# DM385/DM388 DaVinci™ Digital Media Processor Silicon Revisions 1.1, 1.0

# Silicon Errata

Literature Number: SPRZ387B
April 2013–Revised December 2013

# Contents

# List of Figures

# List of Tables

# DM385/DM388 DaVinci™ Digital Media Processor Silicon Revisions 1.1, 1.0

## 1 Introduction

This document describes the known exceptions to the functional specifications for the DM385/DM388 DaVinci™ Digital Media Processor. The updates are applicable to the AAR package. For additional information, see the *DM385/DM388 DaVinci™ Digital Media Processor* data manual (literature number: SPRS821).

For additional peripheral information, see the latest version of the *DM38x DaVinci™ Digital Media Processor Technical Reference Manual* (Literature Number: SPRUHG1).

The advisory numbers in this document may not be sequential. Some advisory numbers may be moved to the next revision and others may have been removed and documented in the device-specific data manual or peripheral user's guide. When items are moved or deleted, the remaining numbers remain the same and are not re-sequenced.

This document also contains Usage Notes. Usage Notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These notes will be incorporated into future documentation updates for the device (such as the device-specific data manual), and the behaviors they describe will not be altered in future device revisions.

### 1.1 Device and Development Support-Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all devices and support tools. Each commercial family member has one of three prefixes: X, P, or null (for example, **X**DM385xxAAR01F). 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 (X/TMDX) through fully qualified production devices/tools (TMS/TMDS).

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all MPUs and support tools. Each device has one of three prefixes: X, P, or null (no prefix). 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/tools (TMDS).

Device development evolutionary flow:

**X —** Pre-production 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.

## 1.2 Package Symbolization and Revision Identification

Figure 1 shows an example of the processor package symbolization. The device revision can be identified by the markings on the top of the AAR package as noted in Figure 1.

Table 1 lists the device revisions for the processor.

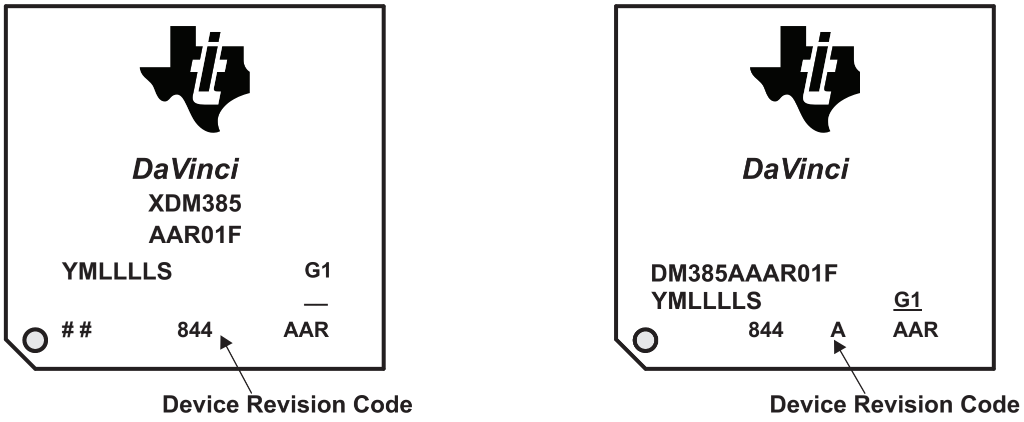


**Figure 1. Example, Device Revision Codes (AAR Package)**

**Table 1. Device Revision Codes**

| DEVICE REVISION CODE | SILICON REVISION | PART NUMBERS/COMMENTS |
|---|---|---|
| Blank | 1.0 | XDM385AAR01F, DM385AAR01F |
| A | 1.1 | DM385AAAR01, DM385AAAR01F, DM385AAARD01F<br>DM385AAAR11, DM385AAAR11F, DM385AAARD11F<br>DM385AAAR21, DM385AAAR21F, DM385AAARD21F<br>DM388AAAR11, DM388AAAR11F, DM388AAARD11, DM388AAARD11F<br>DM388AAAR21, DM388AAAR21F, DM388AAARD21, DM388AAARD21F<br>DM388AAARD11FP, DM388AAARD21FP |

# 2 Silicon Revision 1.0 Usage Notes

## *2.1 Usage Notes for Silicon Revision 1.0*

Usage Notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These notes will be incorporated into future documentation updates for the device (such as the device-specific data manual), and the behaviors they describe will not be altered in future device revisions.

### 2.1.1 DDR3: JEDEC Compliance for Maximum Self-Refresh Command Limit

On silicon revision 1.0, when using DDR3 EMIF Self-Refresh, it is possible to violate the maximum refresh command requirement specified in the JEDEC standard DDR3 SDRAM Specification (JESD79-3E, July 2010). This requirement states that the DDR3 EMIF controller should issue no more than 16 refresh commands within any 15.6 µs interval.

To avoid this requirement violation, when using the DDR3 EMIF and Self-Refresh (setting LP_MODE = 0x2 field in the PMCR), the SR_TIM value in the PMCR ***must*** to be programmed to a value greater than or equal to 0x9.

### 2.1.2 Some PLLs Only Support Even M2 Post Dividers

On silicon revision 1.0, for some top-level PLL instances, odd values of the M2 Post Divider are ***not*** supported. For proper device operation, the M2 Post Divider for these PLL's ***must always*** be programmed to an even value of 2 or greater (for example, 2, 4, and more up to 126).

The top-level PLL's which ***do not*** support odd M2 Post Dividers are

- PLL_HDVICP
- PLL_L3L4
- PLL_DDR
- PLL_HDVPSS
- PLL_AUDIO
- PLL_MEDIACTL
- PLL_VIDEO0
- PLL_VIDEO1
- PLL_VIDEO2

**Note:** PLL_ARM, PLL_USB, and the embedded SERDES PLL's ***do not*** have this restriction.

### 2.1.3 DDR2, DDR3 and DDR3L Require Software Leveling

On all silicon revisions, DDR2, DDR3 and DDR3L require software leveling to tune the device I/Os to the timing characteristics of a particular board design. Hardware leveling is not supported. For details on software leveling, see the TI813x-DDR-Init-U-Boot Embedded Processors Wiki Page

# 3 Silicon Revision 1.1 Usage Notes and Known Design Exceptions to Functional Specifications

## *3.1 Usage Notes for Silicon Revision 1.1*

Usage Notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These notes will be incorporated into future documentation updates for the device (such as the device-specific data manual), and the behaviors they describe will not be altered in future device revisions.

### 3.1.1 DDR3: JEDEC Compliance for Maximum Self-Refresh Command Limit

On silicon revision 1.1, when using DDR3 EMIF Self-Refresh, it is possible to violate the maximum refresh command requirement specified in the JEDEC standard DDR3 SDRAM Specification (JESD79-3E, July 2010). This requirement states that the DDR3 EMIF controller should issue no more than 16 refresh commands within any 15.6 µs interval.

To avoid this requirement violation, when using the DDR3 EMIF and Self-Refresh (setting LP_MODE = 0x2 field in the PMCR), the SR_TIM value in the PMCR ***must*** to be programmed to a value greater than or equal to 0x9.

### 3.1.2 Some PLLs Only Support Even M2 Post Dividers

On silicon revision 1.1, for some top-level PLL instances, odd values of the M2 Post Divider are ***not*** supported. For proper device operation, the M2 Post Divider for these PLL’s ***must always*** be programmed to an even value of 2 or greater (for example, 2, 4, and more up to 126).

The top-level PLL’s which ***do not*** support odd M2 Post Dividers are

- PLL_HDVICP
- PLL_L3L4
- PLL_DDR
- PLL_HDVPSS
- PLL_AUDIO
- PLL_MEDIACTL
- PLL_VIDEO0
- PLL_VIDEO1
- PLL_VIDEO2

**Note:** PLL_ARM, PLL_USB, and the embedded SERDES PLL’s ***do not*** have this restriction.

### 3.1.3 DDR2, DDR3 and DDR3L Require Software Leveling

On all silicon revisions, DDR2, DDR3 and DDR3L require software leveling to tune the device I/Os to the timing characteristics of a particular board design. Hardware leveling is not supported. For details on software leveling, see the TI813x-DDR-Init-U-Boot Embedded Processors Wiki Page

## 3.2 Silicon Revision 1.1 Known Design Exceptions to Functional Specifications

This is a list of the device silicon revisions 1.1 and 1.0 known design exceptions to functional specifications.

### Table 2. Silicon Revision 1.1 Advisory List

## Advisory 1.1.3 — *Debugger Cannot Access Memory Controller Memories if Functionally Powered Down*

**Revision(s) Affected:** 1.1, 1.0

**Details:** When the Media Controller memories are powered down by the functional application code, they cannot be accessed or powered-up from the Debugger.

**Workaround:** During debugging, if the debugger is attempting to access the Media Controller memories ensure the functional application code keeps them powered on.

## Advisory 1.1.5 *DPLL_ARM: Cortex™-A8 MPU DPLL_ARM Reconfiguration Does Not Work*

**Revision(s) Affected:** 1.1, 1.0

**Details:** If the user doesn't follow the TINITZ activation procedure, when reconfiguring the DPLL_ARM, then the DPLL_ARM clock output could **stop**.

**Workaround:** To avoid a DPLL_ARM clock stop condition, when activating the TINITZ bit, the following procedure must be followed:

1. Before the software can change the TINITZ bit from "1" to "0", the user ***must*** first poll the FREQLOCK status bit, and wait to see a "1". Once FREQLOCK = 1, then the user can change the TINITZ bit from "1" to "0".
2. If a lock has never been requested on the DPLL_ARM since power-on-reset, then software can change the TINITZ bit from "1" to "0" regardless of the status of the FREQLOCK.

### Advisory 1.1.6 ***PCIe: PCIESS Slave Port Bursts Must be 16-Byte Multiples and 16-Byte Boundary Aligned***

**Revision(s) Affected:** 1.1, 1.0

**Details:** Data is corrupted if 16-Byte multiples and 16-Byte boundary transfer rules are not met.

**Workaround:** The User ***must*** ensure that all EDMA transfers to PCIESS slave are aligned to 16-Byte boundaries and any misaligned accesses are performed as single access.

## Advisory 1.1.7 *PCIe: STATUS_COMMAND and SECSTAT Register Status Bits Are Incorrect*

**Revision(s) Affected:** 1.1, 1.0

**Details:** When a non-posted transaction is invoked between a Requester (in RC mode) and a Completer and the request packet is rejected by the Completer for any reason, the Completer transmits a completion TLP with an unsupported request as the reason for the rejection. This event is designed to be communicated to the user via both the Status and Command (STATUS_COMMAND) and Secondary Status and IO Base/Limit (SECSTAT) registers. However, the status bits within these registers are not updated to reflect the event status.

The "Received Master Abort" bit in the STATUS_COMMAND register and the "RX_MST_ABORT" bit in the SECSTAT register are not asserted when an RC port receives a "completion with unsupported request completion status".

The "Received Target Abort" bit in the STATUS_COMMAND register and the "RX_TGT_ABORT" bit in the SECSTAT register are not asserted when an RC port receives a "completion with completer abort status".

**Workaround:** Once the Header portion of the Completion Transport Layer Packet (TLP) is parsed and placed within the PCIe Header Log registers, software can read the Completion Status field (HEADER_LOG1[bits 23:21]) to determine the captured error condition (for example, 001b for an *Unsupported Request (UR) Completion Status* and 100b for an *Completer Abort (CA) Status*).

Software should read the completion status and this can be used for error detection.

## Advisory 1.1.9 — *USB OTG VBUS: SRP on USB "Initial B" Device Not Supported*

**Revision(s) Affected:** 1.1, 1.0

**Details:** USB OTG PHY (inverted internal SESSEND signal) prevents the use of SRP by the "Initial B" device. In other words, the Host Negotiation Protocol (HNP) part of the OTG protocol is not affected and the role changing capability during the times the VBUS power remains active is functional. However, during times that the "Initial A" device has removed VBUS power, for power savings, the "Initial B" device would be unable to signal the "Initial A" device to resume power on the VBUS.

**Workaround:** Full OTG functionality is not in place and the USB SS should not be used in an OTG environment where role changing is required.

## Advisory 1.1.10 *PCIe Gen2 Mode: PCIESS Corruption of Round Trip Latency Time and Replay Time Limit Bits (PL_ACKTIMER Register)*

**Revision(s) Affected:** 1.1, 1.0

**Details:** When the PCIe is operating in Gen2 mode (5-Gbps rate per PCIe link in each direction), writing to either of these bits, "*Round Trip Latency Time Limit*" (RND_TRP_LMT) or the "*Replay Time Limit*" (RPLY_LIMT), in the PL_ACKTIMER register will cause the value of the bit field that was not being updated to also be modified, corrupting the register contents.

**Workaround:** Ensure that any updates to either the "*Round Trip Latency Time Limit*" (RND_TRP_LMT) or the "*Replay Time Limit*" (RPLY_LIMT) bits in the PL_ACKTIMER register are made only when the PCIe is operating in Gen1 mode (2.5-Gbps rate per PCIe link in each direction).

## Advisory 1.1.11 *MMC/SD/SDIO 0/1/2: MMCHS_STAT Card Error Interrupt Status (CERR) Not Asserted*

**Revision(s) Affected:** 1.1, 1.0

**Details:** After responses of type R1/R1b for all cards, and responses of type R5/R5b/R6 for SD and SDIO cards, the software has to read two registers: MMCHS_CSRE and MMCHS_RSP10 and compare them bit-by-bit to see if any two corresponding bits (i.e., bits in the same position) are "1". If any corresponding bits are "1", then the host controller indicates a card error interrupt status has occurred, and would, normally, set the MMCHS_STAT CERR bit to avoid the host driver reading the response register (MMCHS_RSP10).

During a data transfer if data was frozen on the last bit of a response when valid card error bits are "set", the MMCHS_STAT register card error interrupt status bit (CERR) will not assert.

**Workaround:** Software should read the MMCHS_CSRE and MMCHS_RSP10 registers. If any corresponding bits in these registers are "1", then a card error has occurred, and software should proceed as if the MMCHS_STAT register CERR interrupt bit was detected.

## Advisory 1.1.12 *ARM Cortex™-A8 MPU: Access Restriction on Reserved Address Space to Avoid System Hang*

**Revision(s) Affected:** 1.1, 1.0

**Details:** Any access to Cortex™-A8 MPU **Reserved** address space 0x4010_0000 through 0x401F_FFFF will lead to a system hang.

**Workaround:** ***Do Not*** access the Cortex™-A8 MPU **Reserved** address space 0x4010_0000 through 0x401F_FFFF.

## Advisory 1.1.13 *HDVICP: SBC FIFO Overflow Issue in Encoder*

**Revision(s) Affected:** 1.1, 1.0

**Details:** This condition exists only for JPEG and VC-1 Encoders where the SL2 stall cycle is long and at very-high bit-rate coding.

If the SL2 stall cycle is more than 8 cycles, an SBC FIFO overflow can happen. The SBC FIFO for Encoder is so small that it cannot keep all the encoded data generated by the Encoder.

The SBC FIFO keeps the encoded bitstreams --- if an overflow occurs, the encoded bitstreams are destroyed.

**Workaround:** Keep lower bit-rate encoding so that FIFO does not overflow. This can be achieved by limiting the number of coefficients having a value ≥ 255 to 26 in an 8x8 block.

### Advisory 1.1.14 *HDVICP Encoder: SBC Buffer Overwrite*

**Revision(s) Affected:** 1.1, 1.0

**Details:** The ECD3 can overwrite the bit-stream data in the SL2 stream page buffer before it gets transmitted to external memory, producing an incorrect bit-stream.

This issue is applies to all Encoders.

**Workaround:** Have an explicit wait on the vDMA to make sure that transfer of the stream buffer page has been completed before the ECD3 transition to the particular page.

## Advisory 1.1.15 *HDVPSS: VIP Scaler Causes Continuous VIP_PARSER Overflow if DDR Bandwidth is Over Consumed*

**Revision(s) Affected:** 1.1, 1.0

**Details:**

This *only* occurs in single-channel capture cases where the VIP scaler is being used.

If the HDVPSS VIP Scaler is being used to downscale external video inline with the VIP_PARSER and there is a momentary DDR bandwidth reduction which causes an overflow of the VIP_PARSER, the VIP Scaler will become out of sync with the external video, resulting in continuous overflows, garbled resulting video, but no lock ups.

The VIP scaler is programmed with an input height/width and output height/width. If the actual input does not match the programmed input values, the design will still try to create the programmed output height/width. While it is “making up” the output for the missing input, it stops requesting data from upstream.

If the 4k-byte buffer for holding captured video data in the HDVPSS VPDMA becomes full due to DDR bandwidth restrictions (can’t get data out fast enough), this will cause an overflow at the VIP_PARSER. This will result in the VIP_PARSER dropping lines/pixels while it is in the overflow state. This results in a frame going to the VIP scaler which is smaller than the programmed size.

The VIP Scaler detects the input is the wrong size, stops requesting data, and starts making up data for what is missing. External video data continues to come in, but because the VIP Scaler has stopped requesting data, it causes another overflow of the VIP_PARSER.

While the VIP Scaler is making up the missing data, it is running as fast as it can, and becomes out of sync with external video coming into the VIP_PARSER. The extra overflow it creates makes the next frame lose lines/pixels too, and this continues on.

The overflow could cause distortions on the data captured by the VIP port.

**Workaround(s):**

**Workaround --- Users who use TI SYS/BIOS M3 FVID2 drivers directly**

Perform the following sequence:

- Periodically call the **IOCTL_VPS_CAPT_CHECK_OVERFLOW** API to check the VIP overflow status.
- Once an overflow is detected, call the **IOCTL_VPS_CAPT_RESET_AND_RESTART** API to reset and restart the appropriate VIP (for example, VIP0, if VIP0_PortA or VIP0_PortB is used).

Both Port A and Port B along with all the VIP blocks (including: CSC, SC, and CHR_DS) are reset for that VIP instance.

**Note:** While performing the reset in that VIP instance, ensure that no module is being accessed in either the M2M path or the capture path when the **IOCTL_VPS_CAPT_RESET_AND_RESTART** API is called.

**Workaround --- Users who use the TI SDK Software package**

Perform the following sequence:

- For every process cycle (data queue-in and queue-out) of the VFCC component, check the VIP overflow status of the appropriate VIP (for example, VIP0, if VIP0_PortA or VIP0_PortB is used) by calling the **IOCTL_VPS_CAPT_CHECK_OVERFLOW** API.
- Once an overflow is detected, call the **IOCTL_VPS_CAPT_RESET_AND_RESTART** API to reset and restart the appropriate VIP.

  No intervention is needed in an OMX application. This workaround has already been handled entirely within the OMX VFCC component (SDK 5.0.1 and later). No additional call is required from the application to handle this error.

**Workaround --- Users who do not use TI's Software**

Perform the following sequence to reset/enable the VIP:

- Disable the VIPx port being used via the ENABLE bit in the VIPx PARSER Port A 0 (offset 0x5504, 0x5A04) and Port B 0 (offset 0x550C, 0x5A0C) registers.
- Assert VIP reset [VIPx_DP_RST] in the HDVPSS CLKC Module Reset register (offset 0x0104).
- Assert the Async FIFO reset [CLR_ASYNC_FIFO_WR and CLR_ASYNC_FIFO_RD] in the VIPx PARSER Port A 0 (offset 0x5504, 0x5A04) and Port B 0 (offset 0x550C, 0x5A0C) registers.
- Make a list of abort descriptors for all VIP VPDMA channels.
- Post this list and wait for it to complete.
- De-assert VIP reset [VIPx_DP_RST] in the HDVPSS CLKC Module Reset register (offset 0x0104).
- De-assert the Async FIFO reset [CLR_ASYNC_FIFO_WR and CLR_ASYNC_FIFO_RD] in the VIPx PARSER Port A 0 (offset 0x5504, 0x5A04) and Port B 0 (offset 0x550C, 0x5A0C) registers.
- Enable the VIPx port via the ENABLE bit in the VIPx PARSER Port A 0 (offset 0x5504, 0x5A04) and Port B 0 (offset 0x550C, 0x5A0C) registers.

## Advisory 1.1.19 *HDVPSS: When HDVPSS VIP is Configured to Write to Tiled Memory Space, the Output Descriptors it Generates will also be Written to Tiled Space*

**Revision(s) Affected:** 1.1, 1.0

**Details:** If the HDVPSS VIP is configured to write frame data to tiled memory space, the descriptors it outputs will also be written to tiled space. The descriptor does not overwrite video data. The only issue (if it is an issue) is that the descriptors will be written to tiled DDR space. The user cannot make HDVPSS VIP write video data to tiled space and their corresponding descriptors write to non-tiled space.

**Workaround:** **Workaround --- Users who use TI SYS/BIOS M3 FVID2 drivers directly**

The current HDVPSS software ***does not*** support read descriptor from tiled memory space. The application can read the frame height and width from the external decoders, instead of depending on the hardware driver to provide these values.

Since both the hardware driver and the application cannot get the FIELD_ID information field-by-field, this workaround does not fix the issue of interlaced source capture not being supported when configured to write to Tiled memory.

**Workaround --- Users who use the TI SDK Software package**

None. Currently, TI’s SDK software package ***does not*** support configuring the VIP to write to Tiled memory space.

Once TI’s SDK software supports this feature, updates will be provided.

**Workaround --- Users who do not use TI's Software**

When developing the software, the applications need take this limitation into consideration.

## Advisory 1.1.21 *Boot: Gigabit Ethernet Boot Will Not Work Reliably*

**Revision(s) Affected:** 1.1, 1.0

**Details:** This issue is related to the time it takes for the AR8031 Gigabit PHY to come up. The Gigabit PHYs take about a second to initialize, before they can detect whether a link is present or not. The ROM code waits only for 10 ms to check if the link is up.

When the Ethernet boot mode is selected, and a power-on-reset ($\overline{\text{POR}}$) is applied, the ROM will check whether the link is up, and times out before the PHY is initialized completely. This results in the Ethernet boot mode failing.

Some PHYs in the market can take up to 5 seconds to initialize.

**Workaround:** There are two workarounds to this problem:

1. Issue a Warm Rest about 5 seconds after a power-on-reset ($\overline{\text{POR}}$). This would allow enough time for the PHY to initialize properly, before the ROM code retries the Ethernet boot mode again. However, this would require additional hardware circuitry.
2. Choose a boot mode in which the Ethernet boot comes after UART boot. The UART boot takes about 3 seconds to time out, so if a boot mode is selected where the UART boot is attempted first, the Gigabit PHY would get enough time to initialize before the ROM attempts to do an Ethernet boot.

## Advisory 1.1.23 — *HDMI: On-Chip HDMI Does Not Operate in 8-/16-bit Mode*

**Revision(s) Affected:** 1.1, 1.0

**Details:** In 16-bit output mode, the “G” (Green) bus is used for Y data and the “B” (Blue) bus is used for Cb/Cr interleaved data. The VOUT1 output connects to the “G” and “B” buses properly. The on-chip HDMI wrapper/PHY correctly uses the “G” for Y, but incorrectly uses “R” bus, instead of the “B” bus, for Cb/Cr. This means the HDMI never receives the Cb/Cr data and therefore, the HDMI display shows the wrong colors on the TV in 16-bit mode.

In 8-bit mode, Y data, as well Cb/Cr data, is sent over the “G” bus in an interleaved format. The HDMI wrapper expects the Y/Cb/Cr data on the “G” bus, but since Y and C are interleaved on the same 8-bit bus, the HDMI needs to operate at 2x pixel clock. Since the on-chip HDMI can operate only at 1x pixel clock, the HDMI cannot operate in 8-bit mode.

This error mainly affects those cases in which the user wants to operate both the on-chip HDMI and VOUT1 in 8-/16-bit simultaneously to show the same video content.

The on-chip HDMI has no issue when operating in 24-bit mode.

**Workaround:** For those cases that the user wants to operate both the on-chip HDMI and VOUT1 simultaneously to show the same video content, perform the following steps:

1. Configure both the on-chip HDMI and VOUT1 to 24-bit mode (RGB888 or YUV444 format).
2. Tie the on-chip HDMI with on-chip HD_COMP DAC to show the same video content simultaneously.
3. Connect both the external HDMI transmitter and the external Video DAC to the VOUT0 output to show the same video content simultaneously.

## Advisory 1.1.24 *HDVPSS VOUT[x]_CLK: Does Not Support Positive-Edge Clocking*

**Revision(s) Affected:** 1.1, 1.0

**Details:** The preliminary Data Manual indicated that VOUT data transitions on the rising-edge of the clock when, in fact, VOUT data transitions only on the negative (falling) edge of the clock.

The actual VOUT[x]_CLK to data/control delay times are now referenced to the falling edge and therefore, timing parameters specified in the device-specific data manual have changed.

**Workaround(s):** There are two options for a Workaround:

1. The external device connected to the HDVPSS interface can capture data on the rising edge.
2. If not 1. Workaround, then external logic can be used to invert (*or delay*) the clock to provide adequate setup and hold times to meet the requirements of the external device.

## Advisory 1.1.29 *Boot: Ethernet Boot ROM Code PHY Link Speed Detection*

**Revision(s) Affected:** 1.1, 1.0

**Details:** The device ROM code relies on the external PHY's *Control Register* (Register 0), specifically bits 0.6 [Speed Selection (MSB)] and 0.13 [Speed Selection (LSB)], to determine the operating speed of the link.

If the external PHY does not update its link speed selection bits to reflect the current operating speed, the ROM code will incorrectly assume the PHY is operating at the speed indicated by the link speed selection bits and configure the device Ethernet MAC to the wrong speed. For example, if the default value of the PHY link speed selection bits indicates 1 Gbps, when the PHY is actually operating at 100 Mbps, the ROM will incorrectly configure the device Ethernet MAC for 100 Mbps mode.

The IEEE 802.3 specification states *When Auto-Negotiation Enable (bit 0.12) is enabled, bits 0.6 and 0.13 can be read or written to, but the state of bits 0.6 and 0.13 have no effect on the link configuration, and it is not necessary for bits 0.6 and 0.13 to reflect the operating speed of the link when it is read.* While some PHYs update the link speed in these bits to reflect the current operating speed, other PHYs do not update these bits because it is not mandatory according to the specification.

**Workaround:** When using Ethernet boot, an external PHY that updates the Register 0 link speed selection bits (0.6 and 0.13) to reflect the current operating speed is required.

## Advisory 1.1.30 *Boot: Ethernet Boot ROM Code Sends an Incorrect Vendor Class Identifier in BOOTP Packet*

**Revision(s) Affected:** 1.1, 1.0

**Details:** When using Ethernet boot, the device ROM code should send a BOOTP request with a unique identifier to distinguish itself from other devices on the same network. Instead, the ROM code sends the same identifier for all devices; hence, the download host attempting to bootstrap the devices can no longer determine which device is requesting the code to be downloaded.

Applications using the DM38x, DM813x, DM810x, DMVA3/4, TNETV803x, DM814x, DM816x, and AM335x devices cannot coexist in the same network if they are booted from Ethernet.

**Workaround:** None.

For some applications, it might be necessary to uniquely identify and service BOOTP packets from a client. The recommended approach to uniquely identify clients is to use the MAC address. Every device comes with a unique MAC address. A list of MAC addresses and the device type can be made available to the host in advance, so that the host can take device-specific action when it receives a BOOTP packet from a MAC address on the Host's list.

## Advisory 1.1.31 *ARM CPU/DMM: SIGBUS Fault Under QNX When Accessing Last 48 Bytes of Physical Memory*

**Revisions Affected:** 1.1, 1.0

**Details:** SIGBUS faults have occurred while running the QNX operating system. If the last 48 bytes of physical memory are considered cacheable and an access is made to one of the bytes and it generates a cache miss, the subsequent cache line fill would cause a SIGBUS fault or *data abort* on the access.

**Workaround:** The data abort can be avoided if the specific cache line fill scenario can be avoided. The fundamental workaround is to ensure that the software executing on the Cortex-A8 does not make a cacheable access at the end of a local interconnect and synchronization agent (LISA) mapping. Options include configuring the high-level operating system (HLOS) not recognize to memory at the end of a LISA mapping or making the cumulative effect of the LISA mappings exceed the physical size of external memory, or a combination of both. The solution in workaround 4 is preferred as a general solution for systems that have less than 2GB of memory. It provides a LISA mapping that would exceed the amount of physical memory and it allows the remaining LISA mappings to precisely reflect actual memory. It also allows HLOS configuration to precisely reflect the actual memory. A variation of mappings that achieves the same effect as workaround 4 would be equally preferred. The solution in workaround 2 is required if the system has 2GB of memory.

1. **Guarantee that the last MMU page of a LISA mapping is uncacheable.**

   This capability cannot be readily guaranteed by the HLOS since an application is typically free to allocate memory and declare it cacheable. A customized memory manager would be required. This solution could cover the entire 2-GB physical memory address space for DDR2/3 less the reserved MMU page(s).

2. **Guarantee the last MMU page size of memory in a LISA mapping is reserved from the HLOS.**

   An HLOS provides a means for defining the location and size of all physical memory. Subsequently, the HLOS can be configured not to recognize a block of memory at the end of physical memory; therefore, a cache line access cannot be made to it. If the LISA mapping covers the entire physical memory, it will exceed the physical size recognized by the HLOS. The smallest block of memory that can be withheld should be aligned on an MMU page boundary. The smallest MMU page for the Cortex-A8 is 4KB.

   **Note:** The HLOS must guarantee that an MMU mapping can never be made to the reserved space. This solution could cover the entire 2-GB physical memory address space for DDR2/3 less any reserved blocks of memory.

   ***Example:*** For 64 MB of physical memory, the last 4K bytes are withheld from the HLOS memory manager. The LISA mappings will cover the actual physical memory.

3. **Configure the LISA mapping to a size larger than the underlying physical memory.**

   If using 64M bytes of physical memory, then configure the LISA mapping for 128M bytes but only allow the HLOS to recognize/access the actual 64M bytes of physical memory. If the access is not to actual physical memory it is in error and the MMU would have to guarantee that the access is caught and an exception issued. This solution can only cover up to 2GB less 128MB of physical memory.

   ***Example:*** For two EMIFs with 64MB of memory on each to be configured as two regions at 0x80000000 and 0xC0000000:

   - MAP_0 and _1 value would be 0x80300100 (128MB at 0x80000000).
   - MAP_2 and _3 value would be 0xC0300200 (128MB at 0xC0000000).

   The HLOS only recognizes 64MB at both 0x80000000 and 0xC0000000.

4. **Configure the lowest level LISA mapping (MAP_0) to cover the entire DDR2/3 external address range and force the EMIF to generate a legal data abort.**

Higher-level LISA mappings will accurately describe actual physical memory while the lowest level mapping (MAP_0) will exceed the size of all physical memory and preclude the data abort scenario. MAP_0 will cover the full 2-GB space and directs any access to a reserved space within the EMIF. The DMM will first map an access according to MAP_1, _2 or _3 settings. These higher-level mappings only cover actual physical memory. If the access is not to actual physical memory it is in error and MAP_0 will cover it and map it to a reserved space in the EMIF which will generate a data abort. A data abort on an access to nonexistent memory is a legal fault and will generate an exception to the HLOS. This solution could cover a physical memory address space of 2GB less 128MB.

***Example:*** For two EMIFs with 64MB of memory on each to be configured as two regions at 0x80000000 and 0xC0000000:

- MAP_0 value would be 0x80720100.
- MAP_1 value would be 0x80200100 (64MB at 0x80000000).
- MAP_2 and _3 value would be 0xC0200200 (64MB at 0xC0000000).

The HLOS only recognizes 64MB at both 0x80000000 and 0xC0000000.

## Advisory 1.1.32 *HDVPSS: Unable to Reset HDVPSS Through PRCM or Using CLKC Module in DSS*

**Revision(s) Affected:**

**Details:** The VPDMA is not being reset by the CLKC module although the CLKC module is resetting other modules.

**Workaround:** None.

## Advisory 1.1.33 *XIP Boot: High-Order Address Handling (GPMC_A[27:13] Pins)*

**Revision(s) Affected:** 1.1, 1.0

**Details:** During the XIP Boot process, the ROM code does not multiplex the high-order address lines GPMC_A[27:13] to their address functions, although the external flash memory device generally needs to see a logic "0" on its higher-order address bits to correctly address memory.

Many of the high-order address pads default to internal pulldowns (IPDs) active; however, some high-order address pins default to internal pullups (IPUs) active, and therefore, need to be driven or pulled low via external hardware for the duration of XIP boot operation.

For more details about how the ROM configures the high-order address pins for various XIP boot options, see the *XIP (NOR) Boot Options* subsection of the *Device Configurations* section of the device-specific data manual.

For more details on pins not specifically configured by ROM, see the *PINCNTLx Registers MUXMODE Functions* table in the *Pin Multiplexing Control* section of the device-specific data manual.

**Workaround:** To isolate the flash memory from the lines that are driven to logic “1” it is recommended to put a bus switch on the GPMC_A[27:13] lines between the device and the NOR flash. At boot time, the NOR address lines A[27:13] can be isolated from the device and driven to "0". Once the initial boot completes and the user code starts executing, the code running from the first few kilobytes of NOR flash can correctly configure the mux for the upper address lines. It can then configure the bus switch so that the NOR flash address lines are connected to GPMC_A[27:13] in the device.

## Advisory 1.1.54 — *HDVPSS: VPDMA Line Limit Feature: Descriptor Reports All Fields as Even, but Captures 30 Even and 30 Odd Fields in Memory*

**Revisions Affected:** 1.1, 1.0

**Details:** The VPDMA input descriptor can be set to only capture a specific height and width of input. This is typically used to keep random captured data that is too large from corrupting adjacent memory regions.

If this feature is used, the input video source is interlaced, and the size of the incoming video is larger than what was set, the field ID reported in the output descriptor from VPDMA is 0 for all fields that are larger than the specified height/width settings.

**Workaround:**

- If larger video is coming in than what the VPDMA was configured for, it is difficult to know if this is a software programming mistake or some other problem.
- The field ID (FID) of each input field is captured in the VIP_PARSER capture port. If software believes the source is interlaced and receives two fields with the same FID, it can query the VIP_PARSER to determine the correct FID. After one such check, it can auto-toggle the FID, or continue to check the VIP_PARSER.

## Advisory 1.1.55 *HPVPSS: VIP Inline Color Space Converter (CSC): In Interlaced Embedded or Discrete Sync Mode, Descriptor Reports All Fields as Even, but Captures 30 Even and 30 Odd Fields In Memory*

**Revisions Affected:** 1.1, 1.0

**Details:** The Color Sapce Converter (CSC) does not pass the field ID information correctly to the VPDMA. If the CSC is being used, the FID reported to the VPDMA and thus in the output descriptor is always 0. The CSC would be used to either:

1. convert interlaced RGB capture data (not a common format) to interlaced YUV422 or YUV420 format (not a common capture type)

   or

2. convert interlaced YUV422 capture data to interlaced RGB format (not a common format).

**Workaround:** The FID of each input field is captured in the VIP_PARSER capture port. If software believes the source is interlaced and receives two fields with the same FID, it can query the VIP_PARSER to determine the correct FID. After one such check, it can auto-toggle the FID, or continue to check the VIP_PARSER.

## Advisory 1.1.58 *HDVPSS: HDVPSS VIP Reset Sequence is Occasionally Unsuccessful if VPDMA is Writing Output Descriptors For VIP Captured Data*

**Revisions Affected:** 1.1, 1.0

**Details:** The HDVPSS VIP reset sequence involves aborting the VIP clients within the VPDMA to reset them. Two registers in each client which should be reset on abort were not. These two registers are associated with the writing of the output descriptors from the HDVPSS VIP.

If any descriptor being used for VIP capture has the write descriptor bit set, and a VIP overflow occurs which requires reset on a frame that is writing a descriptor, and immediately after the reset is performed and a DDR-induced stall occurs during the first frame captured, the HDVPSS VPDMA hangs. Only a hard reset of the entire DSS can be performed to clear this hang.

A stall on the first frame after reset can only occur if the first frame after the reset is performed is sent to DDR. If the first frame after reset completes without stall to the end of frame, the two registers get set to their proper state.

**Workaround:** The first frame captured after completion of the VIP reset sequence should have both the drop data and write descriptor bits set, which keeps the contents of that frame from going to DDR, thus avoiding the potential stall condition.

## Advisory 1.1.59 *HDVPSS: Occasionally, Chip Lockup Occurs When HDVPSS VIP is Performing Single-Channel Capture, Sending Tiled Data to DDR and Connect/Disconnect Events are Occurring*

**Revisions Affected:** 1.1, 1.0

**Details:** In tiled mode, the VPDMA VIP client multiplies the captured line width by either 2 or 4 (depending on tiled container size), and then a downstream CDMA module divides this by 2 or 4. An issue causes the VIP client to load the tiled mode status of the next descriptor before the complete frame is sent. If the next descriptor is not tiled, this causes the client to think it is not tiled, and if the captured width is less than either 2 or 4 128 byte words, the CDMA divides this number, producing a 0, which is sent as the OCP burst size, resulting in chip lockup disconnect.

The above condition can only occur under three circumstances:

1. The HDVPSS VIP scaler is scaling to a line width where the modulo 128 of the width is less than 2 or 4 bytes (depending on container size).
2. Connect/disconnect events are occurring such that line widths getting to the HDVPSS VPDMA VIP clients can have a width that matches the above condition.
3. Frame sizes are less than the sampling period of the descriptor pacing period. Normally, it is assumed that the sampling period is 2x of the frame period, but when receiving shorter frames caused by connect/disconnect, this is not true.

**Workaround:** Always keep the mode bit the same for all descriptors. If the software wants to change from tiled to non-tiled or non-tiled to tiled, then it must abort the client and give new descriptors with the new mode bit. If performing tiled transfers and drop data bit is set, the mode bit **must** be set. In this case the descriptor address must be converted to a tiled address and not use the normal L3 memory map.

## Advisory 1.1.60 *HDVPSS: Occasionally, During Connect/Disconnect and When Line or Width Limit Feature Not Used, Any Memory Areas Can Be Overwritten*

**Revisions Affected:** 1.1, 1.0

**Details:** When connect/disconnect events occur, it is possible for any size frame to be captured by the HDVPSS VIP. If the captured frame is corrupted by the connect/disconnect event such that a very long or very wide frame is captured, and the VPDMA maximum width/maximum height feature is not used (set to unlimited), memory areas outside of those allocated by software can be written to.

If adjacent memory areas are program code, this can cause random behavior depending on exactly what is written.

**Workaround:** Ensure the VPDMA maximum width/maximum height values are set to something other than *unlimited*. However, the maximum size to which these can be set is 1920x1080.

## Advisory 1.1.63 — *HDVPSS: VIP Single-Channel Capture Using Tiled Output Can Lead To VIP Lockup if Connect/Disconnect Events Occur*

**Revisions Affected:** 1.1, 1.0

**Details:** When HDVPSS VIP is configured to output tiled mode, the VPDMA has to line buffer either two or four lines (depending on container size) of data to perform the tiling operation. The VPDMA assumes all lines have the same length. When a connect/disconnect events occur, any line can have any length. If the last of the set of lines (last of two or last of four) is either short or long compared to the previous lines, the VPDMA gets into a hung state, resulting in VIP overflow.

**Workaround:** Use the VIP reset sequence to clear the overflow. This will result in 2-3 frames lost on the channel where the event occurred.

## Advisory 1.1.66 *PCI Express (PCIe): PCIe Boot Fails When Connected to Some PCs*

**Revisions Affected:** 1.1, 1.0

**Details:** The ROM code cannot handle hot-reset or disable-link packets. It has been observed on some motherboards on some slots, that the PC BIOS issues hot-reset or disable-link transactions on the PCIe bus. When the device sees these transactions, the PCIe controller gets reset automatically.

This reset causes the PCIe controller to lose the configuration that was programmed by ROM and, as a result, the PCIe boot does not complete. This issue is seen only when interfacing with some PCs. It is not seen when one device acting as root complex tries to boot another device acting as endpoint.

**Workaround:** If possible, use another slot on the PC.

If it is not possible to change slots, do not use PCIe boot. Boot from a SPI flash and the code running from the SPI flash can configure the PCIe and also handle reconfiguring the PCIe controller when a hot reset or disable link is detected.

## Advisory 1.1.70 *USB: An Attached Non-Compliant USB Device That Responds to a Spurious Invalid Short Packet can Lock-Up the Bus*

**Revision(s) Affected:** 1.1, 1.0

**Details:** The integrated USB PHY (analog transceiver) has a timing error that turns on its receiver too early and occasionally detects the end of its own transmit data as receive data. This causes the USB controller to transmit an invalid short packet. Normally this invalid short packet would be ignored by the attached USB device and the data transmission would continue as expected.

At least one mass storage class USB device has been found to be non-compliant to the USB specification, by responding to this packet. This non-compliant response ("NACK") to the invalid short packet violates USB protocol and the bus to hang.

Poor signal integrity of the differential signal pair used to connect the attached USB device may contribute to this issue. Impedance discontinuities and mismatched terminations on the differential signal pair may cause reflections to propagate longer than expected which allows the transceiver to detect these reflections of its own transmit data as receive data.

**Workaround:** None.

Only attach USB devices compliant to the USB specification to prevent an unexpected response to any invalid short packets.

It is also recommended that the USB DP/DM signals are routed as a 90-Ω differential pair transmission line with minimum impedance discontinuities and proper terminations to minimize reflections.

### Advisory 1.1.71 *PCIe Boot: ROM Code Delay in Waiting for SERDES PLL to Lock Does Not Meet Specifications Requirement*

**Revision(s) Affected:**

**Details:** Currently the ROM code waits only for about 1 µs between enabling the LDO and locking the SERDES PLL. The PLL spec states that the delay must be 50 µs. This 1 µs delay works but, could be a borderline case - so the delay needs to be increased to 50 µs.

**Workaround:** No workaround.

### Advisory 1.1.76 *UART: Extra assertion of the FIFO Transmit DMA Request, UARTi_DMA_TX*

**Revision(s) Affected:** 1.1, 1.0

**Details:** UART TX default configuration would result in extra DMA req assertion when FIFO tx_full is switched from high to low. This is because of the TX THRESHOLD added in ES2.0 compared to ES1.0

A UART TX with a DMA THRESHOLD default configuration of 64 bytes would result in an extra DMA request assertion when FIFO tx_full is switched from high-to-low.

**Workaround:** The S/W workaround is to use TX_THRESHOLD + TRIGGER_LEVEL ≤ 63 (TX FIFO Size - 1).

## Advisory 1.1.78 ***Boot: PCIe Boot Mode Does Not Work By Default***

**Revision(s) Affected:** 1.1, 1.0

**Details:** In this device, the PCIe Boot mode does not work by default due to conflicting PCIe base address register and target boot address.

**Workaround:** The Root Complex (RC) should write the value 0x22C3D to the location 0x4031CF84 followed by writing the value 0x12345678 to 0x4031B7FC. The RC must then wait for 5 ms. The RC will then have to perform a re-enumeration. On the RC side, this can involve steps such as:

- Polling the PCI config register to read Vendor ID and Device ID
- Saving and assigning resources

On the EP side, writing the values will cause the EP to re-initialize the PCIe controller automatically.

Once the link is setup, the RC must update the boot flag location (0x4031B7FC) once more.

### Advisory 1.1.85 *EMAC and Switch Subsystem: Reset Isolation Feature is Not Supported*

**Revision(s) Affected:** 1.1, 1.0

**Details:** The Ethernet Media Access Controller and Switch (CPSW) subsystem may lock up if the Reset Isolation feature is enabled when a warm reset is applied while the host port is transmitting data. Since most warm reset sources can be asynchronous events, this lock-up condition can only be prevented by not enabling this feature. The Reset Isolation feature is not enabled by default.

**Workaround:** There is no workaround for this issue.

## Advisory 1.1.88 *Control Module, Pin Configuration (PINCNTLx): ROM Modifies Bit 19*

**Revision(s) Affected:** 1.1, 1.0

**Details:** Bit 19 within each PINCNTL[270:1] register is a RESERVED bit which should not be modified by any software during device operation. The reset value is specified to ensure datasheet minimum and maximum I/O timings are satisfied across all of the various functions mapped to the corresponding pin. The ROM bootloader initialization incorrectly modifies bit 19 for some of the PINCNTLx registers. Table 3 and Table 4 identify the PINCNTLx registers, per boot mode, affected by the ROM bootloader incorrectly modifying bit 19 from its default value.

**Note:** The ROM bootloader will attempt successive boot methods if a previous boot mode fails. One or more boot modes may apply depending on the configured BTMODE[4:0] pins.

### EMAC

**Table 3. EMAC Boot**

| BOOT MODE OPTIONS (BTMODE[9:8] PINS) | DESCRIPTION | PINCNTLx REGISTER |
|---|---|---|
| 10b | RGMII | 238, 239, 241, 242, 244 |

### MMC

**Table 4. MMC Boot**

| BOOT MODE OPTIONS | DESCRIPTION | PINCNTLx REGISTER |
|---|---|---|
| – | – | 2-6 |

**Note:** In addition to the ROM bootloader software, the following TI software packages may potentially modify bit 19 of the PINCNTLx registers. If any of the following TI software packages are used, care should be taken to properly reset bit 19 to the default value during device operation:

- U-BOOT
- LINUX Kernel (board init files, video drivers)
- HDVPSS Firmware

**Workaround:** To prevent potential datasheet I/O timing violations, software should set bit 19 for all PINCNTLx registers to the reset value defined in the device-specific data manual. Software should also ensure that bit 19 of all PINCNTLx registers is not modified during device operation.

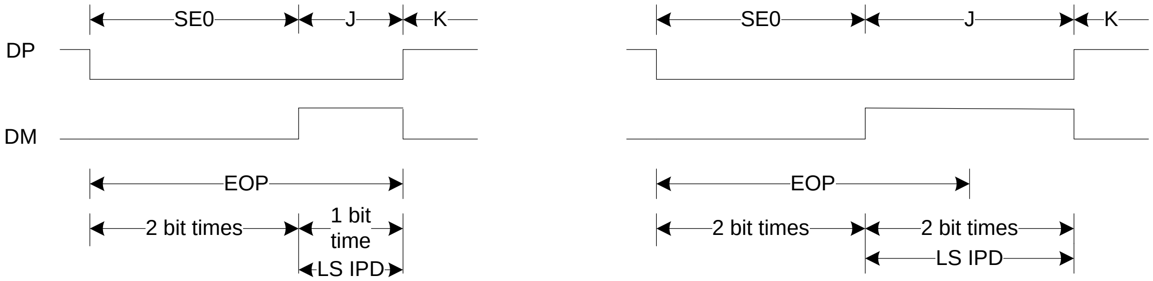
### Advisory 1.1.89 *USB: Incorrect Low Speed Inter-Packet Delay between the Receiver-to-Transmitter Transactions*

**Revision(s) Affected:** 1.1, 1.0

**Details:** The issue appears when connecting a specific USB Low Speed (LS) mouse to the processor’s USB port which is in Host mode. In this scenario the Host sends an IN token to the mouse (Device) which sends the data back to the Host. The Host acknowledges the reception of the data with an ACK having an inter-packet delay (IPD) of 1. The mouse does not see the ACK (because it came too soon); therefore, it does not know that the Host has received the data. The Host sends a second IN token to the Device, expecting a second Data packet, but it receives the first Data packet instead. This is because the Device re-sent the first Data packet. Now the Host and Device are out of sync.

The problem is due to the length of the IPD between the Receiver-to-Transmitter (Rx-to-Tx) transactions. The Host IPD between Rx-to-Tx transactions for LS transactions should range from 2 to 6.5 LS bit times. It has been verified that the controller’s LS IPD for Rx-to-Tx is only 1. This is shown in the following figure.

IPDs are measured from the SE0-to-J transition at the end of EOP to the J-to-K transition that starts the next packet.



IPD=1 and does NOT meet the IPD minimum standard

IPD=2 and DOES meet the IPD minimum standard

The impact is that this specific mouse will not work with the processor’s USB port.

**Workaround:** There are many USB mice on the market that work with an Rx-to-Tx IPD of 1. Switching from the problem mouse to a different mouse will solve the problem. Another solution is to connect a hub between the Host USB port and the Device USB port. The hub will add additional IPD to allow the original mouse to work.

## Revision History

This silicon errata revision history highlights the technical changes made to the current revision.

**Scope:** Applicable updates relating to the device has been incorporated.

### Revision History

| SEE | ADDITIONS/MODIFICATIONS/DELETIONS |
|---|---|
| Table 1 | Added devices DM388AAARD11FP and DM388AAARD21FP. |

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

# IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

| **Products** | | **Applications** | |
|---|---|---|---|
| Audio | www.ti.com/audio | Automotive and Transportation | www.ti.com/automotive |
| Amplifiers | amplifier.ti.com | Communications and Telecom | www.ti.com/communications |
| Data Converters | dataconverter.ti.com | Computers and Peripherals | www.ti.com/computers |
| DLP® Products | www.dlp.com | Consumer Electronics | www.ti.com/consumer-apps |
| DSP | dsp.ti.com | Energy and Lighting | www.ti.com/energy |
| Clocks and Timers | www.ti.com/clocks | Industrial | www.ti.com/industrial |
| Interface | interface.ti.com | Medical | www.ti.com/medical |
| Logic | logic.ti.com | Security | www.ti.com/security |
| Power Mgmt | power.ti.com | Space, Avionics and Defense | www.ti.com/space-avionics-defense |
| Microcontrollers | microcontroller.ti.com | Video and Imaging | www.ti.com/video |
| RFID | www.ti-rfid.com | | |
| OMAP Applications Processors | www.ti.com/omap | **TI E2E Community** | e2e.ti.com |
| Wireless Connectivity | www.ti.com/wirelessconnectivity | | |

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