Errata

AM275x Processors Silicon Errata, Silicon Revision 1.0

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

The Usage Notes Matrix lists all usage notes and the applicable silicon revisions(s). Advisories Matrix list all advisories, modules affected, and the applicable silicon revision(s).

Table 1-1. Usage Notes Matrix

ID	TITLE	SILICON REVISIONS AFFECTED
		AM275x 1.0
R5F	i2284 - Cortex-R5F Does Not Support NMI as Advertised	YES
OSPI	i2351 - OSPI Controller does not support Continuous Read mode with NAND FLASH	YES
C71x	i2424 - PLL Programming Sequence Can Introduce PLL Instability	YES

Table 1-2. Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED
		AM275x 1.0
ECC_AGGR	i2049 - Potential IP Clockstop/Reset Sequence Hand due to Pending ECC Aggregator Interrupts	YES
RAT	i2062 - RAT: Error Interrupt Triggered Even When Error Logging Disable is Set	YES
C71x	i2120 - C71x: SE Hangs on Non-Parity Error Detection in Transposed Streams with LEZR	YES
PSIL	i2137 - PSIL: Clock stop operation can result in undefined behavior	YES
OSPI	i2189 - OSPI: Controller PHY Tuning Algorithm	YES
IA	i2196 - IA: Potential deadlock scenarios in IA	YES
C71x	i2199 - C71x: SE returning incorrect data when non-aligned transposed stream crosses AM1 circular buffer boundary	YES
OSPI	i2249 - OSPI: Internal PHY Loopback and Internal Pad Loopback clocking modes with DDR timing inoperable	YES
PRG	i2253 - PRG: CTRL_MMR STAT registers are unreliable indicators of POK threshold failure	YES
MCAN	i2278 - MCAN: Message Transmit order not verified from dedicated Tx Buffers configured with same Message ID	YES
MCAN	i2279 - MCAN: Specification Update for dedicated Tx Buffers and Tx Queues configured with same Message ID	YES
USART	i2310 - USART: Erroneous clear/trigger of timeout interrupt	YES
USART	i2311 - USART Spurious DMA Interrupts	YES
MMCSD	i2312 - MMCSD: HS200 and SDR104 Command Timeout Window Too Small	YES
C71x	i2377 - RAT input signals rbytecnt and wbytecnt calculation is wrong	YES
OSPI	i2383 - OSPI: 2-byte address is not supported in PHY DDR mode	YES
CPSW	i2401 - CPSW: Host Timestamps Cause CPSW Port to Lock up	YES
C71x	i2427 - Safety: RAM SEC can cause Spurious RAM writes resulting in L2 & MBOX memory corruption	YES
C71x	i2431 - BCDMA: Rx Channel can lockup in certain scenarios	YES
eMMC	i2435 - Boot: ROM timeout for eMMC boot too long	YES
C71x	i2436 - BCDMA RX_IGNORE_LONG setting in RX CHAN CFG register doesn't work	YES
CPSW	i2438 - CPSW: Host to Ethernet Checksum Generation with VLAN ADD/Remove	YES
Pulsar	i2449 - Pulsars do not have RAT MMR Parity, Mismatch with Diagnostic RAT5	YES
PWM	i2455- PWM: The eint interrupt from all three PWM are not routed to main Pulsar	YES

1.1 Devices Supported

This document supports the following devices:

- AM275x

Reference documents for the supported devices are:

- AM275x Processors Technical Reference Manual (SPRUJC6)
- AM275x Processors data sheet (SPRSPB0)

2 Silicon Revision Usage Notes and Advisories

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

2.1 Silicon Usage Notes

i2284 ***Interrupts: Cortex-R5F Does Not Support NMI as Advertised***

Details

NMI has the purpose of preventing an indeterminate latency response to a critical interrupt which can be caused by software disabling interrupts for some amount of time. As such definition of NMI is that software cannot initiate masking the interrupt at that level.. which is FIQ on the Cortex R5F.

The R5F has the NMFI option that can be configured in the SEC MMR, however **all** interrupts are maskable in VIM. Thus a critical interrupt can still be blocked in VIM and not verified to have any sort of finite interrupt response time.

Essentially this defeats the purpose.

Other Cortex R5 implementations do support NMI by not having the ability to disabled the interrupt on one or more critical channels as part of the interrupt controller definition.

Workaround

None

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. 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 Can Introduce PLL Instability***

Details:

PLL programming sequence has been changed to provide 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.

i2424 (continued) ***PLL: PLL Programming Sequence Can Introduce PLL Instability***

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 don't likely 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. Provide 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.

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

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:

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.

i2120 ***C71x: SE Hangs on Non-Parity Error Detection in Transposed Streams With LEZR***

Details:

The C71x Streaming Engine's (SE) pipeline for returning formatted data and return report internal error information is always monitoring the tags for the data that it is working on. When an error is detected for a line of data used to format data back to the CPU, all fetching side execution for queuing up commands to go to UMC, uTLB, and the formatting pipeline back to CPU is halted.

In general operation, the only tags monitored for errors are the ones being used for the current command. For transposed mode, this is all tags touched by the current array column. A gap in suppressing internal tag monitoring causes the formatting pipeline to monitor tags that it is not currently working on while creating zero vectors for the LEZR feature. If the SE's fetching side encounters and records an error for a future column, the formatting side may notice it and halt the fetching side before the command for that column has been committed for formatting.

i2120 (continued) ***C71x: SE Hangs on Non-Parity Error Detection in Transposed Streams With LEZR***

Errors are only reported back to the CPU for commands that are internally committed for formatting, thus halting internal execution before committing the column results in no error being reported to the CPU. Because the SE has halted fetching operations without reporting an error, the CPU proceeds to hang, waiting for either return data or an error from the SE, until an unrelated external event or interrupt occurs.

Workaround(s):

The only 100% workaround is to not use stream templates with both LEZR and transposed mode enabled.

i2137 ***PSIL: Clock stop operation can result in undefined behavior***

Details:

The clock stop interface is a request/acknowledge interface used to coordinate the handshaking of properly stopping the main clock to the module. Attempting a clock stop on the module without first performing the channel teardowns or clearing of global enable bits will result in module-specific behavior that may be undefined.

The impacted modules are PDMA, SA2UL, Ethernet SW, CSI, UDMAP, ICSS, and CAL.

Workaround(s):

Before attempting to perform a clock stop operation, software is required to teardown all active channels (via UDMAP “real time” registers in the UDMAP, or PSIL register 0x408 in PSIL based modules), and after this is complete, also clear the global enable bit for all channels (via PSIL register 0x2 in both the UDMAP and PSIL based modules).

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 verify 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, 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.

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.

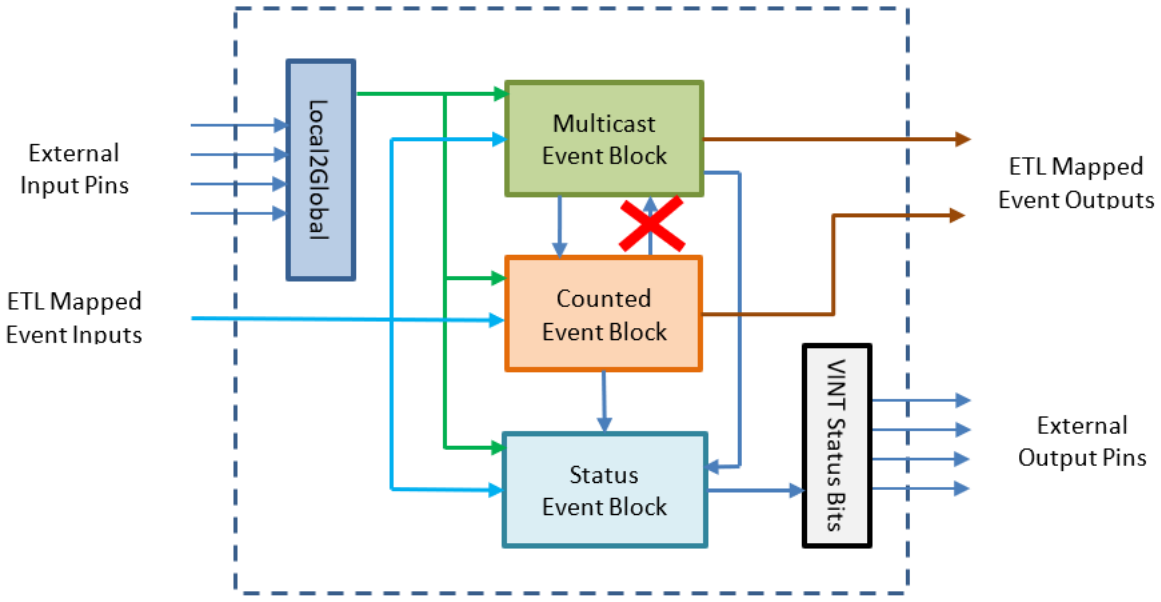

Figure 2-1. Interrupt Aggregator Version 1.0

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

i2199 ***C71x: SE returning incorrect data when non-aligned transposed stream crosses AM1 circular buffer boundary***

Details:

When AM1 refers to a larger circular buffer size than AM0, SE can reuse the wrong 64B line of data during non-aligned transposed streams. This occurs when one of the rows being transposed crosses the AM1 circular buffer boundary, but not the AM0 boundary.

Workaround(s):

Have the transposed stream either be fully aligned, meaning that the start address and all scaled DIM values be multiples of 64B, or to not configure AM1 to be a larger circular addressing buffers size than AM0.

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 2-1. 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

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).

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

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

initial clear, the ESM flags can be used as a reliable indicator of failure (or no failure) from the POKs.

i2278 ***MCAN: Message Transmit order not verified 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 can 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.

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.

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

Details:

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

Workaround(s):

i2310 (continued) ***USART: Erroneous clear/trigger of timeout interrupt***

For CPU use-case.

- If the timeout interrupt is erroneously cleared:
  - This is Valid since the pending data inside the FIFO can retrigger 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 can retrigger the timeout interrupt
  - User must provide that RX timeout behavior is in periodic mode by setting EFR2 bit6 to 1
- If timeout interrupt is erroneously set:
  - This can cause DMA to be torn down by the SW driver
  - Valid since next incoming data can cause 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).

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.

i2377 ***RAT input signals rbytecnt and wbytecnt calculation is wrong***

Details:

RAT logic inside R5SS uses rbytecnt and wbytecnt to determine if read or write access crosses the region boundaries configured by SW. The rbytecnt and wbytecnt values sent to the RAT are computed in the R5SS, but the equations used are incorrect.

This results in the RAT not detecting or erroneously detecting region boundary crossings.

Workaround(s):

Because the Cortex-R5 AXI main interface transactions do not cross a 32 byte boundary, there are no issues with region boundaries equal to or greater than 32 bytes. The issue only occurs when the region boundary is less than 32 bytes. Therefore software can not set region boundaries that are less than 32 bytes.

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.

i2427 *RAM SEC can cause Spurious RAM writes resulting in L2 & MBOX memory corruption*

Details:

In case when a memory encounters a single-bit error during a RAM read data either due to a read or a partial write transaction, the RAM will enter a state which could lead to a later spurious write to the RAM if the next "memory read" is due to a subsequent partial write transaction. If the "memory read" is instead due to an actual memory read transaction, then the lingering bad internal state would be cleared and there wouldn't be any possibility of a later spurious write. The spurious write would be to the last memory address written prior to the partial write transaction which triggers the spurious write. The issue is only applicable to MBOX & L2.

Figure 2-2 lists possible scenarios where the issue is applicable (Example 1,2,3) and not applicable (Example 4,5,6) for more clarity. Transaction# are for illustration and doesn't necessarily represent the exact cycle each operation occurs. [SEC – Single bit Error Correction, DED – Double Bit Error Detection]

Ex #	Transaction 1	Transaction 2+N N=0,1,2,3..	Transaction 2+N+1	Transaction 2+N+2
1	Read or Partial Write Addr A (SEC) ← read with SEC	Full Write Addr X ← last write prior to partial write Note: N=0	Partial Write ← Triggers spurious write	Spurious write to Addr X with Transaction 1 corrected read data of Addr A
2	Read or Partial Write Addr A (SEC) ← read with SEC	Full Write Addr B Full Write Addr C Full Write Addr D ← last write prior to partial write Note: N=2	Partial Write ← Triggers spurious write	Spurious write to Addr D with Transaction 1 corrected read data of Addr A
3	Read or Partial Write Addr A (SEC)	Partial Write Addr B Note: N=0	Spurious write to Addr A with Transaction 1 corrected read data of Addr A (Addr A is overwritten with the RAM content prior to the Transaction 1 Partial write)	
4	Read Addr A (SEC)	Partial Write Addr B Note: N=0	No Spurious write to Addr A with Transaction 1 corrected read data of address A (no data corruption)	
5	Read or Partial Write Addr A (SEC)	Read ← Clears bad internal state Note: N=0	No spurious writes with all command combinations in subsequent cycles	
6	Read or Partial Write Addr A (SEC)	Full Write Addr B Note: N=0	Read ← Clears bad internal state	No spurious writes will all command combinations in subsequent cycles

Figure 2-2.

Workaround(s):

One of the below Options can be used as workaround.

Option 1:

Disable ECC, Applicable only for non-safety application.

Option 2:

Disallow Partial writes to the memory (only perform full line writes)

In case of L2, if the L2 space is cacheable the core will perform only full line writes and this issue is not applicable.

Option 3:

The application can treat all SEC errors like a DED (no correction only detection even in case of single bit error) since there is a possibility of RAM data corruption if application can't control the transactions immediately after a single bit error on a read or partial write transaction.

Note

Prior statements about using the ECC CTRL - SEC Counter as an indicator of normal SEC issue vs spurious write are NOT VALID. After a spurious write, the ECC CTRL SEC Counter can still be 1.

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" paramater can be set to non zero value for channel teardown to function properly and cleanup the internal state memory. Otherwise there can be channel lockups on subsequent runs. The PDMA Z count can 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 can 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.

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 can take 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

i2438 *CPSW: Host to Ethernet Checksum Generation with VLAN ADD/Remove*

Details:

When the CPSW host to Ethernet checksum generation is enabled on HW and a VLAN tag is added or removed on Ethernet egress, a packet from host to Ethernet is corrupted and sent as garbage with a GOOD CRC – which is not acceptable.

Workaround(s):

i2438 (continued) ***CPSW: Host to Ethernet Checksum Generation with VLAN ADD/Remove***

VLAN tags must not be added or removed on Ethernet egress for packets that have a generated checksum.

i2449 ***Pulsars do not have RAT MMR Parity - Mismatch w. Diagnostic RAT5***

Details:

Values stored in Pulsar RAT MMRs are not parity protected while stored. This means a bit flip in the MMR even while parity protected is not detected, so there is no protection from permanent or transient errors. Checking at the initiator parity when the parity is calculated dynamically from values stored in the MMR at the time of the read only covers the interconnect.

Workaround:

User need to perform a Software readback of MMR values during runtime.

i2455 ***PWM: The eint interrupt from all three PWM are not routed to main Pulsar***

Details:

EPWM's eint interrupts are only routed to two C7xs and Device Manager (DM) R5. However, those interrupts are not routed to Main Pulsar R5FSS0 and R5FSS1.

Workaround(s):

Among any of the processors receiving eint interrupt, use one of the following methods:

- DM R5 to manage EPWM0-EPWM2 eint
- DM R5 Interrupt Service Routine to pass the PWM eint interrupt to Main Pulsar. There are two examples of how to the pass the eint interrupts below:

1. Translate the PWM eint into GPIO interrupt using dedicated GPIO pins and programming the MAIN GPIO mux to route the interrupts to MAIN Pulsar
2. Translate the PWM eint into an IPC message through Mailbox to Main Pulsar

3 Trademarks

All trademarks are the property of their respective owners.

4 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from December 2, 2024 to July 1, 2025 (from Revision * (December 2024) to Revision A (July 2025)) **Page**

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265