Errata

AM273x Sitara Microcontrollers Silicon Errata

ABSTRACT

This document describes the known exceptions to the functional specifications (advisories) of the AM273x microcontroller. 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 Silicon Usage Notes and Advisories Matrices 2
Devices Supported 3
2 Usage Notes and Advisories 4
Silicon Usage Notes 4
Silicon Advisories 6
3 Revision History 18

1 Silicon 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. Silicon Usage Notes Matrix

NUMBER	TITLE	SILICON REVISIONS AFFECTED		
		AM273x 1.0	AM273x 1.1	AM273x 1.2
Aurora	i2293 - Aurora Interface does not operate at maximum rated frequency if Clock lane is required or in Bypass mode of operation	YES	YES	YES
Clocks	i2324 - No synchronizer present between GCM and GCD status signals	YES	YES	YES
DSP	i2295 - Memory filter protection within DSP cannot filter access based on PrivID	YES	YES	YES
ESM	i2300 - ESM: nerror gets asserted in safety-enabled test cases when warm reset is asserted multiple times	YES	YES	YES
QSPI	i2364 - Access to address beyond 8MB is not supported in mem map mode	YES	YES	YES
PLL	i2389 - Recommended PLL configuration if locked below 1GHz	YES	YES	YES
	i2390 - Recommended HWA memInit Sequence	YES	YES	YES

Table 1-2. Silicon Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED		
		AM273x 1.0	AM273x 1.1	AM273x 1.2
Aurora	i2299 - Aurora IP first pattern should not be sync	YES	YES	YES
	i2344 - Valid udp size range is AURORA_TX_UDP_SIZE > 4	YES	YES	YES
CPSW	i2345 - Ethernet Packet corruption occurs if CPDMA fetches a packet which spans across memory banks	YES	YES	YES
CSI	i2297 - CSI Careabouts	YES	YES	YES
DMM	i2315 - DMM Careabouts while in Trace mode	YES	YES	YES
	i2318 - DMM cannot write to region which only supports privilege mode writes	YES	YES	YES
DSP	i2298 - DSP PBIST Reset changing DSP L2 clock	YES	YES	YES
	i2341 - Unallocated space access to DSP L2 - DSP IP is not blocking access to reserved space causing aliasing and L2 parity error	YES	YES	YES
DSS	i2289 - Unaligned access from DSS CM4 could cause data integrity failure and hang	YES	YES	YES
EDMA	i2288 - EDMA transfer that spans M1+M2 memories of HWA could result in data corruption	YES	YES	YES
L3	i2294 - Subsequent memory initialization configuration of L3 Bank D will not trigger a memory initialization	YES	YES	YES
MDO	i2301 - MDO: SW marker inserted at FIFO threshold location gets missed	YES	YES	YES
	i2302 - MDO: Issue seen in potential interoperability with receiver supporting on Strict Alignment User Flow Control Stripping during overflow message transmission in Aurora 64B/66B Protocol.	YES	YES	YES
	i2309 - MDO: HWA vbusm2ram sniffer address allocation logic is incorrect	YES	YES	YES
	i2329 - MDIO interface corruption (CPSW and PRU-ICSS)	YES	YES	YES

Table 1-2. Silicon Advisories Matrix (continued)

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED		
		AM273x 1.0	AM273x 1.1	AM273x 1.2
MiBSPI	i2336 - MibSPI in Peripheral Mode in 3- or 4-Pin Communication Transmits Data Incorrectly for Slow SPICLK Frequencies and for Clock Phase = 1	YES	YES	YES
	i2338 - Spurious RX DMA REQ From a Peripheral Mode MibSPI	YES	YES	YES
	i2339 - MibSPI RX RAM RXEMPTY Bit Does Not Get Cleared After Reading	YES	YES	YES
	i2340 - MibSPI RAM ECC is not read Correctly in DIAG Mode	YES	YES	YES
RAM	i2342 - 2D Stats sample value RAM processor write back issue during FFT execution on HWA	YES	YES	YES
R5FSS	i2162 - The Same Interrupt Cannot be Nested Back-2-Back Within Another Interrupt	YES	YES	YES
SPI	i2337 - A Data Length Error is Generated Repeatedly in Peripheral Mode When IO Loopback is Enabled	YES	YES	YES
PLL	i2387 - GCM circuit glitch during clock source switch	YES	YES	YES
	i2392 - Race condition in mem-init capture registers resulting in events miss	YES	YES	YES
	i2394 - Race condition in interrupt and error aggregator capture registers resulting in events miss	YES	YES	YES
CRC	i2386 - CRC 8-bit data width and CRC8-SAE-J1850 and CRC8-H2F possible use in CAN module is not supported	YES	YES	YES
MBOX	i2404 - Race condition in mailbox registers resulting in events miss	YES	YES	YES

Devices Supported

This document supports the following devices:

- AM273x

Note

All Advisories and Usage Notes are applicable to AM273x Si versions 1.0, 1.1 and 1.2 as only ROM features are updated. Following table lists the changes of ROM updates/features across Si versions.

Sl No	Si Version	Features	Comments
1	1.0	GP device	No Secure boot support
2	1.1	• Secure Boot support • Enabled HSM DMA for Crypto accelerators • Remove flash reset command support • Devboot mode support	Only ROM update
3	1.2	• Enable Assets access needed by TIFS • Update PBIST status info to Assets during ROM hand off • Support for different memory part variants	Only ROM Update

2 Usage Notes and Advisories

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

Silicon Usage Notes

i2293 ***Aurora Interface does not operate at Max rated frequency if Clock lane is required or in Bypass mode of operation***

Details

Below is the Max frequency and Skew

Mode	Value
Aurora - No Clock Lane	Max data rate : 900Mbps Multi lane skew between lanes : 600ps
Aurora + Clock Lane	Max data rate : 625Mbps Skew between Data and Clock lane : 230ps
Bypass + Clock + Frame Clock Lane	Max data rate : 450Mbps Skew between Data/Frame Clock and Clock Lane : 230ps

Workaround

None

i2295 ***Memory filter protection within DSP cannot filter access based on PrivID***

Details

Because these bits are tied to 0x0 internally, there is no option to use these signals as attributes for the transfer in memory protection within the DSP.

Workaround

None

i2300 ***ESM: nerror gets asserted in safety-enabled test cases when warm reset is asserted multiple times.***

Details

Glitches are observed on some ESM error lines just after warm reset is asserted. These glitches get captured in the ESM status registers as valid errors. After de-asserting warm reset, these errors are still pending, since these registers are cleared on power-on reset. When nerror is unmasked, these errors are seen at the nerror PAD.

Workaround

To avoid this issue, the safety enable sequence should clear all the ESM status registers at the start.

i2324 ***No synchronizer present between GCM and GCD status signals***

Details:

There is no synchronizer in between GCM and GCD, so the clock configuration register reads may be incorrect momentarily.

Severity:

Minor

Workaround(s):

Poll for the status registers change until it reflects the programmed SRC_SEL and DIV values.

i2364 ***QSPI: Access to address beyond 8MB is not supported in mem map mode***

Details:

Address lines going out from SoC interconnect to QSPI controller are 23. Hence this limits the usage of QSPI flash memory to 8MB per chip select in memmap mode.

Workaround(s):

None

i2389 ***PLL: Recommended PLL configuration if locked below 1GHz***

Details:

If PLL is locked to less than 1 GHz following settings should be used to achieve low jitter clock out from PLL.

Workaround(s):

Recommendation for Sigma Delta settings

SD divider should be programmed to 0x4
[MSS_TOPRCM:PLL_CORE_FRACDIVPLL_CORE_FRACDIV_REGSD]

Recommendation for PLL CTRL settings

SELFFREQDCO field should be programmed to 0x2
[MSS_TOPRCM:PLL_CORE_CLKCTRLPLL_CORE_CLKCTRL]

i2390 ***Recommended HWA memInit Sequence***

Details

A race condition exists in the H/W : When the S/W clears a Done status bit , if in the same cycle H/W tries to set a different Done status bit, the later Done status bit is not latched by the register and is missed causing the S/W to hang.

Workaround

The S/W shall wait for all the Done status bits (MEM_INIT_DONE[13:14]) to be set before attempting to clear the status (MEM_INIT_DONE[13:14])

Or

Attempt to perform Memint sequentially one after another.

Or

After starting memInit poll for the (MEM_INIT_STATUS[13:14] bits to go low, then only clear the status (MEM_INIT_DONE[13:14]) .

Silicon Advisories

i2162 ***R5FSS: The Same Interrupt Cannot be Nested Back-2-Back Within Another Interrupt***

Details:

The nesting (preemption) of the same high priority interrupt inside a low priority interrupt is not possible for the second and subsequent times. The second occurrence of the high priority interrupt has to wait until the program exits the lower priority interrupt service routine (ISR). The issue only occurs if the high priority interrupt following a current preemption is the same as the one which caused the original preemption. If a different interrupt preempts the low priority ISR before the second occurrence of the original higher priority interrupt then there is no issue. This issue impacts both Vector Interface Method and MMR Interface Method of interrupt handling in VIM. The issue impacts both FIQ and IRQ interrupts.

Workaround(s):

A software workaround exists. The objective of the SW workaround is to prevent back-2-back activation of the same interrupt, thereby removing the necessary condition of the bug. This can be achieved by reserving the highest priority level (Priority-0), and using that priority for a dummy interrupt (any one out of 512 interrupts available in R5FSS), and calling this dummy interrupt inside every ISR. Further, the R5FSS core itself need not enter this dummy ISR (it can be masked), only the handshake with VIM around this dummy ISR needs to happen.

A sample pseudo-code is shown below. If required, TI can provide the necessary drivers which implement this workaround.

```
any_isr_routine {
...
1:      set I/F bit in CPSR ; //so R5FSS cannot be interrupted again. I for
irq, F for fiq
2:     Trigger dummy_intr; //writing 1'b1 to Interrupt RAW Status/Set
Register bit in VIM corresponding to the chosen dummy_intr
3:       rd_irqvec; //Read IRQVEC register in VIM to acknowledge dummy_isr
4:       clear dummy_isr; //writing 1'b0 to Interrupt RAW Status/Set Register
bit in VIM corresponding to the chosen dummy_intr
5:       wr_irqvec;//Write to IRQVEC register in VIM to denote end of interrupt
6:       clear I/F bit in CPSR;
…
}
Note: Depending on where the workaround code is inserted in the ISR, step 1 &
6 may not be needed.
```

i2162 (continued) ***R5FSS: The Same Interrupt Cannot be Nested Back-2-Back Within Another Interrupt***

The draw-backs with this workaround are, Priority-0 cannot be used (only Priority 1-15 are available), and the added latency in ISR execution.

i2288 ***EDMA transfer that spans M1+M2 memories of HWA could result in data corruption***

Details

Any EDMA transfer that spans M1+M2 memories of HWA may result in data corruption without any notification of error from the SoC.

As per TPTC IP Spec, a TR is supposed to access a single target end point. M0/M1 memory banks of HWA are available via single target point and M2/M3 memory banks of HWA are available as another target point (different from that of M0/M1). Hence if a single TR is used to access a buffer spanning M1 and M2 memories of the HWA (i.e. a single buffer spanning 2 different target points), the spec is not being adhered to. This errata is explicitly highlighting this spec requirement.

Workaround

Split the access into 2 TRs so that a single TR does not span M1+M2. The 2 TRs can be chained.

i2289 ***Unaligned access from DSS CM4 could cause data integrity failure and hang***

Details

The DSS HWA CM4 cannot perform an access to an address space outside its subsystem, which is:

- Not aligned to the 32 bit boundary OR
- Not a multiple of 32 bit such 8/16 bit access

This includes and is not limited to DSP L2, DSS L3, and MSS L2 spaces.

Workaround

Ensure address is 32 bit aligned and accessed in multiples of 32 bit.

DMA the data from L2/L3 to a location inside the HWA CM4 Subsystem and then access from CM4.

i2294 ***Subsequent memory initialization configuration of L3 Bank D will not trigger a memory initialization***

Details

Memory initialization for DSS_L3 Bank D is not a write pulse high but read-write bit DSS_CTRL::DSS_L3RAM_MEMINIT_START::L3RAM3_MEMINIT_START

Workaround

To trigger a subsequent memory initialization, write 0x0 to the field before writing 0x1 to trigger memory initialization.

1. Write 0x0 to DSS_CTRL::DSS_L3RAM_MEMINIT_START::L3RAM3_MEMINIT_START
2. Write 0x1 to DSS_CTRL::DSS_L3RAM_MEMINIT_START::L3RAM3_MEMINIT_START

i2297 ***CSI Careabouts***

Details

Data corruption can occur when CSI attempts to make writes to PCR/HWA/HSM/DMM, which are not 8 byte aligned.

Workaround

CSI2 to DMM transfer should be with 1 line ping-pong enabled when DMM is operating in CSI2 mode.

Access from CSI2 to any PCR or HWA or HSM space needs to have 1 of the following constraints:

- CSI2 payload size per line should be multiple of 8 bytes.
- CSI2 should operate in 1-line ping-pong configuration.

i2298 ***DSP PBIST Reset changing DSP L2 clock***

Details

During self-test operation of DSS subsystem by DSP, L2 memories are in functional mode and clock should not be disturbed. PBIST_ST_KEY register has to be set to access PBIST controller. MMR registers are for configuration of memory self-test and control for the select line of the clock mux added on L2 clock path.

Configuring the register will create the glitch at the L2 clock mux as it dynamically switches the select line while both clocks are active.

Workaround

DSP PBIST is done by MSS.

i2299 ***Aurora IP first pattern should not be sync***

Details

Aurora has a feature called sync compression, where one can compress the number of sync patterns sent based on a configuration register value. If the very first frame is a sequence of sync pattern, and the sync pattern compression is configured to n (as example), then ideally aurora should output only n sync patterns. Due to the bug, only one sync pattern will go as output.

Workaround

None

i2301 ***MDO sw marker inserted at FIFO threshold location gets missed***

Details

Measurement Data Output (MDO) is used to capture the transactions on the bus connected from different interfaces of the AM273x device and transmit outside over LVDS (4-data lanes). MDO is comprised of a sniffer, FIFO, and an aggregator. The corresponding sniffer module sniffs a bus interface and accumulates data in the FIFO. When a FIFO threshold is reached, the data is sent out to the aggregator as a burst transfer.

An MDO source can also inject a marker indicator along with its data for tracking or other related purpose. If a marker is inserted such that it is a part of the last element of the FIFO threshold location, it will be missed.

This happens only when a sniffer other than *Sniffer 0* is used for transfer.

i2301 (continued) ***MDO sw marker inserted at FIFO threshold location gets missed***

Workaround

Multiple back to back markers (>1) can be sent out by the user to ensure at least one of them is registered by the receiver. The same sniffer configurations should be programmed to Sniffer 0 registers. This way the markers would be sent out and registered by the receiver. This workaround is only beneficial where Sniffer 0 is not in use and is idle for replicating other sniffer configurations.

Note

These workarounds are only required when using markers in operation. There are no restrictions on the sniffers when markers are not in use.

i2302 ***MDO: Issue seen in potential interoperability with receiver supporting on Strict Alignment User Flow Control Stripping during overflow message transmission in Aurora 64B/66B Protocol.***

Details

Measurement Data Output (MDO) is used to capture the transactions on the bus connected from different interfaces of the AM273x device and transmit outside over Aurora LVDS Interface (4-data lanes). MDO is comprised of a sniffer, FIFO, and an aggregator. The MDO sniffer module is responsible for monitoring the hardware interfaces in the chip and capturing the transactions on the bus which are within the configured addressing region of interest.

Data loss due to overflow can occur at the sniffer. This overflow information is sent as an interrupt to the CPU and the Aurora Tx IP. A User-Flow-Control (UFC) packet is generated by the Aurora TX IP in case of a data overflow condition in order to notify the user of this error condition. This is an error scenario and is not expected to occur in normal transfer functionality. At this stage, the data integrity is already comprised.

Aurora IP only supports UFC packet generation as per Section 6.6 of Aurora 64B/66B Protocol Specification, i.e. the UFC header block precedes the UFC data blocks. Strict Alignment User Flow Control Stripping (refer to Section 6.7 of Aurora 64B/66B Protocol Specification) is currently not supported.

Workaround

The input data rate for the MDO should be less than the output data rate so as to keep the effective data rate within the limits to avoid any overflow condition.

i2309 ***MDO - HWA vbusm2ram sniffer address allocation logic is incorrect***

Details

For MDO, while sniffing the 128KB HWA memory, the address allocation for sniffer logic is incorrect, since the address is translated by 512KB whenever 32KB boundary is crossed:

0 -32KB	no translation
32KB-64KB	address is translated by 512KB [22:19] = [18:15]
64KB-96KB	address is translated by 512KB [22:19] = [18:15]
96KB-128KB	address is translated by 512KB [22:19] = [18:15]

Workaround

While programming the address, snif_waddr[18:15] must be the same as snif_waddr[22:19].

START :END WORKAROUND START :END

i2309 (continued) ***MDO - HWA vbusm2ram sniffer address allocation logic is incorrect***

0 -32KB 0x00000000:0x00007FFF 0x00000000:0x00007FFF

32KB-64KB 0x00008000: 0x0008FFFF 0x00088000: 0x0008FFFF

64KB-96KB 0x00010000:0x00017FFF 0x00110000:0x00117FFF

96KB to 128KB 0x00018000:0x0001FFFF 0x00198000:0x0019FFFF

i2315 ***DMM Careabouts while in Trace mode***

Details

DMM can only support 32 bit/64 bit writes as initiator on interconnect.

Workaround

None

i2318 ***DMM cannot write to region which only supports privilege mode writes***

Details

All DMM writes are user-mode writes. DMM cannot write to a region which only supports privilege mode writes. This is applicable for both Trace mode and Direct Data mode.

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

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

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.

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)***

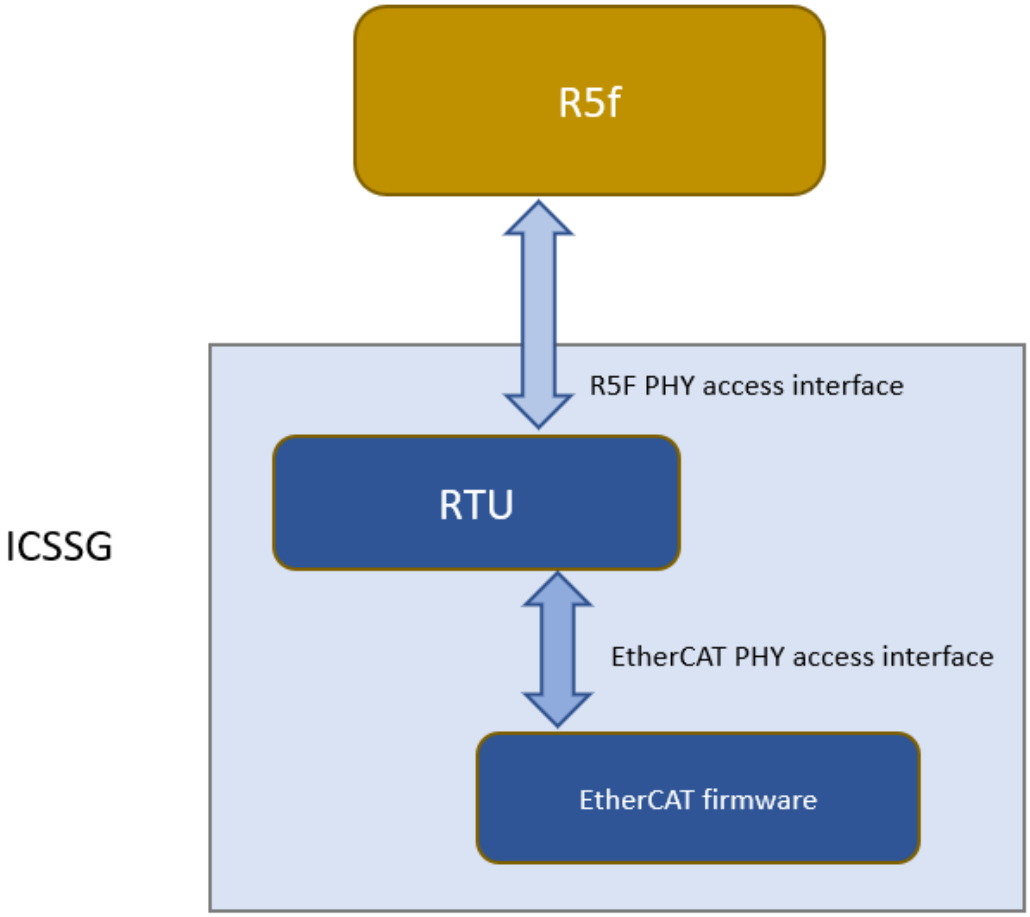

Figure 2-1. MDIO Emulation via Manual Mode using PRU Core

i2336 ***MiBSPI: MibSPI in Peripheral Mode in 3- or 4-Pin Communication Transmits Data Incorrectly for Slow SPICLK Frequencies and for Clock Phase = 1***

Details:

The MibSPI module, when configured in multibuffered peripheral mode with 3-functional pins (CLK, SIMO, SOMI) or 4-functional pins (CLK, SIMO, SOMI, nENA), could transmit incorrect data when all the following conditions are met:

- MibSPI module is configured in multibuffered mode
- Module is configured to be a peripheral in the SPI communication
- SPI communication is configured to be in 3-pin mode or 4-pin mode with nENA
- Clock phase for SPICLK is 1
- SPICLK frequency is MSS_VCLK frequency / 12 or slower

Workaround(s):

The issue can be avoided by setting the CSHOLD bit in the control field of the TX RAM (Multi-Buffer RAM Transmit Data Register). The nCS is not used as a functional signal in this communication; hence, setting the CSHOLD bit does not cause any other effect on the SPI communication.

i2337 ***SPI: A Data Length Error is Generated Repeatedly in Peripheral Mode When IO Loopback is Enabled***

Details:

When a DLEN error is created in Peripheral mode of the SPI using nSCS pins in IO Loopback Test mode, the SPI module re-transmits the data with the DLEN error instead of aborting the ongoing transfer and stopping. This is only an issue for an IOLPBK mode peripheral in Analog Loopback configuration, when the intentional error generation feature is triggered using CTRLDLENERR (IOLPBKTSTCR.16).

Workaround(s):

After the DLEN_ERR interrupt is detected in IOLPBK mode, disable the transfers by clearing the SPIEN (bit 24) in the SPIGCR1 register and then, re-enable the transfers by resetting the SPIEN bit.

i2338 ***MibSPI: Spurious RX DMA REQ From a Peripheral Mode MibSPI***

Details:

A spurious DMA request could be generated even when the SPI Peripheral is not transferring data in the following condition sequence:

- The MIBSPI is configured in standard (non-multibuffered) SPI mode, as a Peripheral
- The DMAREQEN bit (SPIINT0.16) is set to enable DMA requests
- The Chip Select (nSCS) pin is in an active state, but no transfers are active
- The SPI is disabled by clearing the SPIEN (SPIGCR1.24) bit from '1' to '0'

The above sequence triggers a false request pulse on the Receive DMA Request as soon as the SPIEN bit is cleared from '1' to '0'.

Workaround(s):

Whenever disabling the SPI, by clearing the SPIEN bit (SPIGCR1.24), first clear the DMAREQEN bit (SPIINT0.16) to '0', and then, clear the SPIEN bit.

i2339 ***MibSPI: MibSPI RX RAM RXEMPTY Bit Does Not Get Cleared After Reading***

Details:

The RXEMPTY flag may not be auto-cleared after a CPU or DMA read when the following conditions are met:

- The TXFULL flag of the latest buffer that the sequencer read out of transmit RAM for the currently active transfer group is 0
- A higher-priority transfer group interrupts the current transfer group and the sequencer starts to read the first buffer of the new transfer group from the transmit RAM
- Simultaneously, the Host (CPU/DMA) is reading out a receive RAM location that contains valid received data from the previous transfers

Workaround(s):

If at all possible, avoid transfer groups interrupting one another. If dummy buffers are used in lower-priority transfer groups, select the appropriate "BUFMODE" for them (like, SKIP/DISABLED) unless, there is a specific need to use the "SUSPEND" mode.

i2340 ***MibSPI: MibSPI RAM ECC is not read Correctly in DIAG Mode***

Details:

A Read operation to the ECC address space of the MibSPI RAM in DIAG mode does not return the correct ECC value for the first 128 buffers, if the Extended Buffer support is implemented, but the Extended Mode is disabled for the particular MibSPI instance.

i2340 (continued) ***MibSPI: MibSPI RAM ECC is not read Correctly in DIAG Mode***

Workaround(s):

None.

i2341 ***DSP: Unallocated space access to DSP L2 - DSP IP is not blocking access to reserved space causing aliasing and L2 parity error***

Details:

The DSP IP is sending out an access to its L2 memory for access beyond the configured DSP L2 memory size of 384KB (reserved space access) ie beyond 0x8085 FFFC.

Reserved Memory locations from 0x80860000 to 0x8087FFFC is accessible to read and write. Memory Locations from 0x80860000 to 0x8087FFFC are aliased at 0x80840000 to 0x8085FFFC and 0x80850000 to 0x8085FFFC is replicated at 0x80870000 to 0x8087FFFC, hence the actual L2RAM is of 384KB only.

If parity is enabled, an L2 Parity error is observed for reads to reserved locations beyond 0x80860000 -0x8087FFFC.

Workaround(s):

Configuring the MPU : (L2MPPA24- L2MPPA31) to 0

Write access to reserved space will be blocked. No Aliasing & No L2 Parity Error. This ensures the data integrity of valid L2 Region is maintained

Read access to reserved space still leads to L2 Parity Error(If Parity is enabled).

Debug access(Read & Write) are not blocked: Still leads to Aliasing + L2 Parity Error : Its not feasible to block the debug access despite configuring the MPPA registers for Protection enabled

Memory Protection Fault Address Register(0184 A000h:: L2MPFAR/0184 AC00h:: L1DMPFAR) are populated with the address which are blocked(beyond 384KB boundary in this case) & still accessed

Address(L2MPFAR/L1DMPFAR) & Status(L2MPFSR/L1DMPFSR) Registers are required to be cleared for the next read using Clear registers(L2MPFCR/L1DMPFCR) with values 1

Observations(Both when L1D Cache Enabled/Disabled)

For Read : MPU Protection Errors are observed on L1D with L1MPFAR registers populated with the blocked address access

For Write : MPU Protection Errors are observed on L2 with L2MPFAR registers populated with the blocked address access

i2342 ***RAM: 2D Stats sample value RAM processor write back issue during FFT execution on HWA***

Details:

If the processor is writing back some values into the 2D stats sample value RAM and simultaneously an FFT paramset is being executed, then the processor write to the 2D sample RAM is corrupted with data being written into incorrect addresses.

Workaround(s):

Disable context switch for the paramset which has MAX 2D enabled.

i2344 ***Aurora: Valid udp size range is AURORA_TX_UDP_SIZE > 4***

Details:

In Aurora 64b66b mode ([TOP_AURORA_TX| AURORA_TX_CONFIG:AURORA_TX_CONFIG_PROTOCOL_SEL = 1), TOP_AURORA_TX:AURORA_TX_UDP_SIZE=4 and AURORA_TX_UDP_CONFIG_PACK_MODE_SEL = 1 (TWP) is not supported.

For Aurora64b66b and TWP pack mode, other udp sizes (eg 1,2,3,5,6,7,8……so on) are supported, only udp size=4 isn't supported.

Workaround(s):

None. For Aurora64b66b and TWP pack mode, other udp sizes (eg 1,2,3,5,6,7,8……so on) are supported, only udp size=4 isn't supported.

Therefore we need to ask users/customers to not use udp_size=4 with this combination. *Therefore valid udp sizes are -*

Aurora 8b10b and Aurora 64b66b-

* AURORA_TX_UDP_CONFIG_PACK_MODE_SEL = 0 (Bytes) : Valid Udp sizes - AURORA_TX_UDP_SIZE = 8, 12, 16, 20.....so on

* AURORA_TX_UDP_CONFIG_PACK_MODE_SEL = 1 (TWP) : Valid Udp sizes - AURORA_TX_UDP_SIZE = 5, 6, 7, 8.....so on

i2345 ***CPSW: Ethernet Packet corruption occurs if CPDMA fetches a packet which spans across memory banks***

Details:

Each memory bank in SoC has a separate memory controller. Even though memory addresses are contiguous, each bank is a separate entity with a separate controller.

If a memory bank received a memory request say 32 bytes and address of memory request is 16 bytes before end of memory bank, the behavior of the memory controller will be:

When the memory controller encounters end of memory bank after 16 bytes it will wrap around and give 16 bytes from the start of the memory bank.

This results in the packet corruption.

Workaround(s):

Ensure from application side single ethernet packet does not span across memory banks.

i2387 ***PLL: GCM circuit glitch during clock source switch***

Details:

GCM circuit [highlighted] is susceptible to glitch when switching the clock source from Crystal to the PLL Clock, causing the phase misalignment between SYS clocks resulting in random behaviors like aborts or hangs or access failures. Reference below for HSDIV0 feeding clock to R5F and SYS Clocks.

i2387 (continued) ***PLL: GCM circuit glitch during clock source switch***

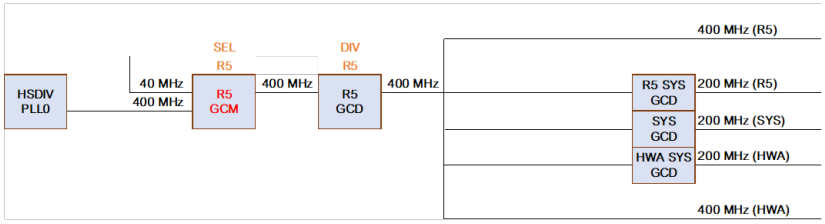

Figure 2-2. PLL

Workaround(s):

1: Use external WDT to reset in case of hang scenarios.

2: Use staggered PLL programming sequence [Step 1 to Step 5] to switch from 40MHz-200MHz – 400MHz in SBL. Application where R5F Running at 400MHz (Core PLL HSDIV0CLKOUT0 output is 400MHz)

Step 1: Program MSS_CR5_CLK_SRC_SEL, MSS_CR5_DIV_VAL and SYS_CLK_DIV_VAL ‘000 ; // Switch back to XTAL

Step 2: Program MSS_CR5_DIV_VAL ‘111; // Suppress the glitch

Step 3: Program SYS_CLK_DIV_VAL ‘111 ; // R5F and SYS clocks are in 1:2 ratio

Step 4: Program MSS_CR5_CLK_SRC_SEL ‘222; // Switch to PLL clock , switch to 200MHz

Step 5: Program MSS_CR5_DIV_VAL ‘000 ; // Switch back to 400MHz

Application where R5F Running at 200MHz (Core PLL HSDIV0CLKOUT0 output is 200MHz)

Step 1: Program MSS_CR5_CLK_SRC_SEL, MSS_CR5_DIV_VAL and SYS_CLK_DIV_VAL ‘000 ; // Switch back to XTAL

Step 2: Program MSS_CR5_DIV_VAL ‘111; // Suppress the glitch

Step 3: Program SYS_CLK_DIV_VAL ‘000 ; // R5F and SYS clocks are in 1:1 ratio

Step 4: Program MSS_CR5_CLK_SRC_SEL ‘222; //Switch to PLL clock , switch to 100MHz

Step 5: Program MSS_CR5_DIV_VAL ‘000 ; //Switch back to 200MHz

i2392 ***Race condition in mem-init capture registers resulting in events miss***

Details:

Potential race condition in capture registers resulting in events getting lost while other events in the same register are being cleared by writing to the register. Following registers are impacted by this issue:

MSS_CTRL ,DSS_CTRL, DSS_HWA_CFG : *MEMINIT_DONE registers

Workaround(s):

Any of the following Workarounds can be used:

Sequentially trigger the mem-init and clear the status before triggering the new mem-init. This is needed if both the status are in the same register.

(OR)

If parallel triggers are must then poll for the all status-bits that got triggered to be 1'b1 and then go and clear the DONE status register

i2392 (continued) ***Race condition in mem-init capture registers resulting in events miss***

(OR)

Check the MEM_INIT_STATUS register after starting the mem-init and wait the status to go -low by checking it in regular interval and finally clear the DONE status register when the status goes low

i2394 ***Race condition in interrupt and error aggregator capture registers resulting in events miss***

Details:

Potential race condition in capture registers resulting in events getting lost while other events in the same register are being cleared by writing to the register. Following registers are impacted by this issue:

MSS_CTRL: *INTAGG_STATUS_REG, *TPCC_ERR/INTAGG_STATUS_RAW

Workaround(s):

Follow below steps in ISR:

1) Before exiting the ISR read the *_ERRAGG_RAW and check the bit-validity by "anding" with *_ERRAGG_MASK.

2) If any bit is set that-implies there is a interrupt/Error which got missed while clearing the *_ERRAGG_STATUS.

3) Service the corresponding bit in ISR and then exit the ISR. So ISR should be exited after both STATUS and "RAW&MASK" are zero

i2386 ***CRC: CRC 8-bit data width and CRC8-SAE-J1850 and CRC8-H2F possible use in CAN module is not supported***

Details:

CRC types CRC8-SAE-J1850 and CRC8-H2F are not supported for 8-bit data width. Minimum data width supported is 16-bit.

Workaround(s):

No workaround. It is recommended to not to use the above mentioned unsupported polynomials.

i2404 ***Race condition in mailbox registers resulting in events miss***

Details:

Potential race condition in capture registers resulting in events getting lost while other events in the same register are being cleared by writing to the register. Following registers are impacted by this issue:

MSS_CTRL: *_MBOX_READ_REQ

MSS_CTRL: *_MBOX_READ_DONE

Workaround(s):

Read the status(READ DONE / READ_DONE_REQ) of the other processor to check any interrupt is in flight before setting up the trigger (WRITE DONE /READ ACK) event (OR)

Re-trigger the (WRITE DONE /READ ACK) event if the status (READ DONE / READ_DONE_REQ) is not received within the given time

Trademarks

All trademarks are the property of their respective owners.

3 Revision History

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

Changes from December 1, 2023 to April 30, 2025 (from Revision B (December 2023) to Revision C (April 2025)) **Page**

- Advisories: Added i2404 to the advisory matrix....2
- MBOX: Added a new item - i2404....17

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