AM65x/DRA80xM Processors Silicon Revision 2.0, 1.0

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.

Topic **Page**

1 Usage Notes and Advisories Matrices

Table 1 lists all usage notes and the applicable silicon revision(s). Table 2 lists all advisories, modules affected, and the applicable silicon revision(s).

Table 1. Usage Notes Matrix

ID	DESCRIPTION	SILICON REVISIONS AFFECTED			
		AM65x		DRA80xM	
		2.0	1.0	2.0	1.0
i2033	Section 3.1.1 — Fail-Safe IO's: Latch-up Risk on Fail-Safe IOs	Yes	Yes	Yes	Yes
i2082	Section 3.1.2 — ADC: High Input Leakage Current May Impact ADC Accuracy		Yes		Yes
i2007	Section 3.1.3 — INTRTR: Spurious Interrupts Generated when Programming Certain Interrupt Routers	Yes	Yes	Yes	Yes

Table 2. Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED			
		AM65x		DRA80xM	
		2.0	1.0	2.0	1.0
DSS	i2000 — DSS: DSS Does Not Support YUV Pixel Data Formats		Yes		Yes
UDMA-P	i2004 — UDMA-P: UDMA-P Host Packet Descriptor's "0x3FFFFF" Packet Length Mode not Functional		Yes		Yes
UDMA-P	i2006 — UDMA-P: UDMA-P Real-Time Remote Peer Registers not Functional Across UDMA-P Domains		Yes		Yes
DDRSS	i2009 — DDRSS: DDR Controller ECC Scrubbing Feature Can Cause DRAM Data Corruption	Yes	Yes	Yes	Yes
On-chip Debug	i2013 — On-Chip Debug: The Assertion of Warm Reset Coinciding with a Debug Configuration Access Targeting the STM Subsystem May Result in a Hang of Said Debug Configuration Access	Yes	Yes	Yes	Yes
On-chip Debug	i2015 — On-Chip Debug: CPTracer Bus Probes MAIN_CAL0_0 and MCU_SRAM_SLV_1 are not able to Distinguish between Secure and Non-secure Transactions	Yes	Yes	Yes	Yes
MCAN	i2017 — MCAN: Message Transmitted with Wrong Arbitration and Control Fields (Early Start of Frame)	Yes	Yes	Yes	Yes
DCC	i2018 — DCC: Incorrect Counter Values in DCC Operation		Yes		Yes
Boot, USB3SS	i2019 — Boot, USB3SS: Boot ROM Does Not Support USB Host MSC (Mass Storage Class) Boot Mode		Yes		Yes
Boot, USB3SS	i2020 — Boot, USB3SS: Boot ROM Does Not Support USB Device Firmware Upgrade (DFU) Boot Mode		Yes		Yes
MSMC	i2021 — MSMC: Non-Coherent Memory Access to Coherent Memory Can Cause Invalidation of Snoop Filter		Yes		Yes
DDRSS	i2022 — DDRSS: Independent Impedance Control for Address/Control and Data Bus Lanes is Not Available		Yes		Yes
RINGACC, UDMA	i2023 — RINGACC, UDMA: RINGACC and UDMA Ring State Interoperability Issue after Channel Teardown		Yes		Yes
MMCSD	i2024 — MMCSD: MMCSD Peripherals Do Not Support HS400	Yes	Yes	Yes	Yes
IO, MMCSD	i2025 — IO, MMCSD: Incorrect IO Power Supply Connectivity Prevents Dynamic Voltage Change on VDDSHV6 and VDDSHV7		Yes		Yes
MMCSD	i2026 — MMCSD: Negative Current from UHS-I PHY May Create an Over-Voltage Condition on VDDS6 and VDDS7 Which Exposes the Device to a Significant Reliability Risk		Yes		Yes
CPSW	i2027 — CPSW: CPSW does not support CPPI receive checksum (Host to Ethernet) offload feature		Yes		Yes
USB3SS	i2028 — USB3SS: SuperSpeed USB Non-Functional		Yes		Yes
Boot, UART	i2030 — Boot, UART: UART Boot Mode Never Times Out		Yes		Yes
DSS	i2032 — DSS: DSS DPI Interface does not support BT.656 and BT.1120 output modes		Yes		Yes
PCIe	i2037 — PCIe: PCI-Express May Corrupt Inbound Data		Yes		Yes
Boot	i2038 — Boot: FAT16 Fails When Root Block Resides in More Than One Cluster		Yes		Yes

Table 2. Advisories Matrix (continued)

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED			
		AM65x		DRA80xM	
		2.0	1.0	2.0	1.0
DSS	i2039 — DSS: Frame Buffer Flip/Mirror Feature Using RGB24/BGR24 Packed Format Can Result in Pixel Corruption	Yes	Yes	Yes	Yes
VTM	i2053 — VTM: Software Reads from On-Die Temperature Sensors Can Be Corrupted		Yes		Yes
RINGACC	i2054 — RINGACC: Reads from GCFG Region Can Cause Spurious RAM ECC Errors	Yes	Yes	Yes	Yes
CC_ARMSS	i2069 — CC_ARMSS: Powering Down CC_ARMSS1 Causes System Data Corruption		Yes		Yes
DCC	i2073 — DCC: Suspend Mode Not Functional		Yes		Yes
USB2PHY	i2075 — USB2PHY: USB2PHY Charger Detect is Enabled by Default Without VBUS Presence		Yes		Yes
CPTS	i2083 — CPTS: GENF (and ESTF) Reconfiguration Issue		Yes		Yes
CPSW	i2084 — CPSW: CPSW Does Not Support Intersperced Express Traffic (IET – P802.3br/D2.0) In 10/100Mbps Mode		Yes		Yes
UART	i2096 — UART: Spurious UART Interrupts When Using DMA	Yes	Yes	Yes	Yes
DSS	i2097 — DSS: Disabling a Layer Connected to Overlay May Result in Synclost During the Next Frame	Yes	Yes	Yes	Yes
SA2_UL	i2098 — SA2_UL: Auth/Decrypt Operations with 2nd Input Thread Does Not Send the DMA Packet Out		Yes		Yes
Cortex-R5F	i2099 — Cortex-R5F: Deadlock Might Occur When One or More MPU Regions is Configured for Write Allocate Mode	Yes	Yes	Yes	Yes
GIC	i2101 — GIC: ITS Misbehavior		Yes		Yes
Safety Modules	i2103 — Safety Modules: Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors	Yes	Yes	Yes	Yes
PCIe	i2104 — PCIe: GEN3 (8GT/s) Operation Not Supported	Yes	Yes	Yes	Yes
OSPI	i2115 — OSPI: OSPI Boot Doesn't Support Some xSPI Modes or xSPI Devices	Yes	Yes	Yes	Yes
HyperFlash	i2119 — HyperFlash: HyperFlash is Not Supported	Yes	Yes	Yes	Yes
Cortex-R5F	i2129 — Cortex-R5F: High Priority Interrupt is Missed by VIM	Yes	Yes	Yes	Yes
Cortex-R5F	i2132 — Cortex-R5F: Interrupt Preemption (Nesting) is Unavailable if Using VIM Vector Interface for Interrupt Handling	Yes	Yes	Yes	Yes
MSMC	i2149 — MSMC: MSMC Scrubber Only Targets Bottom 16 of 32 Ways of SRAM/L3$	Yes	Yes	Yes	Yes

2 Nomenclature, Package Symbolization, and Revision Identification

2.1 Device and Development-Support Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all microprocessors (MPUs) and support tools. Each device has one of three prefixes: X, P, or null (no prefix) (for example, AM6546). Texas Instruments recommends two of three possible prefix designators for its support tools: TMDX and TMDS. These prefixes represent evolutionary stages of product development from engineering prototypes (TMDX) through fully qualified production devices and tools (TMDS).

Device development evolutionary flow:

X — Experimental device that is not necessarily representative of the final device's electrical specifications and may not use production assembly flow.

P — Prototype device that is not necessarily the final silicon die and may not necessarily meet final electrical specifications.

null— Production version of the silicon die that is fully qualified.

Support tool development evolutionary flow:

TMDX — Development-support product that has not yet completed Texas Instruments internal qualification testing.

TMDS — Fully-qualified development-support product.

X and P devices and TMDX development-support tools are shipped against the following disclaimer:

"Developmental product is intended for internal evaluation purposes."

Production devices and TMDS development-support tools have been characterized fully, and the quality and reliability of the device have been demonstrated fully. TI's standard warranty applies.

Predictions show that prototype devices (X or P) have a greater failure rate than the standard production devices. Texas Instruments recommends that these devices not be used in any production system because their expected end-use failure rate still is undefined. Only qualified production devices are to be used.

For additional information how to read the complete device name for any AM654x, AM652x, and DRA80xM devices, see the specific-device Datasheet (SPRSP08, SPRSP31).

2.2 Devices Supported

This document supports the following devices:

- AM6526
- AM6528
- AM6546
- AM6548
- DRA802M
- DRA804M

2.3 Package Symbolization and Revision Identification

Figure 1 shows an example of package symbolization.

Table 3 lists the device revision codes.

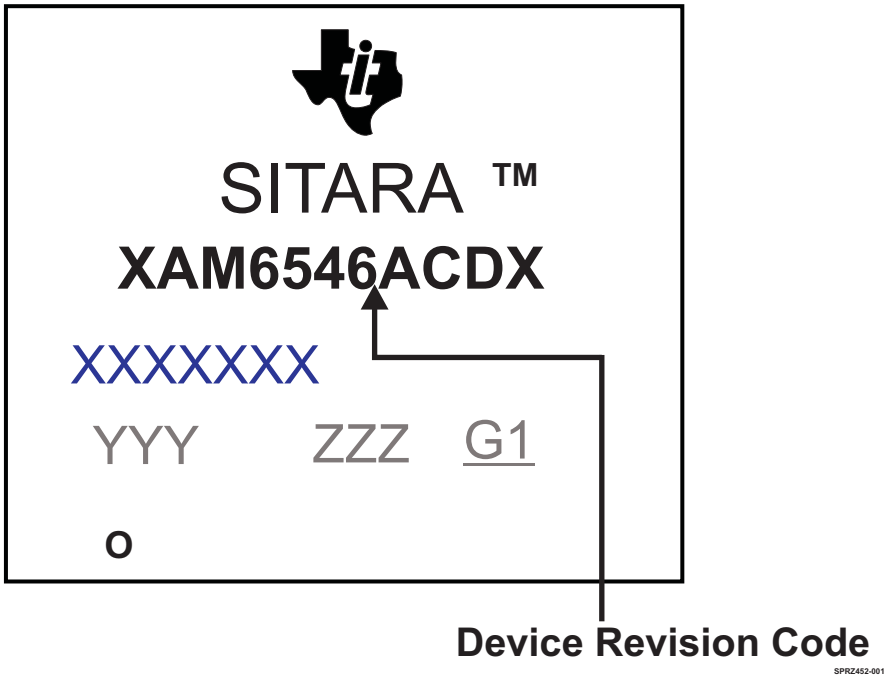

Figure 1. Package Symbolization

Table 3. Revision Identification

DEVICE REVISION CODE	SILICON REVISION	COMMENTS
BLANK	1.0	Available as null.
A	2.0	

3 Silicon Revision 2.0, 1.0 Usage Notes and Advisories

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

3.1 Silicon Revision 2.0, 1.0 Usage Notes

3.1.1 Fail-Safe IO's: Latch-up Risk on Fail-Safe IOs

AM65x silicon revision 1.0 and 2.0 and DRA80xM silicon revision 1.0 and 2.0 incorporate fail-safe I/O's on several pins (I2C0_SCL, I2C0_SDA, I2C1_SCL, I2C1_SDA, and NMIn). The fail safe I/O's tolerate voltage applied on the pins before their respective I/O supply voltage is ramped up. There is a potential latch up risk based on design reviews of the fail-safe I/O pins when driven high during functional mode. This latch-up risk is not yet confirmed through silicon characterization. To avoid the risk of a latch-up condition, the following steps should be implemented, depending on the mux mode used on the fail safe I/O. If the fail safe I/O is used in an I2C mux mode, then an external pull-up resistor (> 1 kOhm) is required on the signal. If the fail safe I/O is used in any other mux mode, then an external series resistor (> 1 kOhm) should be placed on this signal (close to the SoC).

3.1.2 ADC: High Input Leakage Current May Impact ADC Accuracy

On AM65x silicon revision 1.0 and DRA80xM silicon revision 1.0, ADC input leakage current may be higher than expected at worst case Process/Voltage/Temperature (PVT) conditions, where process variation and operating temperature are the major contributors. Leakage current is larger for strong process devices operating at elevated temperatures.

There is also a dependency on the potential applied to an ADC input. Leakage current flows out of the ADC when applying a potential equal to VSS, flows into the ADC when applying a potential equal to VDDA_ADC_MCU, and the direction change occurs at approximately 42% of VDDA_ADC_MCU. Magnitude of leakage current has a non-linear function to the applied potential, where it increases exponentially as the applied potential approached VSS or VDDA_ADC_MCU.

Significant error can be introduced in ADC measurements when high impedance sources are connected to inputs with high leakage. This occurs because the input leakage current introduces a voltage drop across the source impedance. For example, the ADC would measure a potential of 1.45 V when measuring a 1.5 V source with 1 kΩ output impedance that is connected to an ADC input with 50 µA of leakage flowing into the ADC input. The error of this measurement would be 50 mV, which is 3.3% lower than the expected value. Reducing the source impedance from 1 kΩ to 100 Ω in this example would reduce the measurement error to 0.33%.

There are design techniques that can be used to minimize the impact of input leakage.

Reduce impedance of sources connected to ADC inputs. For example, it may be necessary to buffer outputs of a high impedance sources with voltage-follower operational amplifier circuits.

Design static DC sources to apply a nominal potential of approximately 42% of VDDA_ADC_MCU. For example, this approach can be used to minimize leakage when monitoring a DC power source via a resistor voltage divider.

3.1.3 INTRTR: Spurious Interrupts Generated when Programming Certain Interrupt Routers

On AM65x silicon revision 1.0 and 2.0 and DRA80xM silicon revision 1.0 and 2.0, programming the MUXCNTL_n registers to configure input-output mapping of interrupt signals may result in a short glitch on the intended output signal, causing a spurious interrupt. Additionally, reprogramming the register to the same value may also cause a glitch.

The *Interrupt Routers* section in the device TRM also describes this behavior, which is applicable to multiple Interrupt Routers (INTRTR) instantiated in the device, including:

- WKUP_GPIOMUX_INTRTR0
- GPIO_INTRTR0
- MAIN2MCU_LVL_INTRTR0

- MAIN2MCU_PLS_INTRTR0
- TIMESYNC_INTRTR0
- CMPEVT_INTRTR0

To prevent system from servicing these spurious interrupts unintentionally, following programming sequence are recommended when configuring INTR interrupt mapping.

In systems where interrupt mapping is static, typically with RTOS or bare-metal programming, the following interrupt configuration sequence shall be followed:

1. Disable interrupt at GIC and VIM for the interrupt sourced from the INTRTR output;
2. Re-configure MUXCNTL_n on the specific INTRTR;
3. Clear spurious interrupt if any by clearing raw status;
4. Enable the interrupt at GIC and/or VIM.

In systems where static interrupt configurations not possible, such as Linux systems with standard GIC drivers, interrupt drivers must detect false interrupt caused by the glitch, and clear the false interrupt. This method can be performed by the following programming sequence:

- If possible to customize interrupt handler, the handler shall first clear any interrupts before the handler being initialized, otherwise,
- The handler must check the interrupt source (such as GPIO status) to check valid event exists, then service the interrupt.

In systems with shared IRQs where multiple INTRTRs map to the same GIC IRQ, the following pseudo code may be used for the global interrupt handler:

```
isr(irq)
{     if (!read_status_reg())        return IRQ_NONE; }
```

In this case, each ISR can check and report to global IRQ handler that it wasn't the cause of IRQ, allowing the global IRQ handler to call the next handler in the list for that IRQ. In case of spurious IRQ, all the handlers (if there are no events) will return IRQ_NONE, which means Linux kernel will report a spurious IRQ on that line as the global handler will report EOI.

3.2 Silicon Revision 2.0, 1.0 Advisories

Table 4. Silicon Revision 2.0, 1.0 Advisory List

Table 4. Silicon Revision 2.0, 1.0 Advisory List (continued)

i2000

DSS: DSS Does Not Support YUV Pixel Data Formats

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: Table 5 lists the pixel data formats that are not supported in the DSS.

Table 5. Unsupported Pixel Data Formats

Format #		Pixel Formats	Pipline support		Component Bit Depth
Packed	Planar		Video	Video_Lite	
0x3E	N/A[1]	YUV422-YUV2	X	X	8/10/12
0x3F	N/A	YUV422-UYVY	X	X	8/10/12
N/A	0x3D	YUV422-NV12	X	X	8/10/12
N/A	N/A	YUV422-NV21	X	X	8/10/12

(1) N/A - Not Applicable

Selecting the above formats (see Table 5) along with using the color conversion logic inside Video/Video_Lite pipeline will result in incorrect color reproduction at the output of DSS.

Workaround(s): Customer may use RGB mode for image capture or configure the input camera to use RGB mode instead of YUV formats referenced above. If RGB mode not available, it is recommended to use processor core or GPU to convert YUV to RGB.

i2004

UDMA-P: UDMA-P Host Packet Descriptor's "0x3FFFFF" Packet Length Mode not Functional

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: In the UDMA-P Host Packet Descriptor (PD Word 0), there is a 22-bit packet length field. This field is typically set to a value between 0x0 – 0x3FFFFE, representing the total byte length of the packet. If the packet length is less than the sum of the buffer length, the packet will be truncated to the size specified in the packet length field. Specifying a packet length that is greater than the sum of the buffers will result in an error.

Alternatively, the packet length field can be set to a special value, 0x3FFFFF, which disables truncation and allows the UDMA-P to transmit as much data as is specified in the Buffer Length fields. This "0x3FFFF" mode is not functional.

Workaround(s): There is no workaround. A valid packet length field must be supplied. This only prevents a user from sending a packet larger than 0x3FFFFE bytes, which is more than large enough for any envisioned use case.

i2006

UDMA-P: UDMA-P Real-Time Remote Peer Registers not Functional Across UDMA-P Domains

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: In the UDMA-P, both RX and TX channels contain a specific set of registers called "Real-time Remote Peer Registers" (UDMA_PEER[0-15]_j registers). These registers provide access to the remote peer device's real time registers within the PSI-L configuration (PSIL_CFG) register space, address 0x400 to 0x40F. The peer device is the device that the UDMA-P channel is communicating with, and is set via the Rx/TX Channel Destination Thread ID Mapping Register (UDMA_THREAD_j register). Once the UDMA_THREAD_j register is configured with the peer's PSI-L thread ID, any access to the UDMA_PEER[0-15]_j registers on the UDMA-P will results in an access to the PSI-L register on the peer corresponding to the offset of the register accessed. For example, peer register 0 maps to peer PSI-L address 0x400, and peer register 1 maps to peer PSI-L address 0x401, and so on. Having these registers allows the UDMA-P driver to access peer registers in the 0x400 to 0x40F range without accessing the configuration proxy IP, which in some environments can be reserved for secure access only.

There are two UDMA-P instances on the device. These instances exist in separate domains called MAIN and MCU. Along with the two UDMA-P instances, individual peripheral devices also reside in both domains. It was originally intended that a UDMA-P instance in one domain would seamlessly work with a peripheral in the other domain. For the most part, this is still the case, but when it comes to using the peer registers, a bug in the internal message routing prevents the UDMA-P peer registers in the MAIN domain from accessing the PSI-L registers of a peripheral in the MCU domain, and prevents the UDMA-P peer registers in the MCU domain from accessing the PSI-L registers of the peripheral in the MAIN domain. Read results across UDMA-P domains

will always be invalid. Writes across UDMA-P domains will still take affect but will take longer than normal.

Workaround(s): Avoid using the UDMA-P peer registers in one domain to access PSI-L registers of a peripheral in the other domain. This can be accomplished in one of two ways:

1. Always allocate a TX/RX channel from the UDMA-P residing in the same domain as the target peripheral.
2. Use the PSI-L proxy module to write PSI-L registers 0x400 through 0x40F instead of the using the UDMA-P Real-time Remote Peer registers.

Solution 1 avoids the problem by making sure that the peer register access is never performed “across domains.”

Solution 2 avoids the problem by not making use of the affected registers. The drawback here is that in some environments the PSI-L proxy module is restricted to secure access only. This means that any register access will have to be done through secure code, increasing overhead.

Note, solution 1 is preferred due to secure access limitation for this resource configuration when implementing the workaround in software.

i2009

DDRSS: DDR Controller ECC Scrubbing Feature Can Cause DRAM Data Corruption

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The DDR controller implements a built-in ECC scrubbing feature that is used for correcting single-bit errors in the DRAM. When this feature is enabled (DDRCTL_ECCCFG0[4] DIS_SCRUB = 0), the controller schedules an ECC scrub operation when a single-bit error is detected. However, in some cases, the ECC scrub operation can lead to accidental auto-correction of two-bit errors in an another DRAM location, thus corrupting the data inside the DRAM.

Workaround(s): The ECC scrubbing feature inside the DDR controller must be kept disabled at all times by setting DIS_SCRUB = 1 in the DDR controller's ECC Configuraion 0 (DDRCTRL_ECCCFG0) register.

i2013

On-Chip Debug: The Assertion of Warm Reset Coinciding with a Debug Configuration Access Targeting the STM Subsystem May Result in a Hang of Said Debug Configuration Access

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: Debug configuration transaction initiators and targets are typically susceptible only to power-on-resets. The STM Subsystem includes the Arm® STM500 and a CoreSight™ CTI that provides triggering support. The orthogonal debug interconnect for the STM Subsystem is affected by warm reset. This results in the possibility of a hang of a debug configuration access that is active when warm reset is active..

Workaround(s): There is no workaround for this issue. The user needs to be aware of the possibility that a debug configuration hang may result in the rare case that the STM Subsystem is being configured while warm reset is active.

i2015

On-Chip Debug: CPTracer Bus Probes MAIN_CAL0_0 and MCU_SRAM_SLV_1 are not able to Distinguish between Secure and Non-secure Transactions

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The CPTracer Bus Probe supports filtering on the properties of bus transactions including a property 'psecure' that identifies a transaction as being secure or non-secure. In the case of bus probe instances supporting MAIN_CAL0_0 and MCU_SRAM_SLV_1, the 'psecure' property will always indicate that a transaction is non-secure even if it were actually secure.

Workaround(s): There are no workarounds for this issue. The user needs to be aware that secure and non-secure transactions are not distinguishable for MAIN_CAL0_0 and MCU_SRAM_SLV_1 CPTracer Bus Probes.

i2017

MCAN: Message Transmitted with Wrong Arbitration and Control Fields (Early Start of Frame)

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The erratum is limited to the case when MCAN is in state "Receiver" (MCAN_PSR[4:3] ACT = "10") with no pending transmission (no MCAN_TXBRP bit set) and a new transmission is requested before the 3rd bit of Intermission is reached and this 3rd bit of intermission is seen dominant. This issue occurs only with disturbed CAN bus.

Under the following conditions, a message with wrong ID, format, and DLC is transmitted:

- MCAN is in state "Receiver" (MCAN_PSR[4:3] ACT = "10"), no pending transmission
- A new transmission is requested before the 3rd bit of Intermission is reached
- The CAN bus is sampled dominant at the third bit of Intermission which is treated as SoF (see ISO 11898-1:2015 Section 10.4.2.2).

Under the conditions listed above, it may happen, that:

- The shift register is not loaded with ID, format, and DLC of the requested message
- The MCAN will start arbitration with wrong ID, format, and DLC on the next bit
- In case the ID wins arbitration, a CAN message with valid CRC is transmitted
- In case this message is acknowledged, the ID stored in the Tx Event FIFO is the ID of the requested Tx message and not the ID of the message transmitted on the CAN bus, no error is detected by the transmitting MCAN
- If the message loses arbitration or is disturbed by an error, it is retransmitted with correct arbitration and control fields.

Workaround(s): **Workaround 1:**

Request a new transmission only if another transmission is already pending or when the MCAN is not in state "Receiver" (when MCAN_PSR[4-3] ACT ≠ "10").

To avoid activating the transmission request in the critical time window between the sample points of the 2nd and 3rd bit of Intermission, following can be done:

- Evaluate the Rx Interrupt flags MCAN_IR[19] DRX, MCAN_IR[0] RF0N, MCAN_IR[4] RF1N which are set at the last bit of EoF when a received and accepted message gets valid. Prevent the transmission of any message within 3 bit times after detecting the Rx interrupt flags.

Workaround 2:

A checksum covering arbitration and control fields can be added to the data field of the message to be transmitted, to detect frames transmitted with wrong arbitration and control fields.

Workaround 3:

Set MCAN_CCCR[0] INIT bit, add transmission request for the message and then clear MCAN_CCCR[0] INIT bit. This needs to be done for each message to be transmitted.

Workaround 4:

Keep the number of pending transmissions always at ">0" by frequently requesting a message so that the condition "no pending transmission" is never met. The frequently requested message may be given a low priority, losing arbitration to all other messages.

In case, where all the nodes present on the CAN bus have this workaround implemented, at least two nodes shall have this frequently sent low priority message with different priorities/identifiers. Rest of the nodes can have one of these priorities/identifiers (same) as the priority of this frequently sent message.

Comparison within available Workarounds is shown in Table 6:

Table 6. Comparison within available Workarounds

Workaround #	Advantages	Disadvantages
1	• Easy to implement. • Fix is only limited to a local node with the erratum.	• Only viable in interrupt context. • Effectiveness is vastly dependent on interrupt service latency. • Application becomes complicated.
2	• Simple and easy to implement.	• All nodes on the bus must implement this WA. Restricting use of device for end points and all the existing nodes needs to be updated to support this WA.
3	• Simple and easy to implement.	• Node will miss any messages received during this sequence. • System (including existing nodes) needs to be updated for detecting the dropped/missed message.
4	• DMA can be used to send the frequently sent low priority message to reduce CPU load. • Most effective and viable amongst available workarounds. • Fix is only limited to a local node with the erratum.	• 100% CAN Bus utilization at all times. • Increased CPU load in case CPU is used to send the message.

i2018

DCC: Incorrect Counter Values in DCC Operation

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: Due to a potential race condition, if the DCCCNT0, DCCCNT1, DCCCNTSEED0, or DCCCNTSEED1 registers are modified while a DCC module is in operation or immediately before a DCC module is put in to operation, the counter may not expire correctly as per the programmed counter value (expiring either late or early). This could result in an incorrect error signal value, indicating either a false pass or fail.

Workaround(s): All DCC configuration fields, especially counter seeds and values, should be programmed while the DCC module is disabled. A user may also read back the last written counter value in the DCCCNT0 and DCCCNT1 registers before enabling the DCC module to ensure that counter values have had ample time to settle before the

DCC module is enabled.

i2019

Boot, USB3SS: Boot ROM Does Not Support USB Host MSC (Mass Storage Class) Boot Mode

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The Boot ROM does not support USB Host MSC (Mass Storage Class) Boot mode.

Workaround(s): None.

i2020

Boot, USB3SS: Boot ROM Does Not Support USB Device Firmware Upgrade (DFU) Boot Mode

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The Boot ROM does not support USB Device Firmware Upgrade (DFU) Boot mode.

Workaround(s): None.

i2021

MSMC: Non-Coherent Memory Access to Coherent Memory Can Cause Invalidation of Snoop Filter

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: Snoop filter can be invalidated when different system masters access the same memory location with different coherency attributes. The MSMC snoop filter retains knowledge of memory cached locally by master (for example, Arm Cortex®-A53 local cache). A coherent transaction performed by another system master through the MSMC will check the snoop filter and ensure the locally-cached data is accessed for transaction, and that the memory and cache is kept coherent. If, however, a non-coherent transaction is performed by another master through MSMC, it will invalidate the snoop filter entry for the locally-cached data and future transactions will not perform a snoop on the cached contents and the memory and cache are no longer coherent.

Workaround(s): The workaround to avoid this scenario is to control the accesses to cacheable memory. During software initialization (for example, exception level startup) on different A53 clusters, the cache view to memory needs to be controlled such that it is consistent between clusters. When one cluster is performing a non-coherent access to memory, software on the other cluster must ensure that it does not have the memory location in its local cache. This applies to any SoC device variants where there are two A53 clusters in the SoC.

For IO transactions with DMA, it is required that all DMA masters perform coherent transactions only to any memory that is cached locally by an A53 core, unless it is accessing a globally non-coherent memory space. This applies to all SoC device variants, irrespective of the number of A53 clusters in the SoC.

i2022

DDRSS: Independent Impedance Control for Address/Control and Data Bus Lanes is Not Available

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The IO impedance calibration control for the address/control segment and the data segment in the DDR PHY are incorrectly mapped to the same set of control registers. This results in requiring a common set of impedance settings for output driver impedance and on-die termination for both the address/control IO signals as well as the data IO signals. As a result, only register DDRPHY_ZQ0PR0 can be used to program output driver impedance and on-die termination for both address/control and data.

Workaround(s): Program DDRPHY_ZQ0PR0 with appropriate values that will satisfy output driver impedance and on-die termination for both address/control and data signals. Use the DDR configuration tool to help determine the most optimal values. This tool will also facilitate the configuration of all DDR controller and PHY registers for optimal performance. Also, ensure that all layout recommendations are met as described in the AM65x/DRA80xM DDR Board Design and Layout Guidelines Application Report (SPRACI2).

i2023

RINGACC, UDMA: RINGACC and UDMA Ring State Interoperability Issue after Channel Teardown

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The Ring Accelerator (RINGACC) and the Unified DMA Controller (UDMA) each maintain their own state information about rings (queues). However, when a ring is reset in the RINGACC, the UDMA is unaware of this operation and its state information for that ring is not reset. As a result, the UDMA may believe a ring still has valid pending entries, while the RINGACC does not, and will potentially read invalid information from a recently reset and enabled ring.

Resetting a ring is a required operation (following DMA channel teardown operations and before re-enabling the DMA channels) when re-configuring an existing ring for another purpose (that is changing the ring mode, DMA channels, channel type, etc.).

Workaround(s): The software workaround is to use the ring-mode doorbell functionality to cause the UDMA occupancy counter for a given ring to increment and wrap back to 0. Since a ring's UDMA occupancy counter is 21-bits wide, and the highest doorbell ring count that can be written per register write is 127, the workaround requires a maximum of ((2**22) / 127) = 33,027 writes to the doorbell register. Note that a negative value cannot be used as the doorbell ring count value as the UDMA ignores negative values (a negative doorbell ring count value is seen by the UDMA as a ring pop count of abs(value) elements which it ignores in its ring state accounting).

To implement the software workaround, the following sequence should be used:

1. Read the ring occupancy (RINGACC_OCC_j [CNT]). If the ring occupancy is not 0, then steps 2-6 need to be executed to implement the workaround.
2. Reset the ring by writing any value in the RINGACC CFG registers (that is RINGACC_SIZE_j = RINGACC_SIZE_j).
3. Read the ring mode (RINGACC_SIZE_j [QMODE]) to determine the "adjusted ring occupancy count" used in step 5.
    a. If the ring is configured for exposed ring mode or messaging mode, the "adjusted

ring occupancy count" is equal to the ring occupancy.

b. If the ring is configured in credentials mode or queue manager (QM) mode, the "adjusted ring occupancy count" is equal to the ring occupancy divided by 2. This is required because when in credentials mode or QM mode, each ring write increases the ring occupancy by 2 elements (one entry for the credentials, one entry for the data). However, the UDMA-P's local occupancy counter only records the number of writes, and the ring occupancy, therefore, needs to be divided by 2 to convert back to the number of doorbell rings needed.

4. Setup the ring in exposed ring/doorbell mode, if not already in this mode (RINGACC_SIZE_j [QMODE] = 0).
5. Ring the doorbell (2**22 – (adjusted ring occupancy count)) times. This will wrap the internal UDMA-P ring state occupancy counter (which is 21-bits wide) to 0. (If possible, ring the doorbell with the maximum count each iteration to minimize the total number of writes.
6. Restore the original ring mode (if not exposed ring mode).

This will ultimately reset the UDMA state information for a ring so that the RINGACC and UDMA are both in the same reset state for that ring. This workaround must be executed only when the DMA channels associated with the ring are disabled.

i2024

MMCSD: MMCSD Peripherals Do Not Support HS400

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The MMCSD peripherals do not support the Multimedia Card HS400 mode.

Workaround(s): None

i2025

IO, MMCSD: Incorrect IO Power Supply Connectivity Prevents Dynamic Voltage Change on VDDSHV6 and VDDSHV7

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The LVCMOS IOs associated with device pins MMC0_SDCD (A23) and MMC0_SDWP (B23) receive power from VDDSHV6 rather than VDDS_OSC1.

The LVCMOS IOs associated with device pins MMC1_SDCD (B24) and MMC1_SDWP (C24) receive power from VDDSHV7 rather than VDDS_OSC1.

VDDSHV6 and VDDSHV7 also provides IO power to MMCSD0 UHS-I PHY and MMCSD1 UHS-I PHY, respectively, which may require dynamically voltage change for some use cases.

LVCMOS IOs detect the voltage applied to their respective VDDSHV power rail while MCU_PORz or PORz reset inputs are asserted to determine if they should operate in 1.8 V mode or 3.3 V mode. The IO voltage operating mode is latched on the rising edge of PORz.

After PORz is de-asserted, VDDSHV6 and VDDSHV7 must remain within the recommended operating range of the applied voltage to avoid potential long term reliability issues.

Since VDDSHV6 and VDDSHV7 shall not change after PORz is released, it is not possible to implement dynamic IO voltage change on the IOs associated with MMCSD0 and MMCSD1.

Advisory i2026 describes a reliability issue when operating VDDSHV6 and VDDSHV7 at 3.3 V, which limits the operation of these IO supplies to 1.8 V.

These four LVCMOS IOs associated with VDDSHV6 and VDDSHV7 should have been powered from VDDS_OSC1, which is a fixed 1.8 V supply.

Workaround(s): For most applications where MMC card detect and/or write protect function is implemented on these pins, the external pull-up resistors associated with these signals need to be powered from the same power source as the respective VDDSHV6 or VDDSHV7 power pins.

For these use cases, PCBs designers should consider including resistor installation options that allow external pull-ups to be powered from the respective VDDSHV6 / VDDSHV7 supply or the VDDS_OSC1 supply. The resistor installation option allows a common PCB design to support current silicon revision and future silicon revisions where these IOs will be powered from VDDS_OSC1.

For use cases other than MMC card detect and write protect, the effect of this IO being sourced from a different supply rail needs to be evaluated/considered for any connectivity to these pins.

i2026

MMCSD: Negative Current from UHS-I PHY May Create an Over-Voltage Condition on VDDS6 and VDDS7 Which Exposes the Device to a Significant Reliability Risk

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The MMCSD0 UHS-I PHY and MMCSD1 UHS-I PHY receives 1.8 V bias power from device pins VDDS6 and VDDV7 respectively. Unexpected paths through the UHS-I PHY allows current to flow from VDDSHV6 to VDDS6 and VDDSHV7 to VDDS7 when the VDDSHV IO supply is operating at 3.3 V.

The UHS-I PHY typically consumes power from its VDDS bias supply. However, the unexpected current flowing from the 3.3 V VDDSHV IO supply may exceed the bias current consumed by the UHS-I PHY. When this occurs, current may flow out of the 1.8 V VDDS bias supply. This negative current may cause the voltage applied to VDDS6 and VDDS7 to increase above the recommended operating range in some operating conditions.

This issue has been observed on a system where SDIO LDO was sourcing the UHS-I PHY bias supply while its VDDSHV IO supply was 3.3 V. This occurs because SDIO LDO was not designed to sink current. Therefore, it is not able to shunt any negative current to VSS. Negative current causes the VDDS bias supply to increase above the recommended bias supply voltage where the UHS-I PHY enters a non-functional state that can only be cleared when the respective MMCSD subsystem is reset.

The negative current is a function of device operating temperature, device process variation, and UHS-I PHY output toggle rate.

This issue can also occur when the IOs associated with MMCSD0 UHS-I PHY and MMCSD1 UHS-I PHY are operating at 3.3 V while configured to one of the other MUXMODES defined in the Datasheet Pin Multiplexing Table (when IOMUX_ENABLE bit in the respective MMCSD0_SS_PHY_CTRL_1_REG or MMCSD1_SS_PHY_CTRL_1_REG register is set to 1 to enable alternate MUXMODES).

Further analysis has indicated a significant reliability risk when operating VDDSHV6 and VDDSHV7 at 3.3 V.

This issue does not occur when the VDDSHV IO supply is operating at 1.8 V.

Workaround(s): There is no workaround which prevents the negative current from producing an overvoltage condition. Therefore, there is no support for operating VDDSHV6 and

VDDSHV7 at 3.3 V.

i2027

CPSW: CPSW Does Not Support CPPI Receive Checksum (Host to Ethernet) Offload Feature

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: CPSW does not support the CPPI receive checksum (Host to Ethernet) offload feature. As such, the CPSW0_P0_CONTROL_REG [0] RX_CHECKSUM_EN bit must remain at the default value of zero, disabling the receive checksum feature.

Workaround(s): Use software to implement receive checksum operations.

i2028

USB3SS: SuperSpeed USB Non-Functional

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: For SR1.0, the USB3.1 GEN1 (5Gbps) portion of USB3 Subsystem 0 (USB3SS0) is non-functional in both SuperSpeed Host and Superspeed Device mode.

For SR2.0, the USB3.1 GEN1 (5Gbps) portion of USB3 Subsystem 0 (USB3SS0) is non-functional only in SuperSpeed Device mode. SuperSpeed Host mode is functional and supported.

The USB2.0 portion of USB3SS0 is unaffected.

Workaround(s): Implement USB2.0 on USB3SS0 for non-functional SuperSpeed modes (Host and Device modes for SR1.0; Device mode for SR2.0). USB3SS0 supports USB2.0 High-Speed (480Mbps), Full-Speed (12Mbps), and Low-Speed (1.5Mbps) modes of operation when operating in USB Host mode, and USB2.0 High-Speed and Full-Speed modes of operation when operating as a USB Device.

i2030

Boot, UART: UART Boot Mode Never Times Out

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: When UART is configured as the primary boot mode and its 180-second timeout expires, the ROM code is expected to switch to the selected backup boot mode. However, this transition to the backup boot mode never occurs.

Instead, the SoC device goes through a warm reset because the watchdog timer expires before the UART boot mode times out. After the warm reset, the SoC device will continue to attempt booting from UART, instead of switching to the backup boot mode.

Workaround(s): Do not select a backup boot mode when UART is the primary boot mode.

i2032

DSS: DSS DPI Interface Does Not Support BT.656 and BT.1120 Output Modes

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The BT.656 and BT.1120 output modes are not supported on the DSS DPI interface. Selecting the above output modes will result in incorrect color reproduction on DSS DPI interface.

Workaround(s): No workaround is available if BT.656 or BT.1120 output modes are required. Alternatively, MIPI DPI 2.0 RGB output mode could be used instead of BT.656 and BT.1120.

i2037

PCIe: PCI-Express May Corrupt Inbound Data

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: When an inbound PCIe TLP spans more than two internal AXI 128-byte bursts, the bus may corrupt the packet payload. This issue affects inbound reads only. Outbound transactions are not affected. No PCIe error is flagged as the protocol itself is correct and only the contained data is corrupted. This corrupt data may cause associated applications or the processor to hang.

Workaround(s): Limit the PCIe MAX_READ_REQUEST_SIZE (MRRS) and the PCIe MAX_PAYLOAD_SIZE (MPS) to 128 bytes by setting:

- PCIE_EP_DEVICE_CONTROL_DEVICE_STATUS / PCIE_RC_DEVICE_CONTROL_DEVICE_STATUS[14-12] PCIE_CAP_MAX_READ_REQ_SIZE register field to 0h, and
- PCIE_EP_DEVICE_CONTROL_DEVICE_STATUS / PCIE_RC_DEVICE_CONTROL_DEVICE_STATUS[7-5] PCIE_CAP_MAX_PAYLOAD_SIZE_CS register field to 0h.

This ensures that no more than two AXI burst transactions will be needed to complete any single PCIe TLP.

i2038

Boot: FAT16 Fails When Root Block Resides in More Than One Cluster

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: Boot ROM will not find the boot file on a FAT16 file system if the file system uses multiple clusters for the boot block. Boot fails if the boot file does not reside on the first cluster. This has been observed when using Ubuntu to create a small FAT16 partition. In this case the cluster size is 4KB, so only 128 entries reside in the first root cluster (each directory entry is 32 bytes). If the boot file resides in file index 128 or later (max size is typically set to 512) the ROM will not find the boot file.

Workaround(s): Use FAT32 partition, instead of FAT16 partition.

i2039

DSS: Frame Buffer Flip/Mirror Feature Using RGB24/BGR24 Packed Format Can Result in Pixel Corruption

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The frame buffer flip/mirror feature in the video pipeline can result in pixel corruption on the display output when the RGB24 or BGR24 packed format is used and the following conditions are met simultaneously:

- Frame buffer flip/mirror feature enabled (DSS0_VID_ATTRIBUTES[12] FLIP = 0x1)
- RGB24 or BGR24 packed format selected (DSS0_VID_ATTRIBUTES[6-1] FORMAT = 0x0B or 0x0C)
- Scaler inside VID pipe is enabled (DSS0_VID_ATTRIBUTES[8-7] RESIZEENABLE = 0x1, 0x2, or 0x3)
- Scale ratio < 0.5, or down-scaling by more than half, this scale ratio is configured by DSS0_VID_FIRH, DSS0_VID_FIRH2, DSS0_VID_FIRV, and DSS0_VID_FIRV2 registers.

These conditions can cause one pixel to be corrupted at the start of a line, for some lines, in a frame.

Workaround(s): If the RGB24 or BGR24 packed format is selected, then use the GPU to implement the flip/mirror operation.

i2053

VTM: Software Reads from On-Die Temperature Sensors Can Be Corrupted

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The VTM_TMPSENSj_STAT[9-0] DTEMP, where j = 0-7, registers can be read in software to determine the last sampled temperature of each of the '*j*' number of on die temp sensors on the SoC. If a read happens on the same exact cycle that a temperature sample is updated then there is a chance that the read data can be corrupted due to incorrect resynchronization between clock domains.

Workaround(s): Software should perform three reads from the VTM_TMPSENSj_STAT[9-0] DTEMP, where j = 0-7, registers. Software should then compute the temperature to be used based on the average of the two samples that are closest to each other.

The software pseudo code is as follows:

```
#define abs(x) (((x)<0)?-(x):(x))
unsigned int get_best_value(unsigned int s0, unsigned int s1, unsigned int s2)
{
        int d01 = abs(s0 - s1);
        int d02 = abs(s0 - s2);
        int d12 = abs(s1 - s2);

        // if delta 01 is least, take 0 and 1
        if ((d01 <= d02) && (d01 <=d12)) {
                return (s0+s1)/2;
        }
        // if delta 02 is least, take 0 and 2
        if ((d02 <= d01) && (d02 <=d12)) {
                return (s0+s2)/2;
        }
        /* in all other cases, take 1 and 2 */
        return (s1+s2)/2;
}

unsigned int get_temp()
{
        unsigned int s0,s1,s2;
        s0 = Read VTM_TMPSENSj_STAT[9-0] DTEMP;
        s1 = Read VTM_TMPSENSj_STAT[9-0] DTEMP;
        s2 = Read VTM_TMPSENSj_STAT[9-0] DTEMP;
        return get_best_value(s0,s1,s2);
}
```

i2054

RINGACC: Reads from GCFG Region Can Cause Spurious RAM ECC Errors

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: A read to the Ring Accelerator (RA) Global Config Region (GCFG) can cause a read of a RAM with an illegal address. This causes the RAM to read random data which will fail the RAM ECC check. This will cause a log and interrupt to be created. The data itself is not used, so there is no functional failure, but the interrupt will make it appear there was a RAM failure.

NOTE: This affects the MCU NAVSS RA only, as the MAIN NAVSS RA has an aligned size so there are no illegal RAM addresses.

Workaround(s): The software that handles the RAM ECC interrupts for MCU NAVSS RA can check the address in the log registers and ignore the error if the address is beyond the limit of the RAM (which is the number of rings supported by the RA). The software can just clear the error.

i2069

CC_ARMSS: Powering Down CC_ARMSS1 Causes System Data Corruption

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: When attempting to power down CC_ARMSS1, programming sequences defined in Power Modes Section of ID062414 are followed as:

- Cluster shutdown mode without system driven L2 flush, or,
- Cluster shutdown mode with system driven L2 flush.

During the execution of these sequences, data corruption can be observed in the rest of the device subsystems, causing device, operating system(s), and application software to hang or malfunction.

Workaround(s): Cluster power down shall not be used. Instead, system may program individual cores to be shut down, by following the programing sequence defined in Power Modes Section of ID062414, but leave the CC_ARMSS1 cluster power domain ON.

i2073

DCC: Suspend Mode Not Functional

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: When DCC is programmed to work in suspend mode (also referred to as continuous mode), a DCC error event may be correctly generated on any of the following conditions:

- Clock1 expires before the COUNT0 reaches 0
- Clock1 expires after both COUNT0 and VALID0 reach 0
- Clock1 not present
- Clock0 not present.

Due to an IP level issue, a false DCC error event may be generated without any of the above conditions. This false event may trigger an unwanted interrupt in the system.

Further, the false event condition will prevent DCC counters to reload after the first comparison, instead of continuously reload as expected in this mode. These two issues effectively render the suspend mode clock comparison non-usable.

Workaround(s): Suspend mode operation may not be used for any DCCs in the devices. Only single-shot mode is supported.

i2075

USB2PHY: USB2PHY Charger Detect is Enabled by Default Without VBUS Presence

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The USB2PHY Charger Detect function is enabled by default after power on without VBUS presence. This causes D+ to be pulled up, violating the Universal Serial Bus Revision 2.0 Specification.

This violation could cause some USB hubs to not respond to the SETUP packet from the USB host and fail in enumeration if the hub is attached to the SoC's USB host port prior to the SoC being powered on.

Workaround(s): Disable the USB2PHY Charger Detect function by setting the PHY register bit USB2PHY_CHRG_DET[28] MEM_DIS_CHG_DET = 1 prior starting the USB host controller.

i2083

CPTS: GENF (and ESTF) Reconfiguration Issue

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: Re-configuring a GENF/ESTF function after having been previously configured has an issue. Issue details:

If GENF re-configuration occurs when the GENF output is logic one then the re-configuration comparison time will be a half-count instead of the full count, and the GENF output will be off by 1/2 cycle. The re-configured cycle will be correct if the GENF output is logic zero when the re-configuration occurs.

Workaround(s): GENF reconfiguration can only happen after a SOC hardware reset.

i2084

CPSW: CPSW Does Not Support Intersperced Express Traffic (IET – P802.3br/D2.0) In 10/100Mbps Mode

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: The CPSW peripheral does not support Intersperced Express Traffic (IET – P802.3br/D2.0) in 10/100Mbps mode.

IET is only supported in 1000Mbps mode.

Workaround(s): None.

i2096

UART: Spurious UART Interrupts When Using DMA

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: Spurious UART interrupts may occur when DMA mode (UART_FCR[3] DMA_MODE) is enabled and DMA is used to read data from RX FIFO. The Interrupt Controller flags that a UART interrupt has occurred; however, the associated UART_IIR_UART[0] IT_PENDING bit remains set to 1, indicating that no interrupt is pending.

Workaround(s): Acknowledge the spurious interrupts for every occurrence. The issue can be avoided by disabling Receive Data Interrupt (RDI) using the UART_IER_UART[0] RHR_IT bit; however, be aware that this also disables RX timeout interrupts, which may not be practical for all use cases.

i2097

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

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

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.

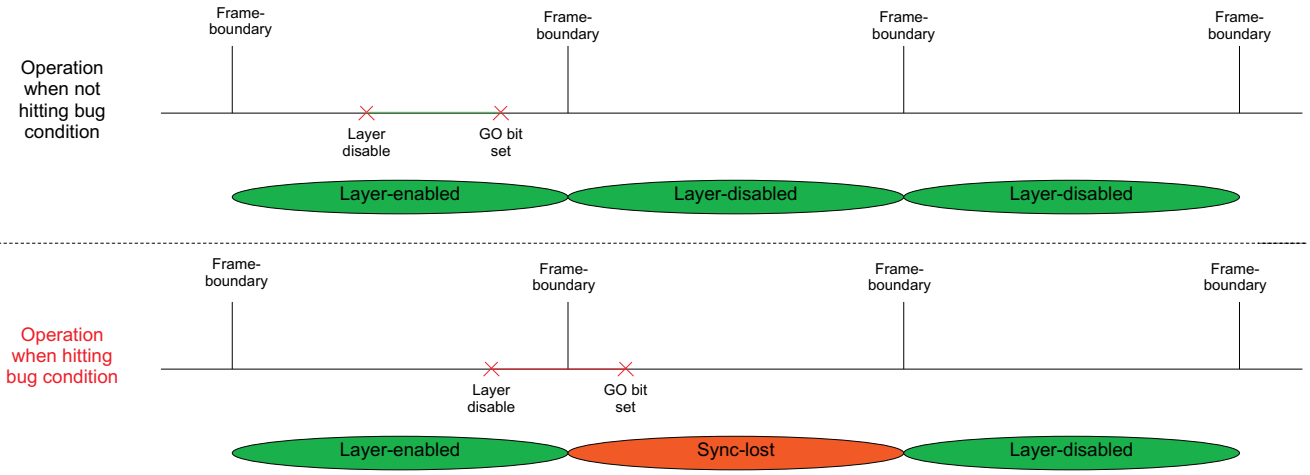

Figure 2. Bug Condition

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 3. 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 3. Workaround Pseudo-code

i2098

SA2_UL: Auth/Decrypt Operations with 2nd Input Thread Does Not Send the DMA Packet Out

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: Thread muxing mode in ETYPE=5 (SA2_UL) can have unpredictable results ranging from lost data to crediting overflow on the UDMAP. This prevents use of destination thread 1, and source threads 2 and 3 of ETYPE=5 (SA2_UL).

Workaround(s): None. Destination thread 1, and source threads 2 and 3 cannot be used on SA2_UL.

i2099

Cortex-R5F: Deadlock Might Occur When One or More MPU Regions is Configured for Write Allocate Mode

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: There are two conditions where R5F can deadlock:

- When software is performing series of store operations to cacheable write back/write allocate memory region and later on software execute barrier operation (DSB or DMB). R5F may hang at the barrier instruction.
- When software is performing a mix of load and store operations within a tight loop and store operations are all writing to cacheable write back/write allocates memory regions, R5F may hang at one of the load instruction.

Workaround(s): Disabling linefill optimization inside Cortex-R5F will eliminate deadlock condition.

To disable the linefill optimization, the software needs to set bit 13 (DLFO) of Auxiliary Control Register (See Cortex-R5F Technical Reference Manual for how to update Auxiliary Control Register).

i2101

GIC: ITS Misbehavior

Revision(s) Affected: AM65x SR 1.0

DRA80xM SR 1.0

Details: GIC AXI master traffic goes through protocol conversion bridge to access memory. Because of the misconfiguration of this protocol conversion bridge AXI read request generated on one particular ARID will not be returned by the bridge.

This will cause all ITS requests on that particular Device ID, for which read access was requested to fail.

Impact of this is if more than 4 or more Device IDs are used in ITS, GIC can use ARID for AXI read which is not supported by protocol conversion bridge and that will result in missed interrupts.

Workaround(s): In the boot sequence, software needs to setup 5 dummy device IDs starting from 0, which are then reserved and can't be used for application, and then send ITS request for the first 2 device IDs. This sequence should use up unsupported ARID which will not be used by GIC during application and no ITS misbehavior would be seen.

Other combination of workaround can be setup 6 dummy device IDs starting from 0 and send ITS request to first device ID, setup 7 dummy device IDs starting from 0 and send ITS request to first 4 device IDs.

Note: This workaround is not implemented in Processor SDK Linux since it needs modifications to Arm® provided / maintained generic driver for GIC in Linux.

i2103

Safety Modules: Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: For functional safety errors, the logged information - ECC_GRP, ECC_BIT, and ECC_TYPE in the Error Status Registers may be incorrect for certain safety checkers. This only applies to safety checkers that map to ECC_GRP = 0,15,31,47,63...(N*16-1). In the case for the DDR Bridge/Controller, the issue only applies to the safety checkers where ECC_GRP = 0,31,63...(N*32-1).

This issue affects all Safety Module instances and their sub-banks. Refer to section Safety Modules of the device TRM.

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 Safety Module instance had the error (thus knowing the IP module), but not which exact safety checker encountered the error.

i2104

PCIe: GEN3 (8GT/s) Operation Not Supported

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: PCI-Express (PCIe) GEN3 (8GT/s) operation is not supported. This restriction applies to both Root Complex and Endpoint modes of operation. PCIe GEN1 (2.5GT/s) and PCIe GEN2 (5GT/s) operation is unaffected.

Workaround(s): No workaround is available. Implement PCIe as GEN1 or GEN2 only.

i2115

OSPI: OSPI Boot Doesn't Support Some xSPI Modes or xSPI Devices

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: For background, the various OSPI and xSPI protocols are described according to bit-width (1 or 8) and data rate (S or D for *S*ingle Data rate or *D*ouble Data rate) for the Command/Address/Data segments of the protocol.

The SoC's ROM OSPI boot mode supports 1S-1S-1S mode and 1S-1S-8S mode.

The xSPI protocol defines 1S-1S-1S mode for general backwards compatibility, and 8D-8D-8D for maximum throughput. The ROM OSPI boot mode is compatible with 1S-1S-1S mode, but is not compatible with 8D-8D-8D mode.

Some SPI Flash memory devices also offer the legacy 1S-1S-8S mode, which is compatible with the ROM OSPI boot mode.

Note that the OSPI IP can in general support 8D-8D-8D mode with an appropriate software driver. The limitation is only for ROM boot which hard codes the 1S-1S-1S and 1S-1S-8S modes.

Workaround(s): If 8-bit data rate is required for boot, a SPI Flash memory device should be carefully selected that is compatible with 1S-1S-8S mode of operation.

If 1-bit data is sufficient for boot, an xSPI Flash memory device should be chosen that explicitly supports the 1S-1S-1S mode at boot. Different memory vendors may only support this mode on specific part variants.

i2119

HyperFlash: HyperFlash is Not Supported

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The Hyperflash interface is not supported.

Workaround(s): None. HyperFlash should not be used.

i2129

Cortex-R5F: High Priority Interrupt is Missed by VIM

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: The VIM will not always interrupt the currently active interrupt when a higher priority interrupt arrives immediately afterwards. In these cases, the higher priority interrupt will only be taken after the completion of the current lower priority interrupt or when an even higher priority interrupt arrives. The impact of this issue is higher than expected interrupt latency for the high priority interrupt. Both Vector Interface (VIC) servicing and MMR Interface servicing modes of VIM are affected.

Workaround(s): This is a problem which affects applications which are latency critical and wants pre-

empting of low priority interrupts with higher priority interrupts. If the application is not latency critical, then the behavior may be acceptable (the high priority interrupt will be eventually taken after the low priority interrupt completes).

Alternatively, user can implement a completely SW managed interrupt servicing scheme, where every ISR (Interrupt Service Routine) shall check for the presence of an active higher priority interrupt (by reading Interrupt Raw Status registers in VIM) and jumping to the ISR corresponding to that interrupt.

i2132

Cortex-R5F: Interrupt Preemption (Nesting) is Unavailable if Using VIM Vector Interface for Interrupt Handling

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: Interrupt preemption, which is the nesting of high priority interrupts inside a low priority interrupt, is unavailable if using VIM Vector Interface for interrupt handling. Nesting of a high priority interrupt within a low priority interrupt will result in corrupted operation of the processor. The issue only impacts Vector Interface method of interrupt handling provided by VIM. It does not impact MMR interface method of interrupt handling. Issues impact both FIQ and IRQ interrupts.

Workaround(s): If using Vector Interface method, user should not set the I/F bit (to enable nesting of interrupts) in CPSR.

If interrupt nesting is required then user should only use MMR interface method for interrupt handling. Note that, MMR interface method incurs an additional latency for Interrupt Service Routine (ISR) entry compared to Vector Interface method.

i2149

MSMC: MSMC Scrubber Only Targets Bottom 16 of 32 Ways of SRAM/L3$

Revision(s) Affected: AM65x SR 2.0, SR 1.0

DRA80xM SR 2.0, SR 1.0

Details: MSMC Scrubber periodically scans through MSMC SRAM/L3$, Snoop Filter, and Tags for correctable 1-bit errors and then corrects them. This is to reduce the probability of multiple 1-bit errors accumulating over time and becoming non-correctable 2-bit errors.

Due to an error in the address decoding, MSMC Scrub transactions only access the lower half of the L3$ Tag ways (0-15). Ways 16-31 are never accessed. The corresponding L3$ Data RAMs will also not be accessed by Scrubber.

Customers will see an increase in probability of accumulating 2-bit detectable/non-correctable errors in upper half (upper 16 ways) of MSMC L3$ Tag and corresponding Data.

This issue does not affect the MSMC SRAM and only applies to L3 Cache.

Workaround(s): There is no complete software workaround.

Software can attempt to periodically flush the L2$ to allow MSMC EDC to be refreshed. This is not a complete workaround, however, since Arm® can silently evict cache lines without alerting MSMC.

4 Modules Affected

Table 7 shows the module(s) that are affected by each usage note.

Table 7. Usage Note by Modules

MODULE	USAGE NOTE
ADC	i2082 — ADC: High Input Leakage Current May Impact ADC Accuracy
Fail-Safe IO's	i2033 — Fail-Safe IO's: Latch-up Risk on Fail-Safe IOs
INTRTR	i2007 — INTRTR: Spurious Interrupts Generated when Programming Certain Interrupt Routers

Table 8 shows the module(s) that are affected by each advisory.

Table 8. Advisories by Modules

MODULE	ADVISORY
Boot	i2019 — Boot, USB3SS: Boot ROM Does Not Support USB Host MSC (Mass Storage Class) Boot Mode
	i2020 — Boot, USB3SS: Boot ROM Does Not Support USB Device Firmware Upgrade (DFU) Boot Mode
	i2030 — Boot, UART: UART Boot Mode Never Times Out
	i2038 — Boot: FAT16 Fails When Root Block Resides in More Than One Cluster
CC_ARMSS	i2069 — CC_ARMSS: Powering Down CC_ARMSS1 Causes System Data Corruption
Cortex-R5F	i2099 — Cortex-R5F: Deadlock Might Occur When One or More MPU Regions is Configured for Write Allocate Mode
	i2129 — Cortex-R5F: High Priority Interrupt is Missed by VIM
	i2132 — Cortex-R5F: Interrupt Preemption (Nesting) is Unavailable if Using VIM Vector Interface for Interrupt Handling
CPSW	i2027 — CPSW: CPSW does not support CPPI receive checksum (Host to Ethernet) offload feature
	i2084 — CPSW: CPSW Does Not Support Intersperced Express Traffic (IET – P802.3br/D2.0) In 10/100Mbps Mode
CPTS	i2083 — CPTS: GENF (and ESTF) Reconfiguration Issue
DCC	i2018 — DCC: Incorrect Counter Values in DCC Operation
	i2073 — DCC: Suspend Mode Not Functional
DDRSS	i2009 — DDRSS: DDR Controller ECC Scrubbing Feature Can Cause DRAM Data Corruption
	i2022 — DDRSS: Independent Impedance Control for Address/Control and Data Bus Lanes is Not Available
DSS	i2000 — DSS: DSS Does Not Support YUV Pixel Data Formats
	i2032 — DSS: DSS DPI Interface does not support BT.656 and BT.1120 output modes
	i2039 — DSS: Frame Buffer Flip/Mirror Feature Using RGB24/BGR24 Packed Format Can Result in Pixel Corruption
	i2097 — DSS: Disabling a Layer Connected to Overlay May Result in Synclost During the Next Frame
GIC	i2101 — GIC: ITS Misbehavior
HyperFlash	i2119 — HyperFlash: HyperFlash is Not Supported
IO	i2025 — IO, MMCSD: Incorrect IO Power Supply Connectivity Prevents Dynamic Voltage Change on VDDSHV6 and VDDSHV7
MCAN	i2017 — MCAN: Message Transmitted with Wrong Arbitration and Control Fields (Early Start of Frame)
MMCSD	i2024 — MMCSD: MMCSD Peripherals Do Not Support HS400
	i2025 — IO, MMCSD: Incorrect IO Power Supply Connectivity Prevents Dynamic Voltage Change on VDDSHV6 and VDDSHV7
	i2026 — MMCSD: Negative Current from UHS-I PHY May Create an Over-Voltage Condition on VDDS6 and VDDS7 Which Exposes the Device to a Significant Reliability Risk
MSMC	i2021 — MSMC: Non-Coherent Memory Access to Coherent Memory Can Cause Invalidation of Snoop Filter
	i2149 — MSMC: MSMC Scrubber Only Targets Bottom 16 of 32 Ways of SRAM/L3$
On-chip Debug	i2013 — On-Chip Debug: The Assertion of Warm Reset Coinciding with a Debug Configuration Access Targeting the STM Subsystem May Result in a Hang of Said Debug Configuration Access
	i2015 — On-Chip Debug: CPTracer Bus Probes MAIN_CAL0_0 and MCU_SRAM_SLV_1 are not able to Distinguish between Secure and Non-secure Transactions

Table 8. Advisories by Modules (continued)

MODULE	ADVISORY
OSPI	i2115 — OSPI: OSPI Boot Doesn't Support Some xSPI Modes or xSPI Devices
PCIe	i2037 — PCIe: PCI-Express May Corrupt Inbound Data
	i2104 — PCIe: GEN3 (8GT/s) Operation Not Supported
RINGACC	i2023 — RINGACC, UDMA: RINGACC and UDMA Ring State Interoperability Issue after Channel Teardown
	i2054 — RINGACC: Reads from GCFG Region Can Cause Spurious RAM ECC Errors
SA2_UL	i2098 — SA2_UL: Auth/Decrypt Operations with 2nd Input Thread Does Not Send the DMA Packet Out
Safety Modules	i2103 — Safety Modules: Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors
UDMA	i2023 — RINGACC, UDMA: RINGACC and UDMA Ring State Interoperability Issue after Channel Teardown
UDMA-P	i2004 — UDMA-P: UDMA-P Host Packet Descriptor's "0x3FFFFF" Packet Length Mode not Functional
	i2006 — UDMA-P: UDMA-P Real-Time Remote Peer Registers not Functional Across UDMA-P Domains
USB2PHY	i2075 — USB2PHY: USB2PHY Charger Detect is Enabled by Default Without VBUS Presence
USB3SS	i2028 — USB3SS: SuperSpeed USB Non-Functional
	i2019 — Boot, USB3SS: Boot ROM Does Not Support USB Host MSC (Mass Storage Class) Boot Mode
	i2020 — Boot, USB3SS: Boot ROM Does Not Support USB Device Firmware Upgrade (DFU) Boot Mode
UART	i2030 — Boot, UART: UART Boot Mode Never Times Out
	i2096 — UART: Spurious UART Interrupts When Using DMA
VTM	i2053 — VTM: Software Reads from On-Die Temperature Sensors Can Be Corrupted

Trademarks

CoreSight is a trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
Arm, Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
All other trademarks are the property of their respective owners.

Revision History

Changes from D Revision (December 2019) to E Revision **Page**

- Added bugs validity for SR 2.0 1
- Section 3.2: Added i2017, MCAN: Message Transmitted with Wrong Arbitration and Control Fields (Early Start of Frame) 11
- Section 3.2: Updated i2028, USB3SS: SuperSpeed USB Non-Functional 17
- Section 3.2: Added i2097, DSS: Disabling a Layer Connected to Overlay May Result in Synclost During the Next Frame 22
- Section 3.2: Added i2101, GIC: ITS Misbehavior 24
- Section 3.2: Added i2103, Safety Modules: Incorrect Reporting of ECC_GRP, ECC_BIT and ECC_TYPE Information for Functional Safety Errors 24
- Section 3.2: Added i2115, OSPI: OSPI Boot Doesn't Support Some xSPI Modes or xSPI Devices 25
- Section 3.2: Added i2119, HyperFlash: HyperFlash is Not Supported 25
- Section 3.2: Added i2129, Cortex-R5F: High Priority Interrupt is Missed by VIM 25
- Section 3.2: Added i2132, Cortex-R5F: Interrupt Preemption (Nesting) is Unavailable if Using VIM Vector Interface for Interrupt Handling 26
- Section 3.2: Added i2149, MSMC: MSMC Scrubber Only Targets Bottom 16 of 32 Ways of SRAM/L3$ 26

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), 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, 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 (www.ti.com/legal/termsofsale.html) 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.

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