OMAP3530/25/15/03 Applications Processor
Silicon Revisions 3.1.2, 3.1, 3.0, 2.1, and 2.0

Silicon Errata

Literature Number: SPRZ278F
February 2008–Revised October 2010

OMAP3530/25/15/03 Applications Processor
Silicon Revision 3.1.2

1 Introduction

This document describes the known exceptions to the functional specifications for the OMAP3530, OMAP3525, OMAP3515, and OMAP3503 applications processors. [See the *OMAP3530/25 Applications Processor Data Manual* (literature number SPRS507) and the *OMAP3515/03 Applications Processor Data Manual* (literature number SPRS505)].

For additional information, see the latest version of the *OMAP35x Technical Reference Manual* (literature number SPRUF98).

The advisory numbers in the document are not sequential. Some advisory numbers have been moved to the next revision. When items are moved, the remaining advisory numbers 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 sheet), and the behaviors they describe will not be altered in future silicon revisions.

1.1 OMAP35x Device and Development Support Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all OMAP processors and support tools. Each commercial OMAP platform member 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	Experimental device that is not necessarily representative of the final device's electrical specifications
P	Final silicon die that conforms to the device's electrical specifications but has not completed quality and reliability verification
NULL	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

X and P 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 (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 Revision Identification

The device revision can be determined by the symbols marked on the top of the package. Figure 1 provides an example of the OMAP35x Applications Processor device markings.

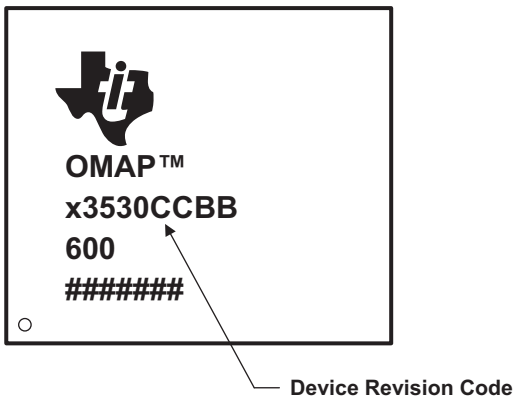


Figure 1. Example, Device Revision Codes for OMAP35x Applications Processor

NOTES:

(A) Non-qualified devices are marked with the letters "X" or "P" at the beginning of the device name, while qualified devices have a "blank" at the beginning of the device name.

(B) "#" denotes an alphanumeric character.

(C) On some "TMX" devices, the device speed may not be shown.

Silicon revision is identified by a code marked on the package. The code is of the format X3530xCBB, where "x" denotes the silicon revision. On TMX devices, if x is"C", then the silicon is revision 3.0. Table 1 and Table 2 list the information associated with each silicon revision on TMX devices.

Table 1. OMAP35x Applications Processor Revision Codes

DEVICE REVISION CODE	SILICON REVISION	COMMENTS
E	3.1.2	Silicon revision 3.1.2 (also referred to as ES3.1.2)
D	3.1	Silicon revision 3.1 (also referred to as ES3.1)
C	3.0	Silicon revision 3.0 (also referred to as ES3.0)
B	2.1	Silicon revision 2.1 (also referred to as ES2.1)

Each silicon revision uses a specific revision of the C64x+ CPU, the C64x+ Megamodule, as well as the ARM Cortex-A8 processor. Table 2 lists the C64x+ CPU and C64x+ Megamodule revision associated with each silicon revision. The C64x+ CPU revision can be read from the REVISION_ID field of the CPU Control Status Register (CSR). The C64x+ Megamodule revision can be read from the REVISION field of the Megamodule Revision ID register (MM_REVID) located at address IVA2.2 subsystem memory address 0181 2000h. The ARM Cortex-A8 variant and revision can be read from the Main ID Register.

The ROM code revision can be read from base address 4001 BFFCh. The ROM code version consists of two decimal numbers: major and minor. The major number is always 14, minor number counts ROM code version. The ROM code version is coded as hexadecimal readable values, e.g. ROM version 14.04 will be coded as 0000 1404h. Table 2 shows the ROM code revision for each silicon revision of the device.

Table 2. Silicon Revision Variables

SILICON REVISION	C64X+ CPU REVISION	C64X+ MEGAMODULE REVISION	ARM CORTEX-A8 VARIANT/REVISION	ROM REVISION
3.1.2	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p7	14.04
3.1	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p3	14.04
3.0	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p3	14.04
2.1	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p2	14.04
2.0	CPU_ID = 10h REVISION_ID = 02h	MM_REVID[REVISION] = 0h	r1p1	14.04

2 Silicon Revision 3.1.2 Usage Notes and Known Design Exceptions to Functional Specifications

2.1 Usage Notes for Silicon Revision 3.1.2

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 silicon revisions.

Note: The peripherals supported on the various OMAP35x Application Processors are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Application Processors, see the device-specific data manuals.

2.1.1 Cortex-A8 Errata List

The *OMAP3530/25/15/03 Applications Processor Silicon Revisions 3.1.2, 3.1, 3.0, 2.1, and 2.0 Silicon Errata* does not cover the advisories associated with the Cortex-A8 processor. For a list of the advisories associated with each version of the Cortex-A8 processor, contact your TI representative for a copy of the *ARM Core Cortex-A8 (AT400/AT401) Errata Notice*. See Table 2 to determine which version of the Cortex-A8 processor is included in each OMAP35x silicon revision.

2.1.2 Performance Limitation On LCD Read/Write Access Through RFBI L4 Port

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, back-to-back accesses for both *Read* and *Write* to the LCD through the L4 interconnect interface of the RFBI module are not supported. A penalty of 1 L4 clock cycle between two consecutive accesses exists.

Read access to the LCD through the RFBI L4 port: The data of a *Read* access is sent back to the initiator of the access only at RECycleTime. RECycleTime is used as a reference event for CS release as well (CS is used as an on-going access notification signal depending on the type of LCD panel connected). This means that any *Read* access to the LCD through the L4 interface of the RFBI will be ended by a CS release (CS going inactive at the end of the access). Therefore, the OMAP35x Applications Processor does not support two consecutive *Read* accesses to the LCD (there is always 1 L4 clock cycle between 2 accesses).

Write access to the LCD through the RFBI L4 port: For a *Write* access, the back-to-back data of a single access that has been split is supported and guaranteed (e.g., one 32-bit write split in two consecutive, back-to-back, 16-bit accesses). In this case, the internal bus CS signal will be kept active until split access completion. However, when there are two different write accesses from the initiator, they will not be seen as back-to-back by the LCD. At the end of the transaction corresponding to one write

access from the initiator, the internal bus CS will be released at WECycleTime, and the next command from the initiator will be accepted. Consequently, any *Write* access to the LCD through the L4 interface of the RFBI will be ended by a CS release (CS going inactive at the end of the access). Therefore, the OMAP35x Applications Processor does not support two consecutive data transmits to the LCD (corresponding to two different and consecutive *Write* accesses from the initiator).

(Internal reference number: 2.1)

2.1.3 IVA2: IDLE Instruction Must be Executed From L1P or L2 SRAM Under Hardware Emulation (OMAP3530/25 only)

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, when the C64x core executes an IDLE instruction, and an emulator access to a register or memory happens simultaneously, the CPU may hang, depending on the relative speed of TCK, GEM clock, and the external memory where the IDLE instruction is fetched. This limitation is ***only*** applicable in *emulation* mode when an external emulator, such as TI XDS560, is connected to the device.

To ensure this race condition does not occur, the IDLE instruction should be executed from L1P SRAM or L2SRAM. This will allow the IDLE instruction to execute safely under hardware emulation.

(Internal reference number: 2.4)

2.1.4 Domain Woken Up by Wake-up Dependency Cannot Transition to Inactive State

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, when a domain is activated up by a wake-up dependency, it cannot transition to an inactive (RET/OFF) state until one of the following conditions are satisfied:

1. If the domain has a sleep dependency with another domain, this domain and the sleep dependency must be also activated (hard-wired or enabled by the software).
2. If the domain is an initiator, it must be taken out of *Standby* mode.

Note: The user should ensure that when a wake-up dependency is enabled for a domain, it is expected to be fully active after the wake-up occurs. This issue happens in a case where a domain has been woken-up, but not activated and used.

(Internal reference number: 2.7)

2.1.5 VENC: Last Data Line Missing in PAL

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, the VENC cannot show the image data at the end of the even frames on line 623. This is a minor violation of the ITU-R BT.470-6. Since lines 23 and 336 are reserved for WSS, the VENC can only show 574 lines of data instead of the 576 lines defined in the standard. One half line is missing for lines 23 (reserved for WSS) and 623, and one full line is missing for line 336 (reserved for WSS).

(Internal reference number: 2.8)

2.1.6 Observability Signals Not Functional in OFF Mode

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, signals are routed to observability pins through a buffer powered by VDD2. Consequently, observability is not functional when VDD2 is powered off. This impacts the observability debug feature, but has no functional drawback.

Note: Maintaining the VDD2 supply on the board has no effect since isolation cells are activated regardless of the supplied voltage.

(Internal reference number: 2.10)

2.1.7 Constraints on Module Clocks When Using DVFS

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, module interface timings can depend on the operating point (OPP). The timing closure is done for one instance of a module at a particular voltage. Operating conditions which work on one OPP are not automatically guaranteed on another module's OPP. For more details on the module interface switching characteristics, see the OMAP35x Applications Processor device-specific data manual.

When Dynamic Voltage and Frequency Scaling (DVFS) is performed, the software should ensure that no timing violations will occur by the two following methods:

- Using software, reconfigure the module accordingly when DVFS is performed to avoid timing violations.
- Use a conservative frequency on the module interface clock corresponding to the performance of the lower OPP used to ensure that DVFS can be performed without additional software management.

(Internal reference number: 2.11)

2.1.8 Retention Voltage Not Supported

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, the retention voltage ***is not*** supported. The TI 65nm process defines a retention voltage which is lower than the lowest OPP voltage. This retention voltage is defined assuming that there is no logic activity at all in the device, which does not apply to the OMAP35x Applications Processor where asynchronous signals (wake-up) can be triggered during the device RET state. Because of this system requirement, the retention voltage does not apply to the OMAP35x Applications Processor, and the lowest OPP voltage should be used instead of a specific retention voltage.

The lowest operating point voltage should be used when the OMAP35x Applications Processor is in *retention* mode.

Note: This voltage can be optimized using SmartReflex before the device goes into *retention* mode.

(Internal reference number: 2.14)

2.1.9 UART: Cannot Acknowledge Idle Requests in Smartidle Mode When Configured for DMA Operations

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, when configured for DMA operations using *smartidle* mode (SYSC[4:3].IDLEMODE = 0x2), the UART module will not acknowledge incoming idle requests. As a consequence, it can prevent L4 from going to idle.

When there are additional expected transfers, the UART should be placed in *force-idle* mode.

(Internal reference number: 2.15)

2.1.10 Unexpected RFBI Latency for High Frame Rate

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, the Remote Frame Buffer (RFBI) state machine architecture can create non-optimized pipelining of the RFBI output data stream. As a result, some idle cycles may be inserted between RFBI accesses to the external panels. The number of idle cycles added depends on the OMAP clock configuration and RFBI configuration. Table 3 shows the minimum additional number of cycles depending on the RFBI configurations.

Table 3. Additional Minimum Cycletime (CS/WE Always Asserted)

RFBI PERFORMANCE	RFBI_CONFIG. CYCLEFORMAT	RFBI_CONFIG. OCPFORMAT	MIN CYCLETIME (Number of OCP cycles)
OCP Slave	1 pixel/ cycle	1 pixel	5
	1 pixel/ 2 cycles	1 pixel	4
	1 pixel/ 3 cycles	1 pixel	4
	2 pixels/ 3 cycles	1 pixel	6
	1 pixel/ cycle	2 pixels	4
	1 pixel/ 2 cycles	2 pixels	4
	1 pixel/ 3 cycles	2 pixels	4
	2 pixels/ 3 cycles	2 pixels	6
Display Controller	1 pixel/ cycle	N/A	4
	1 pixel/ 2 cycles	N/A	3
	1 pixel/ 3 cycles	N/A	3
	1 pixel/ 3 cycles	N/A	6

(Internal reference number: 2.20)

2.1.11 GPIO Drives Random Values When Device Comes Back from OFF Mode

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, the GPIO modules in the PER (Peripheral) domain drives random values when the device comes back from OFF mode with VDD2 voltage shut down.

Regardless of the active configuration of a GPIO (input or output), when coming back from OFF mode, GPIO can randomly drive the line high or low.

Root cause: The isolation at the boundary of the peripheral domain is done through isolation latches which are supplied by VDD2. When VDD2 is shut down, the content of the latches is lost. When the device is waking-up from the OFF mode, the padconf is automatically restored by the Hardware and then the isolation latches drives the pin until isolation is released (PER is woken-up). There is no reset value for the isolation latches thus leading to random value driven by the GPIO.

This does not impact GPIO from the WU domain.

Only GPIO implements this type of isolation latches and hence other functions are not impacted by this limitation.

Workaround 1 allows the software to work in SW supervised mode while Workaround 2 takes the benefit of HW dependency. For both the workarounds, SW is in charge to configure the internal pull in the padconf register to the desired value.

Workaround 1: Change padconf mode to safe mode before initiating the transition to OFF. This would avoid line being driven by isolation latch and the desired level is maintained by the pull.

Workaround 2: Create a WU dependency between WU domain and PER domain (PM_WKDEP_PER[4]: EN_WKUP=1). This will WU the PER domain then release the isolation latch before the isolation cell is released at pad boundary. The reset value of the GPIO module configures the GPIO in input and then the pull ensures the that line is driven to the right value until GPIO module is restored by the applicative SW.

(Internal reference number: 2.23)

2.1.12 Access to IVA Boot ROM at OPP1 Is no Longer Supported

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, accesses to IVA ROM code at low temperature and low voltage (VDD1; OPP1) cannot be guaranteed across the process. In case a failing access happens, the system behavior is unpredictable. Consequently, no access to IVA internal boot ROM at OPP1 is allowed.

Do not access IVA ROM or boot the IVA at any OPP except OPP1.

(Internal reference number: 2.24)

2.1.13 Maximum 12 bits Output (1 bit sign + 11 bits of value) Is Supported on CAVLD of iVLCD

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, while decoding DC coeffs of an Intra16x16 MB, the iVLCD is not able to handle coeff with range greater than 11 bits. It is observed that the value returned by the iVLCD will be the equivalent of (actual_value & 0x07FF). For e.g., value of 0x08A7 will be decoded as 0x00A7. However, the bits consumed by the iVLCD is correct and the decoding continues.

As per the standard, normal imagery can generate DC coeffs with range greater than 11 bits (especially at low QP) and some of the test cases are failing due to this issue. Since the error is in DC it will not only affect the MB in question, but propagate to neighboring MBs in the slice. The impact is visually noticeable.

This issue occurs very rarely and only if QP=1.

This special case must be handled with C64x+ software which increases code size and add extra MHz.

(Internal reference number: 2.25)

2.1.14 Maximum Pixel Rate on Resizer With A-Law Decompression and Horizontal Zoom Greater than 2

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, failure conditions: SDRAM->PRV->RSZ->SDRAM path used with A-Law decompression cannot be used with Horizontal zoom greater than 2. RSZ can only accept pixel rate at up to OCP freq/(2*hzoom). The Maximum horizontal zoom hzoom=4. However, the PRV read rate can only be slowed down to OCP freq/4.

Perform memory -> RSZ -> memory upscaling when Horizontal zoom greater than 2. Don't use PRV-> RSZ path when A-Law decompression is used in PRV.

(Internal reference number: 2.28)

2.1.15 Extra Power Consumption at vdds_mmc1/vdds_sim in Device Off Mode

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, IO cells which are supplied by vdds_mmc1 and vdds_sim are dual voltage buffer. CONTROL_PBIAS_LITE.PBIASLITEPWRDNx bit controls enable/disable of these PBIAS cell. When PBIASLITEPWRDNx=1 (PBIAS is enabled), additional ~30uA at 1.8V or ~300uA at 3.0V power is consumed by the cell. This additional power consumption is negligible while the device is active, but it can be significant during device off mode.

To reduce the power consumption, the SW should do the following configuration before the OFF mode entry.

- Disable PBIAS cell (PBIASLITEPWRDNZx bit = 0).
- + If there is no external pulls or the line is not driven by the external IC, enable the pull down and ensure that the PAD is in input mode by setting the muxmode in safe_mode or in gpio(input).

This workaround cannot be applied for the cases the PAD needs to maintain high level during device off mode. Neither an internal pull-up nor high level driving works while PBIAS is disabled.

(Internal reference number: 2.29)

2.1.16 Extra Power Consumed When Repeated Start Operation Mode Is Enabled on I2C Interface Dedicated for Smart Reflex (I2C4)

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, on the I2C interface dedicated for Smart Reflex communication with the Power Management IC (I2C4), when Repeated Start operation mode is enabled (PRM_VC_I2C_CFG[SREN]='1'), the I2C lines (SCL and SDA) are always driving a low state between two I2C commands (i.e. when there is no I2C traffic).

Knowing that there are external Pull-Up attached on these lines, this will impact the power consumption.

Setting PRM_VC_I2C_CFG[SREN] register bit to '0' will allow the I2C4 lines driving a high state between two I2C commands.

(Internal reference number: 2.30)

2.1.17 OMAP HS Devices Are Not Recovering From Warm Reset While in OFF Mode

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, a deadlock situation occurs if warm reset occurs during OFF mode or after WU from OFF but before ROM code execution is completed on HS device.

Root cause: It was not captured as a ROM code requirement to support a Warm reset during OFF mode.

Failure mechanisms:

1. If warm reset occurs before SSM restoration, then SSM generates secure violation (resulting in warm reset assertion) as soon as ARM is accessing other location than secure ROM. (This always occurs if warm reset happens during OFF mode).
2. If warm reset is asserted after SSM restoration but before ROM code completion then some part of the restoration normally performed by ROM code is missing which can result in unpredictable behavior (lockup or crash happening later on).

Condition of occurrence: The only source of warm reset during OFF mode are: MPU WD, Secure WD and external warm reset: sys_nreswarm pin. With accurate sw configuration, and no security attack, MPU and secure watchdog are not expected to expire during OFF mode (either WDs are disabled before system transitioning to OFF mode or a timer will WU system before to reload WD before expiration). The only remaining relevant warm reset source during OFF mode is sys_nreswarm.

No workaround needs to be implemented if sys_nreswarm is used as OMAP output only; this issue should never happen.

If sys_nreswarm is used as an OMAP input and is potentially asserted while OMAP is in OFF mode then the only way to recover is to generate a power on reset. This can be programmed in the warm_reset sequence if TWL4030/5030 PMIC is used.

(Internal reference number: 2.31)

2.1.18 DPLL3 Recall and Long Relock Time

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, temperature drift is impacting DPLL3 relock time. High temperature drift (positive or negative delta t > 20C) can extend relock time from 40 up to 150 clock cycles. This could cause issues in low power scenarios when the DPLL3 is idle most of the time and needs a fast wakeup to meet application timing.

The issue is fixed by disabling DPLL3 automatic control (bypass or stop mode). Automatic control should be enabled again before doing a transition to retention or off modes.

(Internal reference number: 2.32)

2.1.19 Transfer of Multiple Command Packets Coming from L4 Interconnect During a Blanking Period in Interleaving Mode

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, in video mode, the command mode packets, provided through the DSI protocol engine OCP port, can be interleaving during the vertical and/or horizontal blanking periods of the video stream sequence.

When TxFifo on the OCP slave port is empty, and if the 1st packet written to TxFifo is less than 13 words when 1 data lane is active or 17 words when 2 data lanes are active, only this packet will be sent out on the HS link during the next blanking period enabled for command packet transfer. This is the only sent packet, because this packet is the only completely written packet at the time of FSM read the last location of this packet from TxFifo. Even if more packets are written in TxFifo before the interleaving starts these packets will not be sent during that blanking period.

No workaround is available. The impact is minor because:

- When interleaving is done on a vertical blanking period (VSA, VFP, VBP), since this blanking are expressed in a number of lines, the remaining packet(s) in TxFifo will be sent on the HS link during the next line blanking interval within the same blanking period or during the next one.
- When the interleaving is done on a horizontal blanking period (HSA, HFP, HBP), the remaining data in TxFifo will be sent on the next blanking period.

(Internal reference number: 2.33)

2.1.20 Downscaling limitations

On OMAP35x Applications Processor silicon revisions 3.1.2 and earlier, Synclost interrupts are observed when performing downscaling and shifting resulting windows on the right of the LCD panel. This interrupt occurs because there is not sufficient time to load the necessary lines buffer before performing downscaling operation. DISPC video pipeline is expected to be ready with the pixels at the end of HSW+HBP period. This limitation is applicable on 3 and 5 taps configurations.

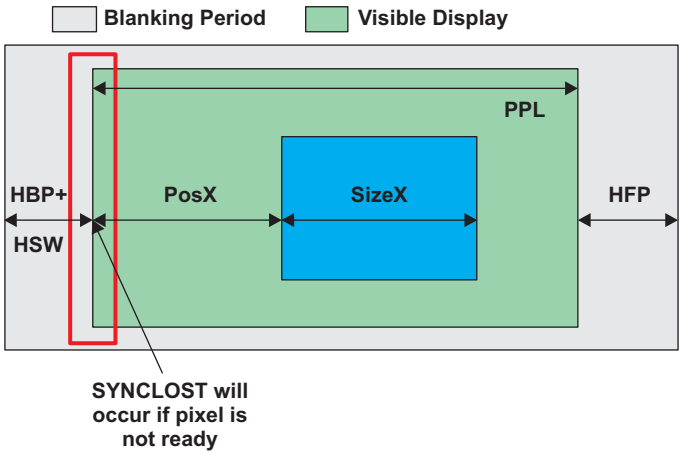


Figure 2. Synclost Interrupt

To ensure proper downscaling operation, these conditions should be met:

- ((PPL-SizeX-PosX) + (HBP+HSW+HFP)) * PCD >= Max(0,Ceil(DS)-2)*(OrgSizeX+1)
- ((PPL-SizeX) + (HBP+HSW+HFP)) * PCD >= Max(0,Ceil(DS)-1)*(OrgSizeX+1)

In case VRFB 90/270 rotation with YUV or RGB16 format, these conditions should be met:

- ((PPL-SizeX-PosX) + (HBP+HSW+HFP)) * PCD >= Max(0,Ceil(DS)-1)*(OrgSizeX+1)
- ((PPL-SizeX) + (HBP+HSW+HFP)) * PCD >= (Max(0,Ceil(DS)-1)*(OrgSizeX+1))* 2

Where:

DS = (OrgSizeY+1)/(SizeY+1)

Ceil(x) is a round-up of x

(Internal reference number: 2.34)

2.2 Silicon Revision 3.1.2 Known Design Exceptions to Functional Specifications

Table 4. Silicon Revision 3.1.2 Advisory List

Table 4. Silicon Revision 3.1.2 Advisory List (continued)

Table 4. Silicon Revision 3.1.2 Advisory List (continued)

Advisory 3.1.1.2 *I2C Module Does Not Allow 0-Byte Data Requests*

Revision(s) Affected	3.1.2 and earlier
Details	When configured as the master, the I2C module does not allow 0-byte data transfers. **Note**: Programming I2Ci.I2C_CNT[15:0]: DCOUNT = 0 will cause undefined behavior.
Workaround(s)	No workaround. ***Do not*** use 0-byte data requests.

Advisory 3.1.1.4 *Delay Required to Read Some GP, WD, and Sync Timer Registers After Wake-Up*

Revision(s) Affected

3.1.2 and earlier

Details

If a General Purpose Timer (GPTimer) is in *posted* mode (TSIRC.POSTED = 1), due to internal resynchronizations, any values read in TCRR, TCAR1, and TCAR2 registers immediately after the timer interface clock (L4) goes from a stopped state to an active state may not return the expected values. This situation is most likely when the OMAP35x Applications Processor wakes up from an idle state.

Notes:

- GPTimer non-posted synchronization mode is not impacted by this advisory.
- This advisory also impacts reads from Watchdog timers WCRR registers.
- All of the watchdog timers support only *posted* internal synchronization mode. There is no capability to change the internal synchronization scheme to *non-posted* mode via software.
- The 32K sync timer CR and 32SYNCNT_REV registers are also impacted by this advisory, since the 32K sync timer is always in *posted* synchronization mode.

Workaround(s)

The software must wait at least 2 timer interface clock cycles + 1 timer functional clock cycle after L4 clock-wakeup before reading TCRR, TCAR1, or TCAR2 registers for GP Timers in *posted* internal synchronization mode, and before reading the WCRR register of the Watchdog timers. The same workaround must be applied before reading CR and 32KSYNCNT_REV registers of the synctimer module.

Advisory 3.1.1.9 *L1D Cache : C64x+ L1D Cache May Lose Data or Hang DMA Operations Under Certain Conditions (OMAP3530/25 only)*

Revision(s) Affected

3.1.2 and earlier

Details

Under certain conditions, parallel loads with predication to the same cache line may cause victims to be dropped and/or DMA to hang.
All of the following conditions ***must*** be true in order for this problem to occur:

1. Two LD instructions in parallel.
2. Both are LDs to the same cache line (upper 26 address bits are the same).
3. The LD using T1 is predicated and the predicate is *false.*
4. The LD using T2 is either not predicated, or is predicated and the predicate is *true.*
5. The cache line is absent from the cache.
6. The two other lines in the same L1D set are valid.
7. The LRU cache line in the set is dirty.

Results:

- L1D informs L2 to expect a victim for the affected set.
- L2 stalls DMAs with addresses that correspond to that set (DMA includes accesses from IDMA and EDMA).

  Note: DMA includes accesses from IDMA, EDMA, and any external masters, such as PCI or other CPUs.

- L1D processes the true-predicated request correctly.
- L1D does not send the indicated victim.

Impact:

- If the load instruction reads a cacheable location:
  - The updated data in the LRU line gets dropped.
  - DMA accesses whose addresses match the affected set hang.
- If the load instruction reads a non-cacheable location:
  - L1D retains the updated data from the LRU line.
  - DMA reads may see stale data if the LRU line's address is in L2 memory.

Workaround(s)

Use Code Gen patch 6.0.3 (available on update advisor) to recompile your source code and avoid this issue. Libraries supplied by TI will be re-released using the 6.0.3 compiler patch. Customer-generated libraries from TI's third-party supplier may also need to be recompiled.

For existing object code and libraries, an available Perl script can determine locations of parallel predicated loads that may fail. The script is available at the same update advisor location as the Code Gen patch.

Advisory 3.1.1.11 *Race Condition May Cause I2C Slave to NACK a Transfer*

Revision(s) Affected 3.1.2 and earlier

Details If the I2C module is configured as a slave, in *autoidle* mode (I2C_SYSC.AUTOIDLE = 1) and the ARDY (I2C.12C_STAT[2]) condition and the START condition are detected in the module at the same time, internal clock gating will be incorrectly applied. This will cause the I2C to NACK (I2C.12C_STAT[1]) the transfer for which the START (I2C.I2C_STA[6]) condition was received. Subsequent transfers will be ACKed as expected.

Workaround(s) **Workaround 1**: Software ***must*** set SYSC_AUTOIDLE to 0. In this case, the failure condition never occurs.

Workaround 2: Ensure that the external I2C master always resends a NACKed transfer via software. If a transfer was NACKed because of this race condition, the next transfer will always be ACKed.

Advisory 3.1.1.12 *MDR1 Access Can Freeze UART Module When in IrDa Mode*

Revision(s) Affected

3.1.2 and earlier

Details

Because of a glitchy structure inside the UART module, accessing the MDR1 register may create a dummy underrun condition and freeze the UART IrDa transmission. Only IrDa modes *Slow Infrared* (SIR), *Medium Infrared* (MIR), and *Fast Infrared* (FIR) are impacted. Even if the bug condition occurs in *UART* mode or *IrDa CIR* mode, it will have no effect. Therefore, UART1 and UART2 are immune to this problem, and only UART3 may exhibit this issue when used in one of the IrDa modes– SIR, MIR, or FIR.

Workaround(s)

To ensure this problem does not occur, the following software initialization sequence must be used each time MDR1 must be changed to one of the three failing *IrDa* modes:

1. If needed, setup the UART by writing the required registers, except MDR1.
2. Set appropriately the MDR1.MODE_SELECT bit field.
3. Wait for 5 L4 clock cycles + 5 UART functional clock cycles.
4. Clear TX and RX FIFO in FCR register to reset its counter logic.
5. Read RESUME register to resume the halted operation.

Advisory 3.1.1.13 *IVA2: Block Cache Operations Word Count (*WC) Must Be Less Than or Equal to 0xFF80 (OMAP3530/25 only)*

Revision(s) Affected 3.1.2 and earlier

Details When performing any block cache operation, such as "Writeback", "Writeback with Invalidate", or "Invalidate", for any memory controller or memory range (e.g., L1P, L2, L1D) the word count programmed ***must be*** less than or equal to 0xFF80. If a value greater than 0xFF80 is desired, then this must be broken into multiple operations. The following registers are affected: L2WWC, L2WIWC, L2IWC, L1PIWC, L1DWIWC, L1DWWC, and L1DIWC.

Workaround(s) No workaround.

Advisory 3.1.1.15 *I2C: RDR Flag May Be Incorrectly Set*

Revision(s) Affected 3.1.2 and earlier

Details Under certain rare conditions, the I2C_STAT[13].RDR bit may be set as well as the corresponding interrupt fire, even when there is no data in the receive FIFO, or the I2C data transfer is still ongoing. These spurious RDR events must be ignored by the software.

Workaround(s) Software must filter out unexpected RDR pulses, using the flowchart illustrated in Figure 3 when receiving an I2C RDR interrupt.

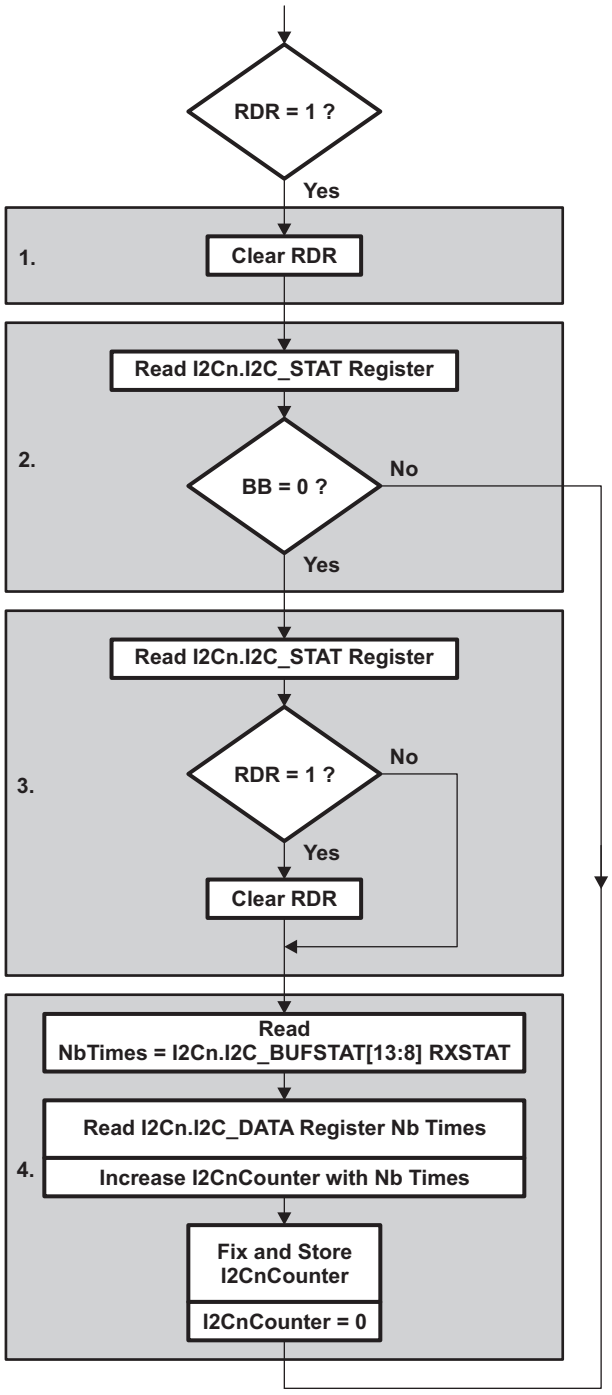


Figure 3. I2C Flowchart

Advisory 3.1.1.16 *IVA2: EDMA Channel Priority Is Not Correctly Enforced (OMAP3530/25 only)*

Revision(s) Affected 3.1.2 and earlier

Details The EDMA channel controller (TPCC) has 2 event queues: Q0 and Q1. Each queue can be mapped to one of the two Transfer Controllers (TPTC): TPTC0 or TPTC1. As explained in the *OMAP35x Technical Reference Manual* (literature number SPRUF98), the events in each event queue will be extracted as soon as the corresponding TPTC is available for a new Transfer Request (TR) to be programmed into the TPTC. However, due to an issue in the IVA2.2 subsystem, the requests queued in Q1 cannot be submitted to their TPTC as long as Q0 is not empty.

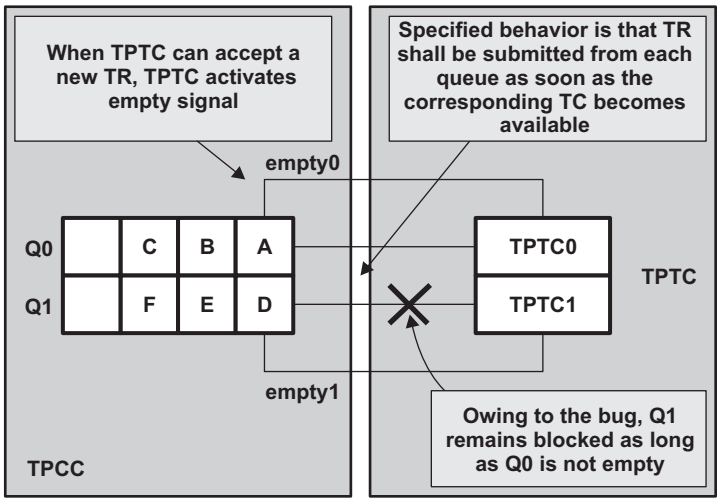


Figure 4. TR Submission Scheme

Workaround(s) Infrequent short transfers (e.g., latency critical synchronized transfers to/from peripheral) must be placed in Q0, and longer transfers (e.g., block copy) in Q1. For example, when a latency critical transfer is placed in EDMA and a longer transfer in QDMA, the following programming model will minimize the impact of this limitation:

- EDMA Event is queued to Q0 (using IVA_TPCC.DMAQNUMn register, n from 0 to 7)
- QDMA Event is queued to Q1 (using IVA_TPCC.QDMAQNUM register)
- Queue to TC Mapping is Q0:TPTC1 Q1:TPTC0 (using IVA_TPCC.QUETCMAP register)
- Queue Priority is Q0 = 0x0, Q1 = 0x7 (using IVA_TPCC.QUEPRI register)

Advisory 3.1.1.29 *Inactive State Management: Impossible to Transition to OFF or RETENTION States*

Revision(s) Affected 3.1.2 and earlier

Details If a power domain meets the conditions to go to an INACTIVE state (i.e., POWERSTATE is programmed to 0x3 (ON) and clock can be shut off), then the domain will transition to INACTIVE state. However, the domain cannot go to RET or OFF state automatically from INACTIVE state, even if software updates the POWERSTATE bit to 0x1 (RET) or 0x0 (OFF). The domain must be transitioned to the ON state before it can transition to the RET or OFF states.

Workaround(s) The following two conditions must be met:

1. Do not use autostate with PM_PWSTCTRL_XXX.POWERSTATE=0x3 (ON) to put power state in INACTIVE. Using autostate from ON to RET (or OFF) transition is not impacted.
2. Perform wake-up event (software must force wakeup) to transition to an active state before transitioning to the RET or OFF states.

Advisory 3.1.1.36 *Inappropriate Warm Reset Generation on Smart Reflex I2C Error*

Revision(s) Affected

3.1.2 and earlier

Details

The Voltage Controller generates an I2C access error when an I2C command is sent to PowerIC during a voltage domain sleep transition. When the access error is generated, an interrupt is generated for software error handling. PRCM generates a warm_reset when an access error is generated by the Voltage Controller during voltage domain wake-up transition. This is to recover the full system (OMAP35x Applications Processor + OMAP Peripherals + Power IC) and avoid a deadlock (OMAP35x Applications Processor wakeup transition stalling due to I2C access error and VDD1/ VDD2 not supplied). Since both VDD1 and VDD2 Voltage Controllers share the same interrupt line, an access error for VDD2 Voltage controller is also propagated to VDD1 Voltage controller.

This bug occurs only with the I2C module used by the Smart Reflex module. Other instances of the I2C module are not impacted by this issue.

In the following scenario,

1. VDD1 is performing a wake-up transition while VDD2 is performing a sleep transition.
2. An I2C access error is generated on VDD2 sleep request and VDD1 and VDD2 share the same error line, so this access error on VDD2 is broadcasted to both VDD1 and VDD2.

The condition for a warm_reset generation is met on VDD1, and Warm_reset is asserted inappropriately.

This issue was detected in simulation, but it is a corner case that is not expected to occur in production, since the I2C access error should never occur if the PCB is safe. This bug is reported in the silicon errata to help developpers in case this kind of behavior is detected, but so far it has not been seen in real silicon.

Workaround(s)

By construction, the issue cannot happen if no I2C command is sent during a VDD2 sleep transition (use SYS_OFF_MODE and Smart Reflex disabled). It will never happen if an I2C access error does not occur (this is the expected behavior).

Advisory 3.1.1.41 *CPU: Back-to-Back SPLOOPS With Interrupts Can Cause Incorrect Operation on C64x+ CPU (OMAP3530/25 only)*

Revision(s) Affected

3.1.2 and earlier

Details

An issue can occur during C64x+ execution when:

1. The DSP code contains 2 contiguous SPLOOP/D/W
2. DSP is interrupted when executing the first SPLOOP/D/W
3. There are less than 2 execute packets between the SPKERNEL of the first SPLOOP/D/W and the SPLOOP/D/W instruction of the second

When this issue occurs, the first SPLOOP/D/W terminates abruptly (i.e. without completing the loop), even though the termination condition is false. The failure mechanism can be seen as a hang or by the first SPLOOP/D/W draining for the interrupt and starting the second SPLOOP/D/W without taking the interrupt or returning to complete the first SPLOOP/D/W.

Workaround(s)

Ensure there are at least two execute packets between the SPKERNEL of the first SPLOOP/D/W and the SPLOOP/D/W instruction of the second.

A fix is implemented in the compiler included in version CGT6.0.6 of the Code Generation Tool (CGT). From this revision of CGT, the compiler now ensures there are two cycles between SPKERNEL and SPLOOP/D/W instructions by adding the appropriate number of NOP instructions following the SPKERNEL instruction.

Advisory 3.1.1.42 *IVA2: DSP Generates False Internal Exception for Multiple Writes (OMAP3530/25 only)*

Revision(s) Affected

3.1.2 and earlier

Details

A false internal exception can be generated by C64x+ if an interrupt happens during DSP code flow and the instructions that will be annulled during pipeline flush are dependant. This issue occurs in the exception detection logic. It examines the DSP instructions during the pipeline flush, even though they have been annulled. The hardware does not detect this and incorrectly assumes that multiple write instructions to the same register with 2 different conditional registers will be executed.

The DSP generates an incorrect internal exception in the following scenario: the CPU is draining the pipeline as part of an interrupt context switch. During this time, it annuls instructions in the pipeline. The first annulled execute packet changes the value of one or more predicate registers. The second annulled execute packet has two or more predicated instructions that use the predicates written in the previous cycle. The values held in the predicate registers appear to cause the instructions in the second annulled execute packet to write to the same register. The conflicting writes would not happen if the first execute packet was not annulled.

Example:

```
ZERO A0
ZERO A1
------------> (interrupt occurs here)
MVK 1, A0; (annulled)
[!A0] MVK 2, A1; (annulled)
|| [!B0] MVK 3, A1; (annulled)
```

Invalid exception triggers as it appears that the last two MVK will both write A1.

Even if this issue appears in a DSP code, it does not affect the code flow and does not produce an unexpected exit routine value.

Workaround(s)

The CPU only recognizes the incorrect exception while it drains the pipeline for an interrupt. As a result, the CPU begins exception processing upon reaching the interrupt handler. The NRP (NMI Return Pointer Register) and NTSR (NMI Task State Register) will reflect the state of the machine upon arriving at the interrupt handler.

Therefore, to identify the incorrect resource conflict exception in software, verify the following conditions at the beginning of the exception handler prior to normal exception processing:

1. Exception occurred during an interrupt context switch.
   - In NTSR, verify that INT=1, SPLX=0, IB=0, CXM=00.
   - Verify that NRP points to an interrupt service fetch packet. That is, (NRP & 0xFFFFFE1F) == (ISTP & 0xFFFFFE1F).
2. The exception is a resource conflict exception. In IERR, verify that RCX == 1 and all other IERR bits == 0.
3. The exception is an internal exception. In EFR, verify that IXF == 1 and all other EFR bits == 0.

Upon matching the above conditions, suppress the exception as follows:

- Clear EFR.IXF by writing 2 to ECR.
- Resume the interrupt handler by branching to NRP.

The above workaround identifies and suppresses all cases of the incorrect resource conflict exception. It resumes normal program execution when the incorrect exception occurs, and has minimal impact on the execution time of program code. The interrupted code sequence runs as expected when the interrupt handler returns.

The workaround also suppresses a particular valid exception case that is indistinguishable from the incorrect case. Specifically, the code will suppress the

exception generated by two instructions with different delay slots (e.g., LDW and DOTP2) writing to the same register in the same cycle, where the conflicting writes occur during the interrupt context switch.

Example of sequence with incorrectly suppressed exception:

```
LDW *A0, A1
DOTP2 A3, A2, A1
NOP
-----------------> interrupt occurs
NOP
NOP ; Both LDW and DOTP2 write to A1 this cycle
```

The workaround will not suppress these valid resource conflict exceptions if the multiple writes occur outside an interrupt context switch. That is, the workaround will not suppress the exception generated by the code above when it executes without an interfering interrupt.

Advisory 3.1.1.53 *GPMC May Stall After 256 Write Accesses in NAND_DATA, NAND_COMMAND, or NAND_ADDRESS Registers*

Revision(s) Affected

3.1.2 and earlier

Details

The GPMC may stall if the following conditions are met:

1. GPMC_CONFIG[0].NANDFORCEPOSTEDWRITE=1.
2. Software performs more than 256 continuous write accesses in NAND_COMMAND_x, NAND_ADDRESS_x or NAND_DATA_x registers.
3. GPMC_STATUS[0].EMPTYWRITEBUFFERSTATUS is always 0 (buffer not empty) during write accesses. This means the software has to write fast enough in GPMC registers in order to never have the write buffer empty.

This mechanism is CS independent. If the software performs 128 write accesses in NAND_DATA_0 followed by 128 write accesses in NAND_DATA_1 then the bug will occur.

Workaround(s)

Accesses performed through the "prefetch and write posting engine" of the GPMC are not impacted by this limitation, and software should use this mechanism for the best performance.

If the prefetch and write posting engine is not used, when GPMC_CONFIG[0].NANDFORCEPOSTEDWRITE=1 and after 255 write accesses in NAND_COMMAND_x, NAND_DATA_x or NAND_ADDRESS_x registers, the software has to wait until GPMC_STATUS[0].EMPTYWRITEBUFFERSTATUS=1 before sending the next 255 write accesses.

Advisory 3.1.1.57 *SPI Dummy DMA RX Request Generation*

Revision(s) Affected 3.1.2 and earlier

Details A dummy DMA RX request is generated as soon as the SPI is configured in *slave* mode and a SPI clock edge is detected. The dummy DMA RX request is generated during module configuration, when the module is not performing an SPI transfer. The dummy DMA RX request occurs as soon as the SPI interface signal sensitivity is changed compared to the default value.

The dummy DMA RX request is generated because the mechanism to avoid dummy data capture on a CS glitch is done regardless of channel activation.

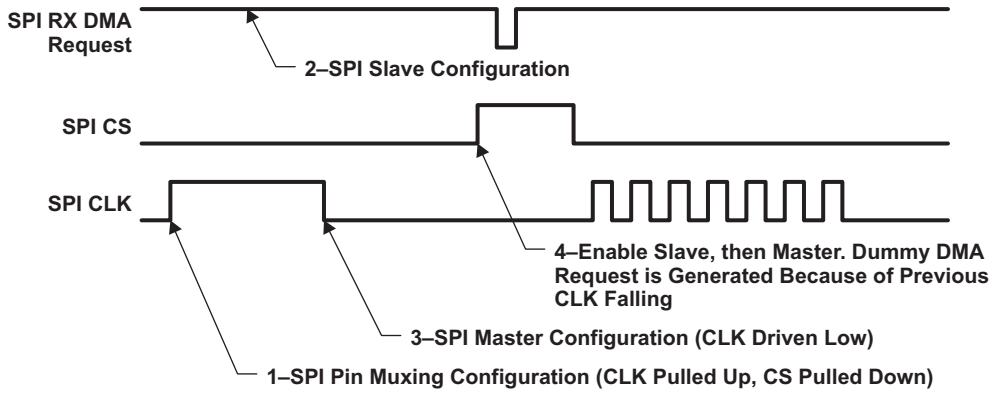


Figure 5. SPI Dummy DMA RX Generation

Workaround(s) Avoid conditions where the SPI is in *slave* mode and the SPI clock toggles (see examples below).

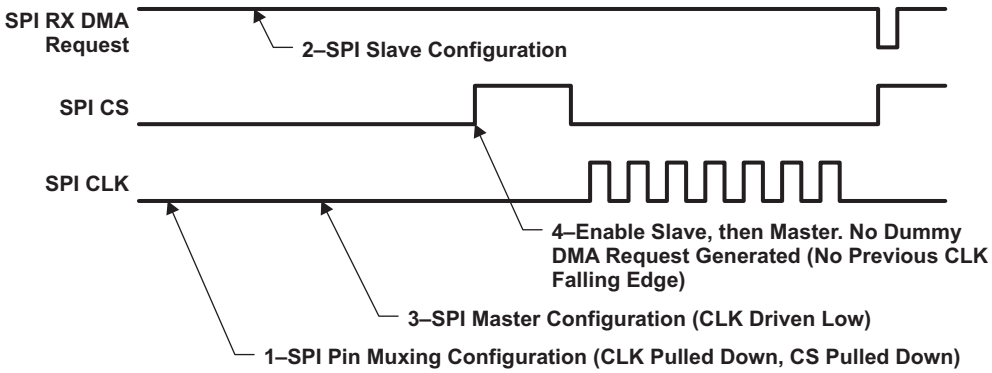


Figure 6. Dummy DMA RX Generation (No Clock Edge)

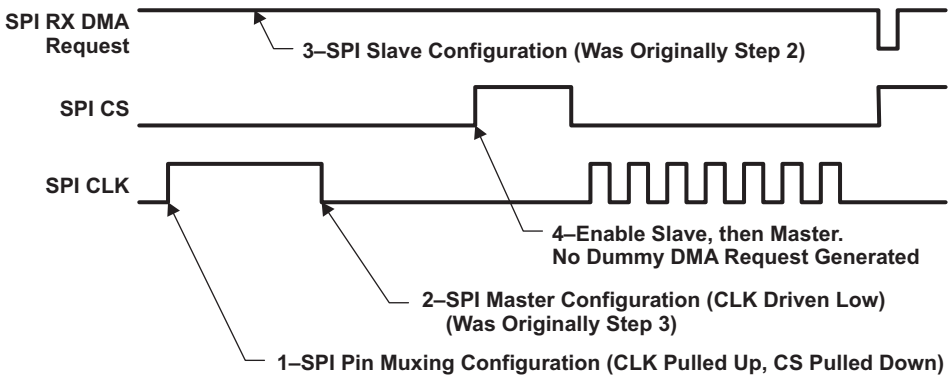


Figure 7. Dummy DMA RX Generation (Slave Mode After Clock Line Driven to Default Value)

Advisory 3.1.1.62 *90 and 270-Degree Rotation DMA Optimization Does Not Function Properly*

Revision(s) Affected 3.1.2 and earlier

Details The DMA optimization functionality has been implemented in the display controller to reduce bandwidth.

The access to the memory in 90- and 270- degree-rotation can be programmed to fetch two pixels per access. When this feature is used, the re-sampling (even if the ratio is 1) must be enabled to store the pixel in the lines buffer. This feature can be used with RGB16 and YUV422 pixels formats. Due to the address generation and the horizontal scaling issue, the DMA optimization does not work properly with RGB16 and YUV422 format for 90-degree and 270-degree rotations. The image is not correctly displayed on the panel.

Workaround(s) No workaround. The bit VIDDMAOPTIMIZATION (bit 20 of DISPC_VID1_ATTRIBUTES or DISPC_VID2_ATTRIBUTES register) must be left to its default value: 0x0.

Advisory 3.1.1.66 *SDMA: DMA4_IRQSTATUS_Lx and DMA4_IRQENABLE_Lx Registers Are Not Secure*

Revision(s) Affected 3.1.2 and earlier

Details The DMA4_IRQSTATUS_Lx and DMA4_IRQENABLE_Lx registers, where x is 0, 1, 2 or 3 are not protected in secure and supervisor modes. Therefore, the channel interrupt line of a secure / supervisor channel can be asserted. However, it is not possible to set or reset an interrupt event.

Workaround(s) No workaround.

Advisory 3.1.1.74 *PRM_VOLTCTRL and PRM_CLKSRC_CTRL Registers Reset on Warm Reset*

Revision(s) Affected 3.1.2 and earlier

Details The PRM_VOLTCTRL and PRM_CLKSRC_CTRL registers are reset on a Warm Reset; however, they should be reset on Cold Reset only. These parameters depend on the device environment only, but the registers must be re-programmed.

Workaround(s) No consequence. If default values are used, the registers must be re-programmed at Warm Reset release.

Advisory 3.1.1.75 *IVA2: CAM/SGX Dependencies (OMAP3530/25 only)*

Revision(s) Affected 3.1.2 and earlier

Details Scenario:

- IVA2 is idled.
- Interrupt is propagated to the IVA2.
- IVA2 INTC (WUGEN) generates a Wake-Up event to the PRCM for IVA2 Wake-Up.
- A Wake-Up dependency is defined between IVA and CAM, or IVA and SGX.
- PRCM wakes-up IVA2, which initiates wake-up for CAM/SGX.
- Interrupt is propagated to IVA2 Core.
- At this point IVA2 Core could initiate a transfer to CAM/SGX even though the CAM/SGX module may not have finished its wake-up sequence, thus resulting in OCP transfer fail.

Workaround(s) The IVA2 software should look at the CAM/SGX clock activity status bit to verify that the CAM/SGX domain is ON before performing any accesses. Next, 10 NOPs should be inserted for additional margin.

Advisory 3.1.1.76 *Erroneous SResp Generation Issued to AES Immediately After Soft Reset*

Revision(s) Affected 3.1.2 and earlier

Details An OCP bus error (SRESP) occurs when an access to the module is performed while the module is coming out of Soft Reset.

Workaround(s) Insert 2 NOPs after Soft Reset assertion.

Advisory 3.1.1.77 *MPU L2 Cache Size Status Register Value Inverted*

Revision(s) Affected 3.1.2 and earlier

Details The MPU L2 Cache Size Status register value is inverted compared to the value given in the spec. The MPU L2 cache size status (CONTROL_FEATURE_OMAP_STATUS[11:10].MPU_L2_CACHESIZE) is inverted:

Expected:

- 00 = 0KB
- 01 = 64KB
- 10 = 128KB
- 11 = 256KB

Current implementation:

- 00 = 256KB
- 01 = 128KB
- 10 = 64KB
- 11 = 0KB

Workaround(s) The software should appropriately handle the inversion.

Advisory 3.1.1.80 *Acting as a Host; For Bulk Split IN Transactions, the MUSBMHDRC Can Transmit Tokens too Close to the SOF Packet, Causing an IPG Error*

Revision(s) Affected

3.1.2 and earlier

Details

This defect will only occur for the following conditions:

- The USBMOTGHS is acting as a host
- BULK IN Split transactions in which the RX polling interval (RXINTERVAL) is large and the peripheral is not responding

This may cause tokens and packets to be transmitted near the SOF causing inter-packet gap violations and data corruption of the packet.

When the USBOTGHS is acting as a host and a Bulk Split IN transaction is in progress, the USBOTGHS transmits an in token and the device responds by sending data. If data is received, the USBOTGHS requests another packet at its scheduled time (send an IN token). However, if the device responds with a NAK by default, then the USBOTGHS will continue to request a packet until stopped by software. However the RXINTERVAL (0x1D) register can be used to limit the number of NAKs received. By default, the USBOTGHS will not limit the number of requests. The error identified by this defect occurs if the USBOTGHS has been transmitting IN tokens for the entire micro frame In this case, the MUSBMHDRC can transmit an IN token that is so close to the SOF that if the device responds by sending a packet, the packet would be corrupted by the transmission of the SOF. The result would be a corrupted packet (which can be identified by the CRC). For all other transactions, this corruption is avoided by use of the HS_EOF1 (0x1C), FS_EOF1 (0x1D) or LS_EOF1 (0x1E) registers. These registers define a blackout that prevents transmission of IN tokens too close to the SOF.

Workaround(s)

Limit the number of times the USBOTGHS will transmit an IN token if no packet is available. Limiting the number of attempts to 2 or 3 attempts will ensure that this corruption will never occur.

Advisory 3.1.1.81 *OCP Error Does Not Get Communicated to USBOTG*

Revision(s) Affected 3.1.2 and earlier

Details The OCP master interface of the sub-chip has a design limitation. This interface connects to an OCP slave on one side and to the AHB master interface of Mentor Graphics® USB On-The-Go (OTG) controller, MUSBMHDRC on the other side. It's a wrapper that converts AHB master read/write requests to equivalent OCP read/write requests and the OCP response/data from OCP slave is expected to be converted to equivalent AHB response/data. In the design, the AHB hresp is always set to OKAY. As a result, an OCP response is never translated to an equivalent AHB response. If an OCP error response is received on the master interface, it will be sent as an AHB OKAY response and not as an AHB Error response.

The DMA controller will continue with the DMA read/write transfer since it is unaware of the OCP error response that occurred for the given read/write transfer. The bus error bit in DMA_CNTL register will never be set.

Workaround(s) In an OCP error scenario, halt the DMA and terminate the DMA transfer.

Advisory 3.1.1.83 *L3 Interconnect Clock Divisor Default Value Must Be Modified Before Configuration of SDRAM Controller*

Revision(s) Affected

3.1.2 and earlier

Details

The SDRAM output clock is gated when an incorrect L3 CLK_SEL ratio is set. This operation is generally transparent and handled at boot time by the ROM code. The workaround should be implemented for the following scenarios:

- GP device External Fast boot is used.
- EMU device External boot is used.

Workaround(s)

For External Boot or External Fast Boot which are both impacted by the issue, CLKSEL_L3 (bit 1:0) = 10b must be set before performing SDRC configuration. In all other cases, this programing is handled by the ROM code and no workaround is necessary.

Advisory 3.1.1.114 *DMA: Drain_IE Reset Value*

Revision(s) Affected 3.1.2 and earlier

Details The Drain_IE bit in the DMA4_CICRi[12] register is not initialized after POR, and its default value can be either '0' or '1' while the documentation specifies '0'.

Workaround(s) Prior to the DMA setup, the software must write 0x0 to this bit to disable the Drain_IE interrupt in the CICR register.

Advisory 3.1.1.115 *sDMA: Channel Is Not Disabled After a Transaction Error*

Revision(s) Affected 3.1.2 and earlier

Details During a destination synchronized transfer on the write port (or source sync with SDMA.DMA4_CCRi[25] BUFFERING_DISABLE = 1), if a transaction error is reported at the last element of the transaction, the channel is not automatically disabled by DMA.

Workaround(s) Whenever a transaction error is detected on the write side of the channel, the software must disable the channel by writing a '0' to DMA4_CCRi[7]: ENABLE bit.

Advisory 3.1.1.121 *USB DEVICE Aborts Remote Wake-Up Sequence When OMAP35x Device Wakes From OFF/RET to ON in USB TLL Mode*

Revision(s) Affected

3.1.2 and earlier

Details

When the OMAP35x device is programmed to go to OFF/RET mode, the power, reset, and clock management module (PRCM) powers down the High-speed USB Host Subsystem. In response, the High-speed USB Host Controller suspends the bus and the USBTLL issues a suspend interrupt.

Before USB Host Controller is switched OFF its register contents are automatically saved to voltage domain retention memory. Also, before the CORE voltage domain is switched to OFF/RET, the USBTLL contents are saved. Once these registers are saved, the OMAP35x device transitions to the OFF/RET state.

When an external device initiates a remote wakeup, the PRCM wakes the OMAP35x device. The CORE domain reset is released and the USBTLL registers are restored. Upon the completion of the USBTLL register restore an ALT interrupt is erroneously generated by the USBTLL to the external device. This breaks the USB remote wakeup protocol and as a result the external device aborts the remote wakeup sequence.

Workaround(s)

To workaround this issue the IdGnd fall interrupts should be disabled by the external device at boot-up. This interrupt can be disabled by clearing the IDGND_FALL bit of the ULPI_USB_INT_EN_FALL_i registers.

Advisory 3.1.1.127 *CONTROL_SWRV_i and CONTROL_MSV Registers Accessible Only in Secure Mode on HS Devices*

Revision(s) Affected

3.1.2 and earlier

Details

The registers CONTROL_SWRV_i and CONTROL_MSV are expected to be readable regardless of the device type, i.e. general purpose (GP) device or high security (HS) device.

The expected behavior is implemented on GP devices. However, on HS devices there is a difference between read accesses performed in secure and non-secure modes:

- Read access to CONTROL_SWRV_i or CONTROL_MSV in non-secure privilege mode always returns 0s.
- Read access to CONTROL_SWRV_i or CONTROL_MSV in secure privilege mode returns the accurate register value.

Workaround(s)

No workaround.

Advisory 3.1.1.128 *Pending Interrupt to Video Sequencer Prevent IVA2 from Going Into Idle Mode*

Revision(s) Affected

3.1.2 and earlier

Details

All the interrupt events either masked or un-masked should be cleared before transitioning the IVA2 domain to standby. This condition is mandatory to allow the idle transition, otherwise the IVA2 is kept active.

This issue may occur if the video sequencer is not used (i.e., kept under reset or already idled) while the IVA2 idle sequence is performed. In this case, if an interrupt is propagated to the video sequencer, it will not be handled by the interrupt handler which is inactive but would avoid the IVA2 standby transition. The interrupt is not processed because the video sequencer is inactive, thus avoiding the standby transition of the whole IVA2 domain.

The following is an example of this issue: When performing an MP3 low-power use case, the video sequencer is not used, kept under reset, and only the DSP domain of the IVA2 sub-system is used. Any interrupt (even masked) to the video sequencer avoids standby transition for the entire IVA2 domain.

Workaround(s)

Any write/read to the video sequencer IRQ Register (SEQ_IRQCLR/SEQ_IRQSTATE, etc.) from the DSP or L3 interconnect resolves this issue and allows the standby transition.

See below sample code example which can be inserted in C source code:

```
// Definition of Constant Values
asm("IVA2_SEQ_BASE_ADDRESS .set 0x00090000");
asm("SEQ_IRQSTATE_OFFSET .set 0x0000004C");
asm("EFI_READ32_REQ_CMD .set 0x02");
//;; Required Code Fragment for Dummy read register in Sequencer
// push registers to be used
asm(" STW A3, *SP--");
asm(" STW A4, *SP--");
asm(" STW B4, *SP--");
asm(" SUB SP, 4, SP"); // secure 1 word for local frame generation
//;; Sequencer ReadRegister() sequence
asm(" ZERO B4");
asm(" MVKH IVA2_SEQ_BASE_ADDRESS, B4");
asm(" DINT");
OMAP3530 silicon errata Revision 5.0
150
TI Confidential
NDA Restrictions
asm(" ADDK SEQ_IRQSTATE_OFFSET, B4");
asm(" NOP");
asm(" EFSW.L1X B4, EFI_READ32_REQ_CMD");
asm(" NOP 9");
asm(" NOP 8");
asm(" EFRW.S1 A3");
asm(" STW A3, *B15[1]"); // if required to save the read data
asm(" NOP 2");
asm(" RINT");
// pop registers
asm(" ADD SP, 4, SP");
asm(" LDW *++SP, B4");
asm(" LDW *++SP, A4");
asm(" LDW *++SP, A3");
```

Advisory 3.1.1.129 *McSPI Can Generate a Wrong Underflow Interrupt*

Revision(s) Affected 3.1.2 and earlier

Details

In the case where:

- A McSPI module is configured as master and is connected to another McSPI module configured as a slave (on the same chip, or on a different chip)
- The CS polarity is changed from the reset state (i.e., changed from CS inactive low to CS inactive high) on the master and slave sides
- The slave is enabled and then the master is enabled according to the programming guide

then the slave McSPI will generate a false underflow as soon as the first channel is enabled on the master McSPI side. This is because the master McSPI sets the right clock and chip select polarities only when the first channel is enabled. As the CS polarity is changed on the master side, this will generate a low-to-high transition on the CS signal. The slave McSPI will detect this transition and will try to load its shift register, which will result in an underflow interrupt being generated because there is no data to load.

If the slave is an external SPI device, then there is no issue. Only slave McSPI modules will be impacted.

If the CS polarity is not changed from its reset state, then there is no issue.

This issue will only occur when performing loopback tests on the same chip between two McSPI modules or when communicating through SPI between two McSPI modules on 2 different chips.

Workaround(s)

The following initialization sequence will solve this issue:

1. On the master side: Set MCSPI_MODULCTRL:SINGLE. Perform the following 3 steps by doing 3 different OCP accesses:
   - Configure channel I in MCSPI_CH(I)_CONF
   - Set MCSPI_CH(I)_CONF:FORCE
   - Reset MCSPI_CH(I)_CONF:FORCE
   - Reset MCSPI_MODULCTRL:SINGLE bit. The SPI bus polarity is now updated.
2. On the slave side : Configure channel 0 : write MCSPI_CH0_CONF Enable channel 0 : set MCSPI_CH0_CTRL:EN
3. On the master side : Enable channel i : set MCSPI_CH(i)_CTRL:EN

Advisory 3.1.1.131 *TV Detect AC Coupling Mode Not Supported*

Revision(s) Affected 3.1.2 and earlier

Details The TV detect in AC coupling mode is not implemented accurately and is not functional; therefore, TV detection in AC mode is impossible.

Workaround(s) Use DC coupling mode only.

Advisory 3.1.1.132 *USB DMA Cannot Handle Concurrent Channels*

Revision(s) Affected 3.1.2 and earlier

Details USB DMA can be used only with one channel (Rx or Tx) active at a time. When more than one channel is active, then DMA transfers cannot be guaranteed.

Workaround(s) **Workaround A**: Use the interrupt mode.

Workaround B: Use Tx DMA mode1 for highest throughput requirement endpoint. Use interrupt mode for others.

Workaround C: Use Rx DMA mode0 for highest throughput requirement endpoint and interrupt mode for others.

Advisory 3.1.1.133 *VC1 En/De-coded Bit Stream Corrupted When iLF Is Used*

Revision(s) Affected 3.1.2 and earlier

Details Due to a hardware issue in an arithmetic operator in the iLF accelerator, the output of the loop filter algorithm is corrupted (truncation error). This operator is not used in any other block of the IVA2.2 subsystem. Other multimedia codecs (H264, RV9, WMV9, …) using iLF are not impacted by this bug. It is safe to use iLF for codecs other than VC1.

Note 1: This bug does not create a noticeable impact on the video quality. Even if there is no visible impact on the video quality for the end-user, using iLF loop filter for VC1 encoding or decoding generates a video stream that is not bit exact as compared to the reference: *SMPTE Standard for Television: Compressed Video Bitstream Format and Decoding Process*.

Note 2: iLF supports the WMV9 implementation, but not the whole VC-1 SMPTE standard, even if VC-1 was originally derived from WMV9.

Workaround(s) iLF accelerator cannot be used for VC1 decoding and will not produce a bit exact output compliant with SMPTE standard. A SMPTE compliant bit stream can be achieved by using IVA2.2 software solution (iLF not used ,and loop filtering handled by DSP software).

Advisory 3.1.1.134 *Unexpected Stalling May Occur During SDMA/IDMA Accesses to DSP L2 Memory*

Revision(s) Affected 3.1.2 and earlier

Details

NOTE: This exception does not apply if there are no IDMA/SDMA accesses to DSP L2 memory (both cache/SRAM and shared SRAM) during run-time.

Background Information:

The C64x+ Megamodule in the IVA2.2 subsystem has a Master Direct Memory Access (MDMA) bus interface and a Slave DMA (SDMA) bus interface (see Figure 8). The MDMA interface provides DSP access to resources outside the C64x+ Megamodule such as external DDR memory, flash memory, and On-chip Memory (OCM) RAM. The MDMA interface is typically used for C64x+ CPU/cache accesses to memory beyond the Level 2 (L2) memory level. These accesses can be in the form of cache line allocates, write-backs, and non-cacheable loads and stores to/from system memories.

The SDMA interface allows other master peripherals in the system to access DSP memories. The memories which may be accessed are Level 1 Data (L1D), Level 1 Program (L1P), L2 cache/RAM, and shared L2 RAM. Examples of masters who may access these memories are EDMA transfer controllers, the system DMA, and the MPU.

Note: The IVA2.2 video hardware accelerator accesses L2 shared RAM through the shared L2 interface (SL2IF). Accesses through the shared L2 SRAM are not susceptible to this issue.

The DSP Internal DMA (IDMA) is a DMA engine within the C64x+ Megamodule which is used to move data between internal DSP memories (L1 and L2) and/or DSP peripheral configuration bus. The IDMA engine shares resources with the SDMA interface.

The C64x+ Megamodule has a L1D cache and L2 cache both implementing write-back data caches. This means that it holds updated values for external memory as long as possible. It writes these updated values to external memory either when it needs to make room for new data, or when specifically requested to do so by the application. These write-backs are called “victims.” The L1D sends its victims to L2. The caching architecture has pipelining which means multiple requests could be pending between L1, L2 and MDMA.

Additional details on the C64x+ Megamodule and its MDMA and SDMA ports can be found in the *TMS320C64X+ Megamodule Reference Guide* (literature number SPRU871).

Exception Description:

Ideally the MDMA and SDMA/IDMA paths operate independently with minimal interference. Normally MDMA accesses may stall for extended periods of time (clock cycles) due to expected system level delays (e.g. due to bandwidth limitations and/or DDR memory refreshes). However, due to the exception, there are cases where SDMA and/or IDMA accesses to L2/L1 experience an additional/unexpected stall during the normal stalls seen by the MDMA interface. For latency sensitive traffic, the SDMA stall can result in missing real time deadlines.

NOTE: SDMA/IDMA accesses to L1P/D will not see any unexpected stall if there are no SDMA/IDMA accesses to L2 (both cache/SRAM and shared SRAM). This is because unexpected SDMA/IDMA stalls to L1 happen only when they are pipelined behind L2 accesses which experience stalls due to the exception.

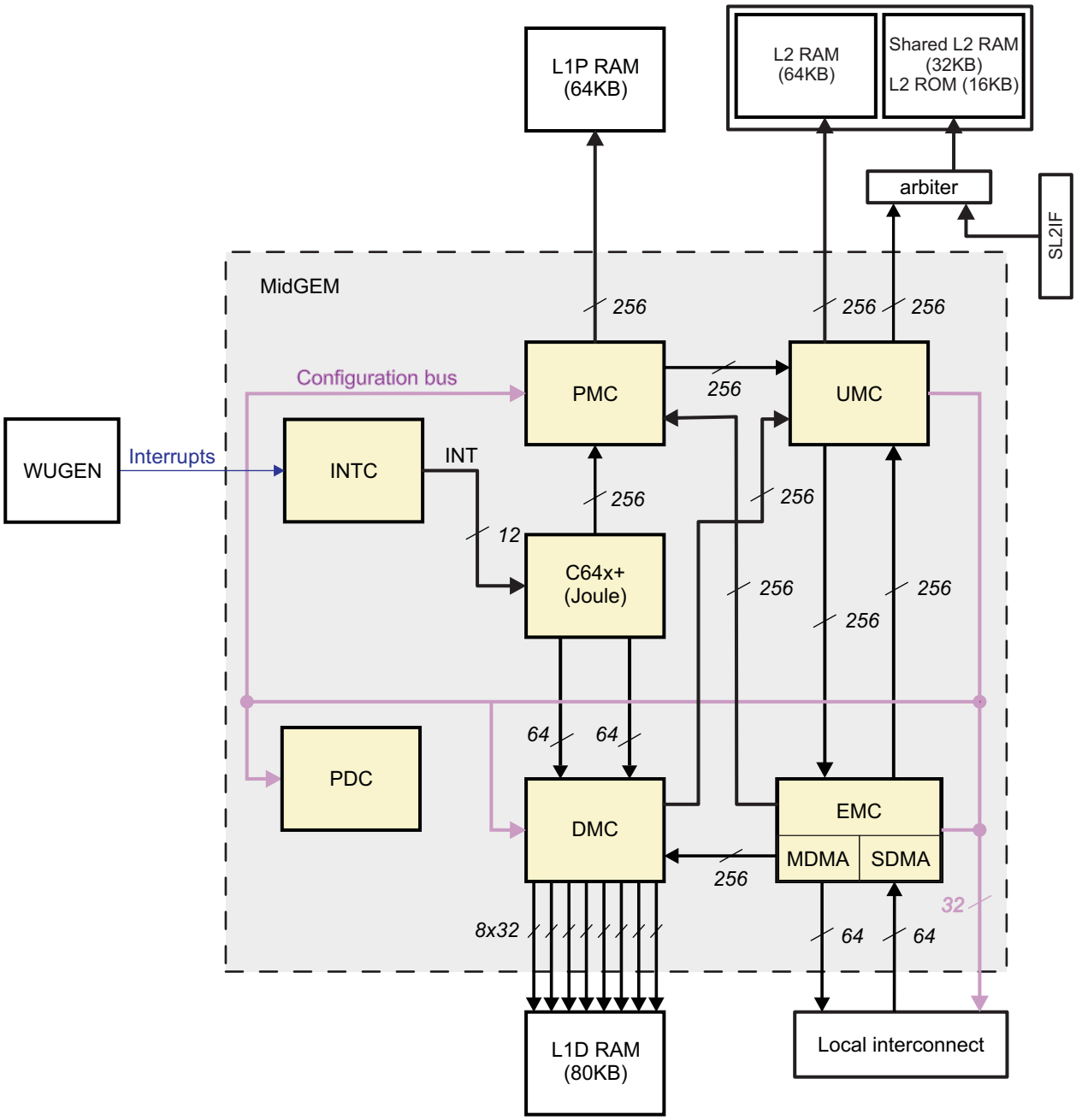


Figure 8. C64x+ Megamodule Block Diagram

In the following scenarios, SDMA/IDMA stalls may possibly occur. Note that each of these scenarios describes expected normal DSP functionality. The only issue is that the SDMA/IDMA access potentially exhibits additional unexpected stalling during each of these scenarios:

1. Bursts of writes to non-cacheable MDMA space (external DDR memory, flash memory, and OCM RAM). The DSP buffers up to 4 non-cacheable writes. When this buffer fills, SDMA/IDMA is blocked until the buffer is non-full. Thus, bursts of non-cacheable writes longer than three writes can stall SDMA/IDMA traffic.
2. Various combinations of L1 & L2 cache activity:

   (a) L1D read miss generating victim traffic to L2 (Cache or SRAM) or external

memory. The SDMA/IDMA may be stalled while servicing the read miss and the victim. If the read miss also misses L2 cache, the SDMA/IDMA may be stalled until data is fetched from external memory to service the read miss.

(b) L1D read request missing L2 (going external) while another L1D request becomes pending. The SDMA/IDMA may be stalled until the external memory access is complete.

(c) L2 victim traffic to external memory during any pending L1D request. SDMA/IDMA may be stalled until external memory access and the pending L1D request completes.

The duration of the IDMA/SDMA stalls is highly dependent on the quantity/characteristics of the L1/L2 cache traffic and MDMA traffic present in the specific system/application and is not simple to quantify. In case # 2a, 2b and 2c from above, stalling may or may not occur depending on specific state of the cache request pipelines and target locations of the traffics. These stalling mechanisms may also interact in various ways to cause combined stalls of larger duration. Due to these factors, it is not straightforward to predict if the stalling will occur and the duration of the stalling.

The occurrence of such IDMA/SDMA stalling and/or any system impact of such stalling will be most likely in systems with very excessive context switching and/or L1/L2 cache miss/victim traffic and in systems with very heavily loaded EMIF, again making it difficult to predict occurrence and precise duration of individual stalls.

Determining if a real-time deadline miss is due to this exception:

The steps outlined below can be used to determine if the issue described here is the culprit of real-time deadline misses for existing applications. Examples of situations in which real-time deadlines may be being missed include loss of McBSP samples and low peripheral throughput.

Step 1. Determine if the transfer missing the real-time deadline is directly accessing L2 or L1D memory. If not, then this issue is not the source of the problem.

Step 2. Next, identify all SDMA transfers to/from L2 memory. An EDMA transfer to/from L2 from/to a McBSP would be an example of such a transfer. Another example would be an system DMA block transfer to/from L2.

If there are no SDMA transfers going into L2, then this issue is not the source of the problem.

Step 3. Redirect all SDMA transfers to L2 memory to other memories. This can be done several ways:

- Temporarily move all the L2 SDMA transfers to L1D SRAM.
- If not all L2 SDMA transfers can be moved to L1D memory, temporarily direct some of the transfers to OCM RAM or DDR memory and keep the rest in L1D memory. Still, there should be no L2 SDMA transfers.

If real-time deadlines are still being missed after implementing any of the options in Step 3, then the issue described here is likely not the cause of the problem. If real-time deadline misses are solved using any of the options in Step 3, then it is very likely this issue is the source of the problem. Refer to “Workaround Description” for ways to work around this issue.

Workaround(s)

Method 1

Entirely eliminate the exception by removing all SDMA/IDMA accesses to L2 SRAM. The EDMA/QDMA, IDMA, system DMA, and other chip masters cannot access L2. There are no issues with the CPU itself accessing code/data in L2. Also, this issue only pertains to SDMA accesses to L2, accesses through the shared L2 interface (SL2IF) are still allowed. Note that it is recommended to always configure the L2 cache/SRAM block as 100% cache.

Method 2

Issues such as dropped McBSP samples can be worked around by moving latency sensitive buffers outside the C64x+ Megamodule. For example, rather than placing buffers for the McBSP into L1/L2, those buffers can instead be placed in other memory

such as OCM RAM.

Method 3

Employ any combination of the following as appropriate/possible to reduce the system impact of the exception:

1. Improve system tolerance on DMA side (IDMA/SDMA/MDMA):
   (a) Understand and minimize latency critical SDMA/IDMA accesses to L2 or L1P/D.
   (b) Directly reduce critical real-time deadlines if possible at peripheral/IO level (e.g. increase word size and/or reduce bit rates on serial ports).
   (c) Reduce DSP MDMA latency by:
      (i) Increase priority of DSP access to external memory such that MDMA latency of MDMA accesses causing stalls is minimized (NOTE: other masters such as system DMA may have real-time deadlines which dictate higher priority than DSP).
      (ii) Do not perform external memory access using ready handshaking during DSP run-time (devices using ready handshaking potentially insert very excessive latency to external memory accesses).
2. Minimize offending scenarios on DSP/Caching side:
   (a) If DSP performing non-cacheable writes is causing issue, insert non-cacheable reads every few writes to allow write buffer to drain.
   (b) Avoid caching from slow memories such as asynchronous memory. Instead, page the data via the EDMA from the off-chip async memory to L2 SRAM or SDRAM space before accessing the data from the DSP. NOTE: paging cannot occur while real-time deadlines must be met.
   (c) Use explicit cache commands to trigger cache write-backs during appropriate times (L1D Writeback All, L2 Writeback All). Conversely, do not use these commands at inappropriate times (when real-time deadlines must be met).
   (d) Restructure program data and data flow to minimize the offending cache activity
      (i) Define the read-only data as “const". The const C keyword tells the compiler that the array will not be written to. By default, such arrays get allocated to the “.const” section as opposed to BSS. With a suitable linker command file, the developer can link the .const section off-chip, while linking .bss on-chip. Because programs initialize .bss at run time, this has the added benefit of reducing the program’s initialization time and total memory image.
      (ii) Explicitly allocate lookup tables and writeable buffers to their own sections. The #pragma DATA_SECTION(label, “section”) directive tells the compiler to place a particular variable in the specified COFF section. The developer can explicitly control the layout of the program with this directive and an appropriate linker command file.
      (iii) Avoid directly accessing data in slow memories (e.g. flash), copy at init time to faster memories
   (e) Modify troublesome code
      (i) Rewrite to use DMAs to minimize data cache writebacks. If the code accesses a large quantity of data externally, consider using DMAs to bring in the data, using double buffering and related techniques (minimizing cache write-back traffic and thus chance for occurrence of the exception).
      (ii) Re-block the loops. In some cases, restructuring loops can increase reuse in the cache, and reduce total traffic to external memory.
      (iii) Throttle the loops. If restructuring the code is impractical, then it becomes reasonable to slow it down. What this does is reduce the likelihood that consecutive SDMA/IDMA blocks “stack up” in the cache request pipelines resulting in a very long stall.

Advisory 3.1.1.138 *ISP: LSC Issue When Used Concurrently With Resizer*

Revision(s) Affected 3.1.2 and earlier

Details Due to faulty arbitration in the SBL (shared buffered logic: local interconnect of the ISP), some LSC coefficients can be incorrectly applied on the image. This issue appears only if the preview resizer is accessing the SBL simultaneously with LSC table prefetch. The issue appears randomly depending on the access sequencing between resizer access and LSC accesses. The occurrence of the issue increases with larger image size and smaller paxel size. The corrupted LSC coefficients are not random values, but value of a neighbor (the artifact is difficult to detect with regular values but can be clearly seen with a dedicated LSC pattern).

Workaround(s) The workarounds consist of avoiding concurrent accesses to the SBL from the resizer and LSC which can be implemented using various methods. TI is currently validating the different options, and this section will be updated once the optimal workaround in terms of efficiency and software complexity has been confirmed.

Advisory 3.1.1.141 *ISP CCDC DRAM Read-Port Issue*

Revision(s) Affected 3.1.2 and earlier

Details The CCP2 Read-port cannot be used to read data from DDR into CCDC as image data is dropped randomly, causing final image corruption. This issue only affects OEMs requiring the data path from DDR through CCDC.

This issue will impact sensors that require throughput greater than 83 Mpix/sec, which is only possible with CSI2-based sensors. There is no issue for CSI2 sensors that require throughput greater than 83Mpix/sec – in this case, processing is OTF. This will not impact CCP2 sensors based on CCP2 throughput limitations.

Workaround(s) Use an alternate data path for ISP. Use read-back path in Preview Module. TI is continuing to study implementation impact for future releases.

Advisory 3.1.1.142 *ISP Lens Shading Correction Issue*

Revision(s) Affected 3.1.2 and earlier

Details The Lens Shading Correction (LSC) module in CCDC cannot be used simultaneously with the Preview-to-Resizer path, as simultaneous accesses can result in data corruption.

The LSC data is randomly dropped, resulting in a color-shifted final image. This issue only occurs when LSC is used simultaneously with Preview-to-Resizer.

Workaround(s) TI is investigating a work-around in which the LSC in CCDC is used along with the Resizer in mem-2-mem operation.

Other possible work-around options for OEM/3P imaging software are possible:

- Use the LSC available in the Preview module.
- Modify the sensor timings to adjust the Resizer/CCDC timing so that the issue does not occur.

Advisory 3.1.1.144 *SDRC Does Not Send Auto-refresh When OMAP Wakes-up From OFF Mode*

Revision(s) Affected

3.1.2 and earlier

Details

The SDRC auto-refresh counter is not automatically triggered when the SDRC configuration is restored after a CORE OFF. Not refreshing the memory may result in data corruption after some time, which would cause system instability.

This issue is caused by the SDRC state machine remaining in an incorrect state after a reset (either warm or cold).

Workaround(s)

A manual MR write command to SDRAM should be issued as soon as the device wakes up from OFF mode. This will automatically transition the SDRC state machine to the correct state i.e., resuming regular auto-refresh commands. This should be implemented in two steps:

- Read the SDRC_MR_p register content.
- Write back the SDRC_MR_p register with the value read in the previous step.

In order to conform to JEDEC constraints regarding AR intervals (see slide JEDEC Standard No. 209), it is mandatory that the SDRC issues automatic self-refresh entries on inactivity periods for HS devices. This should be done by enabling automatic self-refresh entries on timeout of Auto_cnt (CLKCTRL field of SDRC_POWER register set to 0x2) *AND* defining this time out value to 1 (AUTOCOUNT field of SDRC_POWER register set to 0x1) prior to any transition to OFF mode.

Note: Because of the SDRC design, an atomic PRECHARGE command is issued automatically prior to any MR command. There is no need to account for concurrent accesses that any other initiator in the SOC could generate.

Advisory 3.1.1.145 *CONTROL_REVISION Register Not Aligned With Silicon Revision*

Revision(s) Affected 2.1, 3.0, 3.1, and 3.1.2

Details CONTROL_REVISION register contains the same value (0x00000010) for each silicon revision.

Workaround(s) Use CONTROL_IDCODE, which is upgraded for each silicon revision and documented in the *OMAP35x Technical Reference Manual (literature number* SPRUF98 .

Advisory 3.1.1.146 *HS USB OTG Software Reset Is Not Fully Functional*

Revision(s) Affected 3.1.2 and earlier

Details Upon performing a software reset by setting the OTG_SYSCONFIG:SOFTRESET bit,

- The hsusb_stp pin will remain low instead of going high as expected.
- The reset command will not be sent to the PHY as it should be (the module should send 0x69 or 0x61 and nothing is actually sent).

There is no real impact on the USB link functionality except for the two items above. Only software resets through the OTG_SYSCONFIG:SOFTRESET bit field are impacted (cold or warm resets are not impacted).

Workaround(s) Use the ULPI RegAddr and ULPI RegData registers to manually send the reset command to the PHY. As soon as this is done, LINK and PHY can begin negotiating and functioning normally.

Advisory 3.1.1.148 *CKE PAD Is Not Set When Initializing External RAM*

Revision(s) Affected 3.1.2 and earlier

Details By default, the CKE pad is in safe mode after Power On Reset. A pull-up is connected to it, so its default level is high. The ROM Code never configures it.

In case the booting image contains a Configuration Header which configures a CHRAM section, the ROM Code can configure external RAM according to the parameters located in CHRAM section. In case the platform boots from NAND, it can allow for example the ROM Code to directly copy the booting image into external RAM. Because the ROM Code does not configure this pad, the CKE signal does not reach the external RAM.

There is no functional impact at boot time. There is no limitation at run-time. The CHRAM section defines the setting for register SDRC_POWER at physical address 0x6D00 0070. The SW designer must ensure that any special feature involving CKE signaling is OFF (at boot time only).

- SDRC_POWER must turn OFF all auto-count feature (CLKCTRL field = 0).
- SDRC_POWER must disable the Power Down mode of the target memory, via CKE (PWDENA field = 0).

Workaround(s) There is no workaround for this issue. The functionality "configuration & boot from external RAM" remains operational following the basic rules indicated above. After booting, the application can properly configure the CKE pad and configure a more optimized refresh policy.

Advisory 3.1.1.149 *ROM Code: SDRC_POWER Register Is Initialized With Hardcoded Value*

Revision(s) Affected

3.1.2 and earlier

Details

In case the booting image contains a Configuration Header which contains itself a CHRAM section, the ROM Code can configure external RAM according to the parameters located in CHRAM section. In case the platform boots from NAND, it can allow for example the ROM Code to directly copy the booting image into external RAM. The CHRAM section contains all the values to initialize the SDRC module. When setting the SDRC_POWER register, the ROM Code does not read the value contained in the Configuration header but applies instead a hard coded value set to 0xC1.

This value configures the SDRC as follows:

- High power/High Bandwidth Mode (HPHB)
- Power down mode feature disabled
- Enable clock
- No auto-count feature turned on
- Enter self-refresh when hardware idle request
- Enter self-refresh when a warm reset is applied
- Auto-count = 0

There is no chance to configure SDRC_POWER register with a different value than the one described above.

Workaround(s)

There is no workaround for this issue.

Advisory 3.1.1.150 *I2C : I2C_STAT:XUDF Is Not Functional in Slave Transmitter Mode*

Revision(s) Affected 3.1.2 and earlier

Details When configured in slave transmitter mode, the I2C_STAT:XUDF will not be set as expected if an underflow occurs. Only slave transmitter mode is impacted. Master transmitter mode is not impacted. The impact is rather low as the user can rely on I2C_STAT:XRDY interrupt status bit instead.

Workaround(s) Use the I2C_STAT:XRDY interrupt bit instead of XUDF when in slave transmitter mode.

Advisory 3.1.1.152 *SDRC Timings Are Not Aligned With JEDEC Standard*

Revision(s) Affected

3.1.2 and earlier

Details

3 Timings do not follow the JEDEC standard:

- tRP: Precharge command period on power down
  - JEDEC standard: 0 to 7
  - Current implementation: Fixed to 2
  - Impact analysis: No impact, tolerated by memory components (checked with memory vendors)
- tRFC: Auto-refresh period on self-refresh
  - JEDEC standard: 0 to 31
  - Current implementation: tRFC depends on the delay with the next SDRC command. tRFC min is 1 clock period. tRFC minimum period on self-refresh can be controlled through tCKE (tRFCmin=tCKE) (in the range of tCKE [0 to 7])
  - Impact analysis: No impact, tolerated by memory components (checked with memory vendors)
- tWR: Write Recovery time on power down
  - Issue description: tWR is asynchronous, collides with power-down entry and corrupts the last write operation
  - Impact analysis: Random system crash

Workaround(s)

Disable power-down PWDENA=0. This will impact power consumption, variably depending on use case, and memory type:

- Use case IDLE: power impact less than 0.1mA
- Use case FULL-ACCESS-RATE: power impact less than 1mA

For specific MEDIUM-ACCESS-RATE use-cases, the power impact can be minimized using an alternate software work-around:

- AUTOCOUNT=0, SRFRONIDLEREQ=1, CLKCTRL=1, EXTCLKDIS=1, PWDENA=0
- Theory: play with AUTOCOUNT value to trade performance against power savings
- Power impact ~1mA

All the above numbers are correlated by memory vendors' measurements. In all uses cases, the proposed software work-arounds are robust and have less than 1mA power impact.

Advisory 3.1.1.155 *I2C: Data Lost on Transmission From Memory to I2C Interface*

Revision(s) Affected 3.1.2 and earlier

Details The I2C is configured as master transmitter. After serving a XRDY/XDR interrupt (FIFO empty), from the data sent on OCP, one, two or several bytes sent from the memory to the I2C interface are lost. The bytes lost are always the first transmitted on the OCP, when serving the XRDY/XDR interrupts. The occurrence of the bug is related to the coincidence of the moment when data is sent on the OCP and the moment when the most significant bit of a byte is sent on the I2C, always when starting serving the XRDY/XDR interrupt.

Ideally, no data should be lost when transmitted from the OCP to the I2C. However, one, two or several bytes at the beginning of a transmission from the OCP to I2C are lost, if the moment when they are put on the OCP coincides with the transmission of the most significant bit of a byte on the I2C.

Workaround(s) A workaround exists for the interrupt mode of operation. Before serving the XRDY/XDR interrupts, until also XUDF status bit is set. This marks the clearance of the internal shift register and polls the Status Register – XUDF bit, from the local host, after receiving an XRDY or XDR Note. For the data transmission using DMA, there is no available software workaround.

Advisory 3.1.1.157 *EHCI Controller- Issue in Suspend Resume Protocol*

Revision(s) Affected: 3.1.2 and earlier

Details: This issue concerns HSUSBHOST controller of OMAP35x and more specifically the EHCI controller (Only HS ports are impacted). The issue occurs after a Suspend when trying to Resume the bus (Host initiating the Resume) and also when doing RemoteWakeup (Device initiating the Resume).

The HSUSBHOST exits the resume/wakeup sequence without checking the USB bus LineState to ensure that it is in High Speed Idle state after having put the PHY in High Speed state. This error scenario will occur when there is a delay in PHY indicating the controller that the bus has switched to High Speed, and during that time the host is about to send an SOF packet.

Both TLL and PHY modes are impacted.

Workaround(s): The clear of the “ForceResume” bit in PORTSC register must be done at the beginning of the Port- SOF counter boundary, knowing that PortSOF counter is out of sync after a Suspend with GlobalSOF counter (PortSOF counter is stopped during Suspend while GlobalSOF counter continues counting). The FrameIndex counter is based on GlobalSOF counter (SOF counters are counting the 125us delay of a micro-frame). Only FrameIndex counter can be accessed by Software.

Two Workarounds can be listed for this issue (See flow diagrams and SW programming sequences below):

- Workaround 1:
  - Advantage: No limitation on the number of HS port usage.
  - Disadvantage: This workaround applies only to the to Suspend/Resume case and Suspend/RemoteWakeup case is not supported. This is because where within the PortSOF counter the HW will set the Resume bit is not known.
- Workaround 2:
  - Advantage: This workaround applies to both Suspend/Resume and Suspend/RemoteWakeup cases.
  - Disadvantage: Limitation to have only one HS port used (No limitation on the number of FS port usage as OHCI controller is not concerned). This is because EHCI controller has to be stopped by SW before clearing portSC.FPR.

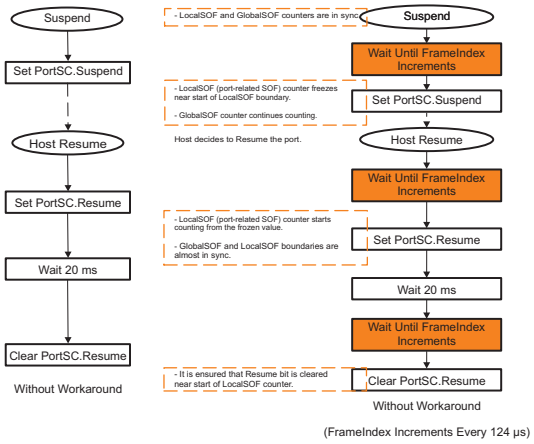
Workaround 1 detailed description: Any Write access to the “Port Suspend” and “Port Force Resume” bits in the PORTSC register must be done at the beginning of a micro-frame. This ensures that the Clear of “ForceResume” bit is done at the beginning of PortSOF counter.

Workaround 1 SW implementation:

- Suspend/Resume case SW implementation:
  1. Read FRINDEX register.
  2. Keep polling FRINDEX register to make sure that the register value has incremented from the value read in Step (1).
  3. Set the PORTSC.suspend bit.
  4. Wait for the required suspend time.
  5. When software is ready to issue resume, Read FRINDEX register.
  6. Keep polling FRINDEX register to make sure that the register value has incremented from the value read in Step (5).
  7. Set PORTSC.FPR bit.
  8. Wait for at least 20ms (as specified by the USB 2.0 Spec).
  9. Read FRINDEX register.
  10. Keep polling FRINDEX register to make sure that the register value has incremented from the value read in Step (9).

11. Clear PORTSC.FPR bit.

- Suspend/RemoteWakeup case SW implementation: NOT APPLICABLE



1.

Figure 9. Workaround 1 Implementation Diagram

Workaround 2 detailed description:

GlobalSOF and LocalSOF counters are in sync as soon as Run/Stop bit is set: Alignment on FrameIndex SOF is required only for the clear of Resume bit.

Workaround 2 SW implementation:

- Suspend/Resume case SW implementation:
  1. Set the PORTSC.suspend bit.
  2. Wait for the required suspend time.
  3. When software is ready to issue resume, set PORTSC.FPR bit.
  4. Wait for at least 20ms (as specified by the USB 2.0 Spec).
  5. Clear the USBCMD.RS bit to 0.
  6. Wait until USBSTS.HCH is set to 1 by HW.
  7. Clear PORTSC.FPR bit.
  8. Set the USBCMD.RS bit to 1.
- Suspend/RemoteWakeup case SW implementation:
  1. Set the PORTSC.suspend bit.
  2. Wait for the required suspend time.
  3. Wakeup event happens and software driver handles the wakeup interrupt.
  4. Wait for at least 20ms (as specified by the USB 2.0 Spec).
  5. Clear the USBCMD.RS bit to 0.
  6. Wait until USBSTS.HCH is set to 1 by HW
  7. Clear PORTSC.FPR bit.
  8. Set the USBCMD.RS bit to 1.

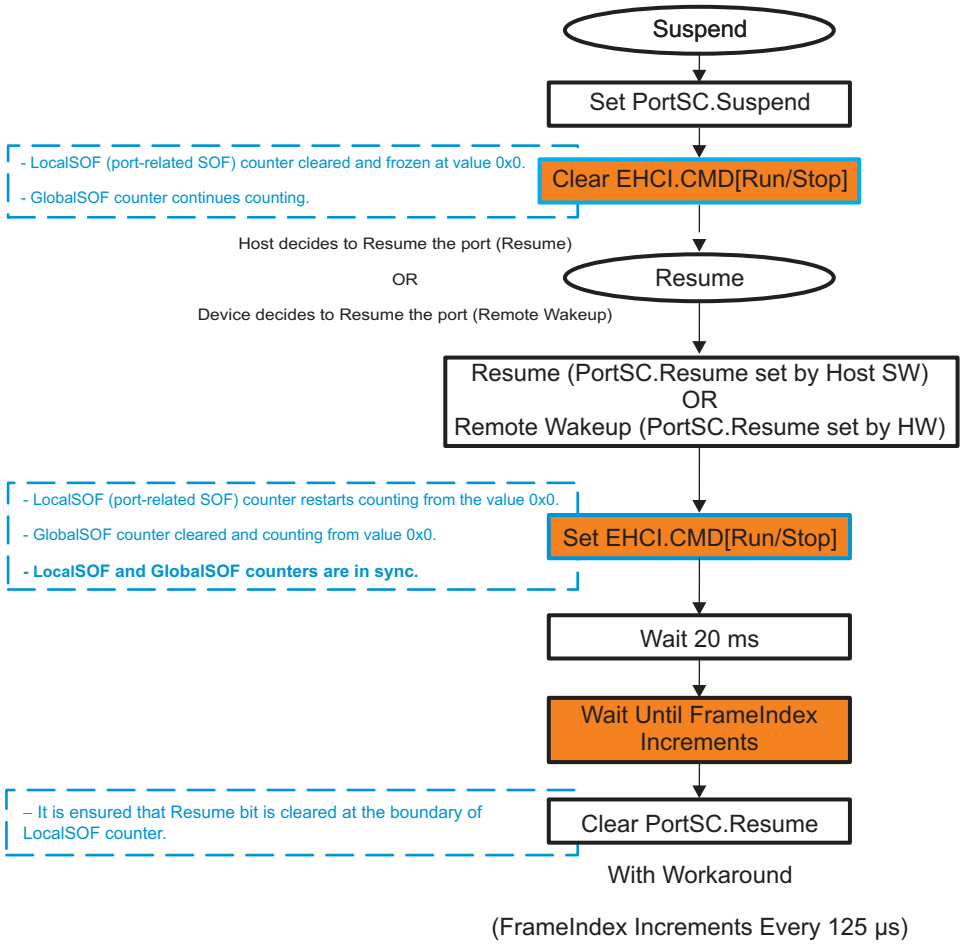


Figure 10. Workaround 2 Implementation Diagram

Advisory 3.1.1.159 *Pull-up Not Maintained On GPIO_28/29 Pin During Padconf Restore*

Revision(s) Affected: 3.1.2 and earlier

Details: A pulse (0.4us) is seen after control module I/O pad configuration restore sequence is completed upon wakeup from Device OFF mode. This pulse is seen on the pin corresponding to GPIO_28 = ETK_D14 and GPIO_29 = ETK_D15 when the line is programmed to maintain the line high through internal pull-up. This "pulse" corresponds to the line being driven temporary by the pull-down instead of expected pull-up.

Condition of occurrence of the issue: The device wakes up from OFF mode and the user wants to maintain the line to high level using the internal pull-up.

Note: This issue does not occur when:

- Device is not going to OFF mode
- Device goes to OFF mode but OMAP is not driving the line (line is driven by other IC)
- Device goes to OFF mode and user wants to maintain the line to low-level

Root cause: This is identified as an issue in the control module save and restore implementation since handshake mechanism between prcm and control module is launched even before the restoration of the last padconf-x register is complete.

Handshake mechanism refers to a signal generated by the control module to the PRCM to notify the completion of the restoration of the pad configuration

The pulse occurs since control module asserts pull-down control of the pad before it re-activates the pull-up control upon the completion of restore sequence. Since pull-down is asserted for some time, the I/O pad toggles from High to Low then to High again causing the pulse on the I/O pad.

Workaround(s): No issue if line is driven low during OFF transitions.

Advisory 3.1.1.160 *GPIO Pad Glitch/Spike Upon Wake-Up From System OFF Mode*

Revision(s) Affected: 3.1.2 and earlier

Details:

When OMAP wakes up from OFF mode, a spurious transition (on the order of a nanosecond) may occur on the pads internally muxed to a GPIO.

The conditions for this spurious transition to occur depend on:

- The GPIO setup (GPIO is input, or output driving 1 or 0)
- The PRCM settings concerning the power domains state transitions (HW or SW)
- The configuration of the pad where the GPIO is muxed

Note: All GPIO blocks including GPIO block1 which is in the WU domain are impacted by this issue.

Root cause: An internal race condition exists between the control signals of the pad upon wake up from system OFF mode, which leads to a possible spurious transition on the signal at the pad's boundary.

The case where a pad OFF mode configuration is used (OFFENABLE =1) but with an override configuration different from the GPIO's configuration in active mode is a valid option and is out of the scope of this errata as we only consider the case where we want the signal to stay constant all the time (same state during active and OFF mode).

Workaround(s):

GPIO belongs to GPIO1 in the WKUP domain (GPIO_0 to GPIO_31)

- Do not use the padconf OFF override: CONTROL_PADCONF_X[9] resp CONTROL_PADCONF_X[25]: OFFENABLE bit='0'

GPIO belongs to GPIO2 to GPIO6 in the PER domain

- Workaround 1 (Recommended option): Use automatic HW transition with Automatic HW wakeup: CM_CLKSTCTRL_PER=0x3 and define a WU dependency between WU domain and PER domain (PM_WKDEP_PER[4]: EN_WKUP=1). Pad Off mode override function must be disabled: CONTROL_PADCONF_X[9] resp CONTROL_PADCONF_X[25]: OFFENABLE bit='0'
  - GPIO maintaining a low level during OFF mode

    It is sw responsibility to ensure that the padconf_x value is written to 0x10C before initiating the transition to OFF (This means GPIO function is configured and pull-down activated on the pad)
  - GPIO maintaining a high level during OFF mode

    Sequence before transition to OFF mode

    1. Save context of the GPIO module.
    2. Configure pull-up in padconf, padconf_x: 0x11C (Note: This can be done before the OFF transition decision is done. One way to proceed is GPIO driver always sets a pull-up in padconf when the GPIO_DATAOUT is set to '1')
    3. Turn GPIO in input GPIO_OE=1
    4. Select Safe mode function, padconf: 0x11F

    Sequence after OFF mode at device wake-up

    1. Complete restoration of the GPIO context
    2. Turn back GPIO in output GPIO_OE=0 (Not needed if step1 and step3 of sequence before transition to OFF mode is done in that order)
    3. Select GPIO function, padconf_x: 0x11C
- Workaround 2 (This option is valid either for SW supervised transition using force wakeup or automatic HW transition) Disable the Pad Off mode override function and ensure that the mode is changed to safe mode before transitioning to OFF mode by following sequence described here after.
  1. Save context of the GPIO module.
  2. Configure pull accordingly to GPIO driving '1' or '0' in padconf. Padconf_x: 0x11C

for GPIO driving a '1' and Padconf_x: 0x10C for GPIO driving a '0'. (Note: This can be done before the OFF transition decision is done. One way to proceed is GPIO driver always set according pull in padconf when the GPIO_DATAOUT is set)

3. Turn GPIO in input GPIO_OE=1
4. Select Safe mode function, padconf: 0x11F for GPIO driving '1' and respectively 0x10F for GPIO driving '0'

Sequence after OFF mode at device wake-up

1. Complete restoration of the GPIO context.
2. Turn back GPIO in output GPIO_OE=0 (Not needed if step1 and step3 of sequence before transition to OFF mode is done in that order)
3. Select GPIO function, padconf_x: 0x11C for GPIO driving '1' and respectively 0x10C for GPIO driving '0'

Advisory 3.1.1.161 *I2C: Wrong RDR Interrupt After Disabling the Module With I2C_EN*

Revision(s) Affected: 3.1.2 and earlier

Details: When the I2C_CON:I2C_EN bit is reset during the I2C module reconfiguration, some synchronization signals are not properly reset. This can generate a wrong RDR interrupt when the next transfer begins (before RRDY interrupt and stop condition).

Workaround(s): During the reconfiguration of the module, perform a software reset of the module (I2C_SYSC:SRST) instead of just resetting the I2C_CON:I2C_EN bit.

Advisory 3.1.1.162 *Voltage Processor TRANXDONE Interrupt Occurs Too Early*

Revision(s) Affected: 3.1.2 and earlier

Details: VPx_TRANXDONE_ST interrupt is generated too early (immediately in worst case) if the HW counter (in Voltage Processor) used to wait for voltage ramp duration is incorrect. The HW counter could be incorrect in the following scenarios:

- The voltage update is done through Voltage Processor force-update command.
- Smart-Reflex was previously enabled.

This limitation impacts the low-to-high OPP transitions as SW waits for TRANXDONE interrupt generation before changing the frequencies. The risk is that high frequency is configured while high voltage is not yet fully ramped. High-to-low OPP transitions are not functionally impacted by this HW limitation.

Workaround(s): Upon Low-to-High OPP transitions, SW shall add a delay (software wait loop) when receiving the VPx_TRANXDONE_ST interrupt and before increasing the frequencies.

This delay needs to be calculated according to:

- PMIC slew rate for voltage ramp-up
- Delta between ‘low’ voltage and ‘high’ voltage
- Delay already introduced by SW execution (ISR, call to set_module_frequency...)

Advisory 3.1.1.164 *SMS ReadEx Deadlock*

Revision(s) Affected: 3.1.2 and earlier

Details: ARM SWAP instructions executed concurrently with another initiator traffic going through SMS VRFB can create a deadlock in some corner cases.

Conditions of occurrence of the issue:

1. ARM executes a SWAP instruction to a non cacheable SDRAM location. This is translated to an OCP ReadEx that should be followed by an unlocking write.
2. Once the ReadEx is issued by the SMS toward the SDRAM controller the arbitration is locked on the ARM SMS thread (class 1 group 0) until unlocking write is received.
3. If VRFB is full before unlocking write is received and accepted by MCU SMS thread (class 1 group 0) then SMS will assert threadbusy_all preventing the unlocking write to be received and accepted by ARM SMS thread (class 1 group 0).
4. VRFB does not de-assert threadbusy_all in case FULL condition is due to a SMS FIFO destination being full and locked (that can not be granted on SDRC arbitration access) by the open ReadEx on the ARM SMS thread (class 1 group 0).

   Note: SWP instruction is not generated by compiler then no risk that swap instruction is generated anywhere outside specific assembly call.

Workaround(s): Replace SWP instructions (locked access) by LDREX/STREX (Exclusive access). Based on ARM documentation LDREX and STREX are supported to shared and non-shared memory. Non-shared memory can be used when the processes to be synchronized are running on the same processor. When the processes to be synchronized are running on different processors, shared memory must be used.

Advisory 3.1.1.166 *HS USB OTG : Idle_req / idle_ack Mechanism Potentially Broken When Autoidle Is Enabled*

Revision(s) Affected: 3.1.2 and earlier

Details: Like any other module, upon assertion of idle_req from the PRCM, the USB OTG module, if it is inactive, will answer immediately by asserting idle_ack. Upon idle_req de-assertion, the module will de-assert idle_ack immediately too.

In the specific case where the USB OTG autoidle feature is enabled at module level (OTG_SYSCONFIG[0]:AUTOIDLE = 1), and where the idle_req is de-asserted very shortly after assertion (less than four L3 clock cycles), the module will never de-assert the idle_ack anymore.

Note:by default after an HW reset (at boot time or upon wakeup from OFF), the OTG_SYSCONFIG:AUTOIDLE bit is automatically set to 1.) This kind of short idle_req pulse can occur randomly during run time, when the CORE is going to retention and is awoken immediately for example. In this case, not de-asserting the idle_ack signal will stop the CORE entering retention again.

This issue is specific to the USB OTG module.

Workaround(s): Disable the autoidle feature at module level (OTG_SYSCONFIG[0]:AUTOIDLE = 0).

If the USB module is not used at all, disable its interface clock at PRCM level. By doing so it will always stay in idle mode.

Perform a dummy read of any USB OTG register. This will re-enable the clock inside the module, which will then de-assert its idle_ack signal.

Advisory 3.1.1.167 *HS USB OTG : OTG_SYSCONFIG:AUTOIDLE Bit Not Correctly Reset*

Revision(s) Affected: 3.1.2 and earlier

Details: Upon any source of HW reset (at boot time or when coming back from OFF mode for example), the reset value of the OTG_SYSCONFIG:AUTOIDLE bit is 1 (autoidle feature enabled). The behavior should be the same when applying a SW reset to the module through the OTG_SYSCONFIG:SOFTRESET bit. Because of an implementation bug, and because the AUTOIDLE and SOFTRESET bits are in the same OTG_SYSCONFIG register, the value defined for the AUTOIDLE bit while performing the SW reset gets over-written into it after SW reset is over.

For example : if the AUTOIDLE bit is 0 and a SW reset is performed. AUTOIDLE bit should be reset to 1.

- If the SW reset is performed by writing 0x2 in OTG_SYSCONFIG, AUTOIDLE will be un-expectedly reset to 0.
- If the SW reset is performed by writing 0x3 in OTG_SYSCONFIG, AUTOIDLE will be reset to 1 as expected.

Workaround(s): There is no workaround. When performing the SW reset, just write the AUTOIDLE bit at the value you expect it to be after reset.

Advisory 3.1.1.168 *IVA iVLCD Cannot Detect Errors*

Revision(s) Affected: 3.1.2 and earlier

Details: The VLD engine for H.264 decoder doesn't throw up the data parsing errors as and when they occur in a macro block, instead the errors are reported in header parsing after decoding few subsequent macro blocks.

This Issue is applicable only for the error streams, which normally occurs in streaming applications.

There is a delay in reporting the data parsing errors from the VLD engine side: On a given erroneous stream, The MSVC reference decoder fails at the macro block (N). On the same stream, iVLCD would not report the error so VLD will miss the error on macro block (N), continue decoding, then finally meet a contradiction in later macro block.

CAVLD decoder will miss specific error patterns, which is happening in this stream. While decoding coeff_token for which nC is 8 or higher, error checking for out-of-table is not activated by mistake.

Workaround(s): There is no S/W workaround. When an error is detected during the header parsing on the DSP, the info about the exact erroneous MB is not available. Hence the decoder has to do the error concealment for all the MB's in that slice on the DSP.

Advisory 3.1.1.169 *UART Not Asserting Its TX DMA Request When RX FIFO Is Not Empty*

Revision(s) Affected: 3.1.2 and earlier

Details: As long as the UART RX FIFO is not empty, the UART will not assert its TX DMA request. This means that in scenario using the DMA in both RX and TX, no UART transmission will occur until the RX fifo is emptied. This can cause deadlock situation if the software leaves some bytes in the RX FIFO.

Workaround(s): Software must make sure to always empty the RX FIFO when using the UART in full duplex mode.

Advisory 3.1.1.170 *IVA2 Does Not Wake-Up After It Goes to IDLE While DMA Request Is Still Asserted*

Revision(s) Affected: 3.1.2 and earlier

Details: IVA2 may go to Retention during a brief period between "a completion of an EDMA write transfer" and "a deassertion of DMA request (MCBSP_DMA for example)" . The deassertion of DMA request could be delayed by making consecutive accesses (Write or Read) to the low-speed peripherals (e.g., consecutive non-posted writes to GPTimerx from DSP or from MPU if an EDMA write transfer and this GPTimerx use same L4 thread, i.e. McBSP1 and GPTimer1) while EDMA is transferring some data.

The DMA request is then de-asserted while the IVA2 has gone to retention but WUGEN memorizes that the DMA request was asserted. When a new DMA is asserted while IVA is in IDLE state then the WUGEN cannot see it because it considers that the DMA is still asserted so it doesn't serve it and doesn't wake-up the IVA2.

Workaround(s): **Workaround 1**: Add a dummy Read access to the same peripheral for each EDMA write transfer for one EDMA request to the peripheral by using EDMA chaining function. When the deassertion of the EDMA request is delayed by other accesses (Write or Read) to the low-speed peripheral, WUGEN sets IVA SS in Standby State without waiting of the actual completion of the EDMA transfer.

If a dummy Read access to the same peripheral is added after each EDMA write transfer to the peripheral then WUGEN sets IVA SS in Standby State after the dummy Read access completion. At this time, the EDMA write transfer should be finished and also the EDMA request is deasserted absolutely. A race between IVA goes to idle and the deassertion of the EDMA request can be avoided.

Note: A dummy Read access to the peripheral should be added for each EDMA write transfer for one EDMA request. Therefore, there is possibility to decrease the performance.

Workaround 2: Enable DMATRUECOMPEN, ITCINTEN and TCINTEN. All EDMA write request is posted-write in default. Enabling DMATRUECOMPEN changes EDMA write request into a non-posted write. The write request which is affected by DMATRUECOMPEN can be enabled/disabled by using ITCINTEN and TCINTEN That is, if ITCINTEN=1 and DMATRUECOMPEN=1 then intermediate write events can be non-posted write requests. And also if TCINTEN=1 and DMATRUECOMPEN=1 then final write event can be a non-posted write request.

Therefore, even the service for EDMA write transfer is delayed by other accesses (Write or Read) to the low-speed peripheral, WUGEN sets IVA SS in Standby State with waiting the completion of this EDMA write transfer by enabling DMATRUECOMPEN, ITCINTEN and TCINTEN.

Note: TCC interrupt is common for both intermediate requests (enabled with ITCINTEN) and final request (enabled with TCINTEN). Therefore, there is no way to get only the interrupt for final request when both ITCENTEN and TCINTEN are enabled.

To generate the interrupt only with final event, additional use of PaRAMs is required. The method depends on the transfer synchronization dimension (that is, ASync or ABsync):

- Case1: In case of Async transfer and Bcnt !=1 or Ccnt !=1, and would like to get the interrupt only with the final event, 3 or more linked PaRAMs are needed.
- Case2: In case of ABsync transfer and Ccnt !=1, and would like to get the interrupt only with the final event, 2 or more linked PaRAM are needed.
- Case3: In other case (i.e. Async transfer and Bcnt =1, Async transfer and Ccnt =1, or ABsync transfer and Ccnt =1), there is no need to care about the PaRAM setting to get the interrupt of final event.

Advisory 3.1.1.173 *ISP: A-LAW Decompression Cannot be Performed in the PREVIEW Module*

Revision(s) Affected: 3.1.2 and earlier

Details: A-LAW decompression cannot be performed in the PREVIEW module if data comes from the CCDC (PRV_PCR[2] SOURCE = 0x0). This is a hardware defect in the PREVIEW module.

Workaround(s): **Workaround 1**: Do not use A-LAW decompression if data path is CCDC->PRV, but you can transfer RAW data from CCDC to PRV.

Workaround 2: A-LAW decompression in the PREVIEW module can be used by changing the data path to CCDC- >memory->PRV instead of CCDC->PRV.

Advisory 3.1.1.174 *SW Reset Done While ISP Processing Is Ongoing Can Cause OCP Protocol Violations*

Revision(s) Affected: 3.1.2 and earlier

Details: The ISP applies a SW reset (ISP_SYSCONFIG[1]:SOFT_RESET or CCP2_SYSCONFIG[1] SOFT_RESET or CSI2_SYSCONFIG[1] SOFT_RESET) immediately without checking if there is ongoing OCP traffic. Incomplete OCP transactions may be generated when the reset occurs in the middle of an L3 OCP burst. Incomplete OCP transactions stalls the L3 interconnect or cause a timeout condition.

Workaround(s): SW needs to first stop all ISP traffic before doing a SW reset. Idea is to block the OCP interface to avoid transfer are on going when reset is applied. To do that, MMU mapping error mechanism is used. MMU is programmed to generate an IRQ when an address which is unmapped in MMU table hit the MMU. Then all MMU tables are trashed so that IRQ will happen on the next OCP access (and access will block the OCP port). Once the IRQ happen, no transfer is on going by construction and reset can be applied.

Following sequence should be applied:

1. Configure the MMU in manual mode so that it requests translation entries from the CPU by triggering an IRQ.
2. When an access arrives, the MMU will stall the OCP port on a clean OCP boundary.
3. The CPU doesn't provide the translation entry: that ensure that the OCP traffic has been stopped.
4. Wait 1000 cycles to make sure all responses have been returned.
5. Issue a SW reset.

Advisory 3.1.1.175 *SBL_PCR [24]CCDCPRV_2_RSZ_OVF Goes High as Soon as RSZ_CNT[28]INPSRC Is Set to 1*

Revision(s) Affected: 3.1.2 and earlier

Details: The CCDCPRV_2_RSZ_OVF event is continuously triggered when the resizer is used in memory to memory mode. This event cannot be masked individually: all SBL events must be masked to prevent continue CPU interruption. That prevent proper monitoring of other SBL events (overflow conditions).

Workaround(s): Overflow events don't need to be monitored continuously. It is enough to monitor them once per frame to decide is a given frame can be used or if it must be dropped and eventually the ISP restarted. Monitoring should be done when then respective end of frame event has occurred (i.e. after a H3A-AF done check the overflow status of the H3A AF FIFO).

Advisory 3.1.1.176 *PRV Pixel Data Read from Memory When CCDC Video Port Is Active*

Revision(s) Affected: 3.1.2 and earlier

Details: The PRV data read from memory behaves incorrectly when it receives a end of frame pulse. This impacts use case where PRV processes a frame from memory while CCDC sends a new one to HIST or H3A.

It is not possible to have concurrently Memory -> PRV and CCDC-> HIST/H3A.

This issue will impact high quality still image capture with multiple and concurrent processing passes.

Workaround(s): Avoid that the CCDC frame ends while PRV is processing data (by synchronizing the modules). Or don't use the failing combination.

Advisory 3.1.1.177 *H3A Buffer Overrun*

Revision(s) Affected: 3.1.2 and earlier

Details: When an overflow occurs during the write of H3A module in H3A buffers in SDRAM, next write of H3A module does not start at the beginning of the buffer but where last overflow occurred. Some write on next buffers can happen and it corrupts next data.

That typically happens when the system is heavily loaded and the Resizer is used in upscaling mode. In fact, the Resizer has the highest priority inside the ISP and H3A has a low priority class. H3A may not gain access to the OCP port when the Resizer outputs data at high rates. That causes FIFO overflows and potentially the buffer overrun.

Workaround(s): H3A FIFO overflows and not expected in normal utilization. As long as no overflows there's no issue.

Workaround 1:

- Rework the use case to not have an overflow
- Increase H3A windows (Paxels' size) to decrease the bandwidth and avoid overflow

Workaround 2:

- Double each H3A buffer size allocated in SDRAM to prevent bad writes on next buffers.

Advisory 3.1.1.178 *Accesses to DDR Stall in SDRC After a Warm-reset*

Revision(s) Affected: 3.1.2 and earlier

Details: In some cases, user is not able to access DDR memory after warm-reset.

This situation occurs while the warm-reset happens during a read access to DDR memory. In that particular condition, DDR memory does not respond to a corrupted read command due to the warm reset occurrence but SDRC is waiting for read completion.

SDRC is not sensitive to the warm reset, but the interconnect is reset on the fly, thus causing a misalignment between SDRC logic, interconnect logic, and DDR memory state.

Root cause description: A corrupted read transaction is issued to a closed row: (address0, bank0) instead of the expected read access, violating protocol.

Failure signature: Once the failure occurs and system has restarted, memory content is not accessible. SDRC registers can be accessed successfully, until 1st access to memory location is performed. After 1st access to memory is done, SDRC is stuck.

Workaround(s): Steps to perform before a SW reset is triggered, if user needs to generate a SW reset and keep DDR memory content:

1. Set SDRC_POWER[SRFRONIDLEREQ]=1 //enable self-refresh on idle request
2. Set PRCM CM_ICLKEN1_CORE[SDRC]=0 //put SDRC in idle
3. Wait until PRCM CM_IDLEST1_CORE[SDRC]=1 //wait until SDRC goes to idle
4. Generate SW reset

Steps to perform after warm reset occurs:

if HW warm reset is the source, apply below steps before any accesses to SDRAM

1. Reset SMS and SDRC
   (a) set SMS_SYSCONFIG[1].SOFTRESET=0x1, wait until SMS_SYSSTATUS[0].RESETDONE = 0x1
   (b) set SDRC_SYSCONFIG[1].SOFTRESET=0x1, wait until SDRC_SYSSTATUS[0].RESETDONE = 0x1
2. Re-initialize SMS, SDRC and memory (if SW warm reset is the source, SDRAM can be accessed reliably with no additional operation since user ensure above sequence before applying the warm reset).

Note: DDR memory content is lost upon HW warm-reset (WDT, secure violation, …). SDRC_POWER[SRFRONRESET] value does not matter.

Advisory 3.1.1.180 *High-Speed USBOTG Short Packet Issue*

Revision(s) Affected: 3.1.2 and earlier

Details: OCP Write transactions to FIFO register gets killed internally causing short packet transmission.

The failure happens when the below events occur in the same cycle.

- RxPktRdy bit of the RX EP (with AutoClear enabled) is being cleared after DMA completes unloading FIFO.
- Data transactions happening in the USB bus.
- FIFO is being loaded by software through OCP slave.

This results in short packet transmission or USB data corruption.

Note: The MCU Reads to FIFO are not impacted by this bug.

Workaround(s): Use only DMA or only OCP Slave to load and unload FIFO. For better throughput good option will be to use DMA for up to 8EPs since the USB core has 8 DMA channels.

For application with more than 8 EPs, assign only RX EPs to MCU and DMA can be assigned either RX or TX EPs.

If MCU needs to load FIFO, it should check no DMA RX channel is active at that time. MCU should disable all the DMA RX channels and then proceed to loading FIFO.

Advisory 3.1.1.181 *Standard OTG Compliance Electrical Tests for HOST Mode Will Fail*

Revision(s) Affected: 3.1.2 and earlier

Details: Below is the sequence performed to observe the issue:

- Perform Enumeration as a HOST.
- Perform USB Reset, Read the device descriptor, check for OTG defined specific test-VID/PID.
- Set test mode according the found PID (Program the test mode register in the USBOTGHS.)

Now the DP-DM data lines are not in the expected states. Hence programming of TestMode J results in the Data-Lines permanently toggle between J and K state.

Workaround(s): Program the PHY OPMODE to 10 using the ulpi extended register access before programming the TESTMODE registers in HOST Mode.

Advisory 3.1.1.183 *GPMC Has Incorrect ECC Computation for 4-Bit BCH Mode*

Revision(s) Affected: 3.1.2 and earlier

Details: The GPMC supports 4- or 8-bit error correction BCH code. 4-bit error mode is using a wrong polynomial, as a result for this mode the GPMC will:

- On page write, generate incorrect ECC parity.
- On page read, generate an incorrect syndrome.

This bug prevents having correct error location.

Workaround(s): No workaround for this issue.

Advisory 3.1.1.185 *HS USB: ECHI and OHCI Controllers Cannot Work Concurrently*

Revision(s) Affected: 3.1.2 and earlier

Details: An issue in the USBHOST memory access arbiter prevents EHCI and OHCI Host Controllers from working simultaneously. As a result one cannot connect a HS and a FS USB devices on the USBHOST.

Workaround(s): No workaround exists for the generic use-case. For low-throughput requirement a SW arbitration scheme can be implemented.

Advisory 3.1.1.186 *MMC OCP Clock Not Gated When Thermal Sensor Is Used*

Revision(s) Affected: 3.1.2 and earlier

Details: Once enabled (CM_FCLKEN3_CORE.EN_TS = 0x1,), the PRCM provide the 32Khz clock to the clock tree made of Thermal sensor+ MMC1/2/3.

As soon as 32Khz clock is provided to MMC module then debouncing operation is started. Once debouncing is completed, then interface clock is ungated at module level and OCP clock is enabled internally to the MMC module (while MMC module is not used). This is creating unexpected over-consumption (100uA/MMC instance measured). The auto gating stays inefficient as long as module is not enabled.

Workaround(s): Each time thermal sensor is used, the MMC instances which are not used should be enabled then disabled (write CM_ICLKEN1_CORE.EN_MMCn = 0x1, CM_FCLKEN1_CORE.EN_MMCn = 0x1, wait until CM_IDLEST1_CORE.ST_MMCn = 0x0 then write CM_ICLKEN1_CORE.EN_MMCn = 0x0, CM_FCLKEN1_CORE.EN_MMCn = 0x0) to avoid over-consumption due to OCP logic being un-necessarily clocked.

Advisory 3.1.1.187 *Context Save Operation Randomly Failing for CONTROL_PAD_CONF_ETK14*

Revision(s) Affected: 3.1.2 and earlier

Details: Context save operation saves the pad configuration register contents to the scratch pad memory, that will be used at the context restore operation after OFF mode exit. Sometimes randomly after context save operation, CONTROL_PAD_CONF_ETK14 register is not saved into scratch pad memory. This register contains pad configuration information for 2 pads : etk_d14 and etk_d15. This is impacting only CONTROL_PAD_CONF_ETK14, all other padconf registers are saved and restored as expected.

The failure signature is this register not saved: the scratch pad memory keep value saved during previous OFF transition. It is not a register content corruption. When waking-up from OFF mode, CONTROL_PAD_CONF_ETK14 is restored with old value (or non-initialized RAM value for first time).

CaseA: Issue occurs randomly when CORE_L4_ICLK/4 (default) (CONTROL_PADCONF_OFF[2]. WKUPCTRLCLOCKDIV=0, default). Caused by the polling for context save completion. Issue does not occur if a delay is added just before SAVEDONE bit polling.

CaseB: Issue occurs always When CORE_L4_ICLK/2 (CONTROL_PADCONF_OFF[2]. WKUPCTRLCLOCKDIV= 1). No workaround available.

Root cause description:

Case A: Failure occurs when OCP port is accessed for context save operation at same time as last save access of the SAR mechanism is performed. The OCP access is delaying the completion of the save of CONTROL_PAD_CONF_ETK14 register. In that case SAR never ended and SAVEDONE bit is wrongly returned.

Case B: Wake-up control clock is CORE_L4_ICLK/2, SAR is wrongly completed before saving last context.

For caseA and caseB, result is current CONTROL_PAD_CONF_ETK14 is not saved in scratchpad @0x480025F8 and previous value is kept in scratchpad.

Workaround(s): Wake-up control clock = CORE_L4_ICLK/4 AND Do not access any SCM registers before context save completed following recommended sequence below.

1. Ts = 2*(1/core_l4_iclk_freq)* X*200 (X=4 when CONTROL_PADCONF_OFF[2]. WKUPCTRLCLOCKDIV= 0)
   - Example: 21.2us for 83MHz CORE_L4_ICLK
2. Enable padconf save operation: CONTROL_PADCONF_OFF[1]:STARTSAVE='1'
3. Wait for Ts Delay + 10% (additional step)
4. Poll SAVEDONEBIT='1'
5. Read padconf_OFF save for etk_d14/15 @0x480028A0 is equal to CONTROL_PADCONF_ETK_D14 @0x480025F8
   (a) If padconf_OFF save equal CONTROL_PADCONF_ETK_D14 then Sequence is completed successfully
   (b) If padconf_OFF save NOT equal to CONTROL_ and Go back to step 1 (restart a new save)

Advisory 3.1.1.188 *I2C4 Does Not Meet I2C Standard AC Timing in FS Mode*

Revision(s) Affected: 3.1.2 and earlier

Details: I2C4_SCL low period is fixed by hardware then cannot be modified by software. Due to IO cell influence, I2C4_SCL AC timing is shorter than expected. As a result the standard AC timing (SCL minimum low period) in FS mode is not met. Please see OMAP3530 Data Manual for exact I2C4 AC timings.

Workaround(s): There is no workaround for I2C4.

I2C4 is dedicated for SmartReflex and expected to connect with Power IC. Design review was done and concluded that there is no problem when OMAP is interfaced with TPS659xx.

Advisory 3.1.1.189 *I2C1 to 3 SCL Low Period Is Shorter in FS Mode*

Revision(s) Affected: 3.1.2 and earlier

Details: Due to IO cell influence, I2C1 to 3 SCL low period can be shorter than expected. As a result, I2C AC timing (SCL minimum low period) in FS mode may not meet the timing configured by software.

Workaround(s): I2C1 to 3, SCL low period is programmable and proper adjustments to the SCLL/HSSCLL values can avoid the issue.

Advisory 3.1.1.190 *Unexpected Warm Reset Assertion on HS Devices*

Revision(s) Affected: 3.1.2 and earlier

Details: For High Secure devices, CONTROL_SEC_CTRL[0:1] are described as OCO (One time Configurable after power on reset Only). In current implementation, these bits are reset when device is coming back from OFF mode. No restoration is performed by the HW when coming back from OFF resulting to CONTROL_SEC_CTRL[0:1] are back to reset value after a WU from OFF mode.

CONTROL_SEC_CTRL[0] is used to freeze secure watchdog count and default value for this bit is SWD NOT freezed.

If the secure Watchdog was frozen before an OFF mode transition, the SWD will be re-enabled again unexpectedly when the device is waking-up from OFF mode. The Secure Watchdog will generate a warm reset when the counter expires.

Note: Issue is systematic. Content of CONTROL_SEC_CTRL[0:1] is correctly maintained to their previous value during OFF period. The reset value is applied after WU when the isolation cells between power domains are released.

Workaround(s): CONTROL_SEC_CTRL[0:1] are not reliable and must not be used. The Secure Watchdog can be disabled by sw. Secure Watchdog is in the wakeup domain then it can be configured only once after power on reset and will kept disabled until next power on reset.

Recommended procedure is to use PA or PPA call to SW SEC_SERVICE_SEC_WD_DISABLE() once after power on reset.

Note: This WA is mandatory for HS devices not using Secure Watchdog functionality (goal it to keep the Secure Watchdog always disabled). No WA is needed in case Secure Watchdog is used in the system and reload of the Secure Watchdog is managed accordingly. This errata is not impacting Secure Watchdog capability.

Advisory 3.1.1.191 *Warm Reset Assertion Time When Warm Reset Happen During OFF Mode*

Revision(s) Affected: 3.1.2 and earlier

Details: Warm_reset is not asserted for same duration depending if it is a secure_Warm_reset (SSM, SWD) or a Global_Warm_reset (all other sources than secure). Global_warm_reset behavior is consistent with TRM description while Secure_Warm_reset has simplified logic (RSTTIME1 only is taken into account).

Impact: Warm_reset released before VDDs are stabilized which can lead to unpredictable behavior (mainly MPU crash). However secure_Warm_reset during OFF should never happen except in case of attack during OFF on a HS and system hanging in that particular context looks acceptable.

Expected behavior: Warm reset assertion duration is (OFFmodeSetupTime+RSTTIME1)*32Khz cycles. This formula is valid for all warm reset source except SWD reset and secure violation reset (SSM).

For WD reset and secure violation reset (SSM) PRM_VOLTSETUP2[15:0]:OFFMODESETUPTIME is ignored and formula is: RSTTIME1*32KHz.

Workaround(s): No WA needed since secure_Warm_reset during OFF is not happening with a correct configuration. In any case RSTTIME1 can be enlarged to match VDD ramp-up requirement.

Advisory 3.1.1.192 *Missed Dependency With McBSP External Clock Avoid Transition to OSWR*

Revision(s) Affected: 3.1.2 and earlier

Details: In case of PER domain, due to a missed dependency if McBSP external clock is the only enabled permanent clock of PER PD, then CORE PD is prevented to transition to idle and therefore into OSWR.

Such issue disappears if at least one other permanent clock of PER PD is enabled.

Note: A permanent clock is a functional clock that can stay active while the corresponding entity managing it, CM, can go into idle. That is, a permanent clock can stay running while CM internal FSMs are clock-gated and even when CORE domain goes to retention.

For WD reset and secure violation reset (SSM) PRM_VOLTSETUP2[15:0]:OFFMODESETUPTIME is ignored and formula is: RSTTIME1*32KHz.

Workaround(s): Enable a permanent clock in PER domain when McBSP external clock is used.

To minimize power consumption impact, a 32KHz clock should be selected and enabled (fclken = ‘1’) without enabling the module (iclken = ‘0’). In such way, only small 32KHz clock tree contributes to power consumption increasing.

Advisory 3.1.1.193 *MPU Cannot Exit From Standby*

Revision(s) Affected: 3.1.2 and earlier

Details: MPU interrupt controller is not able to sort the input interrupt under idlereq pulse application. The sequence which creates this situation is:

1. When there is a pulse of idlereq (one or two clock cycles) applied after coming out of idle state.
2. There is change in input interrupt.

Impact: The input interrupt can not be sorted until the internal OCP clock starts running. Interrupt to CPU will be delayed till the OCP clock starts running. The OCP clock can start running if Another Idlereq pulse greater than two clock cycles.

Workaround(s): Disabling auto-gating (INTCPS_SYSCONFIG[0]:AUTOIDLE=0) feature which will allow the change in idlereq to be sampled. This can be done right before executing the idle instruction to avoid power consumption impact.

Advisory 3.1.1.194 *sDMA FIFO Draining Does Not Finish*

Revision(s) Affected: 3.1.2 and earlier

Details: There is an issue when sDMA channel is disabled on the fly, sDMA enters standby even through FIFO Drain is still in progress. SW WA is to put sDMA in NoStandby before a logical channel is disabled, then put it back to SmartStandby after the channel finishes FIFO draining. The issue only happens when FIFO draining is used and sDMA is configured SRC sync, BufferingEnabled and SmartStandby.

Workaround(s): Put sDMA in NoStandby before a logical channel is disabled, then put it back to SmartStandby right after the channel finishes FIFO draining. This issue can be avoided when one of the conditions (sDMA FIFO drain function enabling, SmartStandby, or On-the-fly channel disabling) is removed.

Advisory 3.1.1.195 *HSUSB Interoperability Issue With SMSC USB3320 PHY*

Revision(s) Affected: 3.1.2 and earlier

Details: After suspend sequence, USB3320 USB PHY goes correctly in low-power mode:

- DP Line goes High and DM line remains Low (J state)
- Rbias Voltage = 0 V

Whereas OMAP HOST controller exit from suspend mode (while it is expected to keep in low power mode).

OMAP Host state (exited from low power mode) is inconsistent with PHY state (low power mode) resulting in a lockup situation.

Resuming the port has no effect as HOST controller has already exited from low-power mode.

Root cause: Delay in assertion of DIR causes USBHOST ULPI interface to exit ULPI Low Power mode. USB3320 USB PHY assert DIR signal 3 clock cycle after STP signal is de-asserted.

Workaround(s): There is no workaround.

Advisory 3.1.1.196 *IVA2 Does Not Wake-up After It Goes to IDLE While an Interrupt Line Is Not Cleared*

Revision(s) Affected: 3.1.2 and earlier

Details: IVA is allowed to transition to sleep as soon as the last pending IRQ was handled and the clear command was sent. If latency is added to clear the interrupt source due to system activity, then IVA2 will transition to sleep before the interrupt source is cleared, meaning that the interrupt line is not de-asserted when IVA goes to sleep but will be released later on.

In that particular condition (interrupt line active when IVA goes to sleep), then this interrupt line is no longer able to generate a WU event to the IVA. The IVA will ignore the activity on that line until it is woken-up by another source.

Workaround(s): Whenever IVA clears interrupt, IVA needs to read back the same register. This ensures the interrupt is always cleared before IDLE entry. This WA is implemented in TI DSP Bridge for GP-Timer and mailbox interrupt. Similar sequence needs to be applied by user in the ISR to all other possible interrupt sources.

Advisory 3.1.1.197 *POWERVR SGX™: MMU Lockup on Multiple Page Miss*

Revision(s) Affected: 3.1.2 and earlier

Details: The POWERVR SGX Bus interface contains an MMU address translation function which works on 4KB page allocations. The page table entries are setup and located in external DRAM memory and are fetched on demand as requests are made, there is a cache within the POWERVR SGX which keeps the most recently used entries. When an internal requester makes a request in virtual space the address is tested with the cached entries.

If there is a hit then the physical high order address bits are returned and combined with the lower address bits of the virtual address bits to form the external physical address access.

If there is a cache miss then the MMU must make an external fetch to memory to update the on chip cache, when this memory fetch completes the original memory access can proceed.

The POWERVR SGX has 7 parallel memory request sources that feed into the MMU translation logic, so at any one time there can be a maximum of 7 parallel request sources that can all exhibit a page miss at the same time. The MMU contains a 3 bit counter to keep track of cache-miss requests (7 requesters).

However there are scenarios where the request sources generate multiple cache miss-requests causing the 3 bit counter to overflow and miss service requests. Ultimately multiple passes through normal and miss phases together with various contributing memory latency can result in the counter not returning to zero and the MMU will lockup in the MMU miss phase. This will block all further accesses and the core will lock up.

This could result in lock up of the POWERVR SGX image processing pipeline and may result in system hang.

Workaround(s): If the core is forced to an idle state (in terms of TA and 3D processing) before a cache invalidate is issued by setting , the occurrence of the lockup is substantially reduced.

Advisory 3.1.1.198 *HS USB OTG: ULPI LINK Possibly Sticks After DPLL3 SW Reset if USB Cable Stays Connected*

Revision(s) Affected: 3.1.2 and earlier

Details: When the USB OTG is physically connected to a host with a cable, and a software cold reset is performed (PRM_RSTCTRL[2] RST_DPLL3), then the ULPI link with the PHY may get stuck. The PHY state machines go into an unknown state.

Root cause: During the relock time of the DPLL3 following the cold reset, the system clock (or system clock divided by 2) is the L3 clock provided to all modules (including the USB OTG module).

At the same moment the USB OTG HW automatically sends ULPI commands to the PHY.

The conjunction of sending ULPI commands while the DPLL is not yet locked can potentially end up in timing violations in the PHY.

The probability for this failure to occur is very low for the following reasons:

- A SW cold reset must be performed, which is not supposed to happen during run time.
- The device embedding OMAP needs to be connected with a cable to a host when the cold reset happens.
- Even if these conditions are met, experiments have shown that the timing violation will likely occur once every several thousand cold resets.

Workaround(s): Turning off USB_V1P8 LDO before issuing the software cold reset will eliminate the issue by turning off the ULPI clock at boot

Advisory 3.1.1.199 *USB Host EHCI May Stall When Exiting Smart-standby Mode*

Revision(s) Affected: 3.1.2 and earlier

Details: When the USBHOST module is set to smart-standby mode, and it is ready to enter the standby state (i.e., all ports are supported and all attached devices are in suspend mode), it may incorrectly assert the Mstandby signal too early while there are ongoing residual OCP transactions.

If this condition occurs, the internal state machine may go to an undefined state and the USB link may be stuck upon the next resume.

Workaround(s): The software should explicitly disable (pause) the USB HOST OCP initiator activity by disabling the schedules (USBCMD[5]ASE = 0, USBCMD[4]PSE=0) just before suspending the connected ports and restoring their state after the USBHOST has entered smart-standby state.

Software workaround sequence:

- Read USBCMD register and save it;
- Clear USBCMD[5]ASE and USBCMD[4]PSE bits;
- Wait for the USBSTS[15]ASS and USBSTS[14]PSS bits to reflect this change;
- Suspend the connected ports;
- Wait for the ports suspend to take effect (~3ms);
- Restore the USBCMD register

Advisory 3.1.1.200 *USB Host EHCI May Stall When Running High Peak-Bandwidth Demanding Use Cases*

Revision(s) Affected: 3.1.2 and earlier

Details:

The USB host module is AHB native. Therefore, there is an AHB2OCP bridge allowing to connect it to the OCP L3 interconnect.

Both AHB and OCP masters are able to generate single accesses (R/W) as well as bursts, depending on the configuration, as well as address ranges.

Under some specific L3 latency conditions, when a USB host write is followed by a USB host single read (not burst read), then the read can be lost in the AHB/OCP bridge. When this happens, the internal state machines of the module go into an undefined state and the EHCI stalls; ongoing transfers are stopped, and new transfers cannot be scheduled anymore.

This situation will only occur when both following conditions happen simultaneously:

- The module is performing a write followed by a single AHB read.
  - This can happen when processing control messages (Transfer descriptions in memory are updated (written) by the host when being processed and an 8-byte command is fetched by the host (2 single AHB reads))
  - This can also happen for any OUT transfer (bulk, isochronous, interrupt) depending on data payload size and maximum Tx packet size parameter (TxMaxP)
- Congestion peaks occur in the system, generating back pressure at the host boundary with the interconnect
  - This can typically happen when high priority initiators like Display Subsystem and/or Camera are running heavy use cases in parallel of USB transfers.

This issue does not impact IN transfers.

Workaround(s):

Define ((payload size) modulo (MaxP size)) >=16 bytes. If ((payload size) modulo (MaxP size))< 16 bytes, then dummy data can be added to the buffer in order to achieve ((payload size)modulo (MaxP size)) >= 16 bytes.

However this is not always possible, typically for control transfers, for which payload size is fixed to 8 bytes.

In this case, it is only possible to reduce the failure occurrence by:

- removing un-necessary control commands (like get_device_state upon suspend exit)
- avoiding enumeration during peak-bandwidth demanding use cases

Once the issue has occurred, the only way to recover will be to reset the USB host module and re-enumerate.

Advisory 3.1.1.201 *USB OTG DMA May Stall When Entering Standby Mode*

Revision(s) Affected: 3.1.2 and earlier

Details: If the OTG module is in SmartStandby Mode (OTG_SYSCONFIG.MIDLEMODE = 0x2) when an OTG DMA channel is enabled (DMA_CNTL.DMA_EN) very near a USB SUSPEND condition, the MStandby signal may assert while there is residual OCP traffic initiated by the DMA. This illegal traffic causes the OTG DMA to stall.

Workaround(s): Ensure the module cannot enter standby mode while DMA transfers are active.

Advisory 3.1.1.202 *DPLL3 in Manual Lock Mode Cannot be Used When CORE Goes to OSWR or OFF State*

Revision(s) Affected: 3.1.2 and earlier

Details: This occurs when the DPLL3 is in Manual Lock mode (CM_AUTOIDLE_PLL[AUTO_CORE_DPLL] = 0).

Due to a wrong isolation cell on the dpll3clkon signal between CM and PRM (TIE-LOW implemented instead of ISO_LATCH), the DPLL3 Lock state signal is not taken into account by the PRM when CORE goes OSWR (Open Switch Retention) or to the OFF state. Consequently, the PRM will consider that the DPLL3 is not locked and the DPLL3 input clock will be stopped.

Impacts:

- CORE OSWR mode: DPLL3 cannot be kept Locked when CORE OSWR state if DPLL3 is set in Manual Lock mode
- CORE OFF mode: Architecture will not support a transition to OFF mode if DPLL3 is set in Manual Lock mode

Workaround(s): CORE OSWR and CORE OFF mode: DPLL3 must be set to Automatic mode when CORE goes to CORE OSWR or OFF state. However, DPLL3 will need to relock after Wakeup.

Advisory 3.1.1.203 *PRCM DPLL Control FSM Removes SDRC_IDLEREQ Before DPLL3 Locks*

Revision(s) Affected: 3.1.2 and earlier

Details: When internal SWAKEUP event occurs while CORE DPLL is going to bypass, PRCM may report that DPLL3 is locked when it is not. This is due to a timing path of an internal signal that is not fitting into one L3 clock cycle. In that case, PRCM DPLL control FSM deasserts the SDRC_IDLEREQ signal before DPLL3 Lock state is set. The consequence is that SDRC is released from IDLE with bypass clock (which is too low), instead of the locked frequency. DLL may or may not lock based on Process Voltage Temperature conditions.

This issue is seen when DPLL3 Automatic mode is enabled: CM_AUTOIDLE_PLL[AUTO_CORE_DPLL] = 1 or = 5.

Workaround(s): **Case 1**: Initialization (Device boot up)

- Disable DPLL3 automatic mode by default (CM_AUTOIDLE_PLL[AUTO_CORE_DPLL] = 0)
- This issue will not be faced since DPLL3 is always locked

Case 2: Before CORE Voltage Domain Sleep Trasition to RETENTION or OFF mode

After Disabling Smart Reflex:

- Reduce DPLL3 M2 Frequency to get L3 running at OPP2 Frequency (by changing M2 Divider value). This is increasing the period duration of one L3 clock cycle.
- Increase CORE Voltage to 1.2V. This is reducing the timing duration of the critical path signal, which will now fit to one L3 clock cycle.
- Enable DPLL3 Automatic Stop mode. This will ensure proper transition to RETENTION or OFF mode.

In summary, L3 = OPP2 + VDD2 = 1.2V combination must be used:

- If OPP3 (L3=166 MHz, VDD2=1.15V):
  - Lower the frequency to 83MHz
  - Increase CORE Voltage to 1.2V
  - Enable DPLL3 Automatic Stop mode
- If OPP2 (L3=83MHz, VDD2=1.05V):
  - Keep the frequency as it is (83MHz)
  - Increase CORE Voltage to 1.2V
  - Enable DPLL3 Automatic Stop mode

Case 3: After CORE Voltage Domain Wakeup Transition from RETENTION or OFF mode

Before Enabling Smart Reflex:

- Disable DPLL3 Automatic mode
- Restore previous DPLL3 M2 Frequency and CORE Voltage values

Notes for all cases:

- Case 2 and Case 3 must be executed if there is a Voltage Domain transition. These cases cannot be executed if only a Power Domain transition is targeted.
- Due to another bug (See), the following two scenarios must to taken into account:
  - Target state is OSWR and voltage transitioning is happening: Case 2 and Case 3 can be applied.
  - Target state is OSWR and voltage transitioning does not happen: Case 2 and Case 3 cannot be applied; Target state must be changed from OSWR to CSWR.

Advisory 3.1.1.204 *PER Domain Reset Issue After Domain-OFF/OSWR Wakeup*

Revision(s) Affected: 3.1.2 and earlier

Details: After Peripheral Power Domain (PER-PD) is woken up from the OFF/OSWR state while Core Power Domain (CORE-PD) is kept active, write or read access to some internal memory (UART3 FIFO and sidetone memory inside McBSP2/3) does not work correctly.

This leads to a corruption of transmit or receive data of UART3, or output from McBSP2/3 sidetone modules. Other logics behave properly.

The cause of the issue is that memory control logic of UART3 FIFO and McBSP2/3 sidetone memory are not reset and remain uninitialized when PER-PD is woken up from OFF or OSWR state. The logics are properly reset when CORE-PD is coming back from OFF/OSWR state. They also get reset by warm-reset, but are not reset by a module level software reset.

Workaround(s): Do not allow PER-PD to go to OSWR/OFF as long as Core-PD is not switched to OFF/OSWR.

When both CORE-PD and PER-PD goes into OSWR/OFF, PER-PD should be brought to active before CORE-PD. This can be done by configuring a wakeup dependency between PER-PD and WKUP-PD (PM_WKDEP_PER.EN_WKUP = 0x1) so that CORE-PD and PER-PD will wake up at the same time.

Even with the above configuration, there is a small possibility that on transition to OFF mode, a wakeup event is generated at a time when PER-PD entered OFF/OSWR but CORE-PD is not. This timing window is very small (about 8 µs at most), but to ensure the correct operation in this corner case, this procedure is recommended. After waking up from CORE OFF/OSWR configuration:

- Check this condition to see if the previous Core-PD transition to OFF/OSWR was aborted (thus not actually reached OFF/OSWR state)
  PM_PREPWSTST_PER[1:0].LASTPOWERSTATEENTERED = 0x0 (PER domain was previously OFF)
  PM_PREPWSTST_CORE[1:0].LASTPOWERSTATEENTERED = 0x3 (CORE domain was previously ON)
- If this corner case is detected, check the UART3 FIFO and/or McBSP2/3 sidetone functionally using their internal loopback features.
  For UART3 the internal loop back mode can be enabled by MCR_REG[4].LOOPBACK_EN
  For McBSP2/3 sidetone feature, digital loop back can be used. Two different words should be used (typically 0x55.. and 0xAA..) in the loop back test to confirm correct behaviour
- If error is detected by the above loopback tests, execute one of recovery sequences:
  - Generate warm-reset
  - Put the Core-PD and PER-PD to OSWR/OFF

Advisory 3.1.1.205 *McBSP Used in Slave Mode Can Create a Dead Lock Situation When Doing Power Management*

Revision(s) Affected: 3.1.2 and earlier

Details: When using McBSP in Slave mode and doing Power management (McBSP going to IDLE state), if the external CLKX clock is not provided by external peripheral component, McBSP cannot exit IDLE state. The consequence is that McBSP registers cannot be accessed anymore (except MCBSPLP_STATUS_REG[CLKMUXSTATUS] register).

There is similar limitation on the CLKR when module is configured as a receiver.

Workaround(s): These workarounds are still valid by replacing CLKX by CLKR. There are three possible workarounds to avoid this situation.

Workaround 1:

Don't use the power management features of McBSP (SIDLEMODE field in MCBSPLP_SYSCONFIG_REG register set to NO-IDLE all the time).

The impact of this workaround is that as McBSP transition to IDLE is avoided (SIDLEMODE=NO-IDLE), PRCM cannot transition the chip to Low Power mode.

Workaround 2:

Keep external CLKX clock always running during application (for example during an Audio Playback):

- At the beginning of the application (e.g. Audio Playback), keep CLKX always running by setting the external peripheral component register accordingly
- At the end of the application (e.g. Audio Playback), unload McBSP2 drivers and set XRST bit from SPCR2 to 0. Then stop CLKX in software (by setting external peripheral component register accordingly).

The impact of this workaround is that the serial clock is kept active during whole application (e.g. Audio Playback).

Workaround 3:

If SIDLEMODE is used (at some value other than NO-IDLE), ensure McBSP registers can be accessed by reading/polling the MCBSPLP_STATUS_REG[CLKMUXSTATUS] register. If registers can not be accessed (CLKMUXSTATUS=1), take corrective actions in software, such as (re)enabling CLKX.

Advisory 3.1.1.206 *DPLL3 Bypass Condition Does Not Consider State of SGX FCLK*

Revision(s) Affected: 3.1.2 and earlier

Details: The state of SGX FCLK is not considered in a DPLL3 bypass condition, so DPLL3 can be put in bypass mode even if SGX FCLK is still active, causing an unexpected SGX FCLK frequency drop. There is no issue when the SGX FCLK source is DPLL4 (Selected through CM_CLKSEL_SGX).

Workaround(s): There are two possible workarounds. The recommendation is to use Workaround #1, as it is more efficient in term of power consumption.

Workaround #1: Clear the CM_AUTOIDLE_PLL[AUTO_CORE_DPLL] bit (that is, Auto Control Disabled) when SGX FCLK is enabled (CM_FCLKEN_SGX[EN_SGX]='1') and set it back to the expected automatic PRCM control configuration when SGX FCLK is disabled. With this workaround, L3 gating will still be possible while DPLL3 is kept in locked state.

Workaround #2: DPLL3 can be forced to stay in a locked state by enabling L3 interface clock for SGX (CM_ICLKEN_SGX[EN_SGX]='1'). However, this will be less efficient in terms of power consumption as the L3 clock tree is enabled.

Advisory 3.1.1.207 *I2C: In SCCB Mode, Under Specific Conditions, the Module Might Hold the Bus by Keeping SCL Low*

Revision(s) Affected: 3.1.2 and earlier

Details: In SCCB mode, if the XRDY/RRDY are not served during the address phase, the module starts to hold the bus by keeping SCL low (FIFO empty or full). Then, after serving these interrupts, the module does not continue the current transfer and blocks in this state.

This bug appears only in applications where the module is used in SCCB mode. The bug is not appearing at all if interrupts are served before the address phase starts quickly enough to avoid entering this context.

Workaround(s): None.

Advisory 3.1.1.208 *I2C: Spurious Wakeup Event When sysclk Period is Higher Than ocpclk Period*

Revision(s) Affected: 3.1.2 and earlier

Details: Expected behavior: The scenario occurs when the IDLEREQ is sent after serving an interrupt/DMA and system clock period is higher than ocp clock period (at least two times higher) and prescaler value > 4 (internal clock is sysclk/(PSC+1)). If there is no other interrupt or DMA request, the module enters the IDLE state and exits this idle with a wakeup generated from some event.

Observed behavior: A spurious wakeup is asserted and maintained while no IRQ/DMA request is generated once the module is brought out of idle.

Workaround(s): If sysclk period and ocpclk period values are close, the prescaler PSC value of the I2C_PSC register should be programmed to any value less than or equal to 4. For higher values, bug may happen. There is no issue for prescaler value PSC = 1 (i.e. High Speed).

Advisory 3.1.1.209 *I2C: Wrong Behavior When a Data With MSB 0 is Put in the FIFO Before a Ttansfer with SBLOCK is Started*

Revision(s) Affected: 3.1.2 and earlier

Details: Expected behavior: Before starting a new transfer from another I2C device, one byte of data is written to TX FIFO. The module is addressed on a 10 bit address as a Slave transmitter (one of his addresses) and I2C clock blocking is enabled. After a repeated start condition SBLOCK is activated again for the second part of the address.

Observed behavior: Given the addressing and SBLOCK conditions defined above, if the data put in FIFO has its MSB 0, the module makes a glitch on the SDA bus on the eighth bit (SDA is set to 0 for a short period) which can be interpreted as an illegal start/stop condition.

Workaround(s): The scenario described is a corner case and it may very seldom happen in applications. To avoid the situation, before a transfer is started on the I2C bus, all interrupts should have been cleared (part of the guideline given in the spec), or when I2C is a transmitter, no data should be placed in the FIFO without receiving the request to do so from the slave.

Advisory 3.1.1.210 *I2C: After an Arbitration Lost the Module Starts Incorrectly the Next Transfer*

Revision(s) Affected: 3.1.2 and earlier

Details: Expected behavior: After a transfer on the I2C bus, where the module lost the arbitration during the address phase, a new transfer as a Master is programmed in I2C_CON by setting MST bit to 1, having the start bit STT in the I2C_CON register still unset. The STT bit can be set after a significant delay to point to the moment in which the transfer starts on the I2C bus. The module should only start the transfer on I2C after setting this STT start bit in I2C_CON.

Observed behavior: The module starts the transfer on I2C before setting STT and immediately after setting the MST bit in the I2C_CON to 1.

Workaround(s): The MST and STT bits inside I2C_CON should be set to 1 at the same moment (avoid setting the MST to 1 while STT = 0).

Advisory 3.1.1.211 *4-cycle Saturating .M Unit Instructions May Mask 2-cycle Saturating. M Unit Instruction Saturation Bit Update*

Revision(s) Affected: 3.1.2 and earlier

Details: If a 4-cycle instruction that can saturate is writing back its result in the same cycle as a 2-cycle instruction that can saturate, only the saturation condition for the 4-cycle instruction is recorded in the CSR.SAT bit and SSR.M1 or SSR.M2 control registers. In other words, if the 4-cycle instruction does not saturate, but the 2-cycle instruction does saturate, the fact that the 2-cycle instruction saturated is lost and is not recorded in the CSR.SAT and SSR.Mx bit-fields.

The list of 4-cycle instructions that can mask the 2-cycle saturation condition is:

- SMPY32
- CMPYR
- CMPYR1
- DDOTPH2R
- DDOTPL2R

The list of 2 cycle instructions that can set the saturation flags are:

- SMPY
- SMPYLH
- SMPYHL
- SSHVL
- SSHVR

Note that the 2 cycle instruction must be started 2 cycles after the 4 cycle instruction in order for the conflict situation to occur. For example:
CMPYR .M1 A0,A1,A2
NOP
SSHVL .M1 A8,A9,A10

If the CMPYR instruction does not saturate, the saturation status of the SSHVL instruction will not be recorded. Note also that the 2 cycle and 4 cycle instructions must be issued on the same .M unit for this interference to occur. If the SSHVL instruction were on the .M2 unit, the saturation bit would be recorded correctly.

Workaround(s): Workaround is to make sure that no code has scheduled any of the instructions from the 4-cycle list to write back in the same cycle as the instructions in the 2-cycle list.

Advisory 3.1.1.212 *SPLOOP CPU Cross-Path Stall*

Revision(s) Affected: 3.1.2 and earlier

Details: This erratum is affecting IVA2 c64x core. When these 3 rules are met, a stall is seen when a SPKERNEL instruction is executed.

- Cross-path instruction rule: An instruction reading a register via the cross path in the first cycle after SPKERNEL instruction.
- Data dependence rule: An instruction in the SPLOOP body that writes to the above cross-path read register. This instruction can be anywhere in the SPLOOP body.
- Functional unit rule: No instruction in parallel with the SPKERNEL instruction that uses the same functional unit as the cross-path read instruction mentioned in the cross-path instruction rule above.

This results in a 1 CPU cycle stall for each iteration of the loop.

3 examples of code that are affected by this issue:

Example 1:

```
SPLOOP 1
MV .S1 A0, A1: stalls every iteration due to MV after loop
SPKERNEL
MV .S2X A1, B2
```

Example 2:

```
SPLOOP 14
MV .S1 A0, A1: stalls every iteration due to MV after loop
NOP 9
NOP 9
NOP 9
NOP 9
SPKERNEL
MV .S2X A1, B2
```

Example 3:

```
SPLOOP 1
MV .S1 A0, A1: stalls every iteration due to MV after loop
SPKERNEL
||NEG .L2 B3, B4: qualifies for rule 3, function unit rule
MV .S2X A1, B2
```

The following 3 examples are not affected by this issue:

Example 1: No stalls, no cross path in instruction after SPKERNEL

```
SPLOOP 1
MV .S1 A0, A1
SPKERNEL
MV.S1 A1, A2
```

Example 2: No stalls, A1 not written to in loop body

```
SPLOOP 1
MV .S1 A0, A2
SPKERNEL
MV .S2X A1, B2
```

Example 3: No stalls, instruction in parallel with SPKERNEL prevents bug since it is in the same unit as the instruction that uses the cross-path.

```
SPLOOP 1
MV .S1 A0, A1
SPKERNEL
```

```
||NEG .S2 B3, B4: masks the bug
MV .S2X A1, B2
```

Workaround(s): Unfortunately, the way SPLOOP code is scheduled is beyond the application developer's control and is controlled by the compiler. Therefore, there are no direct workarounds for non-assembly source code. The fix is included in these compiler releases (versions that are listed refer to c6x compiler within Code Composer Studio):

- 6.0.25 or later
- 6.1.15 or later
- 7.0.2 or later
- 7.1.0B2 or later
- 7.2.0A or later

3 Silicon Revision 3.1 Usage Notes and Known Design Exceptions to Functional Specifications

3.1 Usage Notes for Silicon Revision 3.1

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 silicon revisions.

Note: The peripherals supported on the various OMAP35x Application Processors are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Application Processors, see the device-specific data manuals.

Silicon revision 3.1 applicable usage notes have been found on a later silicon revision. For more details, see Section 2.1, *Usage Notes for Silicon Revision 3.1.2*.

3.2 Silicon Revision 3.1 Known Design Exceptions to Functional Specifications

Some silicon revision 3.1 applicable advisories have been found on a later silicon revision. For more details, see Section 2 *Silicon Revision 3.1.2 Known Design Exceptions to Functional Specifications*.

Table 5. Silicon Revision 3.1 Advisory List

Advisory 3.1.1.165 *Unexpected Cold-Reset Is Generated When Device Is Coming Back from OFF Mode*

Revision(s) Affected: 3.1 and earlier

Details:

CASE A: When entering OFF_MODE, if a wake-interrupt is received in a specific window (3 sysClk cycles wide), a cold reset could be generated.

CASE B: When device goes to OFF mode without hitting caseA already described, a cold reset could be generated while device is waking-up (precisely during Efuse sensing). The timing of the WU event is not playing any role on the occurrence of the issue (No case-B window). Case-B issue is related only with the WU from OFF sequence which is a pure HW sequence.

During wake-up from OFF-MODE, PRCM (PRM logic) is not protected from efuse bits toggling during EFUSE-bit shifting.

Exposed window duration:

- 600 last bits of the fuse chain in CASE B
- Entire efuse chain (~2500 short sensitive window)

The failure depends on Fuse content. Any transition '1-'to-'0' or ''0'-to-'1' during efuse shifting can cause a cold reset then a downgrade of OMAP35xx depending on value detected during the shifting.

The failure also depends on SysClk frequency. The sysClk frequency defines what bits of the fuse chain will be detected.

Workaround(s):

There is no workaround for CASE A, but probability of occurrence is very low because the interrupt should happen during a 3-system clock cycles window while the system is transitioning to OFF mode.

The signature of the occurrence of CASE A is OMAP3530/25 rebooting in 3515/03 mode. This can be detected through the CM_CLKSEL_SGX register which becomes read only. In this case, the system can only recover through a power on reset.

The workaround for case B consists of enabling the TLL Save and Restore mechanism: set PM_PWSTCRTL_CORE:SAVEANDRESTORE to 0x1. Setting this bit is sufficient; there is no need to enable OMAP3530 TLL or USB module clocks either. Enabling this bit will change the HW sequence of events executed upon wake up from OFF mode, thus avoiding the condition for this issue.

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

4.1 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 silicon revisions.

Note: The peripherals supported on the various OMAP35x Application Processors are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Application Processors, see the device-specific data manuals.

Some silicon revision 3.0 applicable usage notes have been found on a later silicon revision. For more details, see Section 2.1, *Usage Notes for Silicon Revision 3.1*.

4.1.1 Display Controller Subsystem (DSS): Limitations Exist When Generating Horizontal and Vertical Timings

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, the display controller registers that control horizontal and vertical output timings (HBP, HFP, HSW, etc.) are not flexible enough to generate waveforms/timings required by some video display standards such as those of the Video Electronics Standards Association (VESA). Table 6 below describes which register fields are used to control the vertical and horizontal output timings, the current register field widths, and the required register field widths.

Table 6. Display Controller Horizontal and Vertical Timing Control Register Fields

Field Name	DSS Register Field	Field Width (Bits)	Required Width (Bits)
HBP	DISPC_TIMING_H[27:20]	8	12
HFP	DISPC_TIMING_H[15:8]	8	12
HSW	DISPC_TIMING_H[5:0]	6	8
VPB	DISPC_TIMING_V[27:20]	8	12
VFP	DISPC_TIMING_V[15:8]	8	12
VSW	DISPC_TIMING_V[5:0]	6	8

4.1.2 Camera ISP: IIR Filters in Auto Focus (AF) Engine Should Only Be Used for Auto-Focus

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, the AF engine IIR filters should only be used for auto-focus purposes. Any other use for the IIR filters is not supported.

4.1.3 High-Speed USB Host Subsystem: Some Limitations Exist When Connecting to External Devices

On OMAP35x Applications Processor silicon revisions 3.0 and earlier, as shown in Figure 11, the OMAP35x device includes a high-speed universal serial bus (USB) OTG controller and a high-speed USB host subsystem.

Note: USB Port 3 is not available on CUS package.

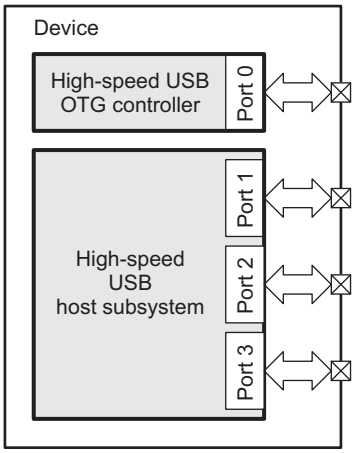


Figure 11. OMAP35x USB Modules

The high-speed USB OTG controller supports a single USB port which uses a UTMI low-pin interface (ULPI) to connect to an off-chip transceiver (12-pin/8-bit single-data rate mode). As shown in Figure 12, USB Port 0 can be connected to the USB 2.0 PHY included in OMAP35x companion chips, e.g. TPS69xxx. Alternatively, USB Port 0 can be connected to discrete USB 2.0 PHYs which include a UTMI low-pin interface (ULPI) and are capable if sourcing an output clock.

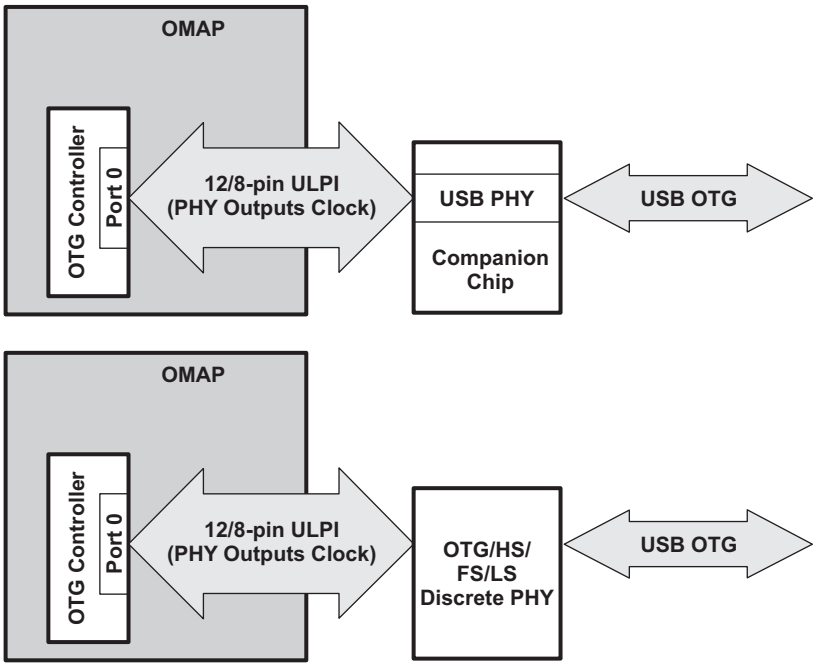


Figure 12. Typical Uses for USB Port 0

As shown in Figure 13, the high-speed universal serial bus (USB) host subsystem supports up to three USB ports, each one of which can owned by one of two controllers:

- The EHCI controller, based on the Enhanced Host Controller Interface (EHCI) specification for USB Release 1.0, is in charge of high-speed traffic (480M bit/s) over either a UTMI port or a UTMI low-pin interface (ULPI) port.
- The OHCI controller, based on the Open Host Controller Interface (OHCI) specification for USB Release 1.0a, is in charge of full-speed/low-speed traffic (12/1.5M bit/s, respectively) over a serial interface.

Both the OHCI and EHCI controllers can operate in parallel. Each of the three USB ports is owned by exactly one of the controllers at any time. Also, when a controller uses the UTMI port, the USB TLL block converts that port to a ULPI transceiver-less link logic (TLL) format.

Please note the following functional limitations when using the high-speed USB host subsystem:

- On silicon revision 3.0 and earlier, if one port is configured in the ULPI mode, then all other ports must use the same configuration. Therefore, the ports must be configured high-speed mode or full-speed/low-speed mode.
- USB Port 3 cannot operate in ULPI mode; it can only operate in serial or ULPI TLL mode. Furthermore, USB Port 3 is not available in the CUS package.

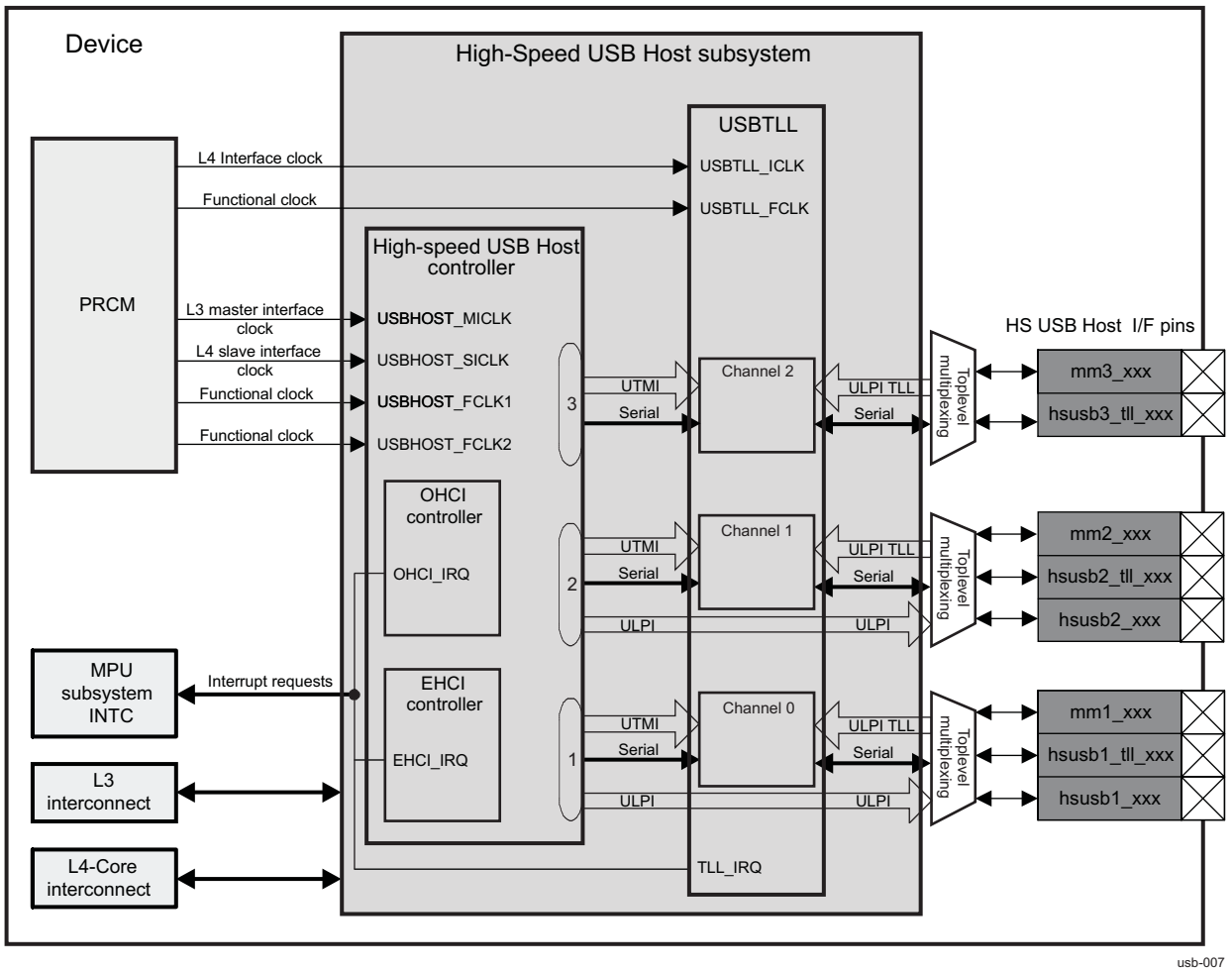
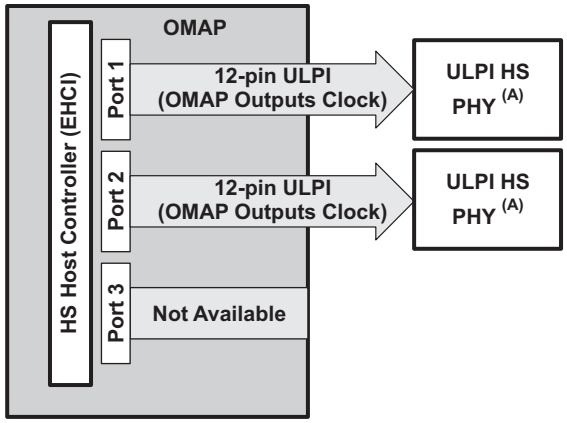


Figure 13. High-Speed USB Host Subsystem Highlight

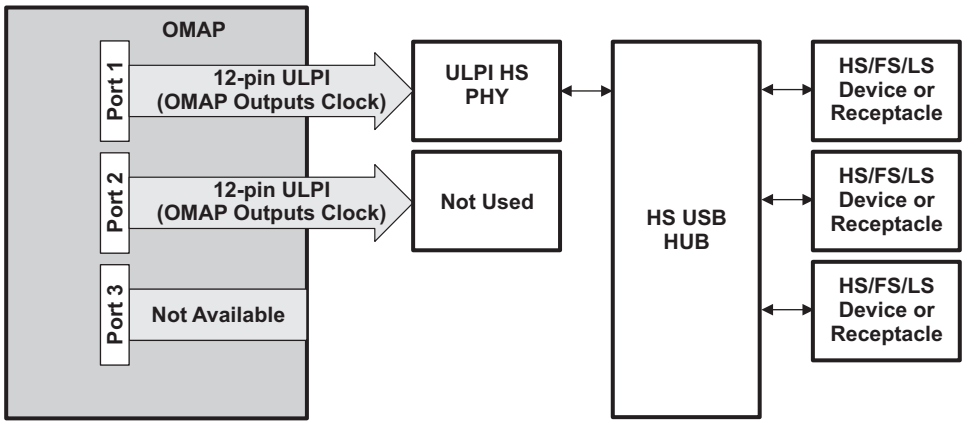
As shown in Figure 14, USB Port 1 and Port 2 can be used to connect to external high-speed PHYs which include a ULPI port. However, in this case, the external USB PHY must be able to accept a source clock generated by the OMAP high-speed USB controller. Also, in this usage model the USB ports cannot support full-speed and low-speed operation. Therefore, using this approach, USB Port 1 and Port 2 cannot provide a fully compliant USB 2.0 Type-A receptacle; a high-speed USB hub would be required in this case; please see Figure 15.

Note: USB Port 3 does not support ULPI mode. USB Port 3 is not available on CUS package.



A. USB port 3 does not support ULPI mode.
Also, USB port 3 not available on CUS package.

Figure 14. Connecting to High-Speed PHYs Using USB Ports



A. USB port 3 does not support ULPI mode. Also, USB port 3 not available on CUS package.

Figure 15. Connecting to a High-Speed USB Hub

4.2 Silicon Revision 3.0 Known Design Exceptions to Functional Specifications

Some silicon revision 3.0 applicable advisories have been found on a later silicon revision. For more details, see Section 3.2,*Silicon Revision 3.1 Known Design Exceptions to Functional Specifications*.

Table 7. Silicon Revision 3.0 Advisory List

Advisory 3.0.1.137 *DPLL Leakage When Placed in Low-Power Stop Mode*

Revision(s) Affected 3.0

Details Due to a DPLL integration issue, leakage is detected on VDDS_DPLL_DLL (MPU/IVA) and VDDS_DPLL_PER (per DPLLs) rails when DPLL is placed in low-power stop mode. This issue is due to DPLL going to fast relock mode instead of going to low-power stop mode.

Notes:

- Device OFF mode and CORE DPLL are not impacted.

Workaround(s) There is currently no workaround in place, but under evaluation, this section will be updated.

Advisory 3.0.1.138 *CPFROM In-Rush Current Issue*

Revision(s) Affected 3.0 and earlier

Details When Vpp is increased to 2.2V to program the CPFROM bits, the in-rush peak current observed on the Vpp line can slew up to more than 250mA in the first ~400ms.

Note: This is a preliminary measurement, and additional characterization is in progress. This issue will only affect OEMs using CPFROM bits, and confirmation regarding whether the Vpp supply rail from PMIC can support the in-rush current transient is pending.

Workaround(s) There is no workaround for this issue.

Advisory 3.0.1.151 *Missed Wake-up Event When Going Into OFF Mode*

Revision(s) Affected 3.0 and earlier

Details When OMAP35x goes to OFF mode, a wakeup event can be missed.

While transitioning to OFF mode, a window existing between GPIO (in peripheral domain) going to OFF and the IO wakeup chain is activated:

- For a 19.2MHz system clock, this window is 4Ms, in which a wakeup event can be missed
- Impacts all device GPIOs except for GPIOs from GPIO block 1: GPIO_[31:0]

This does not impact a wakeup event connected to the wakeup domain.

Workaround(s) Use a GPIO from block 1 to trigger an OFF Wakeup event.

Advisory 3.0.1.152 *USB TLL Save-and-Restore Issue*

Revision(s) Affected 3.0 and earlier

Details The USB TLL Save-And-Restore (SAR) + USBHOST TLL locks up system during CORE_OFF and suspend to RAM. A hardware issue in the TLL block causes a deadlock during the automatic hardware context saving of the TLL block. On second hardware SAR, the save mechanism does not function as expected; this is causes a lock-up in the SAR.

This issue impacts the OHCI TLL, PHY, and EHCI TLL modes. This does not impact HS PHY mode.

Workaround(s) There is no workaround for EHCI mode (HS TLL).

OHCI (FS TLL or FS PHY modes) : In OHCI mode, software Context / Restore is functional and will be integrated as part of mainline releases. This is a full workaround for the hardware issue.

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

This section describes the usage notes and advisories that apply to silicon revision 2.0 of the OMAP35x Applications Processor.

Note: The peripherals supported on the various OMAP35x Applications Processor devices are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Applications Processor, see the device-specific data manuals.

5.1 Usage Notes for Silicon Revision 2.1

5.2 Silicon 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 4.2, *Silicon Revision 3.0 Known Design Exceptions to Functional Specifications*.

Table 8. Silicon Revision 2.1 Advisory List

Advisory 2.1.1.120 *Peripheral Boot Issue When Using External Crystal*

Revision(s) Affected 2.1 and earlier

Details **Note**: This issue only applies in cases where the sys_clkout1 pin of the OMAP35x device is required during boot.

The SYS_BOOT[6] pin is not accessible via the register map, therefore the ROM code cannot change the pad configuration according to the XTAL mode.

Furthermore, the clock is never provided to the SYS_CLKOUT1 pin and not propagated to the companion chip, prohibiting the ROM code to accurately program the companion chip for peripheral boot (USB or MMC boot).

Workaround(s) No workaround.

Advisory 2.1.1.122 *ROM Code: Incomplete ASIC ID*

Revision(s) Affected 2.1 and earlier

Details The length of the root key hash has been extended on the OMAP35x architecture. During peripheral booting the ROM Code sends out an ASIC ID which - among other information - contains also the root key hash. The length of this hash field in the ASIC ID is correctly set to 20bytes but the 4 last bytes are all zero.

There is no known limitation as the root key hash field inside the ASIC ID is not necessary for peripheral booting.

Workaround(s) There is no workaround for this issue.

Advisory 2.1.1.123 *ROM Code: Context Restore Failure if OCM RAM Is OFF After Wakeup*

Revision(s) Affected 2.1 and earlier

Details The context restore procedure will fail if the On-Chip Memory (OCM) RAM is OFF after wakeup.

The OCM RAM will be off after wakeup if bits MEM1ONSTATE and MEM2ONSTATE in the PM_PWRSTCTRL_CORE register are set to OFF (0x0) before going to sleep.

Workaround(s) The application software must make sure that the bits MEM1ONSTATE and MEM2ONSTATE are set to ON (0x3) or RETENTION (0x1) before putting the CORE domain to sleep. There is no need to set the MEMORYCHANGE bit in the same register.

Advisory 2.1.1.128 *MMC: Multiple Block Read Operation Issue*

Revision(s) Affected 2.1 and earlier

Details The multiple block read transfer, in polling and interrupt mode, does not work correctly on MMC1 and MMC2. A Data CRC error is generated due to some corrupted data, when the read buffer (two 512-bytes portions) of the MMC controller is full.

If the buffer is not free, the MMC controller stops the clock and the card stops to send data to . The clock will be re-enabled only when one portion will be emptied and then the transfer will restart. The output clock and data enable are generated on same internal clock edge. The data enable signal must be re-asserted before the first clock edge of the input clock (feedback clock) in order to sample the data correctly.

Hold buffers have been added on data enable signal that makes data enable arrival after the first clock edge. It leads that the first data is not sampled on clock restart.

This issue depends on the pattern written and the output data width (1-, 4- or 8-bits mode). For example, Pattern written: 0x2800, Pattern read: 0x0801.

No data are read by the MPU into the read buffer in order to fill the 2 portions. The data enable signal is de-asserted and the output clock is disabled.

- In 1-bit mode, the failure occurs on the second data of third block when the MPU starts to read the data.
- In 4-bits mode, the failure occurs on the third data of third block when the MPU starts to read the data.

This issue is related to the integration of the IP and is not a functional bug. Therefore, MMC3 is not impacted.

Workaround(s) There is no workaround for this issue.

Advisory 2.1.1.135 *Isolation Issue Between SIM_VDDS and VDDS*

Revision(s) Affected 2.1 and earlier

Details There is no functional impact, but there is leakage current of approximately 20mA from SIM_VDDS to VDDS due to an isolation issue.

Workaround(s) There is no workaround for this issue.

Advisory 2.1.1.141 *CPU Revision Code Register Does Not Return Same Value for ES2.0 and ES2.1 Samples*

Revision(s) Affected 2.1

Details The CPU revision code register should return the same value for both ES2.0 and ES2.1 samples. However, the version of the cortexA8 core is incorrectly read. The results of the read are shown in Table 9.

Table 9. CPU Revision Code Register Readings

	EXPECTED READING	ACTUAL READING
ES2.0 Silicon ID	0x411fc081- r1p1	0x411fc081- r1p1
ES2.1 Silicon ID	0x411fc081- r1p1 + bug fixes	0x411fc082 - r1p2

Workaround(s) There is no workaround for this issue.

Advisory 2.1.1.145 *ROM Code: MMC1 Interface Configured in 8 Bits Mode Is Not Recommended at Boot Time.*

Revision(s) Affected

2.1 and earlier

Details

By default, the ROM Code is able to boot from MMC1, configured in 1 bit mode interface. In case the booting image contains a Configuration Header which configures a CHMMCSD section, the MMC1 interface can be configured in 8 bits mode. So the ROM Code is supposed to be able to switch from 1bit bus width to 8 bits bus width at the booting phase on MMC. The DAT4-DAT7 pins of MMC1 are powered by the MMC1a power supply. The ROM Code does not turn ON this power supply.

Moreover, pad interface 3V power supply voltage stability is not checked and can lead to booting instability.

Configuring the MMC1 interface at boot-time in 8-bit mode with a configuration header should be avoided. Functionality seems OK but the absence of power supply may cause an untimely ageing of IO pads.

Workaround(s)

There is no workaround.

6 Silicon Revision 2.0 Usage Notes and Known Design Exceptions to Functional Specifications

This section describes the usage notes and advisories that apply to silicon revision 2.0 of the OMAP35x Applications Processor.

Note: The peripherals supported on the various OMAP35x Applications Processor devices are different. The user should only refer to usage notes and advisories pertaining to features supported on the specific device. For a complete list of the supported features of the OMAP35x Applications Processor, see the device-specific data manuals.

6.1 Usage Notes for Silicon Revision 2.0

Silicon revision 2.0 applicable usage notes have been found on a later silicon revision. For more details, see Section 5.1, *Usage Notes for Silicon Revision 2.1*.

6.2 Silicon Revision 2.0 Known Design Exceptions to Functional Specifications

Some silicon revision 2.0 applicable advisories have been found on a later silicon revision. For more details, see Section 5.2, *Silicon Revision 2.1 Known Design Exceptions to Functional Specifications*.

Table 10. Silicon Revision 2.0 Advisory List

Advisory 2.0.1.52 *VENC: DAC1 and DAC2 Inversion*

Revision(s) Affected 2.0

Details

As stated in the Display Subsystem chapter of the *OMAP35x Technical Reference Manual* (literature number SPRUFA4), the video encoder DAC1 is used for luminance/composite output and the video encoder DAC2 is used for chrominance output. The video encoder DAC1 also includes a TV detection feature. The TV detection functionality is available on the OMAP35x Applications Processor video encoder DAC1, but is not available on DAC2. Due to an integration error in the video encoder, DAC1 is used for chrominance output, and DAC2 is used for luminance output. For proper TV detection in *composite video* mode, DAC1 should have been used for luminance, and DAC2 for chrominance.

However, due to an integration error in the video encoder, DAC1 is used for chrominance output and DAC2 is used for luminance output. Proper care should be taken when connecting to external devices (see workaround).

Furthermore, for proper TV detection in composite video mode (CVBS), DAC1 should be used for luminance and DAC2 for chrominance. Therefore, as a direct consequence of implementation, the TV detect feature is not available for composite video mode. Figure 16 shows the DAC implementation.

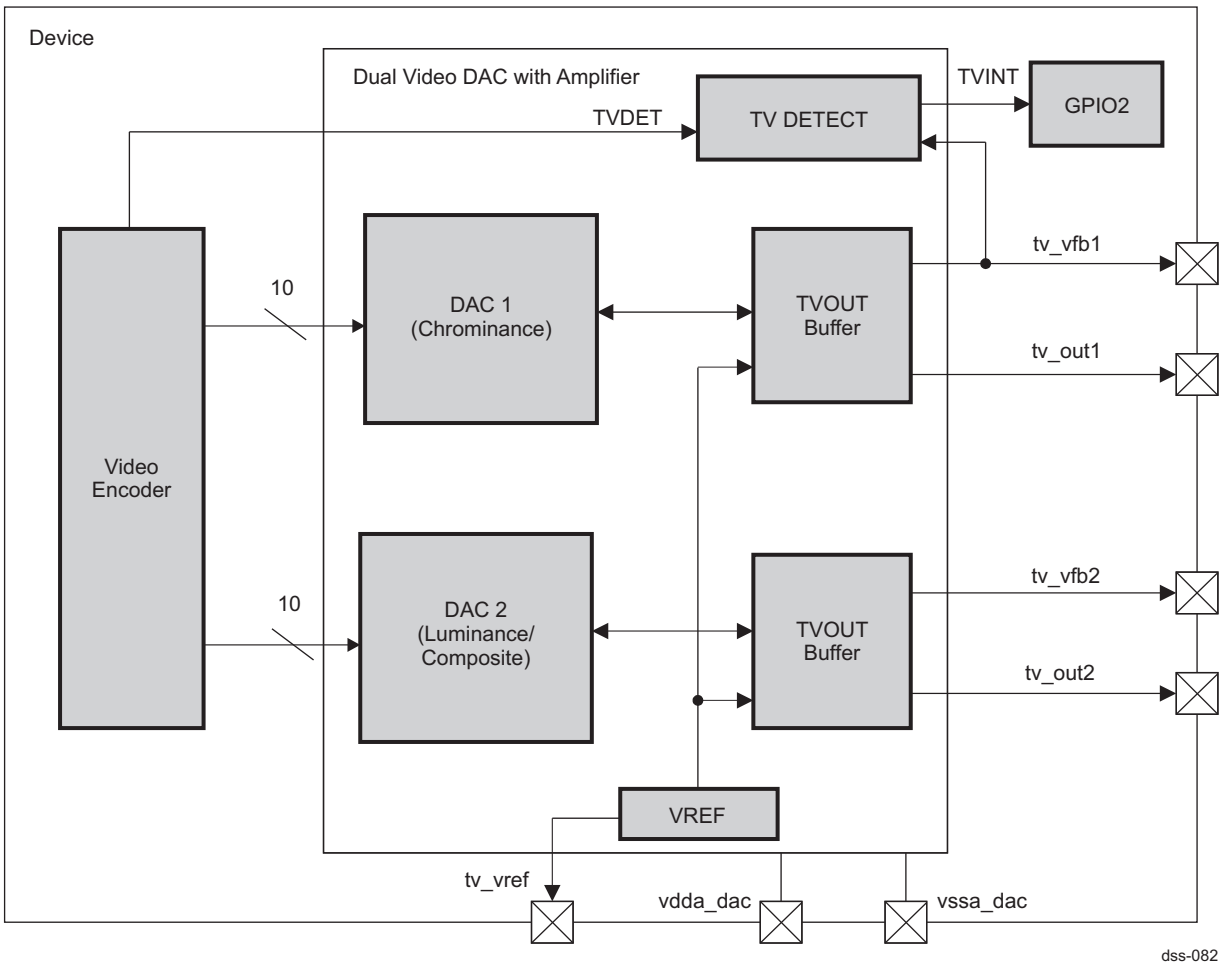


Figure 16. DAC Implementation

Workaround(s)

For proper TV out functionality, the external connection to the TV set must follow the implementation shown in Figure 17.

There is no workaround for the TV detect functionality. This feature is unavailable in *composite* video mode.

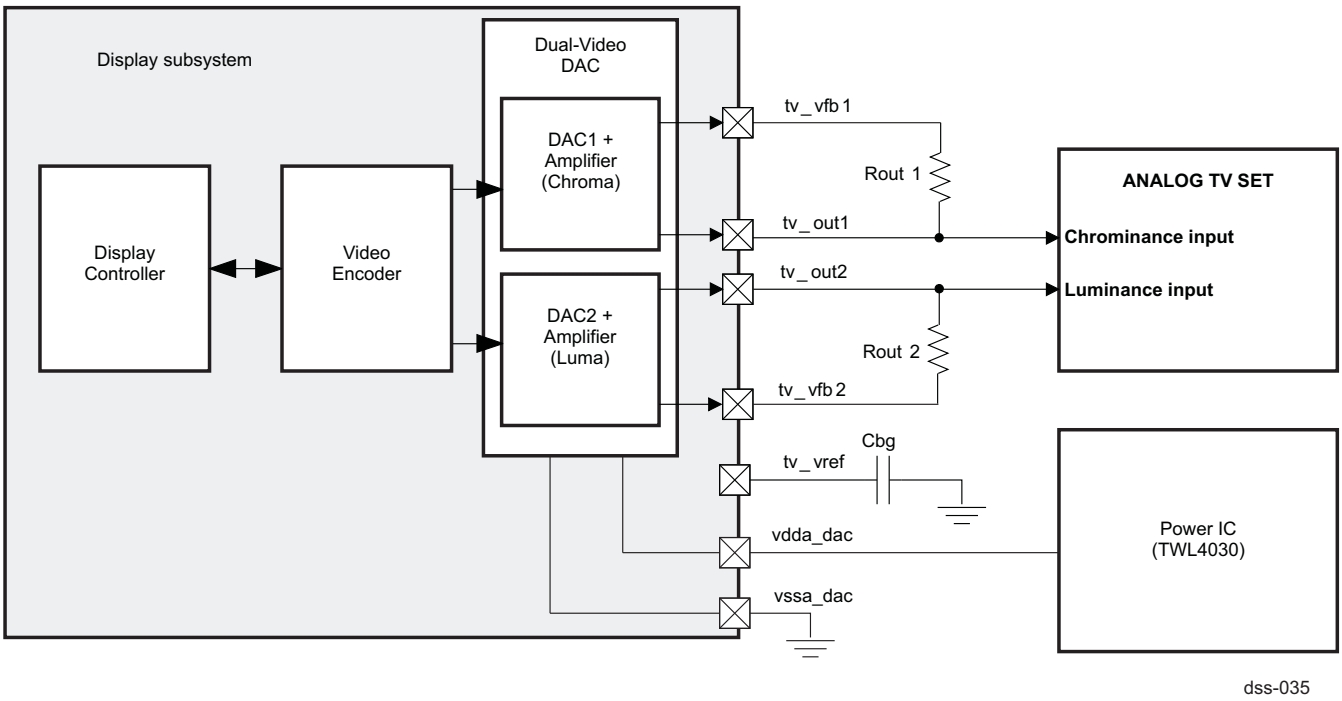


Figure 17. External Connection to TV Set

Advisory 2.0.1.80 *Device Stalls After a Warm Reset*

Revision(s) Affected 2.0

Details The PRCM fails to release the MPU reset after a number of Global Warm Reset iterations. This failure is random and can occur on any device type at any SYS_CLK frequency (i.e., 12 MHz, 19.2 MHz, 26 MHz, etc). However, the issue only occurs when DPLL3 (CORE DPLL) and DPLL4 (PER DPLL) are locked prior to generating a global software warm reset. This issue is not seen when DPLL3 is configured in *bypass* or *locked* mode and DPLL4 is in *low-power stop* mode before the Global Warm Reset has been applied.

Note: This issue affects all sources of Global Warm Reset (i.e,. global software warm reset, watchdog timer reset, external global warm reset, IcePick warm reset, etc.).

Workaround(s) Avoid Global software Warm Reset assertion or any source of Global Warm Reset while both DPLL3 and DPLL4 are locked.

Advisory 2.0.1.82 *SDI: Incorrect Control of SDI Analog Complex IO*

Revision(s) Affected 2.0

Details There is an incorrect signal connection to control the SDIv2 analog complex IO. As a result, *FlatLink3G* mode is not functional. Other serial modes (CMADS 1000, 2000) are not impacted.

Workaround(s) There is no workaround for this issue.

Advisory 2.0.1.104 *ROM Code: MPU Is Stuck in Secure Mode After Warm Reset on GP Device*

Revision(s) Affected 2.0

Details After any Warm Reset, the chip will branch to secure ROM Code. Since the secure ROM code does not exist on GP devices, the MPU will remain stuck in a loop. No debug can be performed.

The ROM code does not correctly manage the different resets. For a Warm Reset, the ROM code interprets the reset as a wake-up reset. The ROM code then tries to proceed to the context restore, interpreting data stored in scratchpad memory. If the data stored in scratchpad contains rubbish, the ROM code will attempt to restore them.

Workaround(s) The scratchpad memory from physical address 0x48002910 to 0x480029FF must be cleared by the application after a Power On Reset.

Advisory 2.0.1.105 *ROM Code: VBUS Detection Timeout too Short*

Revision(s) Affected 2.0

Details The VBUS detection timeout in ROM Code is too short (1ms) compared to the T2 debouncing time (30ms) which prevents the start of USB booting after board Power-On Reset. The USB boot functions correctly on subsequent nRESPWRON since T2 holds the VBUS status.

Workaround(s) Apply a second nRESPWRON to the MPU. The feasability of this workaround depends on the capability of the hardware.

Advisory 2.0.1.106 *USB: Peripheral Booting Dependent Upon System Clock Frequency*

Revision(s) Affected	2.0
Details	When using TPS69xxx PHY, USB peripheral booting is functional only with a 26MHz system clock.
Workaround(s)	Use another PHY (ISP1504) or 26MHz clock frequency.

Advisory 2.0.1.108 *ROM Code: OCM RAM Is Not Turned ON Correctly After Wakeup*

Revision(s) Affected 2.0

Details The internal RAM can be turned OFF, even when the chip is completely ON, running the normal application. After a wake-up, the ROM code needs to access internal RAM. The ROM code detects the RAM has been turned off and then tries to turn it on. This operation is not performed correctly, causing the OCM RAM not to be turned ON. The ROM code loops infinitely, waiting for the RAM to be ON.

Workaround(s) Before applying any sleep mode, the application must ensure that the OCM RAM is configured to be ON when the chip is ON.

Advisory 2.0.1.109 *ROM Code: Semaphore Is Not Initialized in Scratchpad Memory*

Revision(s) Affected 2.0

Details The booting image can contain a configuration header that can enable the ROM code to configure SDRC. When trying to configure the SDRC, the ROM code grabs a semaphore that is stored in scratchpad memory, but this semaphore was not initialized at POR.

The value stored in this register may prevent the ROM code from booting or performing wake-up reset when SDRC restoration is necessary.

Workaround(s) To prevent any wake-up problem, the software must delete the semaphore content after POR. However, there is no workaround to prevent any freeze at POR if a configuration header is defined in the booting image.

Advisory 2.0.1.110 *ROM Code: MMC Booting Using 4 /8-Bit Bus May Fail*

Revision(s) Affected 2.0

Details The booting image that is stored on the MMC can contain a software booting configuration (SWBCFG) [a specific data header that is located just before the booting code]. This SWBCFG is interpreted by the ROM code. The SWBCFG contains 2 important fields: MMC clock and MMC bus width. When setting the MMC bus width to 4, or 8 bits, the ROM code may fail to boot on certain MMC or SD devices.

This failure depends on the MMC type. When switching to 4 or 8-bit bus width, the busy signal (present on DAT0 line) is asserted by the MMC. Some MMC devices need more than 100 Ms to be operational after having changed the bus width. After 60 Ms, the ROM code tries to read data from MMC without checking the busy signal level, and the read operation fails. MMC1 and MMC2 are impacted.

Workaround(s) Use the MMC in *1-bit* mode at booting phase.

Advisory 2.0.1.114 *Cortex-A8 r1p1 Exhibits Higher Level of Missed Branch Prediction*

Revision(s) Affected 2.0

Details Due to an integration issue (test signal not tied to the right value), a higher level than expected of missed branch predictions are generated.

Workaround(s) There is no workaround.

Appendix A Revision History

This silicon errata revision history highlights the technical changes made to the OMAP35x device.

Table 11. *OMAP35x* **Revision History**

SEE	ADDITIONS/CHANGES/DELETIONS
Section 2.1	Added the following Usage Notes to Silicon Revision 3.1.2 • 2.1.19- DPLL3 Recall and Long Relock Time • 2.1.20- Transfer of Multiple Command Packets Coming from L4 Interconnect During a Blanking Period in Interleaving • 2.1.21- Downscaling Limitations
Section 2.2	Added the following Advisories to Silicon Revision 3.1.2 • 3.1.1.57- SPI Dummy DMA RX Request Generation • 3.1.1.77- MPU L2 Cache Size Status Register Value Inverted • 3.1.1.80- Acting as a Host; For Bulk Split IN Transactions, the MUSBMHDRC Can Transmit Tokens too Close to the SOF Packet, Causing an IPG Error • 3.1.1.81- OCP Error Does Not Get Communicated to USBOTG • 3.1.1.201- USB OTG DMA May Stall When Entering Standby Mode • 3.1.1.202- DPLL3 in Manual Lock Mode Cannot be Used When CORE Goes to OSWR or OFF State • 3.1.1.203- PRCM DPLL Control FSM Removes SDRC_IDLEREQ Before DPLL3 Locks • 3.1.1.204- PER Domain Reset Issue After Domain-OFF/OSWR Wakeup • 3.1.1.205- McBSP Used in Slave Mode Can Create a Dead Lock Situation When Doing Power • 3.1.1.206- DPLL3 Bypass Condition Does Not Consider State of SGX FCLK • 3.1.1.207- I2C: In SCCB Mode, Under Specific Conditions, the Module Might Hold the Bus by Keeping SCL Low • 3.1.1.208- I2C: Spurious Wakeup Event When sysclk Period is Higher Than ocpclk Period • 3.1.1.209- I2C: Wrong Behavior When a Data With MSB 0 is Put in the FIFO Before a Ttansfer with SBLOCK is Started • 3.1.1.210- I2C: After an Arbitration Lost the Module Starts Incorrectly the Next Transfer • 3.1.1.211- 4-cycle Saturating .M Unit Instructions May Mask 2-cycle Saturating. M Unit Instruction Saturation Bit Update • 3.1.1.212- SPLOOP CPU Cross-Path Stall

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

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

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI 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 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. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

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