

# Chip Errata for the i.MX23

This document details all known silicon errata for the i.MX23. [Table 1](#) provides a revision history for this document.

**Table 1. Document Revision History**

Rev. Number	Date	Substantive Change(s)
2	04/2010	New errata: <a href="#">2349</a>
1	11/10/2009	New errata: <a href="#">5835</a> , <a href="#">5837</a>
0	09/01/2009	Initial release. New errata: <a href="#">2229</a> , <a href="#">2725</a> , <a href="#">2727</a> , <a href="#">2745</a> , <a href="#">2747</a> , <a href="#">2748</a> , <a href="#">2749</a> , <a href="#">2752</a> , <a href="#">2765</a> , <a href="#">2783</a> , <a href="#">2786</a> , <a href="#">2798</a> , <a href="#">2811</a> , <a href="#">2814</a> , <a href="#">2815</a> , <a href="#">2816</a> , <a href="#">2829</a> , <a href="#">2836</a> , <a href="#">2847</a> , <a href="#">2854</a> , <a href="#">2858</a> , <a href="#">4559</a> , <a href="#">4733</a>

[Table 2](#) provides a cross-reference to match the revision code in the processor version register to the revision level marked on the part.

**Table 2. Revision Level to Part Marking Cross-Reference**

i.MX23 Revision	Device Marking	Processor Version Register (HW_DIGCTRL_CHIPID_REVISION)
TA4 <sup>1</sup>	TA4	0x3

<sup>1</sup> TA4 is equivalent to revision 1.3

**Table 3** summarizes all known errata and lists the corresponding silicon revision level to which it applies. A ‘Y’ entry indicates the erratum applies to a particular revision level, while a ‘—’ entry means it does not apply.

**Table 3. Summary of Silicon Errata and Applicable Revision**

Errata	Name	Silicon Rev. TA4 <sup>1</sup>
2229	Insufficient SD/MMC read delay leads to premature timeouts and can cause some cards to fail booting.	Y
2725	Abort when the CPU accesses external memory while it is in self-refresh mode does not seem to work properly.	Y
2727	I2C 9th Clock Pulse (ACK) not generated when RETAIN_CLOCK set.	Y
2745	EMI interrupts are not available.	Y
2747	Undetectable data corruption can occur during IN transfer in isochronous mode.	Y
2748	AMBA-AHB Estimator clock crossing issue.	Y
2749	USB: Port change interrupt not generated.	Y
2752	PXP YBASE Calculation Incomplete	Y
2765	Chip locks up when switching to synchronous mode if ref_cpu is not active.	Y
2783	Audioout_ctrl_edge_sync bit is not used (audioin_ctrl_edge_sync is used for ADC and DAC)	Y
2786	JTAG crash when HCLK is less than 24 MHz.	Y
2798	Phase error between left/right ADC channels.	Y
2811	Unreliability of VDD5V_GT_VDDIO functionality	Y
2814	The DCDC converters inadvertently turn on when 5 V is removed while the DCDC_XFER bit is clear.	Y
2815	The hardware functionality to disable battery brownouts when 5 V is present always does NOT follow the VBUSVALID_5VDETECT value.	Y
2816	The PWDN_5VBRNOUT functionality trips during certain events even though 5 V voltage is valid.	Y
2829	HW_I2C_CTRL1_OVERSIZE_XFER_TERM_IRQ set at end of every transaction	Y
2836	APBH/APBX do not flush out the remaining FIFO data after a terminate	Y
2847	BCH soft reset may cause bus master lock up.	Y
2854	The DCDC fails to shut down when the battery is below the battery brownout level and PWD_BATTBRNOUT is set, due to the presence of a 5 V supply.	Y
2858	USB controller may access a wrong address for the dTD (endpoint transfer descriptor) and then hang.	Y
4559	ROM Fails to Boot from BA-NAND	Y
4733	SD Boot does not support MMC with CSD_STRUCTURE version in EXT_CSD register	Y
5835	5 V disconnections can cause large droops on the VDDIO and VDDD supply rails under certain operating conditions	Y
5837	Setting the ENABLE_DCDC bit in the HW_POWER_DCDC4P2 or HW_POWER_5VCTRL registers can result in false brownout detection.	Y
2349	The DAC DMA halts unexpectedly	Y

<sup>1</sup> TA4 is equivalent to revision 1.3.

## **2229                      Insufficient SD/MMC read delay leads to premature timeouts and can cause some cards to fail booting.**

### **Description:**

The SD/MMCC read delay being used by ROM is not sufficient, which leads to premature timeouts that cause some cards to fail booting. According to the SD specifications, a minimum of 100 ms or one hundred times the "typical delay" should be used. In a failing SD cards case, the "typical delay" calculated from the TAAC and NSAC fields is 1.5 ms. So the read response delay for this card should have been 100 ms, but ROM was using a ~4-ms delay.

After debugging in ROM, a particular failing card started working when the delay was modified to 13 ms. Four out of five 2-Gbyte Sandisk blue SD cards tested could not boot.

This issue is only reproducible with SD 2.0 2-GB devices and is not reproducible with MMC, eMMC and older SD 1-GB devices.

### **Workaround:**

The only workaround is to patch ROM using other bootable devices like I<sup>2</sup>C EEPROM, SPI EEPROM, and so on. The patch should first call Init of the ROM SSP driver, then update the pSpContext->timeout value to 1000000 (1-sec timeout).

This patch binary image is available on Freescale.com. It is called "IMX23 EVK SD ROM PATCH", with a description of "EEPROM Patch Workaround for Errata #2229".

**2725                      Abort when the CPU accesses external memory while it is in self-refresh mode does not seem to work properly.****Description:**

If the EMI is in self-refresh mode, any external data accesses to the memory cause the system to hang. External access cannot be completed when the EMI is in this mode, so the accesses stall the system bus.

**Workaround:**

Do not allow external data access of the system bus when the EMI is in self refresh mode. Software must enforce this scenario.

**2727 I2C 9th Clock Pulse (ACK) not generated when RETAIN\_CLOCK set.****Description:**

When RETAIN\_CLOCK is set, the ninth clock pulse (ACK) is not generated. However, the SDA line is read at the proper timing interval. If RETAIN\_CLOCK is cleared, the ninth clock pulse is generated.

Also, the HW\_I2C\_VERSION register incorrectly states the version is 1.2. It should be 1.3.

**Workaround:**

HW\_I2C\_CTRL1[ACK\_MODE] has default value of 0. It should be set to 1 to enable the fix for this issue.

**2745 EMI interrupts are not available.****Description:**

The i.MX23 memory controller is capable of generating an interrupt under the following conditions:

- A single access outside the defined physical memory space is detected
- Multiple accesses outside the defined memory space are detected.
- DRAM initializations are complete
- DLL unlock condition is detected.

This interrupt is not used with the i.MX23 interrupt controller.

**Workaround:**

For software to detect and react to any of the four controller interrupt conditions, the software must read DRAM control register 16 and examine the values in the int\_status field. Typically, however, this feature should have minimal value in a system with memory management, where physical memory mappings can be controlled via page tables. As such, Freescale does not recommend using these status bits or interrupts for the purpose of physical memory range control.

**2747                      Undetectable data corruption can occur during IN transfer in isochronous mode.****Description:**

Undetectable data corruption can occur during IN transfer in isochronous mode.

**Workaround:**

There is a safe window for the software to prime the isochronous endpoint for an IN transfer that prevents the occurrence of this erratum.

For high-speed, this window is impossibly small. Do not use high-speed isochronous endpoints in high-speed.

However, for full-speed, the window is more reasonable. This time window corresponds to the amount of time in a frame allocated for the asynchronous schedule. The USB specification requires that no more than 90% of any frame be allocated for periodic (isochronous and interrupt) full-/low-speed transfers, so in the worst case the last 100  $\mu$ s of the FS frame will be allocated for the asynchronous schedule. If the software primes the endpoint from 900  $\mu$ s to 999  $\mu$ s after receiving an SOF interrupt, this erratum never occurs.

**2748 AMBA-AHB Estimator clock crossing issue.****Description:**

USB internal AHB bus transfers can be lost when using anything other than AHB unspecified length incrementing burst types.

**Workaround:**

Only use type unspecified INCR bursts, which is guaranteed by this setting:

`HW_USBCTRL_SBUSCFG[AHBBRST] = 0x0` (this is the default reset value)

This type of burst mode results in the highest external memory efficiency for the i.MX23.



**2749 USB: Port change interrupt not generated.****Description:**

In self-powered device mode, the USB controller may not generate a port change interrupt when VBUS goes away.

**Workaround:**

Use VBUSVALID interrupts in POWER module. Set POLARITY\_VBUSVALID to 0x0 to check for 5 V disconnected. See Chapter 32, “Power Supply,” in the *i.MX23 Applications Processor Reference Manual* (document number IMX23RM), for details.

## 2752 PXP YBASE Calculation Incomplete

### Description:

Chapter 17, “Pixel Pipeline,” in the *i.MX23 Applications Processor Reference Manual* (order # IMX23RM), states the following: “The [crop] XBASE and YBASE provide the coordinates of the first block to be displayed from the source image and the WIDTH and HEIGHT parameters specify an effective size of the resulting image in the output buffer.”

This statement is correct for non-interlaced, and interlaced\_output. However, when doing interlacing input to one field, the crop\_ybase value gets shifted by 2 to match the size of the new image. This results in an incorrect index into the original S0 image.

### Workaround:

To get the correct location for cropping while doing interlaced input, the value of the crop\_ybase register needs to be doubled to get the desired index location into the S0 image.

**2765 Chip locks up when switching to synchronous mode if ref\_cpu is not active.****Description:**

In order to switch the EMI clock from asynchronous to synchronous mode, both the xtal\_ref and the cpu\_ref must be active, even if both BYPASS\_CPU and BYPASS\_EMI are set in the HW\_CLKCTRL\_CLKSEQ register.

**Workaround:**

If cpu\_ref is inactive, then activate the cpu\_ref before attempting to switch to synchronous mode.

**2783                    Audioout\_ctrl\_edge\_sync bit is not used (audioin\_ctrl\_edge\_sync is used for ADC and DAC)****Description:**

The EDGE\_SYNC bit in AUDIOOUT\_CTRL register has no effect on the DAC.

**Workaround:**

The EDGE\_SYNC bit in AUDIOIN\_CTRL register should be used for both DAC and ADC.

**2786 JTAG crash when HCLK is less than 24 MHz.****Description:**

The chip can appear to “crash” when using JTAG to debug code and the clocks are changed such that HCLK is set to less than 24 MHz. The “crash” is not actually a chip problem in that the chip does continue to run. But it is no longer possible to halt or stop at a breakpoint via the debugger, so it seems that the chip has died.

**Workaround:**

The workaround is to set the debugger JTAG target clock to 700 kHz or lower, depending on how low HCLK is set. The lower HCLK is set, the lower the JTAG target clock must be set. For example, a JTAG target clock setting of 100 kHz is recommended for HCLK at 3 MHz. If HCLK is 12 MHz, 700 kHz is recommended. Where HCLK is from 12 MHz up to 24 MHz, no higher than 700 kHz is recommended.

This is only required when using JTAG to debug portions of code where HCLK can be below 24 MHz (stopping at breakpoints, stepping, and so on). For downloading code via JTAG or debugging where HCLK is at or above 24 MHz, the JTAG target clock can be at 1 MHz.

**2798 Phase error between left/right ADC channels.****Description:**

There is a phase error of 1/2 sample between the left and right ADC (62.5  $\mu$ s at 8 KSPS and 11.34  $\mu$ s at 44.1 KSPS).

**Workaround:**

The phase error can be minimized (but not eliminated) by running at a higher sample rate.

## 2811

**Unreliability of VDD5V\_GT\_VDDIO functionality****Description:**

Due to unreliability of the VDD5V\_GT\_VDDIO functionality, the chip should default to using VBUSVALID detection and only boot up until the default VBUSVALID\_TRSH level has been reached. Otherwise, software can start running before a reliable state has been reached. Currently, software has to handle this case by spinning in a loop, waiting for one of the following conditions to occur:

- VBUSVALID level is reached
- A good battery voltage plus PSWITCH is pressed
- RTC\_ALARM is true
- AUTO\_RESTART bit is true

**Workaround:**

There are inherent weaknesses with using the VDD5V\_GT\_VDDIO method as the hardware 5-V detection method. This method depends on a fixed offset between VDDIO and the VDD5V voltage. So the VDD5V\_GT\_VDDIO status changes as the VDDIO level changes. Secondly, as with any linear regulator, the VDDIO linear regulator supply strength is based on its supply voltage, which is VDD5V. As the VDD5V level decreases, so does the VDDIO supply strength. If the VDDIO rail has too much load and the VDD5V level decreases, the VDDIO output voltage can decrease as well. If the VDDIO voltage drops with VDD5V, then a case could exist in which VDD5V\_GT\_VDDIO never becomes false. One method for countering this particular weakness is to enable and properly handle VDDIO brownouts. However, this method produces other complications such as causing device shutdown on a 5-V disconnect in some cases where the device could have performed a 5-V to battery transfer and avoided a shutdown. Or this method could also necessitate adding functionality to the VDDIO interrupt handler to differentiate between brownouts caused by 5-V disconnects versus not.

To avoid these weaknesses with the VDD5V\_GT\_VDDIO functionality, the hardware can be programmed to use the VBUSVALID functionality for the 5-V status by setting the VBUSVALID\_5VDETECT bit. However, great care must be taken to coordinate this change properly.

Keep in mind that the PWD\_BATTBRNOUT functionality is broken and always uses VDD5V\_GT\_VDDIO as an internal enable/disable mechanism (see erratum [2815](#)). A problem can occur when the 5-V detection is changed from VDD5V\_GT\_VDDIO to VBUSVALID if the VDD5V\_GT\_VDDIO status is true but the VBUSVALID\_TRSH level being used is higher than the VDD5V level. If this occurs, then the VBUSVALID status being false will cause the system to believe that a 5-V disconnection has occurred. A hardware 5-V disconnection event causes the DCDC to automatically turn on, regardless of the battery voltage level, which could result in the DCDC operating from a voltage lower than the minimum operating voltage specified in Chapter 2, “Characteristics and Specifications,” in the *i.MX23 Applications Processor Reference Manual* (order # IMX23RM). This, in turn, could result in physical damage to the DCDC converter itself. To avoid this situation, always be sure that the VDD5V voltage level is higher than the

VBUSVALID\_TRSH level programmed. Likewise, when raising the VBUSVALID\_TRSH from one level to another, be sure that the VDD5V is higher than the target level.



**2814                      The DCDC converters inadvertently turn on when 5 V is removed while the DCDC\_XFER bit is clear.**

**Description:**

If the DCDC\_XFER bit is clear, the DCDC convert should not automatically turn on when 5 V is removed. Instead, a power down should occur if 5 V is removed, and DCDC\_XFER and ENABLE\_DCDC are zero.

**Workaround:**

Because the DCDC will automatically be enabled by hardware once 5 V is removed, proper protection must be configured so the device shuts down if the battery voltage is below the minimum voltage limit, as specified in Chapter 2, “Characteristics and Specifications,” in the *i.MX23 Applications Processor Reference Manual* (order # IMX23RM). To do this, leave the PWDN\_5VBRNOUT bit set until sufficient battery voltage is present. Then, the software can properly enable the power supply to prepare for a DCDC transfer on the loss of 5 V. After that preparation is complete, software can clear the PWDN\_5VBRNOUT bit.

**2815                      The hardware functionality to disable battery brownouts when 5 V is present always does NOT follow the VBUSVALID\_5VDETECT value.**

**Description:**

The hardware functionality to disable battery brownouts when 5 V is present does NOT follow the VBUSVALID\_5VDETECT value. However, it should in order to coordinate with the rest of the hardware (and software).

**Workaround:**

The automatic hardware functionality to disable hardware battery brownout shutdowns once 5 V is connected must use VDD5V\_GT\_VDDIO instead of following the VBUSVALID\_5VDETECT configuration. The developer should take this into consideration when developing the power protection scheme. At some point in the application, software handling of battery brownouts is enabled, and hardware battery brownouts are enabled, which makes the effects of this problem go away.

**2816                      The PWDN\_5VBRNOUT functionality trips during certain events even though 5 V voltage is valid.**

**Description:**

The PWDN\_5VBRNOUT bit will incorrectly power down the chip when specific power bits are modified. The 5 V brownout comparator is injected with a small amount of noise when these bits are set, causing the output to glitch momentarily.

**Workaround:**

1. Save the current value of HW\_POWER\_5VCTRL\_\_PWDN\_5VBRNOUT.
2. Clear HW\_POWER\_5VCTRL\_\_PWDN\_5VBRNOUT.
3. Modify HW\_POWER\_5VCTRL\_\_ENABLE\_DCDC.
4. Restore value of HW\_POWER\_5VCTRL\_\_PWDN\_5VBRNOUT.

**2829 HW\_I2C\_CTRL1\_OVERSIZE\_XFER\_TERM\_IRQ set at end of every transaction****Description:**

The HW\_I2C\_CTRL1\_OVERSIZE\_XFER\_TERM\_IRQ is set at the end of every transaction. This IRQ is not used in master mode and therefore should be ignored.

**NOTE**

If HW\_I2C\_CTRL1\_OVERSIZE\_XFER\_TERM\_EN is set, the HW\_I2C\_STAT\_ANY\_ENABLED\_IRQ is also set at the end of any MASTER transaction since HW\_I2C\_CTRL1\_OVERSIZE\_XFER\_TERM\_IRQ will be high.

**Workaround:**

Ignore the HW\_I2C\_CTRL1\_OVERSIZE\_XFER\_TERM\_IRQ bit and do not set HW\_I2C\_CTRL1\_OVERSIZE\_XFER\_TERM\_EN.

## 2836 APBH/APBX do not flush out the remaining FIFO data after a terminate

### Description:

Both APBX and APBH bridge DMAs are affected by this erratum. The issue occurs when a peripheral terminates a DMA transaction abruptly without sending all the data the corresponding DMA channel was set up to receive. There is a possibility of leftover data remaining in the DMA channel's FIFO; that data is not written into memory and is lost.

For example, if a timeout occurs in UART RX, it issues a `dma_terminate` to the corresponding APBX DMA channel. If the DMA channel does not receive all the data that it was set up to receive, and there was data lingering in the DMA channel's FIFO, the remaining data is not written to memory.

### Workaround:

Do not set HALTONTERMINATE.

A possible workaround to recover the data is to set up a dummy descriptor to read the remaining data in the channel's FIFO. It should be used only if the application can tolerate the small latency introduced by the workaround.

1. At the end of every descriptor, check the difference between `AHB_BYTES` and `APB_BYTES` in the `DEBUG2` register of the DMA channel.
2. A difference of non-zero means there is data in the channel's FIFO. Set up a dummy descriptor with `XFER_COUNT = AHB_BYTES - APB_BYTES` only if the difference is non-zero.
3. Set `CMDWORDS` in the dummy descriptor to be zero; that is, DMA should not be writing PIO words into the peripheral.
4. Do not increment the semaphore counter in the DMA channel.
5. After the channel is kicked off, poll the `AHB_BYTES` bit field in its `DEBUG2` register. When it goes to zero, the dummy descriptor is done reading the data from the channel's FIFO and storing it in memory.
6. Reset the DMA channel; otherwise, it will hang.

**2847 BCH soft reset may cause bus master lock up.****Description:**

The BCH bus master may lock up if the soft reset bit (HW\_BCH\_CTRL\_SFTRST) is set. HW\_BCH\_CTRL\_CLKGATE may be set and cleared as desired to clock gate the block, as long as HW\_BCH\_CTRL\_SFTRST is never set.

**Workaround:**

Do not assert HW\_BCH\_CTRL\_SFTRST. To reset the register set, write desired values to the BCH registers.

**2854**                    **The DCDC fails to shut down when the battery is below the battery brownout level and PWD\_BATTBRNOUT is set, due to the presence of a 5 V supply.**

**Description:**

If the DCDC is trying to source off a depleted battery while PWD\_BATTBRNOUT is set, the DCDC should shut down.

**Workaround:**

The software workaround is to use the software battery brownout.

## 2858 USB controller may access a wrong address for the dTD (endpoint transfer descriptor) and then hang.

### Description:

Currently, software checks the Active bit in dTD to see whether it is finished. If the Active bit is 0, then software frees the allocated memory for the dTD.

Hardware sequence after all data of a dTD is transferred is as follows:

1. Update dTD. (This includes three DWs AHB write access. The Active bit is cleared in the first DW write.)
2. Update qHead (this includes three DWs AHB write access)
3. Read the dTD again to check if software added a new dTD (this is a SINGLE AHB read). At the same time, send out an interrupt if needed.

After step 1, if software finds the Active bit is cleared, then the dTD memory space is freed and this space may be allocated for other thread use. In step 3, hardware may get a wrong dTD.

This issue does not occur if some delay is added before freeing the dTD memory space.

This issue only occurs in USB INCR8 mode, because steps 1 and 2 have 6 SINGLE AHB transfers in INC8 mode, but only 2 burst AHB transfers in INCR mode.

This issue only occurs when the dTD list is used, because if only one dTD is used, the software only checks the Active bit after an interrupt is received (step 3). But when the dTD list is used, the software may check the entire list after the interrupt for the first dTD is received, and hardware just finished the transfer of the second dTD at this time.

The [Figure 1](#) illustrates this issue:

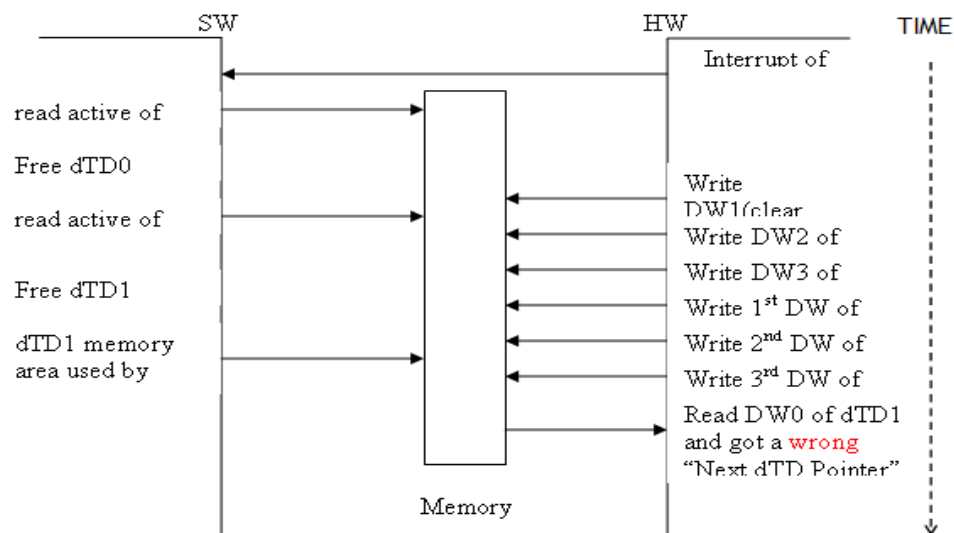


Figure 1. dTD Issue Sequence of Events



**Workaround:**

Postpone freeing the current dTD; free it when its next dTD can be freed, so the last completed dTD (followed by an ACTIVE dTD) is always freed when the next IOC irq comes.

## 4559 ROM Fails to Boot from BA-NAND

### Description:

In ROM code, the definition of the data structure returned by ONFi BA-NAND's parameters page command is not enclosed within `#pragma pack(1)` compiler directive. As such, data read from the BA-NAND does not correspond to actual fields in the data structure, causing improper values in the `m_u16SectorSize` field. Due to an incorrect sector size calculation in ROM, boot from BA-NAND fails. The ROM executes the following steps after reading the parameter page's data:

1. It reads the first 2 KB of data present on sector 0 by issuing start lba address 0. Each lba is 512 bytes. The first sector contains the MBR.
2. Micron's BA-NAND sector size is 4 KB. To skip the next 2 K of sector 0, ROM sends an abort cmd to the BA-NAND device.
3. Next, ROM will prepare to read the config block present at sector 1 or lba 8.
4. ROM will calculate the next sector address in lba units. However, because the sector size is incorrect from the parameters page, an incorrect lba address for sector 1 will result. Reading from this incorrect address results in unidentified data and the boot will fail.

### Workaround:

To work around this issue, a simple patch should be present on the first 2 K of firmware data that should be loaded first and called. The patch should do the following to fix the problem.

```
MyNandItf.BANandHalAbort(0); // to abort reading next 2K of first sector
pNandContext->zSerializerState.m_u32PageDataSize = 4096; // fix the sector size
pNandContext->zSerializerState.m_u32SectorDataRead = 0; // initialize to 0
```

The actual firmware should be programmed starting at the next 4 K from the beginning of the firmware area.

**4733 SD Boot does not support MMC with CSD\_STRUCTURE version in EXT\_CSD register****Description:**

The ROM is not able to boot an eMMC with the CSD\_STRUCTURE version in the EXT\_CSD register. See the code in sd\_ddi.c. It should check for CSD\_STRUCTURE==3 and then look in the EXT\_CSD register for the actual CSD\_STRUCTURE version. See the MMC specification, v. 4.2 or greater.

**Workaround:**

The only workaround is to patch ROM by providing the fix in other boot mode devices like I<sup>2</sup>C EEPROM, or SPI EEPROM, and so on.

**5835**                      **5 V disconnections can cause large droops on the VDDIO and VDDD supply rails under certain operating conditions****Description:**

When 5 V is disconnected, the DCDC does not switch its input from the VDD4P2 source to the battery quickly enough to prevent the DCDC output from drooping under heavy loads. One condition that exhibits the problem is a disconnect from 5 V with a 3.5-V battery voltage and a ~250-mA load on VDDIO.

**Workaround:**

To prevent a significant dip, the VDD4P2 must switch off as soon as possible when 5 V is lost to alert the DCDC to use battery power. To do this, configure the VDD5V\_DROOP interrupt as an FIQ and set the droop interrupt threshold to a voltage level above the VBUSVALID threshold. Inside the VDD5V\_DROOP handler, clear the ENABLE\_DCDC and ENABLE\_4P2 bits in the HW\_POWER\_DCDC4P2 register.

## 5837                    **Setting the ENABLE\_DCDC bit in the HW\_POWER\_DCDC4P2 or HW\_POWER\_5VCTRL registers can result in false brownout detection**

### **Description:**

When the ENABLE\_DCDC bit in HW\_POWER\_DCDC4P2 is set, a glitch is propagated through the brownout comparators. If the glitch is sufficiently large, it can cause a false brownout detection. The VDDD, VDDA, VDDIO, and VBUSVALID comparators are all susceptible to the glitch.

### **Workaround:**

The sequence below is needed to work around this issue prior to setting the ENABLE\_DCDC bit in HW\_POWER\_DCDC4P2:

1. Disable the power rail brownout interrupts (clear HW\_POWER\_CTRL VDDA, VDDD, VDDIO ENIRQ bits).
2. Set the HW\_POWER\_5VCTRL PWRUP\_VBUS\_CMPS bit (may already be set).
3. Set the HW\_POWER\_5VCTRL VBUSVALID\_5VDETECT bit to 0.
4. Set the HW\_POWER\_5VCTRL VBUSVALID\_TRSH to 0x0 (2.9V).
5. Disable VBUSDROOP status and interrupts (clear VDD5V\_DROOP\_IRQ).
6. Set the ENABLE\_DCDC bit in HW\_POWER\_DCDC4P2.
7. Wait 100  $\mu$ s.
8. Check VBUSVALID\_IRQ bit. If it is set, then set and clear the PWD\_CHARGE\_4P2 bit to repower on the 4P2 regulator because it is automatically shut off on a VBUSVALID false condition. It may be helpful to ramp up the CHARGE\_4P2\_ILIMIT value at this point to gradually draw power from 5 V rail. If HW\_POWER\_5VCTRL ENABLE\_DCDC is already set, the DCDC will we sourcing charge from 4P2 as soon as PWD\_CHARGE\_4P2 is toggled.
9. Clear VBUSDROOP, VBUSVALID, and the output rails IRQ bits as needed.
10. Restore the output rail ENIRQ bits, the VBUSVALID\_THRSH level, VBUSVALID\_DETECT value, VBUSVALID ENIRQ, and VBUSDROOP ENIRQ to their original values.

The sequence below is needed to work-around this issue prior to setting the ENABLE\_DCDC bit in HW\_POWER\_5VCTRL. Note that the below workaround assumes the usual requirements for setting the HW\_POWER\_5VCTRL ENABLE\_DCDC bit are met (that is, the hardware and/or software battery brownout protection mechanism is enabled to properly protect the system against the DCDC sourcing from the battery if the voltage if the battery voltage is too low.).

1. Disable the power rail brownout interrupts (clear HW\_POWER\_CTRL VDDA, VDDD, VDDIO ENIRQ bits).
2. Set the HW\_POWER\_5VCTRL PWRUP\_VBUS\_CMPS bit (may already be set).
3. Set the HW\_POWER\_5VCTRL VBUSVALID\_5VDETECT bit to 1.
4. Set the HW\_POWER\_5VCTRL VBUSVALID\_TRSH to 0x0 (2.9V).

5. Disable VBUSDROOP status and interrupts (clear VDD5V\_DROOP\_IRQ).
6. Set the ENABLE\_DCDC bit in HW\_POWER\_5VCTRL.
7. Wait 100  $\mu$ s.
8. Check VBUSVALID\_IRQ bit. If it is set, and 5 V is present and the 4P2 rail was enabled, then repeat the sequence for enabling the 4P2 regulator and DCDC from 4P2. This bit indicates that the DCDC has tried to source from the battery, even if 4P2 sourcing is enabled. This is because a VBUSVALID false condition automatically disables the 4P2 regulators and DCDC source, so the DCDC then falls back to battery sourcing.
9. Clear VBUSDROOP, VBUSVALID, and the output rails IRQ bits as needed.
10. Restore the output rail ENIRQ bits, the VBUSVALID\_THRSH level, VBUSVALID\_5VDETECT value, VBUSVALID ENIRQ, and VBUSDROOP ENIRQ to their original values.

## 2349 The DAC DMA halts unexpectedly

### Description:

Some devices can experience an issue where audio output from the DAC halts unexpectedly. This occurs because of stopping and starting the DAC output several times.

A device having such an issue may show one of the following symptoms when a playback is attempted:

- no audio output (as the DAC DMA is halted)
- audio output is available at the beginning of playback for few seconds and then it is disabled (as the DAC DMA is halted)

### Workaround:

Set the `HW_AUDIOOUT_CTRL DMAWAIT_COUNT` to 31 (recommended to be set during the peripheral initialization) whenever the `AUDIOOUT` uses the DMA.

There are no known side effects for this workaround.

## **How to Reach Us:**

### **Home Page:**

[www.freescale.com](http://www.freescale.com)

### **Web Support:**

<http://www.freescale.com/support>

### **USA/Europe or Locations Not Listed:**

Freescale Semiconductor, Inc.  
Technical Information Center, EL516  
2100 East Elliot Road  
Tempe, Arizona 85284  
1-800-521-6274 or  
+1-480-768-2130  
[www.freescale.com/support](http://www.freescale.com/support)

### **Europe, Middle East, and Africa:**

Freescale Halbleiter Deutschland GmbH  
Technical Information Center  
Schatzbogen 7  
81829 Muenchen, Germany  
+44 1296 380 456 (English)  
+46 8 52200080 (English)  
+49 89 92103 559 (German)  
+33 1 69 35 48 48 (French)  
[www.freescale.com/support](http://www.freescale.com/support)

### **Japan:**

Freescale Semiconductor Japan Ltd.  
Headquarters  
ARCO Tower 15F  
1-8-1, Shimo-Meguro, Meguro-ku  
Tokyo 153-0064  
Japan  
0120 191014 or  
+81 3 5437 9125  
[support.japan@freescale.com](mailto:support.japan@freescale.com)

### **Asia/Pacific:**

Freescale Semiconductor China Ltd.  
Exchange Building 23F  
No. 118 Jianguo Road  
Chaoyang District  
Beijing 100022  
China  
+86 10 5879 8000  
[support.asia@freescale.com](mailto:support.asia@freescale.com)

### **For Literature Requests Only:**

Freescale Semiconductor  
Literature Distribution Center  
P.O. Box 5405  
Denver, Colorado 80217  
1-800 441-2447 or  
+1-303-675-2140  
Fax: +1-303-675-2150  
[LDCForFreescaleSemiconductor@hibbertgroup.com](mailto:LDCForFreescaleSemiconductor@hibbertgroup.com)

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale, the Freescale logo, CodeWarrior, ColdFire, PowerQUICC, StarCore, and Symphony are trademarks of Freescale Semiconductor, Inc. Reg. U.S. Pat. & Tm. Off. CoreNet, QorIQ, QUICC Engine, and VortiQa are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. ARM is the registered trademark of ARM Limited.

© 2009-2010 Freescale Semiconductor, Inc.

Document Number: i.MX23CE

Rev. 2

04/2010

