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

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

Table 1-2. Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED	
		AM64x/AM243x 1.0	AM64x/AM243x 2.0
Boot	i2328 — Boot: USB MSC boots intermittently	YES	YES
Boot	i2257 — xSPI boot mode redundant image boot failure	YES	NO(1)
Boot	i2307 — Boot: ROM does not properly select OSPI clocking modes based on BOOTMODE	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
DebugSS	i2317 — Debug is not available after TRST pin de-assertion when device pin EMU1 is repurposed as MCU_OBSCLK0	YES	NO(1)
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
DMA	i2285 — BCDMA: Blockcopy Gets Corrupted if TR Read Responses Interleave with Source Data Fetch	YES	NO(1)
DMA	i2320 — BCDMA and PKTDMA: Descriptors and TRs required to be returned unfragmented	YES	NO(1)
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(1)
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
JTAG	i2228 — JTAG: TAP used by Debuggers may be inaccessible if TRSTn device pin is never asserted	YES	NO(1)
MCAN	i2279 — MCAN: Specification Update for dedicated Tx Buffers and Tx Queues configured with same Message ID	YES	YES
MDIO	i2329 — MDIO: MDIO interface corruption (CPSW and PRU-ICSS)	YES	YES
OSPI	i2189 — OSPI: Controller PHY Tuning Algorithm	YES	YES
OSPI	i2303 — OSPI: OSPI0_LBCLK0 pin has input floating by default	YES	NO(1)
PCIe	i2236 — PCIe: SERDES output reference clock cannot be used	YES	NO(1)
PCIe	i2241 — PCIe: The SerDes PCIe Reference Clock Output can exceed the 5.0 GT/s Data Rate RMS jitter limit	YES	NO(1)
POK	i2277 — POK: De-Glitch (filter) is based upon only two samples	YES	NO(1)
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
USB	i2091 — 2.0 PHY hangs if received signal amplitude crosses squelch threshold multiple times within the same packet	YES	NO(1)
USB	i2134 — USB: 2.0 Compliance Receive Sensitivity Test Limitation	YES	NO(1)

(1) Pending validation

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.
  - 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

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

- 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.

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.

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.

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, 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. In order to guarantee 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 the application note spract2 (link: https://www.ti.com/lit/spract2). To sample data under some PVT conditions, it is necessary

i2189 (continued) ***OSPI: Controller PHY Tuning Algorithm***

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.

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,

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

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.

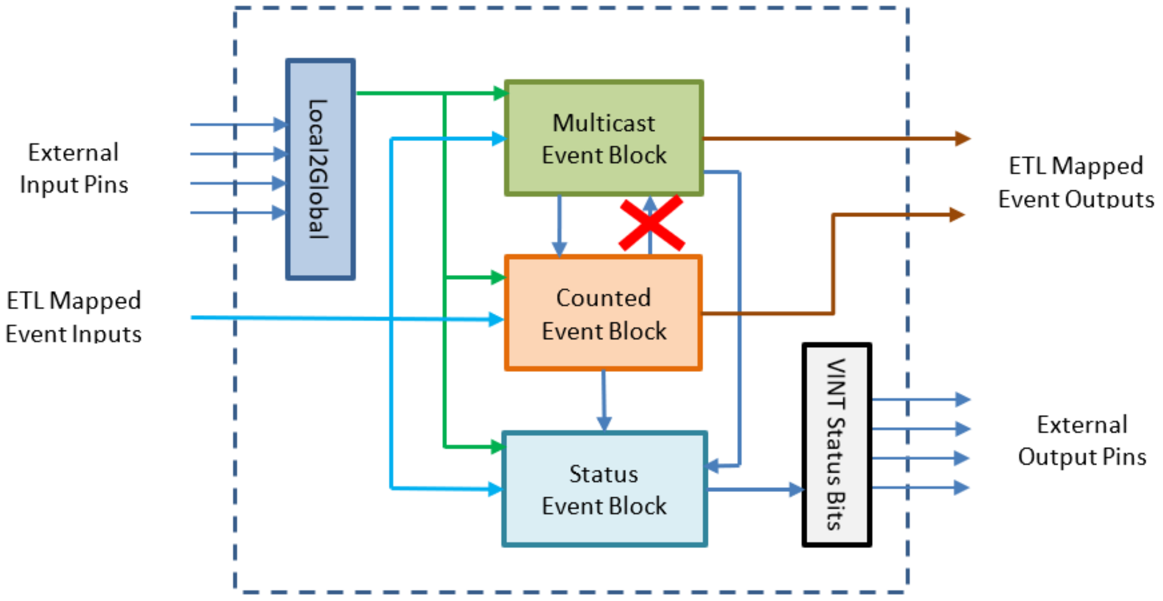

Figure 2-1. Interrupt Aggregator Version 1.0

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***

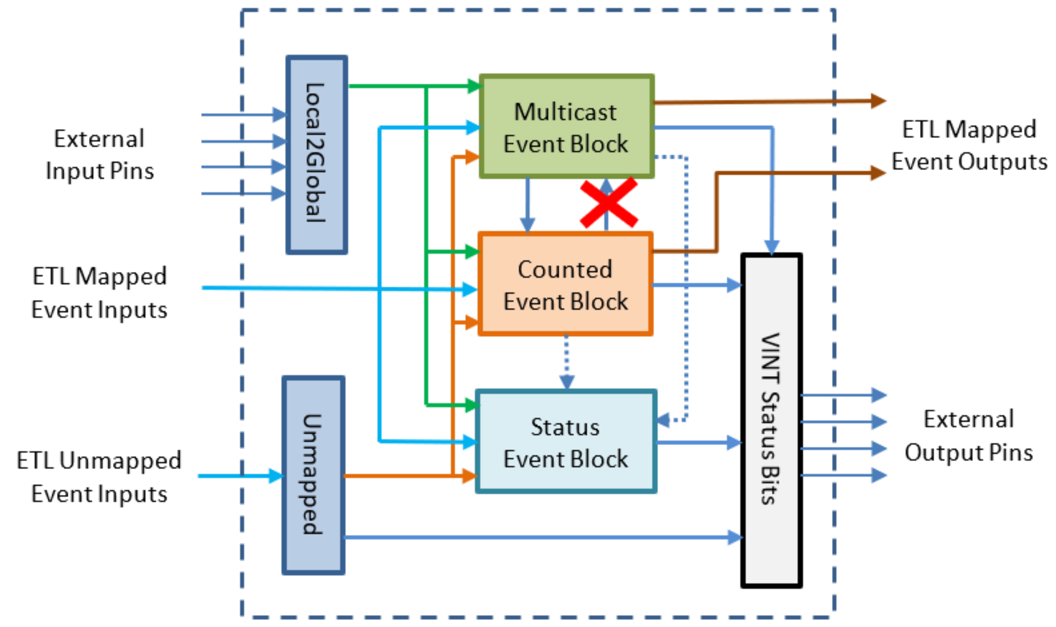

Figure 2-2. Interrupt Aggregator Version 1.1

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:

[AM64x] 3-port CPSW

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

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.

- 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).

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

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.

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

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.
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.

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
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 OK since the pending data inside the FIFO will 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

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

- 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 OK since the next periodic event will retrigger 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 will cause DMA to be torn down by the SW driver

-OK since next incoming data will 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).

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.

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 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.

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

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.

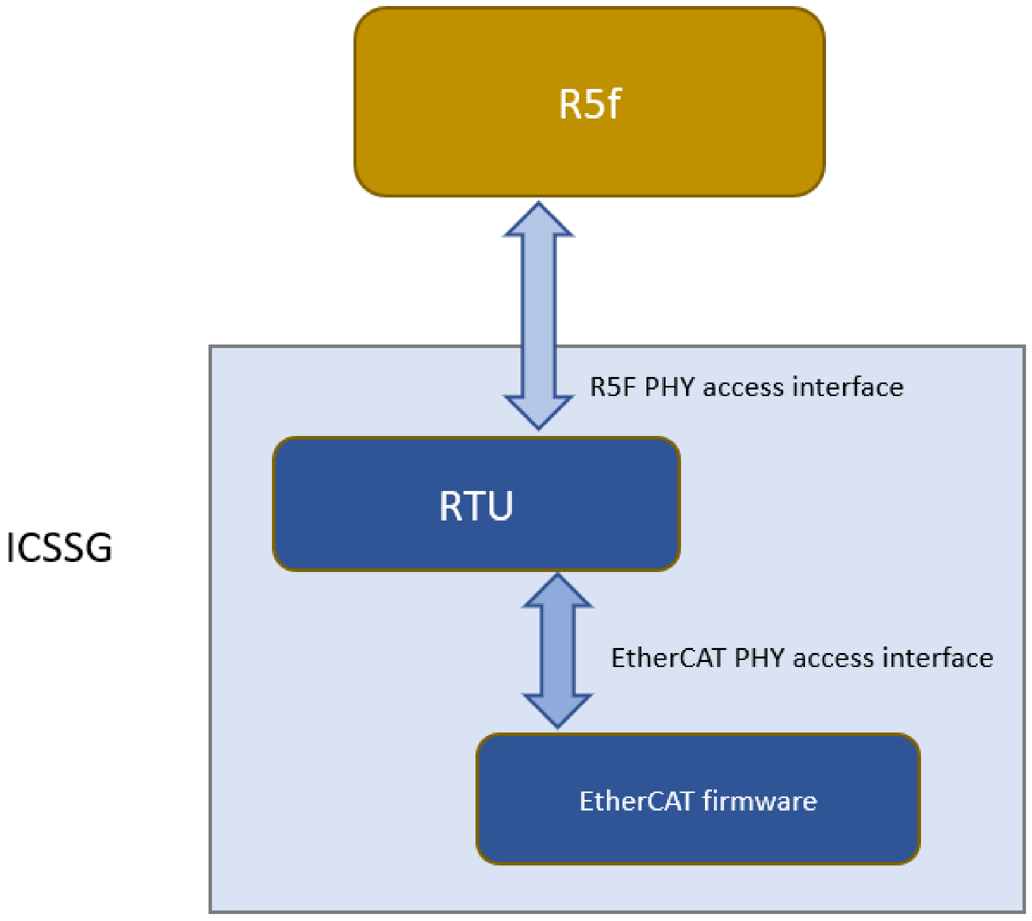

Figure 2-3. MDIO Emulation via Manual Mode using PRU Core

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.

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

Trademarks

Revision History

Changes from May 1, 2022 to July 31, 2022 (from Revision D (April 2022) to Revision E (July 2022)) **Page**