Errata

AM62x Processor 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

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
		AM62x 1.0 , AM625SIP 1.0
Boot	i2372 — ROM doesn't support select multi-plane addressing schemes in Serial NAND boot	YES
OSPI	i2351 — OSPI: Controller does not support Continuous Read mode with NAND Flash	YES
PLL	i2424 — PLL: PLL Programming Sequence May Introduce PLL Instability	YES

Table 1-2. Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED
		AM62x 1.0 , AM625SIP 1.0
BCDMA	i2431 — BCDMA: RX Channel can lockup in certain scenarios	YES
Boot	i2307 — Boot: ROM does not properly select OSPI clocking modes based on BOOTMODE	YES
Boot	i2328 — Boot: USB MSC boots intermittently	YES
Boot	i2366 — Boot: ROM does not comprehend specific JEDEC SFDP features for 8D-8D-8D operation	YES
Boot	i2371 — Boot: ROM code may hang in UART boot mode during data transfer	YES
Boot	i2410 — Boot: ROM may fail to boot due to i2409	YES
Boot	i2413 — Boot: HS-FS ROM boots corrupted ROM boot image	YES
Boot	i2414 — Boot: Ethernet PHY Scan and Bring-Up Flow doesn't work with PHYs that don't support Auto Negotiation	YES
Boot	i2415 — Boot: UART Backup Boot Authentication Failure w/ xSPI Primary Boot Mode	YES
Boot	i2416 — Boot: FAT boot partition with ESP flag doesn't boot	YES
Boot	i2417 — Boot: GPMC NAND configured to slower clock speed	YES
Boot	i2418 — Boot: Secure ROM Panic due to Certificate Info not present	YES
Boot	i2419 — Boot: When disabling deskew calibration, ROM does not check if deskew calibration was enabled	YES
Boot	i2420 — Boot: XSPI Boot time is not consistent in SFDP mode	YES
Boot	i2421 — Boot: fatTiny GPT handling causes data abort	YES
Boot	i2422 — Boot: ROM timeout for MMCSD filesystem boot too long	YES
Boot	i2423 — Boot: HS-FS ROM applies debug access restrictions to all address space covered by the efuse controller firewall	YES
Boot	i2435 — Boot: ROM timeout for eMMC boot too long	YES
CPSW	i2208 — CPSW: ALE IET Express Packet Drops	YES
CPSW	i2401 — CPSW: Host Timestamps Cause CPSW Port to Lock up	YES
DDR	i2232 — DDR: Controller postpones more than allowed refreshes after frequency change	YES
DDR	i2244 — DDR: Valid stop value must be defined for write DQ VREF training	YES
DEBUG	i2283 — Restrictions on how CP Tracer Debug Probes can be used	YES
DMA	i2320 — BCDMA, PKTDMA: Descriptors and TRs required to be returned unfragmented	YES
DSS	i2097 — DSS: Disabling a layer connected to Overlay may result in synclost during the next frame	YES
ECC_AGGR	i2049 — ECC_AGGR: Potential IP Clockstop/Reset Sequence Hang due to Pending ECC Aggregator Interrupts	YES
Internal Diagnostic Modules	i2103 — Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors	YES
Interrupt Aggregator	i2196 — IA: Potential deadlock scenarios in IA	YES
MCAN	i2279 — MCAN: Specification Update for dedicated Tx Buffers and Tx Queues configured with same Message ID	YES

Table 1-2. Advisories Matrix (continued)

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED
		AM62x 1.0 , AM625SIP 1.0
MCAN	i2278 — MCAN: Message Transmit order not guaranteed from dedicated Tx Buffers configured with same Message ID	YES
MDIO	i2329 — MDIO: MDIO interface corruption (CPSW and PRU-ICSS)	YES
MMCSD	i2312 — MMCSD: HS200 and SDR104 Command Timeout Window Too Small	YES
OSPI	i2189 — OSPI: Controller PHY Tuning Algorithm	YES
OSPI	i2249 — OSPI: Internal PHY Loopback and Internal Pad Loopback clocking modes with DDR timing inoperable	YES
OSPI	i2383 — OSPI: 2-byte address is not supported in PHY DDR mode	YES
PRG	i2253 — PRG: CTRL_MMR STAT registers are unreliable indicators of POK threshold failure	YES
RAT	i2062 — RAT: Error Interrupt Triggered Even When Error Logging Disable Is Set	YES
RESET	i2407— RESET: MCU_RESETSTATz unreliable when MCU_RESETz is asserted low	YES
RTC	i2327 — RTC: Hardware wakeup event limitation	YES
USART	i2310 — USART: Erroneous clear/trigger of timeout interrupt	YES
USART	i2311 — USART Spurious DMA Interrupts	YES
USB	i2134 — USB: 2.0 Compliance Receive Sensitivity Test Limitation	YES
USB	i2409 — USB: USB2 PHY locks up due to short suspend	YES

1.1 Devices Supported

This document supports the following devices:

- AM62x
- AM625SIP

Reference documents for the supported devices are:

- AM62x Processors Technical Reference Manual (SPRUIV7)
- AM62x Processors Data Sheet (SPRSP58)
- AM625SIP Processors Data Sheet (SPRSP98)

2 Silicon Usage Notes and Advisories

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

2.1 Silicon Usage Notes

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.

i2372 ***Boot: ROM doesn't support select multi-plane addressing schemes in Serial NAND boot***

Details:

The ROM bootloader does not support certain multi-plane Serial SPI NAND flash memories that require the read from cache/buffer command to comprehend changing the cache/buffer/plane number to access the correct data.

Workaround(s):

Carefully review the addressing requirements of a candidate flash memory for references to a special bit for selecting a plane/buffer/cache in the read from cache/buffer command. Do not use memories that have such a requirement.

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 it 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

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

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:

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

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.

i2097 ***DSS: Disabling a Layer Connected to Overlay May Result in Synclost During the Next Frame***

Details:

Disabling a layer (for example VID1) connected to an OVR (that is toggling DSS_VID_ATTRIBUTESx[0] ENABLE from 1 to 0) may result in synclost during the next frame. The synclost may result in a corrupted or blank frame (all pixel data sent out of DSS during the frame is 0x0). The occurrence of synclost is dependent on the timing of setting the GO bit (that is DSS_VP_CONTROL[5] GOBIT to 1) vis-à-vis the disabling of the layer. If the “disable layer” MMR write operation and “set GO bit” MMR write operation happens within the same frame boundary, no synclost occurs. If the operations happen across the frame boundary, then synclost occurs (for one frame). The design automatically recovers and returns to normal operation from the next frame after GO bit is set, see Figure 2-1.

i2097 (continued) ***DSS: Disabling a Layer Connected to Overlay May Result in Synclost During the Next Frame***

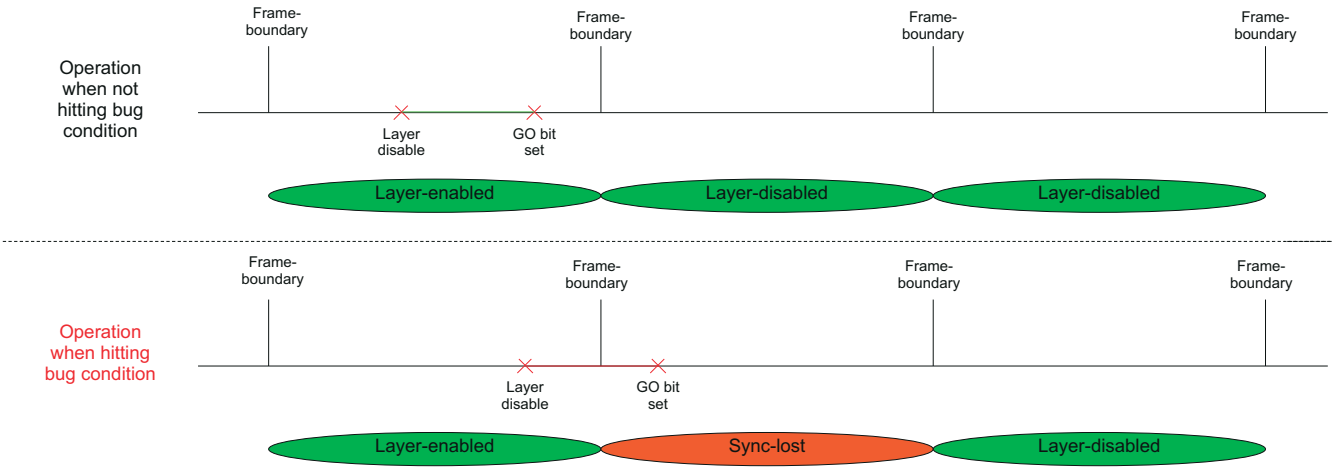

Figure 2-1. Bug Condition

Workaround(s):

A simple software workaround exists. In the workaround, prior to disabling a layer on the OVR, it is moved to the “non-visible” area of the OVR (for example: DSS_OVR_ATTRIBUTES_x[17-6] POSX = max_value_of_posx or DSS_OVR_ATTRIBUTES_x[30-19] POSY = max_value_of_posy). This avoids the synclost when the layer is disabled.

A sample software workaround pseudo-code is shown on Figure 2-2. In this case, the regular “disable layer” MMR write operation and “set GO bit set” MMR write operation are replaced with macros which implement the software workaround.

```
macro disable_layer (overlay n , layer m)
set OVR[n].ATTRIBUTES2[m].POSX = posx_max;
set OVR[n].ATTRIBUTES2[m].POSY = posy_max;
global_ovr_layer_disable_tracker[n][m] = 1;
endmacro

macro set_go_bit (vp n)
if(|(global_ovr_layer_disable_tracker[n])//any bit set
{
   set VP[n].CONTROL.GOBIT  = 1;
   Wait for 10 DSS FUNC CLK cycles;
   for (i=0;i<NUM_LAYERS;i++)
   {
     if(global_ovr_layer_disable_tracker[n][i])
     {
        Clear OVR[n].ATTRIBUTES[i].ENABLE = 0;
        global_ovr_layer_disable_tracker[n][i] = 0;
     }
   }
}
set VP[n].CONTROL.GOBIT  = 1;
endmacro
```

- Replace layer disable MMR write operation with a macro which positions the layer to the non-visible area of the OVR
- Track which layers are disabled. This will be used while GO bit is set

- Replace GO bit set MMR write operation with this macro
- First, set GO Bit for the changes in “disable_layer” macro (and any other earlier changes) to take effect
- After the first GO bit set, few idle_cycles (10 DSS functional clock cycles) are necessary before we move to the second step

- In the second step, actually disable the layers based on the previously tracked information
- Set the GO bit for the second time for the disable of the layers to take effect

Figure 2-2. Workaround Pseudo-code

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).

i2103 (continued) ***Internal Diagnostics Modules: Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors***

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.

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):

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 SPRACT2. To sample data under some PVT conditions, it is necessary 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.

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 2-3 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.

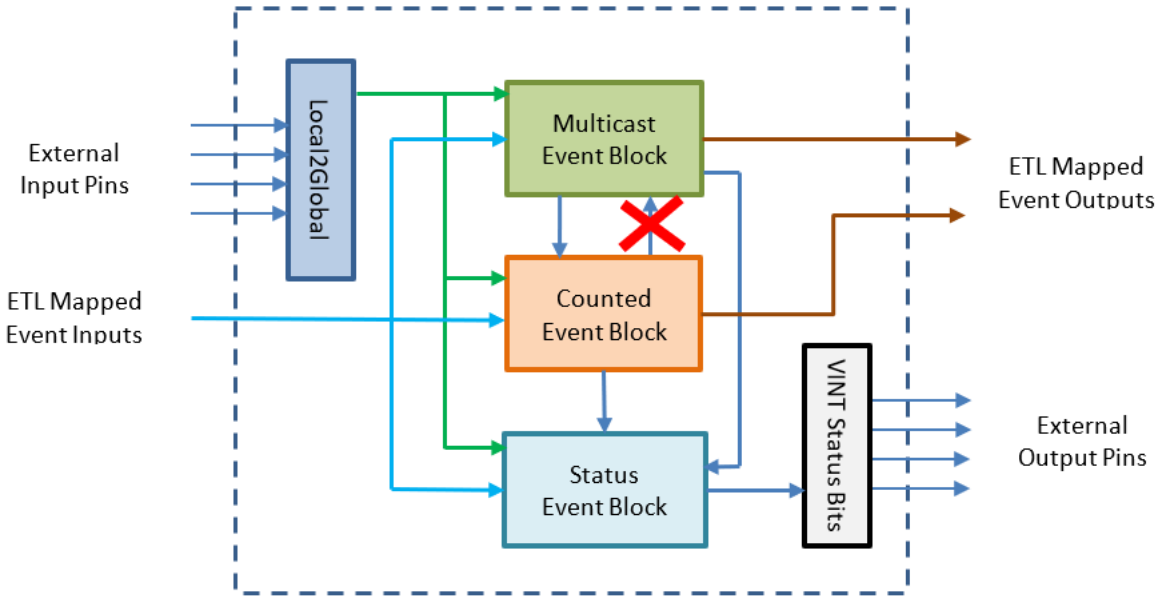

Figure 2-3. Interrupt Aggregator Version 1.0

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

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.

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 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
  - 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 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
  - Valid 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).

i2327 ***RTC: Hardware wakeup event limitation***

Details:

The RTC hardware wakeup event cannot get used if software is unable to unlock the RTC within one second after the reset to the RTC is released.

All other functionality of the RTC (eg, time of day) is not affected by this errata

Workaround(s):

None

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.

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

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

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 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

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

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)***

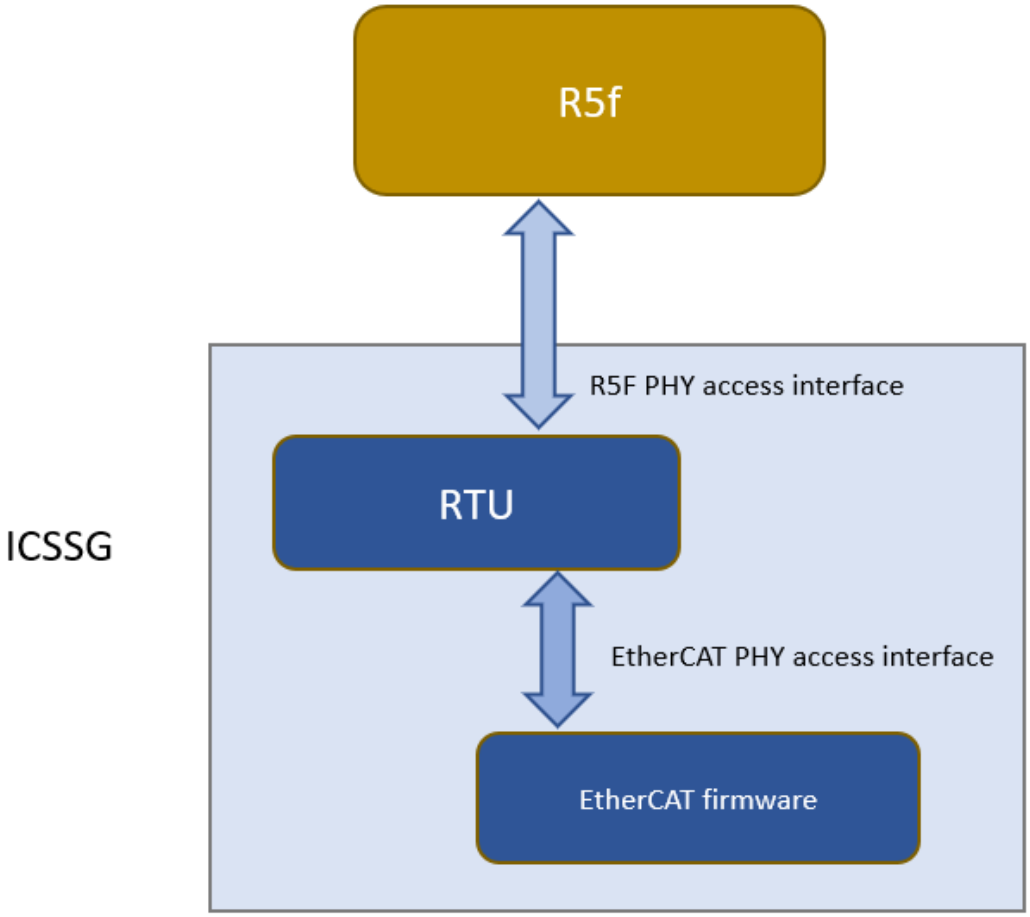

Figure 2-4. MDIO Emulation via Manual Mode using PRU Core

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

Details:

This issue impacts the following Module:

[AM62x] 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:

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

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.

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

i2278 ***MCAN: Message Transmit order not guaranteed 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

i2278 (continued) ***MCAN: Message Transmit order not guaranteed from dedicated Tx Buffers configured with 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.

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.

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 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

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

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.

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.

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: WKUP_CTRL_MMR1_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.

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.

Affected probes:

AM62x: GPMC, FSS

Workaround

none

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.

i2407 ***RESET: MCU_RESETSTATz unreliable when MCU_RESETz is asserted low***

Details:

MCU_RESETSTATz goes high periodically for a short duration and then low again while MCU_RESETz is still asserted low. This issue is seen only when MCU_RESETz is

i2407 (continued) ***RESET: MCU_RESETSTATz unreliable when MCU_RESETz is asserted low***

asserted low for greater than 100us. The device remains in reset while MCU_RESETz is low; the advisory only applies to the signal MCU_RESETSTATz.

Workaround(s):

Any one of the following could be used as a workaround for this advisory

- Do not use MCU_RESETz in a functional system. MCU_RESETz can still be used for debug, realizing the errata limitation.
- Limit the maximum low duration of MCU_RESETz to less than 100us.
- Use Main Domain RESETSTATz instead of MCU_RESETSTATz. MCU_RESETz also causes a Main reset, so Main Domain RESETSTATz could be used for device reset observation. Consult the datasheet for RESETSTATz timing specifications.
- For new designs, the circuits which produce Main Domain reset and MCU Domain reset should be combined with an AND gate as an input to RESETz. Also connect MCU Domain reset circuit to the MCU_RESETz input. This will provide full functionality of MCU warm reset using MCU_RESETz and MCU_RESETSTATz to indicate status of MCU domain reset. RESETz will be triggered on either a MAIN domain reset or MCU domain reset by using the AND gate.

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 PLL_REG12.pll_ldo_ref_en field (bit 5) in PHY2 region to '1'.
3. Set PLL_REG12.pll_ldo_ref_en_en field (bit 4) in PHY2 region to '1'.
4. Proceed with normal USB controller initialization.

i2410 ***Boot: ROM may fail to boot due to i2409***

Details:

Due to i2409, the ROM may fail to boot in USB boot mode after a warm reset. If the USB 2.0 PHY locks up, the ROM does not implement any of the workarounds listed in i2409, and thus the ROM will hang and fail to boot.

Workaround(s):

The advisory described in i2409 should be avoided by implementing one of the workarounds described in the advisory in software.

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):

i2415 (continued) ***Boot: UART Backup Boot Authentication Failure w/ xSPI Primary Boot Mode***

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

i2416 ***Boot: FAT boot partition with ESP flag doesn't boot***

Details:

If the FAT boot partition has the "ESP" flag set, which basically changes the partition ID to 0xEF, ROM would assume the partition is invalid causing boot to fail.

Workaround(s):

This partition type is not supported, make the boot partition type is FAT.

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.

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

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

i2421 ***Boot: fatTiny GPT handling causes data abort***

Details:

Attempting to boot from a GPT formatted filesystem causes the Public ROM (R5) to go into a Data Abort.At which point boot would hang until the Watchdog timer kicks in.

Workaround(s):

This GPT partition table type is not supported, make sure the partition table is MBR and the boot partition type is FAT.

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):

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

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

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.

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 will take 10 seconds.

Workaround(s):

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

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 would not complete and will terminate 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 should be set to non zero value in order 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.

Trademarks

All trademarks are the property of their respective owners.

Revision History

Changes from July 1, 2024 to December 11, 2024 (from Revision E (July 2024) to Revision F (December 2024)) **Page**

- Added Usage Note i2424; PLL: PLL Programming Sequence May Introduce PLL Instability....4
- Added Advisory i2431; BCDMA: RX Channel can lockup in certain scenarios....24

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