Errata

AM62Ax Sitara™ Processors Silicon Errata, Silicon Revision 1.0

ABSTRACT

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

Table of Contents

1 Usage Notes and Advisories Matrices

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

Usage Notes Matrix

ID	DESCRIPTION	SILICON REVISIONS AFFECTED
		AM62Ax 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

Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED
		AM62Ax 1.0
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
c71x	i2199 — C71x: SE returning incorrect data when non-aligned transposed stream crosses AM1 circular buffer boundary	YES
CPSW	i2208 — CPSW: ALE IET Express Packet Drops	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
Interrupt Aggregator	i2196 — IA: Potential deadlock scenarios in IA	YES
MCAN	i2278 — MCAN: Message Transmit order not guaranteed from dedicated Tx Buffers configured with same Message ID	YES
MCAN	i2279 — MCAN: Specification Update for dedicated Tx Buffers and Tx Queues configured with same Message ID	YES
MMCHS	i2312 — MMCHS HS200 and SDR104 Command Timeout Window Too Small	YES
OSPI	i2189 — OSPI: Controller PHY Tuning Algorithm	YES
OSPI	i2249 — OSPI: Failing OSPI DDR PHY Internal Pad Loopback and No Loopback timing modes	YES
RAT	i2062 — RAT: Error Interrupt Triggered Even When Error Logging Disable Is Set	YES
USART	i2310 — USART: Erroneous triggering of timeout interrupt	YES
USART	i2311 — USART Spurious DMA Interrupts	YES
USB	i2134 — USB: 2.0 Compliance Receive Sensitivity Test Limitation	YES

1.1 Devices Supported

This document supports the following devices:

- AM62Ax

Reference documents for the supported devices are:

- AM62Ax Processors Technical Reference Manual (SPRUJ16)
- AM62Ax Processors Data Sheet (SPRSP77)

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: Controller does not support Continuous Read mode with NAND Flash***

Details: The SoC and OSPI controller 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 should 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.

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

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

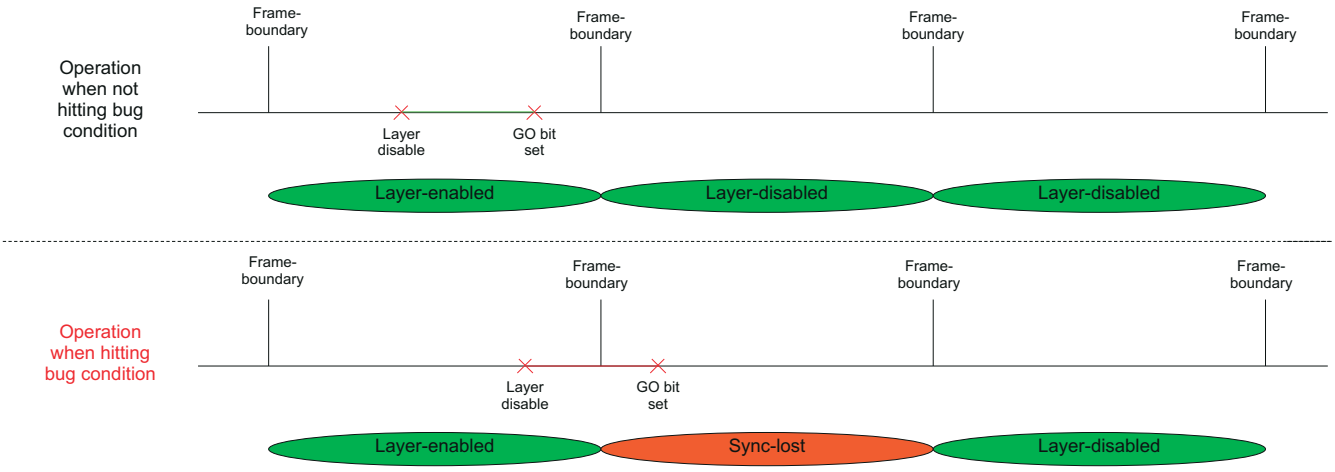

Figure 2-1. Bug Condition

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

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

i2134 (continued) ***USB: 2.0 Compliance Receive Sensitivity Test Limitation***

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 the application note spract2 (link: https://www.ti.com/lit/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,

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

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.

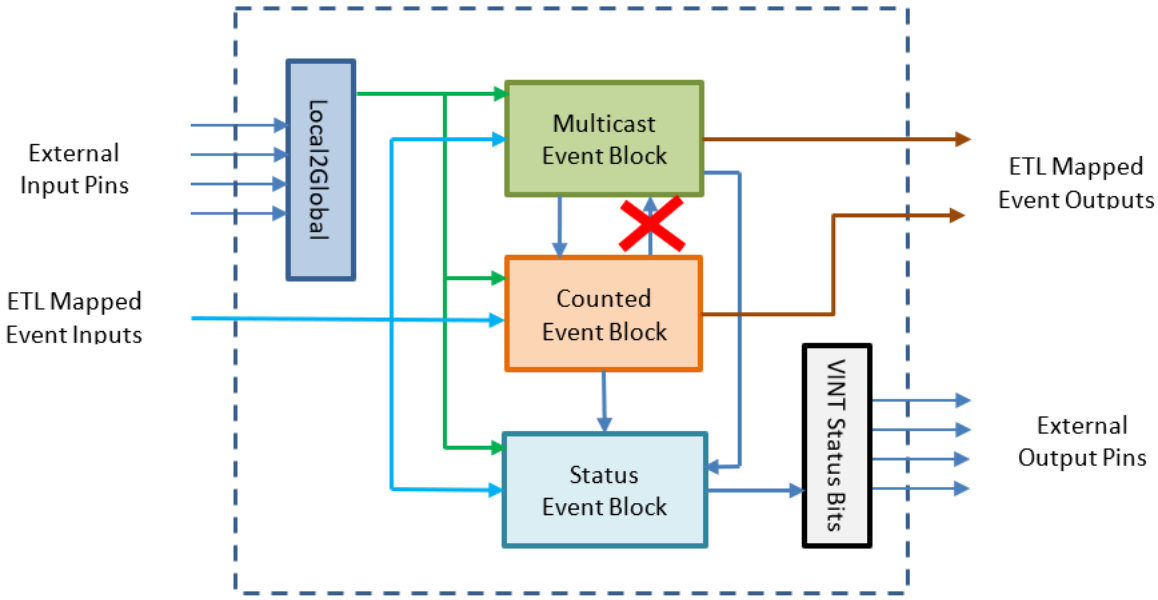

Figure 2-3. Interrupt Aggregator Version 1.0

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

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

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

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

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

Details: This issue impacts the following Module:

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.

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

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.

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
- Tx requests for these Tx Buffers are submitted sequentially with delays between each

Workaround Workaround #1:

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

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

Workaround #2:

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

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

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

Workaround Workaround #1:

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

Workaround #2:

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

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

Details: The USART 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

- 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

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

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).

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 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.

Trademarks

All trademarks are the property of their respective owners.

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