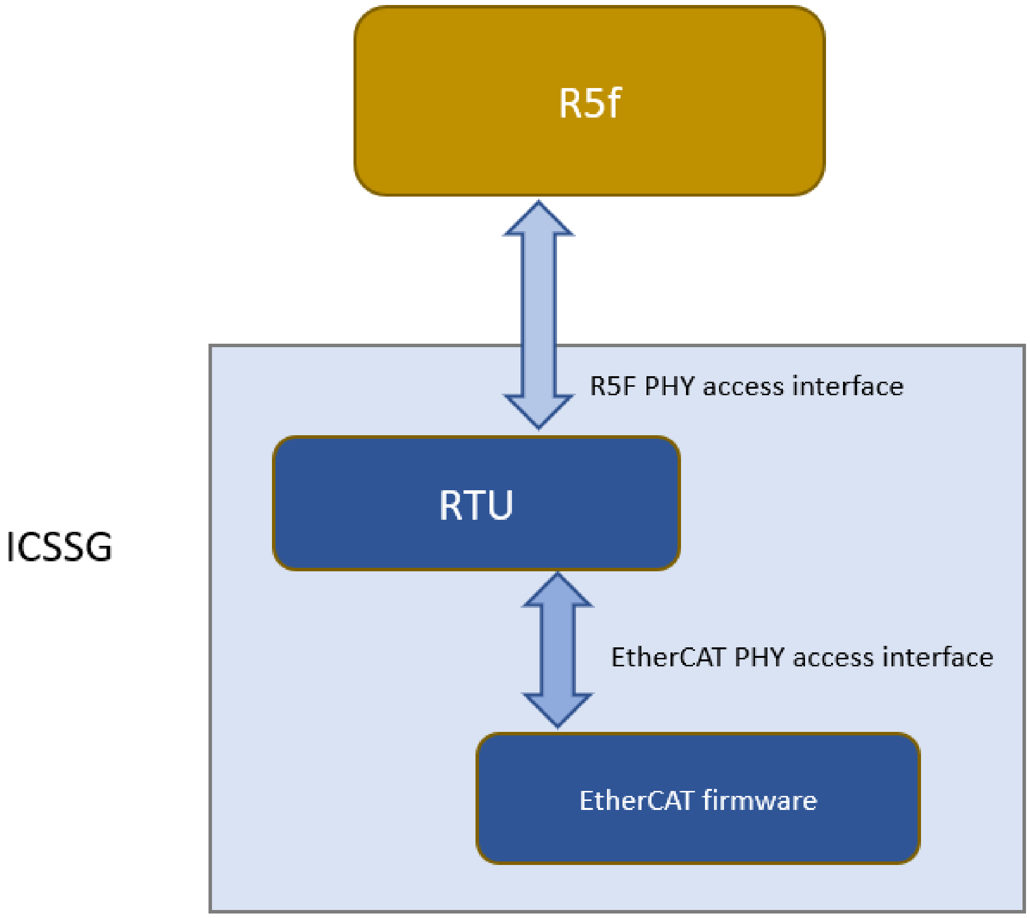

**Figure 2-3. MDIO Emulation via Manual Mode using PRU Core**

***i2331*** ***CPSW: Device lockup when reading CPSW registers***

**Details:**

A device lockup can occur during the second read of any CPSW subsystem register after any MAIN domain power on reset (POR). A MAIN domain POR occurs using the hardware MCU_PORz signal, or via software using CTRLMMR_RST_CTRL.SW_MAIN_POR or CTRLMMR_MCU_RST_CTRL.SW_MAIN_POR. After these resets, the processor and internal bus structures may get into a state which is only recoverable with full device reset using MCU_PORz.

Due to this errata, Ethernet boot should not be used on this device.

**Workaround(s):**

To avoid the lockup, a warm reset should be issued after a MAIN domain POR and before any access to the CPSW registers. The warm reset realigns internal clocks and prevents the lockup from happening. The warm reset should be triggered using CTRLMMR_MCU_RST_CTRL.SW_MCU_WARMRST. This will issue a warm reset to the entire device (both MCU and MAIN domains), and the device will re-initiate the boot process. Note that reset status bits distinguishing cold and warm resets would not be useful due to this workaround.

***i2331*** (continued) ***CPSW: Device lockup when reading CPSW registers***

If M4F is used as a safety processor, please consult your TI representative for workaround information.

If the application never accesses any CPSW register, this errata does not apply.

There is no workaround for Ethernet boot from CPSW0. Ethernet should not be used as a boot source on affected devices

### *i2243* — *PCIe: Timing requirement for disabling output refclk during L1.2 substate is not met*

**Details**

PCIe base specification requires Refclk to reach idle electrical state within 100ns of CLKREQ# deassertion when entering L1.2 substate (please refer TL1O_REFCLK_OFF parameter).

This timing requirement cannot be met when sourcing Refclk from the device since hardware does not automatically gate Refclk. Refclk gating has to be performed by software by writing to PHY_EN_REFCLK field in SERDES_RST register.

As a result, Refclk cannot be gated in L1.2 substate. Generally, allowing Refclk to run in L1.2 substate is not expected to cause any functional issues. However, if gating Refclk within 100ns is required by the system, L1.2 substate cannot be supported.

**Workaround**

Use an external Refclk generator to supply the PCIe reference clock

### *i2249* — *OSPI: Internal PHY Loopback and Internal Pad Loopback clocking modes with DDR timing inoperable*

**Details**

The OSPI Internal PHY Loopback mode and Internal Pad Loopback mode uses “launch edge as capture edge” (same edge capture, or 0-cycle timing).

The programmable receive delay line (Rx PDL) is used to compensate for the round trip delay (Tx clock to Flash device, Flash clock to output and Flash data to Controller).

In the case of internal and IO loopback modes, the total delay of the Rx PDL is not sufficient to compensate for the round trip delay, and thus these modes cannot be used.

The table below describes the recommended clocking topologies in the OSPI controller. All other modes not described here are affected by the advisory in DDR mode and are not recommended clocking topologies.

**Table 2-2. OSPI Clocking Topologies**

| Clocking Mode Terminology | CONFIG_REG.PHY_MODE_ENABLE | READ_DATA_CAPTURE.BYPASS | READ_DATA_CAPTURE.DQS_EN | Board implementation |
|---|---|---|---|---|
| No Loopback, no PHY | 0 (PHY disabled) | 1 (disable adapted loopback clock) | X | None. Relying on internal clock. Max freq 50MHz. |
| External Board Loopback with PHY | 1 (PHY enabled) | 0 (enable adapted loopback clock) | 0 (DQS disabled) | External Board Loopback (OSPI_LOOPBACK_CLK_SEL = 0) |
| DQS with PHY | 1 (PHY enabled) | X (DQS enable has priority) | 1 (DQS enabled) | Memory strobe connected to SOC DQS pin |

**Workaround**

None. Please use one of the unaffected clocking modes based on the table in the description

### *i2256* — *PCIe: MSI or MSI-X does not trigger interrupt if address is not aligned to 8-bytes*

**Details**

While operating as Root Complex, PCIe MSI or MSI-X reception causes a write to GIC ITS register space in order to trigger interrupt to CPU cores. If the address used in the MSI or MSI-X transaction is NOT aligned to 8-bytes, the interrupt to CPU cores does not occur.

***i2256*** (continued) ***PCIe: MSI or MSI-X does not trigger interrupt if address is not aligned to 8-bytes***

For example, MSI or MSI-X address with lower bits set to 0x10 does result in an interrupt whereas 0x14 fails.

**Workaround**

None available. MSI or MSI-X transaction has to be issued to 8-byte aligned ITS address.

***i2274*** ***DDR: Including DDR in BSCAN causes current alarm on the DDR supply***

**Details**

BSCAN causes current alarm trips when DDR is included. Customers using BSCAN should be warned of this issue to preclude the DDR in the scan chain during boundary scan. This only affects device packages which have DDR interface pinned out.

**Workaround**

Remove DDR from the scan chain when performing boundary scan. If DDR interface is not pinned out, this errata does not apply.

***i2278*** ***MCAN: Message Transmit order is not ensured from dedicated Tx Buffers configured with same Message ID***

**Details**

The erratum is limited to the case when multiple Tx Buffers are configured with the same Message ID (TXBC.NDTB > 1).

Under the following conditions, a message may be transmitted out of order:

- Multiple Tx Buffers configured with the same Message ID
- Tx requests for these Tx Buffers are submitted sequentially with delays between each

**Workaround**

Workaround #1:

After writing the Tx messages with same Message ID to the Message RAM, request transmission of all these message concurrently by single write access to TXBAR. Make sure none of these messages have a pending Tx request before making the concurrent request.

Workaround #2:

Use the Tx FIFO instead of dedicated Tx Buffers (set bit MCAN_TXBC[30] TFQM = 0 to use Tx FIFO) for the transmission of several messages with the same Message ID in a specific order.

***i2306*** ***ROM Code: Need to turn off internal termination resistors in SERDES***

**Details**

The SERDES implementation in this device has internal termination resistors enabled by default. During PCIe boot, the ROM code does not disable these termination resistors, which results in reduced voltage swing of the PCIe reference clock, potentially less than the minimum limit defined for a PCIe reference clock. This can result in PCIe boot failure.

**Workaround**

None

## *i2363* *Boot: Peak voltage transitions may exceed PCIe spec during PCIe boot*

**Details:**

When booting from a PCI-Express (PCIe) GEN1 link, transmitter peak differential output voltage transitions may exceed the PCIe Base Specification Revision 4.0 maximum permitted transition voltage by as much as 5% when the SERDES supply rails (VDDA_0P85_SERDES0, VDDA_0P85_SERDES0_C, VDDA_1P8_SERDES0) are also operating at maximum voltages. PCIe GEN1 transmitter peak differential output voltage is fully compliant when the SERDES supply rails are operating at nominal or minimum permitted voltages. PCIe GEN2 links are unaffected by this erratum.

**Workaround(s):**

None

## *i2312* *MMCSD: HS200 and SDR104 Command Timeout Window Too Small*

**Details:**

Under high speed HS200 and SDR104 modes, the functional clock for MMC modules will reach up to 192 MHz. At this frequency, the maximum obtainable timeout through of MMC host controller using MMCSD_SYSCTL[19:16] DTO = 0xE is (1/192MHz)*2^27 = 700ms. Commands taking longer than 700ms may be affected by this small window frame.

**Workaround(s):**

If the command requires a timeout longer than 700ms, then the MMC host controller command timeout can be disabled (MMCSD_CON[6] MIT=0x1) and a software implementation may be used in its place. Detailed steps as follows (in Linux):

1. During MMC host controller probe function (omap_hsmmc.c:omap_hsmmc_probe()), inform processor that the host controller is incapable of supporting all the necessary timeouts.

2. Modify the MMC core software layer functionality so the core times out on its own when the underlying MMC host controller is unable to support the required timeout.

## *i2371* *Boot: ROM code may hang in UART boot mode during data transfer*

**Details:**

Due to advisory i2310, it is possible for ROM code execution to hang during UART boot. The software workaround presented in i2310 is not implemented in ROM, and thus an erroneous timeout interrupt can be triggered in an unexpected state. This can prevent the ROM from being able to clear this interrupt and therefore hang.

This can manifest any time UART boot mode is used or when UART is used as the boot interface to enable production flows such as UniFlash or programing eFuses with OTP Keywriter.

**Workaround(s):**

None. Another boot interface should be used.

## *i2366* *Boot: ROM does not comprehend specific JEDEC SFDP features for 8D-8D-8D operation*

**Details:**

JEDEC spec JESD216 - SERIAL FLASH DISCOVERABLE PARAMETERS (SFDP) details the parameter table used in certain serial flash devices to describe features and

***i2366*** (continued) ***Boot: ROM does not comprehend specific JEDEC SFDP features for 8D-8D-8D operation***

how to communicate/configure the device. The ROM interprets relevant portions of the SFDP for a device's features (such as a how to change from 1S-1S-1S to 8D-8D-8D mode), but does not properly comprehend a flash device that requires:

- A swapped byte order in 8D-8D-8D mode compared to 1S-1S-1S mode
- A command extension that in 8D-8D-8D mode that requires a different command than the first byte sent (such as an inversion of the opcode or another unique byte)

**Workaround(s):**

Review the SFDP table of any candidate flash memory that is compliant with JEDEC JESD216; in most cases vendors do not publish this table and can instead be requested from the flash vendor. If the 18th DWORD of the JEDEC Basic Flash Parameter table has bit 31 with a value of "1b", then the memory must be programmed with a swapped byte order from the factory or programmed with the SoC. If bits [30:29] have a value other than "00b" then it will not work with any bootmodes in 8D-8D-8D mode. Avoid using any 8D-8D-8D bootmodes with that flash device as a result.

***i2138*** ***PSIL: Configuration accesses and source thread teardowns may cause data corruption***

**Details:**

When performing a tear down on one source thread, a single data phase on a different source thread may be lost. This affects all PSIL_ENDPT modules that have more than 1 source thread (ICSSG/CSI/SA2UL)

Also, if a configuration response is sent out by a PSIL Endpoint Gasket while data is also being sent out, the response will cause data corruption. This can affect data transmitted long after the configuration response occurs. This affects all PSIL_ENDPT users that require width adaption where their PSIL port is less than 128-bit (SA2UL).

**Workaround(s):**

All PSIL source threads must be idled by disabling the source of Rx traffic before attempting a configuration access or a teardown of any source thread. For ICSSG/CSI, idling of PSIL source threads can also be done by pausing all source threads.

***i2253*** ***PRG: CTRL_MMR STAT registers are unreliable indicators of POK threshold failure***

**Details**

The POK overvoltage and undervoltage flags in the CTRL_MMR PRG STAT registers are unreliable indicators of whether the POK has seen a failure. As a result, they are being marked as Reserved in the device Technical Reference Manual (TRM).

Register names affected: CTRLMMR_MCU_PRG_PP_x_STAT

**Workaround**

The filtered POK output updates ESM flags.

Upon POK initialization (i.e. enable), the ESM flags should be cleared (due to comparisons carried out during the bandgap and / or the POK settling time). After this initial clear, the ESM flags can be used as a reliable indicator of failure (or no failure) from the POKs.

## i2259 — *DDR: Possibility of starvation of low priority commands*

**Details**

DDR bridge allows prioritization of commands within a given thread, based on CBA priority. This may cause lower priority commands within a given thread to be blocked for a long time in the case where DDR bandwidth is usurped by higher priority commands.

**Workaround**

System needs to ensure that high priority traffic does not usurp DDR bandwidth by managing traffic using Class of Service.

## i2283 — *Restrictions on how CP Tracer Debug Probes can be used*

**Details**

Some CP Tracer bus probes do not receive the full SoC physical address but only a minimal set that was relevant to endpoint being monitored. This limits the usefulness of the probe in the SoC Analysis > Traffic Profiling feature in CCS.

1) Address Filtering / Matching : User would typically input the full 36b/40b (depending on device) address for any address-qualified bus probe jobs.

2) Decoding of transaction trace : User would expect that the address provided in the decoded stream was the full 36b/40b physical address of the transaction.

3) There is an additional issue for SOCs that use address aliasing (AM64x uses address aliasing with two SRAM banks).

Affected probes:

AM64x: GPMC, OSPI, SRAM

**Workaround**

none

## i2305 — *ICSSG: PRU RAM WRT during active FDB lookup write data corruption*

**Details**

The PRU Filter Data Base (FDB) RAM write can cause FDB RAM to get corrupted.

**Workaround**

1. Do not allow PRU FDB writes during New Packets, which can be achieved by taking the port down

OR

2. PRU FDB RAM write can only happen during safe zone, ie check for start of frame (SOF0/SOF1) and calculate safe zone depending on transfer speeds (100MHz or 1GHz)

## i2326 — *PCIe: MAIN_PLLx operating in fractional mode, which is required for enabling SSC, is not compliant with PCIe Refclk jitter limits*

**Details:**

The MAIN_PLLx, which optionally supplies the 100MHz PCIe Refclk for SERDES and external components, does not comply to the PCIe Refclk jitter limits when configured in fractional mode. Fractional mode is required for enabling SSC, therefore SSC mode is not compliant to the PCIe Refclk jitter limits.

**Workaround(s):**

When sourcing the 100MHz PCIe Refclk from the MAIN_PLLx, the MAIN_PLLx should be configured in integer mode only (DACEN = 0, DSMEN = 0). This prevents the use of SSC

***i2326*** (continued) ***PCIe: MAIN_PLLx operating in fractional mode, which is required for enabling SSC, is not compliant with PCIe Refclk jitter limits***

for PCIe Refclk, which requires the PLL to operate in fractional mode. If SSC is required on the PCIe interface, an external Refclk generator with SSC should be used to provide the SERDES 100MHz Refclk.

***i2368*** ***ICSSG: Device lockup when reading registers in ICSSG module***

**Details:**

A device lockup can occur during the second read of any ICSSG register after any MAIN domain power on reset (POR). A MAIN domain POR occurs using the hardware MCU_PORz signal, or via software using CTRLMMR_RST_CTRL.SW_MAIN_POR or CTRLMMR_MCU_RST_CTRL.SW_MAIN_POR. After these resets, the processor and internal bus structures may get into a state which is only recoverable with full device reset using MCU_PORz.

**Workaround(s):**

To avoid the lockup, a warm reset should be issued after a MAIN domain POR and before any access to the ICSSG registers. The warm reset realigns internal clocks and prevents the lockup from happening. The warm reset should be triggered using CTRLMMR_MCU_RST_CTRL.SW_MCU_WARMRST. This will issue a warm reset to the entire device (both MCU and MAIN domains), and the device will reinitiate the boot process. Note that reset status bits distinguishing cold and warm resets would not be useful due to this workaround.

If M4F is used as a safety processor, please consult your TI representative for workaround information.

If the application never accesses any ICSSG register, this errata does not apply.

## *i2383* *OSPI: 2-byte address is not supported in PHY DDR mode*

**Details:**

When the OSPI controller is configured for 2-byte addressing in PHY DDR Mode, an internal state machine mis-compares the number of address bytes transmitted to a value of 1 (instead of 2). This results in a state machine lockup in the address phase, rendering PHY DDR mode non-operable.

This issue does not occur when using any Tap mode or PHY SDR mode. This issue also doesn't occur when using 4 byte addressing in PHY DDR mode.

**Workaround(s):**

For compatible OSPI memories that have programmable address byte settings, set the amount of address bytes required from 2 to 4 on the flash. This may involve sending a specific command to change address bytes and/or writing a configuration register on the flash. Once done, update the amount of address bytes sent in the controller settings from 2 to 4.

For compatible OSPI memories that only support 2-byte addressing and cannot be re-programmed, PHY DDR mode will not be compatible with that memory. Alternative modes include:

- PHY SDR mode
- TAP (no-PHY) DDR mode
- TAP (no-PHY) SDR mode

## *i2401* *CPSW: Host Timestamps Cause CPSW Port to Lock up*

**Details:**

The CPSW offers two mechanisms for communicating packet ingress timestamp information to the host.

The first mechanism is via the CPTS Event FIFO which records timestamps when triggered by certain events. One such event is the reception of an Ethernet packet with a specified EtherType field. Most commonly this is used to capture ingress timestamps for PTP packets. With this mechanism the host must read the timestamp (from the CPTS FIFO) separately from the packet payload which is delivered via DMA. This mode is supported and is not affected by this errata.

The second mechanism is to enable receive timestamps for all packets, not just PTP packets. With this mechanism the timestamp is delivered alongside the packet payload via DMA. This second mechanism is the subject of this errata.

When the CPTS host timestamp is enabled, every packet to the internal CPSW port FIFO requires a timestamp from the CPTS. When the packet preamble is corrupted due to EMI or any other corruption mechanism a timestamp request may not be sent to the CPTS. In this case the CPTS will not produce the timestamp which causes a lockup condition in the CPSW port FIFO. When the CPTS host timestamp is disabled by clearing the tstamp_en bit in the CPTS_CONTROL register the lockup condition is prevented from occurring.

**Workaround(s):**

Ethernet to host timestamps must be disabled.

CPTS Event FIFO timestamping can be used instead of CPTS host timestamps.

## *i2409* — *USB: USB2 PHY locks up due to short suspend*

**Details:**

The USB 2.0 PHY may hang in response to a USB wake-up event that occurs within 3 microseconds of the USB controller entering suspend. This PHY hang can only be recovered via a power cycle as warm reset is ineffectual.

**Workaround(s):**

Note: this workaround is only applicable if USB is not the primary boot mode. If USB is the primary boot mode, no workaround is available.

In order to prevent this issue from occurring, a specific order of operations must be observed during the USB controller initialization process:

1. Remove USB controller reset via the LPSC.
2. Set USB controller suspend_residency_enable field in SUSP_CTRL to '1'.
3. Proceed with normal USB controller initialization

## *i2291* — *Debug: ATB Hang during Cortex-A53 trace*

**Details**

A debugger hang may occur if an Advanced Trace Bus (ATB) flush request is sent to a Cortex-A53 core which has its clock gated via the associated PSC in the disabled or software-reset disabled (SwRstDisable) state. The issue does not occur if the Cortex-A53 core is in a powered down state.

**Workaround**

Do not perform debug trace activities on a Cortex-A53 core when in a disabled or software-reset disabled (SwRstDisabled) state. If only one Cortex-A53 core is being used in the dual core cluster, the other Cortex-A53 core should be in a powered down mode. This will allow debug trace of the active Cortex-A53 core.

## *i2413* — *Boot: HS-FS ROM boots corrupted ROM boot image*

**Details:**

ROM supports an image format in which both boot loader and TIFS images are present. This is called a combined image.

On HS-FS devices, when the combined image is signed with an RSA key, ROM is expected to:

- Skip the integrity check on the boot loader components
- Perform integrity check and signature verification on TIFS components.

Due to a bug in ROM, ROM is skipping the integrity check on the TIFS components on an HS-FS device when a non-degenerate RSA key is used.

**Workaround(s):**

Sign the X509 certificate with the RSA degenerate key for enabling the integrity check of all the components (bootloader and TIFS)

### i2414 — *Boot: Ethernet PHY Scan and Bring-Up Flow doesn't work with PHYs that don't support Auto Negotiation*

**Details:**

ROM Ethernet (either RGMII or RMII) boot mode relies on PHY auto-negotiation to complete before checking for link status. Hence PHY that do not support auto-negotiation cannot work with this boot mode.

**Workaround(s):**

None, a PHY supporting auto-negotiation is required.

### i2415 — *Boot: UART Backup Boot Authentication Failure w/ xSPI Primary Boot Mode*

**Details:**

On HS-SE device type using a flash based primary boot mode which support redundant boot address like OSPI boot mode and a secondary boot mode like UART. Under the following condition:

Boot a valid image from backup boot media (UART) with below configuration:

1. Primary Image at 0x0 => Bad Image (fails authentication)
2. Redundant Image at 0x40_0000 => Valid TIFS image but not a ROM boot (fails authentication)
3. Backup boot mode => Valid Image (Expected Image to boot)

ROM is not able to boot the Valid image from the secondary boot mode, like UART boot media.

Under normal circumstance,  after each time an image failed to boot, Secure ROM has to reset all the internal state machine for the next Retry operation.

When trying operate on the TIFS certificate, Secure ROM doesn’t reset all the necessary variables after the Image failed at Redundant offset.

Hence, during Backup boot flow Secure ROM was not able to authenticate the Certificate/ Image binary.

Due to this, Boot fails at UART backup boot for a Valid Image binary as well.

**Workaround(s):**

None, other than making sure the image located at the redundant offset is a complete boot certificate and not just a TIFS/SYSFW certificate.

### i2417 — *Boot: GPMC NAND configured to slower clock speed*

**Details:**

When using GPMC NAND boot mode the GPMCFCLKDIVIDER field of the GPMC_CONFIG1 register bit [1:0] (i.e. GPMCFCLKDIVIDER) gets set to 1 which causes a divide by 2 for the GPMC_FCLK.

The ROM uses very conservative CONFIG timing values anyways so end result may not really adversely affect throughput.

**Workaround(s):**

None.

***i2418*** ***Boot: Secure ROM Panic due to Certificate Info not present***

**Details:**

In normal boot flow ( other than Full Combined Image flow), if Certificate info ( Extended info or Legacy info ) is not present, Secure ROM will enter to an infinite loop. This condition will be observed when a certificate is given to the SOC, which has no certificate info present in it. Secure ROM will panic ( stuck in infinite loop ) on below further conditions:

1. Certificate info is not present
2. Address translation Failure
3. Hash computation failure

**Workaround(s):**

Ensure that Certificate into is present (Extended info or Legacy info).

***i2419*** ***Boot: When disabling deskew calibration, ROM does not check if deskew calibration was enabled***

**Details:**

If PLL Deskew calibration is being disabled, the ROM driver code intends to check if deskew calibration is enabled and if lock has failed. However the current code has an assignment in an if condition. As a result, it does not check if deskew calibration is enabled before clearing the config bit. There is no functional issue.

**Workaround(s):**

None

***i2420*** ***Boot: XSPI Boot time is not consistent in SFDP mode***

**Details:**

When using xSPI boot with SFDP enabled (i.e. booting in DDR mode at 25Mhz) there is boot time variation across cold or warm boot. The issue is related to asynch bridge crossing in the OSPI subsystem, it is causing a race condition between:

1. OSPI IP finishing its prefetch of data
2. The next read transaction being submitted to the OSPI IP by the TI OSPI wrapper.

This introduces enough of a delay to cause the OSPI controller to de-assert chip select hence slowing down the overall transfer.

**Workaround(s):**

None

***i2422*** ***Boot: ROM timeout for MMCSD filesystem boot too long***

**Details:**

Due to a bug in ROM if attempting to boot in SD/MMC boot (filesystem mode) from an eMMC device that is empty or erased (or factory fresh) the normal boot timeout to switch to backup boot mode will not occur as the boot gets stuck in an infinite loop until the watchdog timer reset kick in.

**Workaround(s):**

***i2422*** (continued) ***Boot: ROM timeout for MMCSD filesystem boot too long***

Need to boot from another primary boot mode to program the eMMC flash.

***i2423*** ***Boot: HS-FS ROM applies debug access restrictions to all address space covered by the efuse controller firewall***

**Details:**

On HS-FS device ROM applies debug restrictions to FWL 33 and 66 which contains secure assets. The debug access restriction was applied to the entire firewall region and not just to the region which applies to the secure asset. This prevent the use of an external emulator to be able to perform initial flash programming for example without requiring TIFS SW to in the picture.

**Workaround(s):**

TIFS SW is needed to be able to request the needed firewall to be open.

***i2431*** ***BCDMA: RX Channel can lockup in certain scenarios***

**Details:**

BCDMA RX chan Teardown can lockup channel and cannot be used for subsequent transfers if none of the TRs have EOP flag set in configuration specific flags field. Subsequently when channel is re-enabled, transfer does not not complete and terminates with various errors in TR response.

**Workaround(s):**

a) When receiving data from a PSIL/PDMA peripheral, EOP flag needs to be set in the each TR's configuration specific flag field and PDMA's 1 X-Y FIFO Mode Static TR "Z" parameter should be set to non zero value for channel teardown to function properly and cleanup the internal state memory. Otherwise it leads to channel lockups on subsequent runs. The PDMA Z count should also match the TR size, so that PDMA delineates each transfer as an individual packet. This is especially problematic in cases like where TRPD has infinite reload count set to perform cyclic transfer using a single set of TRs in streaming mode, in which case each TR could potentially be the last one.

b) If the usecase doesn't allow for PDMA Z count to be set in advance or packet EOP cannot be set then alternate is to use PKTDMA in single buffer mode instead of BCDMA.

***i2433*** ***ICSS: Reading the 64-bit IEP timer does not have a lock MSW logic when LSW is read***

**Details:**

IEPx 64-bit timestamp can be incorrect when lower 32-bit data is 0xFFFFFFFC or above (at 250MHz). In this case the upper 32-bit value is updated but lower value is the old number. The issue is seen when IEP counter (IEP_COUNT_REG1 : IEP_COUNT_REG0) is read back-to-back from ICSS PRU cores.

**Example 1:**

1st read : 0x000000D0(Upper):0xFFFFFFFC(lower)

2nd read : 0x000000D0(Upper):0x00000028(lower)

**Example 2:**

1st read : 0x000000D7(Upper):0xFFFFFFFC(lower)

***i2433*** (continued) ***ICSS: Reading the 64-bit IEP timer does not have a lock MSW logic when LSW is read***

2nd read : 0x000000D7(Upper):0x0000002C(lower)

**Example 3:**

1st read : 0x000000D6(Upper):0xFFFFFFF0(lower)

2nd read : 0x000000D7(Upper):0xFFFFFFFC((lower)

As shown above, this leads to timer increment behavior that is non-monotonic or timer differences to be unusually large as in Example 3 . This is due to 1 cycle race condition when loading 64-bit value from IEPx counter.

**Workaround(s):**

Note: these workarounds exist in SDK9.2 and later

Workaround in C for PRU:

```
uint64_t timestamp = (uint64_t) (0x2E0010);
```

/* Workaround starts here */

```
if ((timestamp & 0xFFFFFFFF) >= 0xFFFFFFFC)

{     timestamp = *(uint64_t*) (0x2E0010); }
```

/* Workaround ends here */

Workaround in assembly for PRU:

```
 ldi32 r4, 0xFFFFFFFC ; 0-4 for 250MHz clock
    ;load 64-bit timestamp to r2:r3
    lbco &r2, c26, 0x10, 8
    qbgt skip_iep_read_errata. r2, r4
    ;re-read IEP if IEP_COUNTER_LOW >= 0xFFFF_FFFC
    lbco &r2, c26, 0x10, 8
skip_iep_read_errata:
```

Workaround in C for R5F, A53:

```
uint64_t getIepTimeStamp64 (void)
{
    uint64_t u64Timestamp1 = (volatile uint64_t)(0x300AE010);
    uint64_t u64Timestamp2 = (volatile uint64_t)(0x300AE010);
    if (u64Timestamp2 > u64Timestamp1)
    {
#ifdef __DEBUG
        if (((u64Timestamp2 >> 32)-(u64Timestamp1 >> 32)) == 1)
        {
            /* HW errata fixed due to picking u64Timestamp1*/
            if ((u64Timestamp2 & 0xFFFFFFFF) >= (u64Timestamp1 & 0xFFFFFFFF))


{                 DebugP_log ("Errata fixed (1): %llx : %llx\r\n",
        u64Timestamp1, u64Timestamp2);             }

        }
#endif
        return u64Timestamp1;
    }
    else
    {
#ifdef __DEBUG
        if ((u64Timestamp2 & 0xFFFFFFFF) < (u64Timestamp1 & 0xFFFFFFFF))
```

## *i2433* (continued) *ICSS: Reading the 64-bit IEP timer does not have a lock MSW logic when LSW is read*

```
{              /* Adjust the IEP MSW in the case running into HW errata
*/
    DebugP_log ("Errata fixed (2): %llx : %llx\r\n", u64Timestamp1,
u64Timestamp2);         }

#endif
        /* HW errata fixed due to picking u64Timestamp2*/
        return u64Timestamp2;
    }
}
```

## *i2434* *ICSS: IEP RX SOF and TX SOF capture register issue in 32-bit shadow mode when nano seconds part is zero*

**Details:**

IEP timestamp can get corrupted if the nano second portion is zero. This issue is seen in 32-bit shadow mode and also seen with HW timestamp capture registers i.e. when IEPx_CAPR0_REG0(Nano seconds part is 0x00000) then IEPx_CAPR0_REG1 (milliseconds part) is retained as old value. Both RX_SOF(IEPx_CAPR0_REG0/1) capture register and TX_SOF(IEPx_CAPR4_REG0/1) capture register can be affected.

Examples:

- previous packet IEP0 timestamp at EOF: 0xE650(ms):0xf4084(ns)
- Next packet IEP0 timestamp from SOF IEP capture register: 0xE650(ms):0x00000(ns)
- Next packet IEP0 timestamp at l2 fifo drain check for 27 bytes: 0xE651(ms):0x1770(ns)

From the above, it is clear that IEP0 timestamp from capture register is incorrect because previous packet eof timestamp should be less than next packet sof timestamp.

**Workaround(s):**

If IEPx_CAPRx_REG0 (nano seconds part) from IEP capture register is 0x00000, then load the millisecond's part from IEPx_CAPRx_REG1.

## *i2435* *Boot: ROM timeout for eMMC boot too long*

**Details:**

Due to a bug in ROM, if attempting to boot in eMMC boot mode (ie, from eMMC boot partitions, sometimes referred to as eMMC alternative mode) from an eMMC device that is empty or erased (or factory fresh), the normal boot timeout to switch to backup boot mode takes 10 seconds.

**Workaround(s):**

Need to boot from another boot mode if this timeout considered too long in the system.

### i2436 — *BCDMA: BCDMA RX_IGNORE_LONG setting in RX CHAN CFG register doesn't work*

**Details:**

RX_IGNORE_LONG flag in RXCHAN CFG register of BCDMA gets ignored and BCDMA reports errors in TR response when remote endpoints don't send EOP to match TR boundary.

**Workaround(s):**

RX_IGNORE_LONG is unusable, so remote endpoint such as PDMA should close packet by sending EOP to match TR boundary (PDMA X*Y*Z should match TR ICNT0*ICNT1*ICNT2*ICNT3)

If infinite stream is desired (PDMA Z=0) then switch to PKTDMA and use Single Buffer Mode

### i2160 — *DDR: Valid VRef Range Must be Defined During LPDDR4 Command Bus Training*

**Details:**

The DDR PHY updates VREF(ca) for the command/address bus during LPDDR4 Command Bus Training (CBT). If VREF(ca) search range is set to invalid values such as no working settings can be found during CBT, the training process could fail or hang.

**Workaround(s):**

Set the following fields to known valid working values before enabling CBT.

For frequency set 0: PI_CALVL_VREF_INITIAL_START_POINT_F0 and PI_CALVL_VREF_INITIAL_STOP_POINT_F0

For frequency set 1: PI_CALVL_VREF_INITIAL_START_POINT_F1 and PI_CALVL_VREF_INITIAL_STOP_POINT_F1

For frequency set 2: PI_CALVL_VREF_INITIAL_START_POINT_F2 and PI_CALVL_VREF_INITIAL_STOP_POINT_F2

Recommendation is to use the nominal VRef value (based on the device programming of drive strength on the processor and termination in the memory) +/- 4%. Please use the online DDR Register Configuration Tool at http://dev.ti.com/sysconfig to program these registers and check the Revision History to ensure this workaround has been addressed in the version of the tool being used.

### i2482 — *Boot: ROM does not provide enough clocks during SD card initialization*

**Details:**

ROM code is not providing 74 clocks before sending first command as specified in the SD Card Physical Layer Specification ver. 2.00. This may cause SD card boot to fail, however, a failure has never been observed due to this errata on affected devices.

**Workaround(s):**

None

## Trademarks

## Revision History

**Changes from December 2, 2024 to October 31, 2025 (from Revision I (December 2024) to Revision J (October 2025))** **Page**

- Added Usage Note i2330: DDRSS Register Configuration Tool Updates........................................................6
- Added Advisory i2436; BCDMA: BCDMA RX_IGNORE_LONG setting in RX CHAN CFG register doesn't work............................................................................................................................................. 41
- Added Advisory i2160; DDR: Valid VRef Range Must be Defined During LPDDR4 Command Bus Training..41
- Added Advisory i2482; Boot: ROM does not provide enough clocks during SD card initialization..................41