TMS320DM8127 DaVinci™ Video Processors Silicon Revisions 3.0, 2.1

Silicon Errata

Literature Number: SPRZ385C
June 2012–Revised March 2014

Contents

List of Figures

List of Tables

TMS320DM8127 DaVinci™ Video Processors Silicon Revisions 3.0, 2.1

1 Introduction

This document describes the known exceptions to the functional specifications for the TMS320DM8127 DaVinci™ Video Processors. The updates are applicable to the CYE package. For additional information, see the *TMS320DM8127 DaVinci™ Video Processors* data manual (literature number SPRS712).

Throughout this document, unless otherwise specified, TMS320DM812x and DM812x, refer to the TMS320DM8127 device. For additional peripheral information, see the latest version of the *TMS320DM814x DaVinci Digital Media Processors Technical Reference Manual* (Literature Number: SPRUGZ8).

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 resequenced.

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 DSP devices and support tools. Each DSP commercial family member has one of three prefixes: TMX, TMP, or TMS (for example,TMS320DM8127SCYE0). 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 (TMX/TMDX) through fully qualified production devices/tools (TMS/TMDS).

Device development evolutionary flow:

TMX — Experimental device that is not necessarily representative of the final device's electrical specifications.

TMP — Final silicon die that conforms to the device's electrical specifications but has not completed quality and reliability verification.

TMS — Fully-qualified production device.

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.

TMX and TMP devices and TMDX development-support tools are shipped against the following disclaimer:

"Developmental product is intended for internal evaluation purposes."

TMS 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 (TMX or TMP) 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 examples of the TMS320DM8127 processor package symbolization. The device revision can be identified by the markings on the top of the CYE package; the "B" *or* "S" between the device number and the package identifier indicates the device revision ("B" for silicon revision 2.1 and "S" for silicon revision 3.0), as noted in Figure 1.

Table 1 lists the device revisions for the TMS320DM8127 processor.

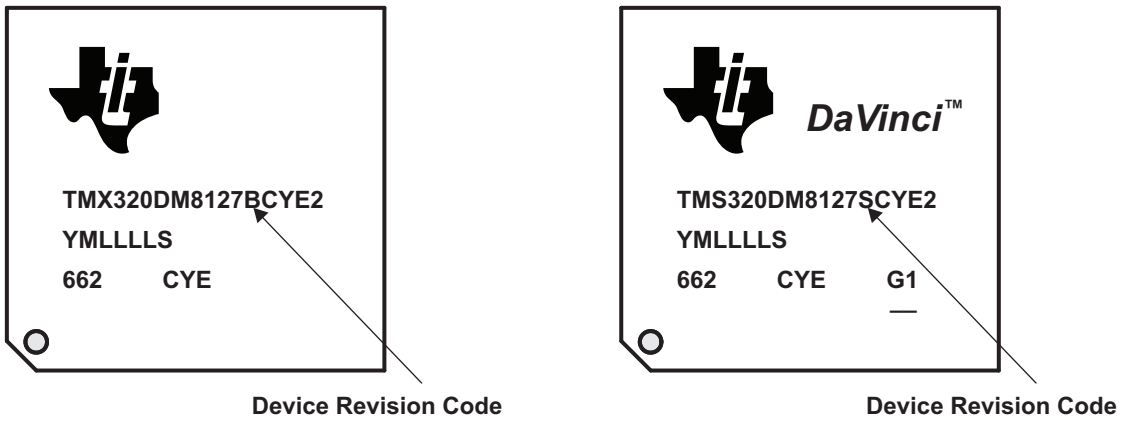

Figure 1. Example, Device Revision Codes for TMS320DM8127 (CYE Package)

Table 1. TMS320DM8127 Device Revision Codes

DEVICE REVISION CODE	SILICON REVISION	PART NUMBERS/COMMENTS
B	2.1	TMX320DM8127BCYE1, TMX320DM8127BCYED1 TMS320DM8127BCYE0, TMS320DM8127BCYED0, TMS320DM8127BCYE1, TMS320DM8127BCYED1, TMS320DM8127BCYE2, TMS320DM8127BCYED2, TMS320DM8127BCYE3, TMS320DM8127BCYED3
S	3.0	TMS320DM8127SCYE0, TMS320DM8127SCYED0, TMS320DM8127SCYEA0, TMS320DM8127SCYE1, TMS320DM8127SCYED1, TMS320DM8127SCYE2, TMS320DM8127SCYED2, TMS320DM8127SCYE3, TMS320DM8127SCYED3

2 Silicon Revision 3.0 Usage Notes and Known Design Exceptions to Functional Specifications

2.1 Usage Notes for Silicon Revision 3.0

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 DM812x silicon revision 3.0 and earlier, 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 0x7.

2.1.2 Some PLLs Only Support Even M2 Post Dividers

On DM812x silicon revision 3.0 and earlier, 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 (e.g., 2, 4, etc. up to 126).

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

- PLL_DSP
- PLL_HDVICP
- PLL_L3
- 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 and DDR3 Requires Software Leveling

On all silicon revisions, DDR2 and DDR3 requires 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 TI DM814x Embedded Processors Wiki Page.

2.2 Silicon Revision 3.0 Known Design Exceptions to Functional Specifications

This is a list of the device silicon revision 3.0 known design exceptions to functional specifications.

Table 2. Silicon Revision 3.0 Advisory List

Table 2. Silicon Revision 3.0 Advisory List (continued)

Advisory 3.0.3 *Debugger Cannot Access Memory Controller Memories if Functionally Powered Down*

Revision(s) Affected: 3.0 and earlier

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 3.0.5 *DPLL_ARM: Cortex™-A8 MPU DPLL_ARM Reconfiguration Does Not Work*

Revision(s) Affected: 3.0 and earlier

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

Revision(s) Affected: 3.0 and earlier

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 3.0.7 *PCIe: STATUS_COMMAND and SECSTAT Register Status Bits Are Incorrect*

Revision(s) Affected: 3.0 and earlier

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 3.0.8 *DDR0/DDR1: LPDDR/DDR2/DDR3 Chip Select 1 (DDR[x]_CS[1]) Feature Not Supported*

Revision(s) Affected: 3.0 and earlier

Details: For both DDR0 and DDR1 modules, DDR[x]_CS[1] feature is not supported.

Workaround: Do Not Use the DDR[x]_CS[1] function! Configure the DDR0/DDR1 peripherals using DDR[x]_CS[0] *only*.

Advisory 3.0.9 *USB OTG VBUS: SRP on USB "Initial B" Device Not Supported*

Revision(s) Affected: 3.0 and earlier

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

Revision(s) Affected: 3.0 and earlier

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

Revision(s) Affected: 3.0 and earlier

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

Revision(s) Affected: 3.0 and earlier

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 3.0.13 *HDVICP: SBC FIFO Overflow Issue in Encoder*

Revision(s) Affected: 3.0 and earlier

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 3.0.14 *HDVICP Encoder: SBC Buffer Overwrite*

Revision(s) Affected: 3.0 and earlier

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 3.0.15 *HDVPSS: VIP Scaler Causes Continuous VIP_PARSER Overflow if DDR Bandwidth is Over Consumed*

Revision(s) Affected: 3.0 and earlier

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 3.0.16 *HDVPSS: Enabling and Disabling the VIP_PARSER Within the HDVPSS VIP in Single-Channel Capture and Discrete-Sync Modes, can Lead to Lockup*

Revision(s) Affected: 3.0 and earlier

Details: This occurs only when in single-channel capture and discrete-sync modes.

In discrete sync or single-channel capture mode, if the VIP_PARSER register within the HDVPSS VIP is disabled and re-enabled, processing elements following the VIP_PARSER register can lock up.

This error could cause the VIP port to lockup and stop capturing any data.

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

Call the IOCTL API (**IOCTL_VPS_CAPT_RESET_AND_RESTART**) to reset and restart the appropriate VIP.

Note: The application should force a reset by calling the IOCTL API before calling the **FVID2_START** API.

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

Perform the following sequence:

- Before calling the **FVID2_START** API, force a reset by calling the IOCTL API (**IOCTL_VPS_CAPT_RESET_AND_RESTART**) on the appropriate VIP (for example, VIP0, if VIP0_PortA or VIP0_PortB is used).

  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 3.0.18 *HDVPSS: Capture of RGB and Converting to YUV or Capture of YUV and Converting to RGB Inline Within the HDVPSS VIP can Lead to VIP Path Lockup if DDR Bandwidth is Over Consumed*

Revision(s) Affected: 3.0 and earlier

Details:

Video capture of RGB data can be converted to YUV data within the HDVPSS VIP. RGB data is first converted to YUV444 data, then converted to YUV422 data using a 444 to 422 converter.

Video capture of YUV422 data can be converted to RGB data within the HDVPSS VIP. YUV422 data is first converted to YUV444 data using a 422 to 444 converter and then converted to RGB data.

The 422 to 444 converter requires a minimum of four pixels per line or it will lock up. The 444 to 422 converter requires a minimum of nine pixels per line or it will lock up.

Normal video will not contain this few a number lines per frame, but it is possible that a momentary DDR bandwidth reduction can cause this to occur. When this line reduction per frame occurs, it will always be proceeded by a VIP_PARSER overflow event.

This can only happen in single channel capture modes when RGB is being converted to YUV inline, or YUV is being converted to RGB in line.

Frequency is dependent on DDR loading. There are 4KB of buffering for captured video data. Overflow at the VIP_PARSER which can lead to this issue will only occur if this 4k-byte buffer overflows.

This error could cause the VIP port to lockup and stop capturing any data.

Workaround:

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

None. Currently, TI's SDK software package ***does not*** support either the capturing of RGB and converting to YUV *or* the capturing of YUV and converting to RGB.

Once TI's SDK software supports these capture/conversion features, updates will be provided.

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

Perform the following sequence to disable and re-enable the HDVPSS 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 3.0.19 *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: 3.0 and earlier

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 3.0.21 *Boot: Gigabit Ethernet Boot Will Not Work Reliably*

Revision(s) Affected: 3.0 and earlier

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 3.0.23 *HDMI: On-Chip HDMI Does Not Operate in 8-/16-bit Mode*

Revision(s) Affected: 3.0 and earlier

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 3.0.24 *HDVPSS VOUT[x]_CLK: Does Not Support Positive-Edge Clocking*

Revision(s) Affected: 3.0 and earlier

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.

For more detailed information on the timing parameter changes, see the *TMS320DM8127 DaVinci™ Video Processors* Data Manual (Literature Number: SPRS712).

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 3.0.25 *DEMMU System MMU May Hang When Used in Table-Walk Mode*

Revisions Affected: 3.0 and earlier

Details: When an access is made through the System MMU toward the end of a virtual page and the page (corresponding to the next incrementing virtual address range) is not resident in the TLB, the System MMU may hang. This hang may cause the requestor (C674x or EDMA) to hang. All subsequent requests that target the System MMU will also hang. Once a hang occurs, the only recovery mechanism is to reset the device.

The hang occurs because an internal counter for the address increments without accounting for potential stalls. Because of this, a request that was actually contained within a page spills into the second page, which is a miss. The System MMU is not designed to handle this page crossing and causes a hang.

Workaround: One of the following workarounds must be used to avoid the hang situation:

1. Leave System MMU in bypass mode. The System MMU can be set to bypass mode by either of the following methods:
   - Set MMU_EN = 0 (default value is 1, enabled) in the MMU_CFG register.
   - Set MMUENABLE = 0 (default value is 0, disabled) in the MMU_CNTL register.

   Note that by default on RESET, the DEMMU is in bypass mode since the MMUENABLE bit in the MMU_CNTL register is 0.

   For potential issues with this workaround, see *System MMU Bypass - GPMC Accessibility Limitations* below.
2. Lock all required entries within the MMU.

   The MMU consists of 32 entries, where each entry can map either a 4KB, 64KB, 1MB, or 16MB range. Thus, this workaround is straight forward if the DSP accesses, at most, 32 different 16MB (or smaller) ranges. It is recommended that the final 1KB at the end of a contiguous range is not used to avoid potential hardware pre-fetching crossing to an unmapped page boundary.

System MMU Bypass - GPMC Accessibility Limitations

In order for the DSP to access GPMC directly, the System MMU must be used to remap the GPMC physical address range to a different virtual address. Therefore, Workaround 1 cannot be used and also support direct DSP access to the GPMC addresses. If Workaround 1 is desired, then the DSP can access the GPMC via indirect means, such as by submitting an EDMA request to access the GPMC. Alternatively, Workaround 2 can be used.

The reason for this limitation is that the DSP views virtual addresses between 0x0000 0000 and 0x10FF FFFF as local addresses (mapped to DSP L1D, L1P, L2, and control registers). The L3 interconnect maps a part of GPMC addresses to the same range. In order for the DSP to directly access GPMC, the DEMMU must be used to remap a chosen virtual address (say 0x2000 0000) to the GPMC physical address of 0x0000 0000.

System MMU Lockdown Example:

- Application usage needs from DDR:
  - 32M GPMC (0x0000 0000 - 0x007 FFFFF)
  - 64M DDR (0x8000 0000 - 0x80 FFFFFF)

- System MMU Entries:

```
– Entry 0: Super-section: GPMC: 0x0000 0000 - 0x00FF FFFF    //16MB
– Entry 1: Super-section: GPMC: 0x0100 0000 - 0x01FF FFFF    //16MB
– Entry 2: Small Page:     PAD: 0x0200 0000 - 0x0200 0FFF    //4KB
– Entry 3: Super-section:  DDR: 0x8000 0000 - 0x80FF FFFF    //16MB
– Entry 4: Super-section:  DDR: 0x8100 0000 - 0x81FF FFFF    //16MB
– Entry 5: Super-section:  DDR: 0x8200 0000 - 0x82FF FFFF    //16MB
– Entry 6: Super-section:  DDR: 0x8300 0000 - 0x83FF FFFF    //16MB
– Entry 7: Small Page:     PAD: 0x8800 0000 - 0x8800 0FFF    //4KB
```

NOTES:

1. The System MMU has only 32 TLB entries. This may limit the usable address range. For example, using super-sections (16MB), the overall addressable range can be 32*16MB = 512MB. The System MMU allows the TLB entry size to be super-section (16MB), section (1MB), large page (64KB) and page (4KB).
2. An OS (such as Linux) often requires full System MMU functionality including the use of hardware table walks (for page misses) and small pages. In this case, crossing page boundaries is unavoidable. Therefore, if the System MMU is used, this approach would require that any buffers shared between Cortex™-A8 and C674x need to be allocated from a contiguous memory rather than allocated in small pages.

Example Code:

```
#define __raw_readl(a)                  (*(volatile unsigned int *)(a))
#define __raw_writel(v, a)              (*(volatile unsigned int *)(a) = (v))
#define __raw_readw(a)                  (*(volatile unsigned short *)(a))
#define __raw_writew(v, a)              (*(volatile unsigned short *)(a) = (v))

#define SYS_MMU_BASE_ADDR                                   0x48010000
#define MMU_TTB     (SYS_MMU_BASE_ADDR + 0x4C)
/* IKD new defines for more MMU registers */
#define MMU_LOCK    (SYS_MMU_BASE_ADDR + 0x50)
#define MMU_LD_TLB  (SYS_MMU_BASE_ADDR + 0x54)
#define MMU_CAM     (SYS_MMU_BASE_ADDR + 0x58)
#define MMU_RAM     (SYS_MMU_BASE_ADDR + 0x5C)

#define MMU_LOCK_BASEVALUE_LSB     10
#define MMU_LOCK_CURRENTVICTIM_LSB 4

#define MMU_CAM_V 0x4
#define MMU_CAM_P 0x8

#define MMU_CAM_SECTION      0x0
#define MMU_CAM_LARGEPAGE    0x1
#define MMU_CAM_SMALLPAGE    0x2
#define MMU_CAM_SUPERSECTION 0x3

#define MMU_CAM_VATAG_MASK   0xfffff000
#define MMU_RAM_PHYADDR_MASK 0xfffff000
/* IKD */

#define TTB_ADDR                                        (0x80000000+54*1024*1024)

#define MMU_TTB_MASK          0xFFFFC000
#define MMU_SECTION_ADDR_MASK 0xFFF00000
#define MMU_CNTL     (SYS_MMU_BASE_ADDR + 0x44)

#define GPMC_PHYSICAL_ADDR    0x08000000
#define GPMC_VIRTUAL_ADDR     0x11000000
#define OCMC0_PHYSICAL_ADDR   0x40300000
#define OCMC0_VIRTUAL_ADDR    0x40300000
#define EMIF0_PHYSICAL_ADDR   0x80000000
#define EMIF0_VIRTUAL_ADDR    0x80000000

/* IKD */
```

```
#define SUPERSECTIN_SIZE      0x1000000
void mmu_setup_locked_entries();
void mmu_lock_section(int curr_entry, unsigned long va, unsigned long pa);

void main() {
                SYS_MMU_TWL();
}
void mmu_lock_section(int curr_entry, unsigned long va, unsigned long pa)
{
  unsigned long lock;

  /* point to the entry & lock entries till the same */
  lock =
    ((curr_entry + 1) << MMU_LOCK_BASEVALUE_LSB) |
    (curr_entry << MMU_LOCK_CURRENTVICTIM_LSB);
  __raw_writel(lock, MMU_LOCK);

  /* setup CAM and RAM */
  __raw_writel((va & MMU_CAM_VATAG_MASK) | MMU_CAM_V | MMU_CAM_P |
                    MMU_CAM_SUPERSECTION, MMU_CAM);

  __raw_writel((pa & MMU_RAM_PHYADDR_MASK), MMU_RAM);

  __raw_writel(1, MMU_LD_TLB); /* load an entry */

}
/*
 * lock some entries in the MMU so TWL does not have to happen
 */
void mmu_setup_locked_entries()
{
  int curr_entry = 0;
  int i;

  /* GPMC - one extra section at the end */
  for(i=0; i<2; i++){
    mmu_lock_section(curr_entry++,
                                      GPMC_VIRTUAL_ADDR + i*SUPERSECTIN_SIZE,
                                      GPMC_PHYSICAL_ADDR + i*SUPERSECTIN_SIZE);
  }

  /* DDR - one extra section at the end */
  for(i=0; i<9; i++){
    mmu_lock_section(curr_entry++,
                                      EMIF0_VIRTUAL_ADDR + i*SUPERSECTIN_SIZE,
                                      EMIF0_PHYSICAL_ADDR + i*SUPERSECTIN_SIZE);
  }

  /* OCMC */
  mmu_lock_section(curr_entry++, OCMC0_VIRTUAL_ADDR, OCMC0_PHYSICAL_ADDR);
}

/* IKD */
int SYS_MMU_TWL()
{
                /* IKD */
                mmu_setup_locked_entries();
                /* IKD */
                __raw_writel(0x2,MMU_CNTL);

                return 0;
}
```

Advisory 3.0.29 *Boot: Ethernet Boot ROM Code PHY Link Speed Detection*

Revision(s) Affected: 3.0 and earlier

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 3.0.30 *Boot: Ethernet Boot ROM Code Sends an Incorrect Vendor Class Identifier in BOOTP Packet*

Revision(s) Affected: 3.0 and earlier

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, "DM814x ROM v1.0", for all devices (i.e., DM812x, DM814x, DM816x, and AM335x, etc.); 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 DM812x, DM814x, DM816x, AM335x, DM38x, DM813x, DM810x, and DMVA3/4 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 3.0.31 *ARM CPU/DMM: SIGBUS Fault Under QNX When Accessing Last 48 Bytes of Physical Memory*

Revisions Affected: 3.0 and earlier

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 [for architecture details, see the DMM Functional Description section of the (literature number]. 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 3.0.32 *HDVPSS: Unable to Reset HDVPSS Through PRCM or Using CLKC Module in DSS*

Revision(s) Affected: 3.0 and earlier

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

Workaround: None.

Advisory 3.0.33 *XIP Boot: High-Order Address Handling (GPMC_A[27:13] Pins)*

Revision(s) Affected: 3.0 and earlier

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

Revisions Affected: 3.0 and earlier

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.

When this feature is used with interlace video inputs, the field ID will not be correctly reported if the size of the incoming video is larger than the specified height/width settings. The field ID reported in the output descriptor will be 0 for all fields that are larger than the specified height and width.

Workaround: There is currently no workaround for this issue. The system must determine the size of the incoming video and set the height /width settings accordingly.

Advisory 3.0.55 *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: 3.0 and earlier

Details: If the Color Space Converter (CSC) is used with interlaced RGB formats, the field ID is not reported correctly to the VPDMA. The field ID reported to the VPDMA is 0 for both fields. This issue occurs when converting from interlaced RGB to interlaced YUV422 and when converting from interlaced YUV422 to interlaced RGB.

Workaround: There is currently no workaround for this issue. Support for interlaced RGB formats is typically not required.

Advisory 3.0.58 *HDVPSS: HDVPSS VIP Reset Sequence is Occasionally Unsuccessful if VPDMA is Writing Output Descriptors for VIP Captured Data*

Revisions Affected: 3.0 and earlier

Details: The HDVPSS VIP reset sequence involves aborting the VIP clients within the VPDMA to reset them. Two registers in the VPDMA capture client are not automatically reset during this abort sequence. Under some conditions, this can stall or hang the VPDMA when capture is initiated following the abort sequence. This stall condition can only occur if the first frame is sent to DDR.

If a VPDMA stall occurs, a hard reset of the entire DSS is required to clear the stall.

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

Use SDK v5.04.00.11 and later.

This VPDMA stall condition is prevented in the HDVPSS VIP reset sequence provided in the current S/W releases.

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

The VPDMA stall condition can be avoided if the VIP reset sequence sets the Drop Data and Write Descriptor bits in the frame capture descriptor prior to the first frame capture following an abort sequence. Setting the two bits in the frame capture descriptor prevents the first frame from being stored to DDR, allows the two registers to be cleared, and prevents the stall condition. The Write Descriptor and Drop Data bits are documented in the Data Packet Descriptor Word 4 in the *TMS320DM814x High-Definition Video Processing Subsystem (HDVPSS)* User Guide (literature number: SPRUHF7A).

Advisory 3.0.60 *HDVPSS: Occasionally, During Connect/Disconnect and When Line or Width Limit Feature Not Used, Any Memory Areas Can Be Overwritten*

Revisions Affected: 3.0 and earlier

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 overwritten.

if program code is overwritten, random behavior may result.

Workaround: Ensure the VPDMA maximum width/maximum height values are set to something other than *unlimited*. Typical applications set width and height settings based on the desired input format. Setting the width and height settings prevents memory corruption when supporting input resolutions up to 1920x1080.

There is no workaround for input resolutions greater than 1920x1080, as these higher resolution formats require a maximum width/maximum height setting of *unlimited*.

Advisory 3.0.62 *HDVPSS: Discrete Sync Interlaced Output Mode: Vertical Sync Output For Odd Fields May Not Correctly Detect Video Signal*

Revisions Affected: 3.0 and earlier

Details: In discrete sync interlaced output mode, the HDVPSS Video Encoder (VENC) does not correctly position the vertical sync (VSYNC) output on a half line boundary for the odd field.

Some devices cannot correctly decode fields due to the incorrectly positioned discrete vertical sync.

Workaround: There is no workaround that provides correct discrete vertical sync output alignment for interlaced formats.

To avoid this issue for interlaced video output formats, it is recommended to use embedded output syncs instead of discrete syncs.

Advisory 3.0.63 *HDVPSS VIP Single-Channel Capture Using Tiled Output Can Lead To VIP Lockup if Connect/Disconnect Events Occur*

Revisions Affected: 3.0 and earlier

Details: When the HDVPSS VIP is configured to output in tiled mode, the VPDMA assumes that all input lines are of equal length when buffering the input lines. When a connect/disconnect events occur, any line can have any length. If the last of the set of lines is either short or long compared to the previous lines, the VPDMA can enter a hung state, resulting in VIP overflow.

Workaround:

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

Use SDK v5.04.00.11 and later.

This VPDMA hung state is prevented in the HDVPSS VIP reset sequence provided in the current S/W releases.

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

The VIP_PARSER overflow bits in the VIP_PARSER_fiq_status register (bits 2 through 9) can be periodically polled by system S/W to check for an overflow condition. If overflow is detected, the VIP must be reset by following the VIP Reset Procedure described below. Note, when a reset is initiated, 2-3 frames will be lost on the channel where the event occurred. Overflow detection and VIP reset are automatically handled by current PSP drivers.

VIP Reset Procedure :

1. 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.
2. Assert VIP reset [VIPx_DP_RST] in the HDVPSS CLKC Module Reset register (offset 0x0104).
3. 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.
4. Make a list of abort descriptors for all VIP VPDMA channels.
5. Post this list and wait for it to complete.
6. De-assert VIP reset [VIPx_DP_RST] in the HDVPSS CLKC Module Reset register (offset 0x0104).
7. 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.
8. 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 3.0.66 *PCI Express (PCIe): PCIe Boot Fails When Connected to Some PCs*

Revisions Affected: 3.0 and earlier

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: There is no PCIe boot workaround that ensures operation for all PC motherboards and slots. To avoid this issue, PCIe boot from a PC slot should be avoided. Booting from SPI flash will allow S/W detection of hot-reset transactions on the PCIe bus and reconfiguration of the PCIe controller.

Advisory 3.0.70 *USB: A USB Device that Responds to a Spurious Invalid Short Packet may Lock up the Bus*

Revision(s) Affected: 3.0 and earlier

Details: The integrated USB PHY (analog transceiver) has a timing error that turns on the receiver too early and makes the USB PHY susceptible to interpreting periodic oscillations on the DP/DM lines 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.

One mass storage class USB device has been found to respond to the invalid short packet and lock up the USB bus.

The primary cause of this issue is poor signal integrity of the differential signal pair used to connect the attached USB device. Impedance discontinuities and mismatched terminations on the differential signal pair can cause reflections on the signal pair to oscillate in a periodic manner. The transceiver may detect these periodic reflections of its own transmit data as receive data.

Workaround: This issue can be avoided through proper layout of the USB signals. It is 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.

Additional protection from this issue can be gained through use of a DP/DM calibration procedure with modified squelch and receiver (RX) threshold settings which will improve USB DP/DM margins.

Advisory 3.0.71 *ROMCODE: PCIe Boot is Unstable*

Revision(s) Affected: 3.0 and earlier

Details: During the PCIe controller initialization, the ROM code locks the SERDES PLL. According to the SERDES PLL specification, there must be a 50-µs delay between enabling the LDO and locking the SERDES PLL. However, the ROM code waits only for 1 µs between enabling the LDO and locking the SERDES PLL.

This violation of the SERDES PLL specification could cause the SERDES PLL to fail to lock at extreme ends of the device operating ranges.

Workaround: Use PCIe boot only as a secondary boot. Boot from SPI flash first. The user bootloader can configure the PCIe module correctly and then initiate the next stage of the boot over to the PCIe.

Advisory 3.0.72 *ROMCODE: ROM Code Does Not Support booting from eMMC Devices of Size 4GB or More on MMC1*

Revision(s) Affected: 3.0 and earlier

Details: During eMMC initialization on MMC1, the ROM code queries the card for operating conditions by issuing CMD1 with ARG = 0. The fixed pattern response expected from the MMC card, if the card capacity is greater than 4GB in size, is 0x**40FF** 8080. As stipulated in Section 7.4.2, *Operating voltage range validation*, of the JEDEC Embedded MultiMediaCard (e·MMC) industry standard (JESD84-A441).

eMMC devices currently available in the market only adhere to Section 7.4.3, *Access mode validation (higher than 2GB of densities)*, and do not adhere to Section 7.4.2, *Operating voltage range validation*, of the JEDEC eMMC standard. When the ROM code issues a CMD1 with ARG = 0, current eMMC devices respond with a fixed pattern of 0x**00FF** 8080, even if the eMMC card capacity is of 4GB or more. Because of this, the initialization fails.

Workaround: There are two workarounds for this issue:

1. If use cases permit, use an MMC device which is less that 4GB in size.
2. If capacity of 4GB or more is needed, use eSD instead of eMMC.

Advisory 3.0.76 *UART: Extra Assertion of the FIFO Transmit DMA Request, UARTi_DMA_TX*

Revision(s) Affected: 3.0 and earlier

Details: When the UART transmit FIFO reaches 64, a tx_fifo_full signal is asserted. The UART tx_fifo_full signal is asserted asynchronously to the dma_request signal which is generated in the OCP clock domain. If a FIFO full condition is allowed, it is possible that an extra dma_request pulse can occur resulting in a TX sync error.

Workaround: This issue can be avoided by keeping the sum of the TX_THRESHOLD and TRIGGER_LEVEL ≤ 63 or 1 less than the FIFO size. If this TX_THRESHOLD sum condition is met, the transmit FIFO will not become full and the extra dma_request pulse will not occur.

TX_THRESHOLD + TRIGGER_LEVEL ≤ 63 (1 less than the FIFO size)

Table 3. UART Registers Required

UART REGISTER	REGISTER DESCRITPION	UART OFFESET ADDRESS
EFR	Enhanced Feature Register	8h
LCR	Line Control Register and Register Portal	Ch
MCR	Modem Control Register	10h
SPR/TLR	Trigger Level Register	1Ch
SCR	Supplementary Control Register	40h
MDR3	Mode Definition Register 3	80h
TXDMA	TX_DMA_THRESHOLD Register	84h

Example setup for TX_THRESHOLD + TRIGGER_LEVEL = 63:

```
Set LCR[7:0] = BFh                // Enable EFR register access
Set EFR[4] = 1                          //Set ENHANCEDEN bit to enable writing to
MCR register

Set LCR[7:0] = 00h                // Enable FCR register access
Set SCR[6] = 1                          // Set TXTRIGGRANU1 bit to set TX trigger
level granularity to 1
Set MCR[6] = 1                // Enable TLR register access
Set TLR[3:0] =0 and FCR[5:4] = 1      // Set Trigger Level to 1
Set MDR3[2] = 1                                   // Set to enable TX DMA threshold
setting
Set TXDMA[5:0] = 3Eh            // Set TX_THRESHOLD = 62
```

Advisory 3.0.79 *EDMA: TC0 and TC1 Read Accesses Always Use Physical Address*

Revision(s) Affected: 3.0 and earlier

Details: The EDMA transfer controllers (TC0 and TC1) are intended to support either physical addressing, or virtual addressing. The EDMA addressing mode is configured with the Control Module MMU_CFG register TC0MMU and TC1MMU bit fields.

Physical addressing is the default configuration and is used in most systems. This addressing uses the direct path between each TC through the L3 interconnect to the target memory or peripheral. The Physical addressing mode works as expected.

Virtual addressing is accomplished via the System MMU that is shared with the DSP. When enabled for Virtual addressing, the EDMA virtual address accesses should loop-back through the L3 interconnect through the System MMU and again through the L3 interconnect (now with Physical Address) to the target memory or peripheral. The TC0 and TC1 write accesses behave as expected and use the MMU virtual:physical translation path. However, the TC0 and TC1 read accesses do not go through the MMU and always use the Physical address; therefore, TC0 and TC1 read cannot be used with virtual addresses.

Workaround: None. Virtual addressing cannot be supported for EDMA read accesses.

Advisory 3.0.80 *Power Sequencing: CVDD versus CVDD_**

Revision(s) Affected: 3.0, 2.1 [Data Manual Revision A and earlier]

Details: The Revision A and earlier Data Manuals permit CVDD_* to ramp up before CVDD and to ramp down after CVDD. This may lead to unintended current flow and should be avoided. The unintended current flow can cause a reduction in the number of Power-On Hours (POH) for a given device by increasing its failure rate over time.

Workaround: CVDD should be powered-up coincidentally or prior to CVDD_*, and CVDD should be powered-down coincidentally or after CVDD_*, as required by the device-specific Data Manual Revision B and later.

For the actual "CVDD_*" power supply names, see the device-specific data manual.

Advisory 3.0.81 *Power Sequencing: 3.3V DVDD* versus VDDA_1P8*

Revision(s) Affected: 3.0, 2.1 [Data Manual Revision A and earlier]

Details: The Revision A and earlier Data Manuals permit the 3.3V DVDD* and VDDA_1P8 supply to ramp at the same time. This may lead to incorrect I/O bias. The incorrect I/O bias level can cause a reduction in the number of Power-On Hours (POH) for a given device by increasing its failure rate over time. This incorrect I/O bias level will remain until the DVDD_DDR[x] supply is ramped up and stable.

Workaround: Ramp the 3.3V DVDD* supplies after the VDDA_1P8 supply is ramped and stable, as required by the device-specific Data Manual Revision B and later.

For the actual "3.3-V DVDD*" power supply names, see the device-specific data manual.

Advisory 3.0.82 *Power Sequencing: DVDD_DDR[x] versus 3.3V DVDD* Supplies*

Revision(s) Affected: 3.0, 2.1 [Data Manual Revision A and earlier]

Details: The Revision A and earlier Data Manuals permit the 3.3V DVDD* supplies to be ramped prior to the DVDD_DDR[x] supply. In this case, the internal pullup/pulldown (IPU/IPD) resistor for some pins may not be turned on until the DVDD_DDR[x] supply is ramped and stable. Once the DVDD_DDR[x] supply is ramped and stable, the IPUs and IPDs will function as expected.

Workaround: Ensure that the DVDD_DDR[x] supply is ramped and stable prior to ramping the 3.3-V DVDD* supplies.

For the actual "3.3-V DVDD*" power supply names, see the device-specific data manual.

Advisory 3.0.85 *EMAC and Switch Subsystem: Reset Isolation Feature is Not Supported*

Revision(s) Affected: 3.0 and earlier

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 3.0.88 *Control Module, Pin Configuration (PINCNTLx): ROM Modifies Bit 19*

Revision(s) Affected: 3.0 and earlier

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 4 and Table 5 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 4. EMAC Boot

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

MMC

Table 5. 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.

3 Silicon Revision 2.1 Usage Notes and Known Design Exceptions to Functional Specifications

3.1 Usage Notes for Silicon Revision 2.1

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.

Silicon revision 2.1 applicable usage notes have been found on a later silicon revision. For more details, see Section 2.1, *Usage Notes for Silicon Revision 3.0*.

3.2 Silicon Revision 2.1 Known Design Exceptions to Functional Specifications

This is a list of the device revision 2.1 known design exceptions to functional specifications. Some silicon revision 2.1 applicable advisories have been found on a later silicon revision. For more details, see Section 2.2, Silicon Revision 3.0 Known Design Exceptions to Functional Specifications.

Table 6. Silicon Revision 2.1 Advisory List

Advisory 2.1.51 *HDVPSS: VIP Capture of One-Line Frame to VIP LO Port Immediately After HDVPSS Reset or VIP Client Abort Causes Lockup*

Revisions Affected: 2.1

Details: As data is captured from the HDVPSS VIP to the VPDMA, it is broken into 128-byte segments for transfer to DDR. At the end of each line a *remainder* for the line is calculated, sent as a DDR transfer, and then the address incremented by the line stride. The VPDMA VIP LO port clients incorrectly send the previous line's remainder as the size of the last transfer of a line instead of the current line. If the line captured by the VIP LO port clients is one line, it uses the last remainder calculated from the previous frame.

If a VIP LO port client abort is performed, or immediately after reset, and a one-line frame goes to the VIP LO port clients as the first or second frame, the reset value of the remainder calculation registers is sent as the burst, and the reset value is 0. The remainder registers are ping-ponged and there are two per channel. This can occur in multichannel or single-channel configurations.

In the multichannel case, any camera source that sends a one-line frame can cause this to occur.

In a single-channel configuration, the VIP LO port captures luma and chroma components separately. If either component has one line immediately after reset, this can occur. This means if the CHR_DS module is used to convert 422 data to 420, a two- or three-line captured frame can also cause this condition, because the chroma component for a two- or three-line frame is downsampled to one line.

Workaround: For single-channel cases, the HDVPSS VIP scaler is typically used and prevents single line frames from reaching the VPDMA, thus, avoiding this issue.

For multichannel cases, there is no workaround. The source creating the multichannel input must ensure that one-line frames do not occur.

Advisory 2.1.59 *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: 2.1

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 2.1.68 *DDR DMM: Continuous Writes From Cortex-A8 Occasionally Starve Other Requestors for DDR Bandwidth*

Revisions Affected: 2.1

Details:

A large number of continuous writes from the Cortex-A8 CPU can potentially consume all of the DDR bandwidth, starving all other requestors. In this state, writes from other masters may be starved, but reads are not. However, reads may be held off behind previous writes awaiting completion. These writes can be cache writebacks or CPU writes. Priority mechanisms, such as interconnect pressure and DDR priority (PEG PRIORITY in DMM), may not be usable when the situation occurs.

All the chip traffic to the DDR is routed via the Dynamic Memory Manager (DMM) and then the External Memory Interface (EMIF). The Cortex-A8 has a dedicated low-latency port (ELLA) to the DMM. All other requestors route the requests via the chip L3 interconnect to the two LISA ports of the DMM. In order to ensure lower latency for Cortex-A8 requests, the DMM always prioritizes Cortex-A8 requests over other requests and, other than prioritizing the Cortex-A8 ELLA port over LISA ports, no other prioritization is performed in the DMM. Writes and reads are prioritized independently, but no reordering of requests is performed in the DMM. The DMM also tags the requests to EMIF with the PEG PRIORITY. Requests in the EMIF may be prioritized as per the EMIF prioritization rules which include PEG PRIORITY [for more details, see the DMM and EMIF sections in the *TMS320DM814x DaVinci Video Processors Technical Reference Manual* (literature number SPRUGZ8)]. In the error condition, the EMIF may be presented by the DMM with only the Cortex-A8 requests, starving all other requestors.

The issue may manifest as loss of bandwidth seen by requestors such as DSP or EDMA. Real-time requestors, such as HDVPSS, may sometimes lose real-time deadlines causing artifacts on the display, such as rolling.

Workaround:

There is no single workaround for the issue. However, specific settings of the Cortex-A8 cache may help avoid this behavior in many systems:

1. Disabling the write allocate cache policy (bit 22) in the Cortex-A8 C9, L2 cache auxiliary control register is seen to change the observed behavior.
2. Setting the write allocate delay disable (bit 24) to disable (1) in the Cortex-A8 C9, L2 cache auxiliary control register has been observed to change the observed behavior.

The workarounds may help by minimizing the chances of a long series of writes from the Cortex-A8.

Note: the L2 cache auxiliary control register cannot be entered by application/OS code; this must be programmed via ROM functions invoked via SMI. The following pseudo code is an example of how to program the L2 cache auxiliary control register:

```
l2_disable_wa:
        stmfd   sp!, {r0 - r12, lr}       @ Store registers r0 -
> r12 on the stack, ROM does not preserve values
        mrc     p15, 1, r0, c9, c0, 2     @ Read the L2 cache auxiliary control
register
        orr     r0, r0, #(1 << 22)        @ OR in the bit to disable write
allocate (bit #22)
        mov     r12, #0
        orr     r12, r12, #(1 << 8)
        add     r12, r12, #2              @ Program ROM Entry point into R12
(0x102 for L2 cache aux ctrl register)
                                          @ R0 contains the value the ROM will
write into the L2 cache aux ctrl register
        .word   0xe1600070                @ opcode of SMC/SMI instruction -
 jump into the ROM
        ldmfd   sp!, {r0 -
 r12, pc}       @ Restore registers from stack that may have been overwritten by
ROM code.
```

Advisory 2.1.87 *Control Module, Pin Configuration (PINCNTLx), 3.3 V Mode Operation: Reduction in Power-On Hours May Occur if the Input Receiver is Disabled*

Revision(s) Affected: 2.1

Details:

Bit 18 within each PINCNTL[270:1] register, currently documented as RESERVED, serves as an active high enable for the input receiver of the corresponding pin. In order for the pin to function as an input or as bidirectional, bit 18 of the corresponding PINCNTLx register ***must*** be set to "1". This bit defaults to "1" after device reset.

Clearing bit 18 to "0" can cause a reduction in the number of Power-On Hours for a given device, if the I/O buffer is operating in 3.3 V mode. PINCNTLx bit 18 should not be modified by any software during device operation.

A reduction in the number of Power-On Hours will ***not*** occur when the I/O is operating in 1.8 V mode.

The ROM bootloader initialization incorrectly clears bit 18 to "0" for some of the PINCNTLx registers. Table 7 through Table 13 identify the PINCNTLx registers, per boot mode, affected by the ROM bootloader incorrectly clearing PINCNTLx bit 18 to "0".

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 7. EMAC Boot

BOOT MODE OPTIONS (BTMODE[9:8] PINS)	DESCRIPTION	PINCNTLx REGISTER
00b	MII	233, 249-258
01b	RMII	233, 240-242
10b	RGMII	233, 238-240, 245-247

FAST XIP (MUX0)

Table 8. FAST XIP (MUX0) Boot(1)

BOOT MODE OPTIONS (BTMODE[14:13] PINS)	DESCRIPTION	PINCNTLx REGISTER
00b	Not muxed	122, 128-132, 243-255
01b	A/A/D muxed	122, 128-132
10b	A/D muxed	

(1) BTMODE[15] (GPMC_Wait enable), BTMODE[12] (GPMC CS0 data bus width), and BTMODE[10] (GPMC configuration) inputs do not affect how the ROM bootloader configures bit 18 of the PINCNTLx register for the Fast XIP boot method.

NAND (All NAND and NANDI2C)

Table 9. NAND Boot

BOOT MODE OPTIONS	DESCRIPTION	PINCNTLx REGISTER
BTMODE[10] = xb BTMODE[12] = xb	All NAND / NANDI2C configurations	122, 128-132

PCIe

Table 10. PCIe Boot

BOOT MODE OPTIONS	DESCRIPTION	PINCNTLx REGISTER
–	PCIE32 / PCIE64	105-108, 220

UART

Table 11. UART Boot

BOOT MODE OPTIONS	DESCRIPTION	PINCNTLx REGISTER
–	–	71

XIP (MUX0)

Table 12. XIP (MUX0) Boot

GPMC CONFIGURATION (BTMODE[10] PIN)	BOOT MODE OPTIONS (BTMODE PINS)	DESCRIPTION	PINCNTLx REGISTER
BTMODE[10] = 0b (GPMC Option A)	BTMODE[14:13] = 00b	Not muxed	122, 128-132, 243-255
	BTMODE[14:13] = 01b	A/A/D muxed	122, 128-132
	BTMODE[14:13] = 10b	A/D muxed	
BTMODE[10] = 1b (GPMC Option B)	BTMODE[14:12] = 000b	Not muxed, 8-bit bus	122,129,130,132, 243-255
	BTMODE[14:12] = 001b	Not muxed, 16-bit bus	122,129,130,132, 244-255
	BTMODE[14:13] = 01b	A/A/D muxed	122,128-130,132
	BTMODE[14:13] = 10b	A/D muxed	

XIP (MUX1)

Table 13. XIP (MUX1) Boot

GPMC CONFIGURATION (BTMODE[10] PIN)	BOOT MODE OPTIONS (BTMODE PINS)	DESCRIPTION	PINCNTLx REGISTER
BTMODE[10] = 0b (GPMC Option A)	BTMODE[14:13] = 00b	Not muxed	117-120, 122, 128-132, 168-175, 231
	BTMODE[14:13] = 01b	A/A/D muxed	122, 128-132
	BTMODE[14:13] = 10b	A/D muxed	
BTMODE[10] = 1b (GPMC Option B)	BTMODE[14:12] = 000b	Not muxed, 8-bit bus	117-120, 122, 129, 130, 132, 168-175, 231
	BTMODE[14:12] = 001b	Not muxed, 16-bit bus	117-120, 122, 129, 130, 132, 168-175
	BTMODE[14:13] = 01b	A/A/D muxed	122,128-130,132
	BTMODE[14:13] = 10b	A/D muxed	

Note: In addition to the ROM bootloader software, the following TI software packages may potentially clear bit 18 of the PINCNTLx registers. If any of the following TI software packages are used, care should be taken to properly set bit 18 to "1" for all PINCNTLx registers during device operation:

- U-BOOT
- LINUX Kernel
- HDVPSS Firmware

Workaround: To prevent a reduction in the number of Power-On Hours and enable the corresponding pins to be used as inputs or bidirectional functions as desired by the application, software should set bit 18 for all PINCNTLx registers to "1" after the initial boot is performed by the ROM bootloader. Software should also ensure that bit 18 for all PINCNTLx registers is not modified during device operation.

A reduction in the number of Power-On Hours occurs ***only*** when the I/O buffer is operating in 3.3 V mode and ***does not*** occur when the I/O buffer is operating in 1.8 V mode.

This advisory will be fixed in a future *major* silicon revision and the device-specific documentation updated to support the PINCNTLx register bit 18 as RXACTIVE versus RESERVED.

Revision History

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

This silicon errata revision history highlights the technical changes made to the SPRZ385 revision to make it an SPRZ385A revision, as well as the technical changes made to the SPRZ385A revision to make it an SPRZ385B revision and includes the technical changes to the SPRZ385B revision to make it an SPRZ385C external release.

Scope: Applicable updates relating to the TMS320DM812x devices have been incorporated.

DM812x Revision History

SEE	ADDITIONS/MODIFICATIONS/DELETIONS
Revision C Changes below:	
Global	Updated/Changed "Digital Media" to "Video" in doc title. Updated/Changed doc for external release including die revision code example and silicon revision 3.0 part numbers.
Revision B (Silicon Revision 3.0) Changes below:	
Global	Added Silicon Revision 3.0 device-specific data
Section 1.2	Package Symbolization and Revision Identification: • Updated/Changed “Figure 1 shows …” paragraph
Figure 1	Example, Device Revision Codes: • Updated figure to show an example of silicon revision 3.0 (“S”)
Table 1	TMS320DM8127 Device Revision Codes: • Added Silicon Revision 3.0 ("C") device-specific data
Section 2	Silicon Revision 3.0 Usage Notes and Known Design Exceptions to Functional Specifications: Added *new* section
Section 2.1	Usage Notes for Silicon Revision 3.0: Added *new* section Moved the following 2.1 Usage Notes to this *new* section: • Section 2.1.1, DDR3: JEDEC Compliance for Maximum Self-Refresh Command Limit • Section 2.1.2, Some PLLs Only Support Even M2 Post Dividers • Section 2.1.3, DDR2 and DDR3 Requires Software Leveling
Section 2.2	Silicon Revision 3.0 Known Design Exceptions to Functional Specifications: Added *new* section Moved **all other** 2.1 Advisories to this section **except** for these: • • Advisory 2.1.51,HDVPSS: VIP Capture of One-Line Frame to VIP LO Port Immediately After HDVPSS Reset or VIP Client Abort Causes Lockup • Advisory 2.1.59, HDVPSS: Occasionally, Chip Lockup Occurs When HDVPSS VIP is Performing Single-Channel Capture, Sending Tiled Data to DDR and Connect/Disconnect Events are Occurring • Advisory 2.1.68,DDR DMM: Continuous Writes From Cortex-A8 Occasionally Starve Other Requestors for DDR Bandwidth • Advisory 2.1.87, Control Module, Pin Configuration (PINCNTLx), 3.3 V Mode Operation: Reduction in Power-On Hours May Occur if the Input Receiver is Disabled
Section 3.1	Usage Notes for Silicon Revision 2.1: • Updated/Changed “Silicon revision 2.1 applicable usage notes …” paragraph
Section 3.2	Silicon Revision 2.1 Known Design Exceptions to Functional Specifications: • Updated/Changed “This is a list of the device revision 2.1 known …” paragraph • Added Advisory 2.1.59, HDVPSS: Occasionally, Chip Lockup Occurs When HDVPSS VIP is Performing Single-Channel Capture, Sending Tiled Data to DDR and Connect/Disconnect Events are Occurring
Revision A (Silicon Revision 2.1) Changes below:	

DM812x Revision History (continued)

SEE	ADDITIONS/MODIFICATIONS/DELETIONS
Global	Updated/Changed "DEMMU" to "System MMU" Updated/Changed "Digital Media Processors" to "Video Processors"
Section 3.2	*Silicon Revision 2.1 Known Design Exceptions to Functional Specifications:* Advisory 2.1.25, System DEMMU May Hang When Used in Table-Walk Mode: • Updated/Changed the System MMU Entries (Entry 1 through Entry 7) code in the System MMU Lockdown Example. Added the following Advisories to this section: • Advisory 2.1.51, *HDVPSS: VIP Capture of One-Line Frame to VIP LO Port Immediately After HDVPSS Reset or VIP Client Abort Causes Lockup* • Advisory 2.1.54, *VPDMA Line Limit Feature: Descriptor Reports All Fields as Even, but Captures 30 Even and 30 Odd Fields in Memory* • Advisory 2.1.55, *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* • Advisory 2.1.58, *HDVPSS: HDVPSS VIP Reset Sequence is Occasionally Unsuccessful if VPDMA is Writing Output Descriptors for VIP Captured Data* • Advisory 2.1.60, *HDVPSS: Occasionally, During Connect/Disconnect and When Line or Width Limit Feature Not Used, Any Memory Areas Can Be Overwritten* • Advisory 2.1.62, *HDVPSS: Discrete Sync Interlaced Output Mode: Vertical Sync Output For Odd Fields May Not Correctly Detect Video Signal* • Advisory 2.1.63, *HDVPSS: VIP Single-Channel Capture Using Tiled Output Can Lead To VIP Lockup if Connect/Disconnect Events Occur* • Advisory 2.1.66, *PCI Express (PCIe): PCIe Boot Fails When Connected to Some PCs* • Advisory 2.1.68, *DDR DMM: Continuous Writes From Cortex-A8 Occasionally Starve Other Requestors for DDR Bandwidth* • Advisory 2.1.70, *USB: A USB Device that Responds to a Spurious Invalid Short Packet may Lock up the Bus* • Advisory 2.1.71, *ROMCODE: PCIe Boot is Unstable* • Advisory 2.1.72, *ROMCODE: ROM Code Does Not Support Booting from eMMC Devices of Size 4GB or More* • Advisory 2.1.76, *UART: Extra Assertion of the FIFO Transmit DMA Request, UARTi_DMA_TX* • Advisory 2.1.79, *EDMA: TC0 and TC1 Read Accesses Always Use Physical Address* • Advisory 2.1.80, *Power Sequencing: CVDD versus CVDD_** • Advisory 2.1.81, *Power Sequencing: 3.3V DVDD* versus VDDA_1P8* • Advisory 2.1.82, *Power Sequencing: DVDD_DDR[x] versus 3.3V DVDD* Supplies* • Advisory 2.1.85, *EMAC and Switch Subsystem: Reset Isolation Feature is Not Supported* • Advisory 2.1.87, *Control Module, Pin Configuration (PINCNTLx), 3.3 V Mode Operation: Reduction in Power-On Hours May Occur if the Input Receiver is Disabled* • Advisory 2.1.88, *Control Module, Pin Configuration (PINCNTLx): ROM Modifies Bit 19*

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