Chip Errata for the i.MX 6Solo/6DualLite

This document details the silicon errata known at the time of publication for the i.MX 6Solo/6DualLite multimedia applications processors.

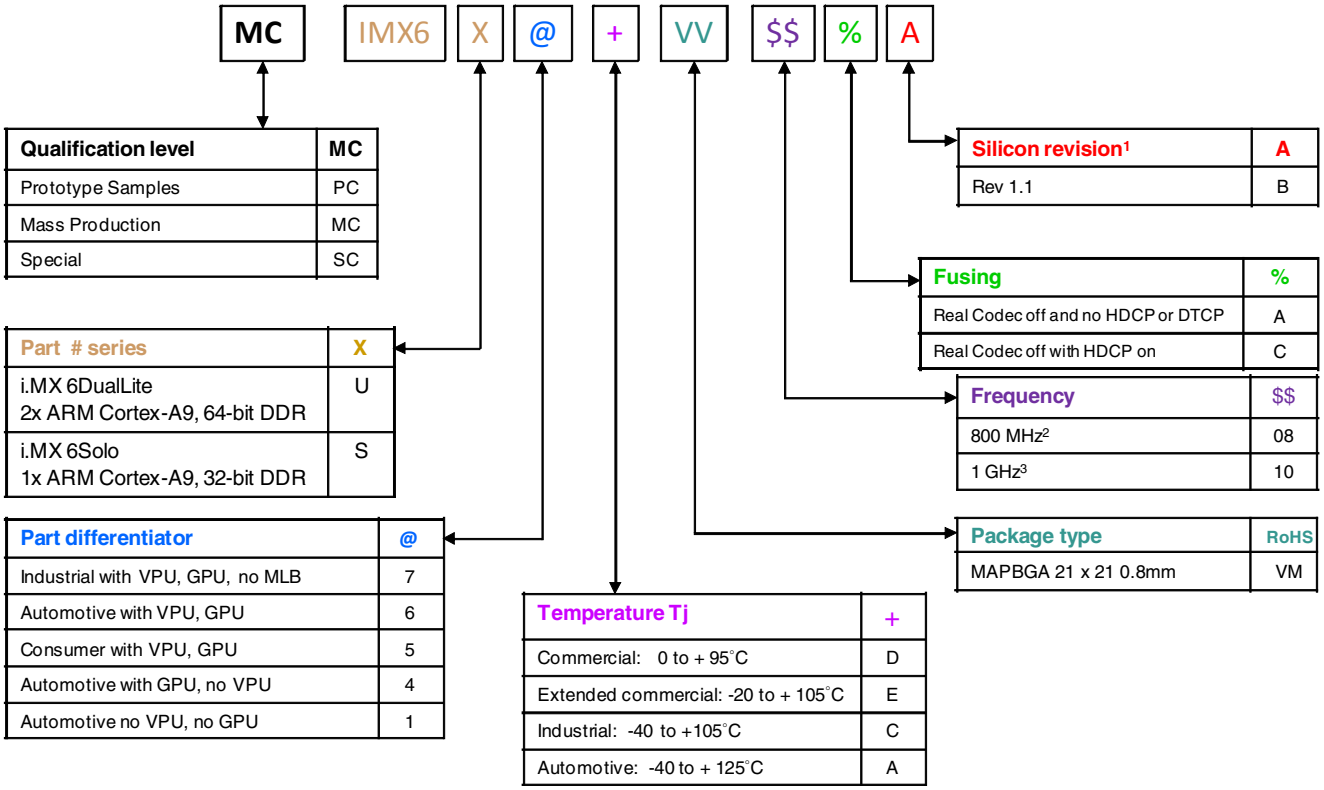
Table 1 provides a revision history for this document.

Table 1. Document Revision History

Rev. Number	Date	Substantive Changes
Rev 2	5/2013	• Deleted ERR004353 • Added the following errata: – ERR006308 – ERR006358 – ERR006687 • Updated the following: – ERR004446 – ERR005829
Rev. 1.1	2/2013	Restored pages omitted in Rev. 1.
Rev 1	1/2013	• Added the following: – ERR006223 – ERR006259 – ERR006281
Rev 0	10/2012	Initial public release.

freescale™ semiconductor

Figure 1 provides a cross-reference to match the revision code to the revision level marked on the device.



1. See the freescale.com\imx6series Web page for latest information on the available silicon revision.
2. If a 24 MHz input clock is used (required for USB), the maximum SoC speed is limited to 792 MHz.
3. If a 24 MHz input clock is used (required for USB), the maximum SoC speed is limited to 996 MHz.

Figure 1. Revision Level to Part Marking Cross-Reference

Table 2 summarizes errata on the i.MX 6Solo/6DualLite.

Table 2. Summary of Silicon Errata

Errata	Name	Solution	Page
Analog			
ERR005852	Analog: Transition from Deep Sleep Mode to LDO Bypass Mode may cause the slow response of the VDDARM_CAP output	No fix scheduled	9
ARM®			
ERR003717	ARM: 740657—Global Timer can send two interrupts for the same event	No fix scheduled	10
ERR003718	ARM: 743622—Faulty logic in the Store Buffer may lead to data corruption	No fix scheduled	12
ERR003719	ARM: 751469—Overflow in PMU counters may not be detected	No fix scheduled	14
ERR003720	ARM/MP: 751472—An interrupted ICIALLUIS operation may prevent the completion of a following broadcast operation	No fix scheduled	15
ERR003721	ARM: 751473—Under very rare circumstances, Automatic Data prefetcher can lead to deadlock or data corruption	No fix scheduled	17
ERR003723	ARM: 751476—May miss a watchpoint on the second part of an unaligned access that crosses a page boundary	No fix scheduled	18
ERR003724	ARM: 754322—Possible faulty MMU translations following an ASID switch	No fix scheduled	19
ERR003725	ARM: 725631—ISB is counted in Performance Monitor events 0x0C and 0x0D	No fix scheduled	21
ERR003726	ARM: 729817—MainID register alias addresses are not mapped on Debug APB interface	No fix scheduled	22
ERR003727	ARM: 729818—In debug state, next instruction is stalled when sdabort flag is set, instead of being discarded	No fix scheduled	23
ERR003728	ARM: 740661—Event 0x74 / PMUEVENT[38:37] may be inaccurate	No fix scheduled	24
ERR003729	ARM: 740663—Event 0x68 / PMUEVENT[9:8] may be inaccurate	No fix scheduled	25
ERR003730	ARM: 743623—Bad interaction between a minimum of seven PLDs and one Non-Cacheable LDM can lead to a deadlock	No fix scheduled	27
ERR003731	ARM: 743626—An imprecise external abort, received while the processor enters WFI, may cause a processor deadlock	No fix scheduled	29
ERR003732	ARM: 751471—DBGPCSR format is incorrect	No fix scheduled	30
ERR003733	ARM: 751480—Conditional failed LDREXcc can set the exclusive monitor	No fix scheduled	32
ERR003734	ARM: 752519—An imprecise abort may be reported twice on non-cacheable reads	No fix scheduled	33
ERR003735	ARM: 754323—Repeated Store in the same cache line may delay the visibility of the Store	No fix scheduled	34
ERR003736	ARM: 756421—Sticky Pipeline Advance bit cannot be cleared from debug APB accesses	No fix scheduled	36
ERR003737	ARM: 757119—Some “Unallocated memory hint” instructions generate an UNDEFINED exception instead of being treated as NOP	No fix scheduled	37
ERR003738	ARM: 751475—Parity error may not be reported on full cache line access (eviction / coherent data transfer / cp15 clean operations)	No fix scheduled	38

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
ERR003739	ARM: 751470—Imprecise abort on the last data of a cache linefill may not be detected	No fix scheduled	39
ERR003741	ARM/PL310: 729815—The “High Priority for SO and Dev reads” feature can cause Quality of Service issues to cacheable read transactions	No fix scheduled	40
ERR003743	ARM/PL310: 754670—A continuous write flow can stall a read targeting the same memory area	No fix scheduled	41
ERR004324	ARM/MP: 761319—Ordering of read accesses to the same memory location may not be ensured	No fix scheduled	42
ERR004325	ARM/MP: 764369—Data or unified cache line maintenance operation by MVA may not succeed on an Inner Shareable memory region	No fix scheduled	43
ERR004326	ARM/MP: 761321—MRC and MCR are not counted in event 0x68	No fix scheduled	45
ERR004327	ARM/MP: 764319—Read accesses to DBGPRSR and DBGOSLSR may generate an unexpected UNDEF	No fix scheduled	46
ERR005175	ARM/MP: 771221—PLD instructions may allocate data in the Data Cache regardless of the Cache Enable bit value	No fix scheduled	47
ERR005183	ARM/MP: 771224—Visibility of Debug Enable access rights to enable/disable tracing is not ensured by an ISB	No fix scheduled	48
ERR005185	ARM/MP: 771225—Speculative cacheable reads to aborting memory region clear the internal exclusive monitor, may lead to livelock	No fix scheduled	49
ERR005187	ARM/MP: 771223—Parity errors on BTAC and GHB are reported on PARITYFAIL[7:6], regardless of the Parity Enable bit value	No fix scheduled	51
ERR005198	ARM/PL310: 780370—DATAERR, TAGERR, and Tag parity errors are incorrectly sampled by the eviction buffer, leading to data corruption	No fix scheduled	52
ERR005200	ARM/MP: 765569—Prefetcher can cross 4 KB boundary if offset is programmed with value 23	No fix scheduled	55
ERR005382	ARM/MP: 775419—PMU event 0x0A (exception return) might count twice the LDM PC ^ instructions with base address register write-back	No fix scheduled	56
ERR005383	ARM/MP: 775420—A data cache maintenance operation that aborts, followed by an ISB and without any DSB in-between, might lead to deadlock	No fix scheduled	57
ERR005385	ARM/MP: 782772—A write to Strongly Ordered memory region, followed by a condition-failed LDREX, might deadlock the processor	No fix scheduled	58
ERR005386	ARM/MP: 782773—Updating a translation entry to move a page mapping might erroneously cause an unexpected translation fault	No fix scheduled	59
ERR005387	ARM/MP: 782774—A spurious event 0x63, “STREX passed,” can be reported on an LDREX that is preceded by a write to Strongly Ordered memory region	No fix scheduled	61
ERR005391	ARM: Debug CTI interrupt can cause a system deadlock when power gating the core	No fix scheduled	62
ERR006259	ARM: Debug/trace functions (PMU, PTM and ETB) are disabled with absence of JTAG_TCK clock after POR	No fix scheduled	63

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
CAAM			
ERR005766	CAAM: CAAM cannot handle interleaved READ data “beats” returned by two different slaves in the system, in reply to two different AXI-ID accesses	No fix scheduled	64
CCM			
ERR006223	CCM: Failure to resume from Wait/Stop mode with power gating	No fix scheduled	65
EIM			
ERR004446	EIM: AUS mode is nonfunctional for devices larger than 32 MB	No fix scheduled	66
ENET			
ERR004512	ENET: 1 Gb Ethernet MAC (ENET) system limitation	No fix scheduled	67
ERR005783	ENET: ENET Status FIFO may overflow due to consecutive short frames	No fix scheduled	68
ERR005895	ENET: ENET 1588 channel 2 event capture mode not functional	No fix scheduled	69
ERR006358	ENET: Write to Transmit Descriptor Active Register (ENET_TDAR) is ignored	No fix scheduled	70
ERR006687	ENET: Only the ENET wake-up interrupt request can wake the system from Wait mode.	No fix scheduled	71
EPDC			
ERR004573	EPDC: Collision status must be read before clearing IRQ	No fix scheduled	72
ERR005313	EPDC: Incorrect data fetched when the buffer update width is 2048 pixels or greater	No fix scheduled	73
ERR005991	EPDC: Memory access may lock up when two continuous unaligned burst accesses return data back to back	No fix scheduled	74
ERR005992	EPDC: EPDC may not detect collision correctly on update buffer boundary when using >16 LUTs AND with unaligned update buffer AND have different LUT updating adjacent area	No fix scheduled	75
EXSC			
ERR005828	EXSC: Protecting the EIM memory map region causes unpredictable behavior	No fix scheduled	76
FlexCAN			
ERR005829	FlexCAN: FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment during the arbitration process	No fix scheduled	77
GPU			
ERR004341	GPU2D: Accessing GPU2D when it is power-gated will cause a deadlock in the system	No fix scheduled	79
ERR005908	GPU2D: Image quality degradation observed for stretch blits when the stretch factor is exactly an integer	No fix scheduled	80
ERR004300	GPU3D: L1 cache performance drop	No fix scheduled	81
ERR004484	GPU3D: L1 cache “Write Address Data” pairing error	No fix scheduled	82

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
ERR005216	GPU3D: Black texels in Android App Singularity 3D	No fix scheduled	83
HDMI			
ERR004366	HDMI: 9000482480—ARM core read operation returns incorrect data	No fix scheduled	84
ERR005172	HDMI: Under certain circumstances, the HDCP may transmit incorrect Ainfo value, causing a failure on the receiver side	No fix scheduled	85
I/O			
ERR004307	I/O: MIPI_HSI, USB_HSIC, and ENET I/O interfaces should not be configured to Differential input mode	No fix scheduled	86
MIPI			
ERR005190	MIPI: CSI2 Data lanes are activated before the HS clock from the CSI Tx side (camera) starts	No fix scheduled	87
ERR005191	MIPI: Corruption of short command packets with Word Count (WC) greater than 16'hFFEE, during video mode transmission by the MIPI Generic Interface	No fix scheduled	88
ERR005192	MIPI: Reverse direction long packets with no payload incorrectly issue a CRC error for MIPI DSI	No fix scheduled	89
ERR005193	MIPI: The bits for setting the MIPI DSI video mode cannot be changed on the fly	No fix scheduled	90
ERR005194	MIPI: On MIPI DSI, there is a possible corruption of the video packets caused by overlapping of the current line over the next line, if the configuration is programmed incorrectly when using the DPI interface	No fix scheduled	91
ERR005195	MIPI: Incorrect blanking packet may be sent by the MIPI DSI interface	No fix scheduled	92
ERR005196	MIPI: Error Interrupt generated by the MIPI CSI interface for certain legal packet types	No fix scheduled	93
ERR005197	MIPI: Null and Blanking data packets activate 'dvalid' signal	No fix scheduled	94
MMDC			
ERR005778	MMDC: DDR Controller's measure unit may return an incorrect value when operating below 100 MHz	No fix scheduled	95
PCIe			
ERR003747	PCIe: 9000436491—Reading the Segmented Buffer Depth Port Logic registers returns all zeros	No fix scheduled	96
ERR003748	PCIe: 9000427578—Root ports with address translation drop inbound requests, without reporting an error	No fix scheduled	97
ERR003749	PCIe: 9000426180—MSI Interrupt Controller Status Register bit not cleared after being written by software	No fix scheduled	98
ERR003751	PCIe: 9000413207—PME Requester ID overwritten when two PMEs are received consecutively	No fix scheduled	99
ERR003753	PCIe: 9000405932—AXI/AHB Bridge Slave does not return a response to an outbound non-posted request	No fix scheduled	100

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
ERR003754	PCIe: 9000403702—AHB/AXI Bridge Master responds with UR status instead of CA status for inbound MRd requesting greater than CX_REMOTE_RD_REQ_SIZE	No fix scheduled	101
ERR003755	PCIe: 9000402443—Uncorrectable Internal Error Severity register bit has incorrect default value	No fix scheduled	102
ERR003756	PCIe: 9000387484—LTSSM: Software-initiated transitions to Disabled, Hot Reset, Configuration, or Loopback states sometimes take longer than expected	No fix scheduled	103
ERR003757	PCIe: 9000448152—Internal Address Translation Unit (iATU): Inbound Vendor Defined Message (VDM) 'ID Match Mode' is not functional	No fix scheduled	104
ERR003758	PCIe: 9000441819—Upstream Port does not transition to Recovery after receiving TS OSs during "ENTER_L2 negotiation"	No fix scheduled	105
ERR003759	PCIe: 9000439510—Internal Address Translation Unit (iATU) can sometimes overwrite Outbound (Tx) Vendor Messages and MSIs	No fix scheduled	106
ERR003760	PCIe: 9000439175—Poisoned Atomic Op requests targeting RTRGT0 receive UR response instead of CA response	No fix scheduled	107
ERR004297	PCIe: 9000336356—Link configuration sometimes proceeds when incorrect TS Ordered Sets are received	No fix scheduled	108
ERR004298	PCIe: 9000471173—Bad DLLP error status checking is too strict	No fix scheduled	109
ERR004299	PCIe: 9000493959—L1 ASPM incorrectly entered after link down event during L1 ASPM entry negotiation	No fix scheduled	110
ERR004321	PCIe: 9000470913—Power Management Control: Core might enter L0s/L1 before Retry buffer is empty	No fix scheduled	111
ERR004374	PCIe: 9000487440—TLP sometimes unnecessarily replayed	No fix scheduled	113
ERR004489	PCIe: 9000505660—PCIe2 receiver equalizer settings	No fix scheduled	114
ERR004490	PCIe: 9000514662—LTSSM delay when moving from L0 to recovery upon receipt of insufficient TS1 Ordered Sets	No fix scheduled	115
ERR004491	PCIe: 9000507633—TLP might be replayed an extra time before core enters recovery	No fix scheduled	116
ERR005186	PCIe: The PCIe Controller Core Does Not Send Enough TS2 Ordered Sets During Link Retrain And Speed Change	No fix scheduled	117
ERR005188	PCIe: The PCIe Controller cannot exit successfully L1 state of LTSSM when the Core Clock is removed	No fix scheduled	118
ERR005189	PCIe: PCIe Gen2/Gen3 Hardware Autonomous Speed Disable Bit In Configuration Register is not sticky	No fix scheduled	119
ERR005723	PCIe: PCIe does not support L2 power down	No fix scheduled	120
ROM			
ERR005645	ROM: Normal SD clock speed (SDR12) not selectable in SD/SDXC boot mode	No fix scheduled	121
ERR005768	ROM: In rare cases, secondary image boot flow may not work due to mis-sampling of the WDOG reset	No fix scheduled	122

Table 2. Summary of Silicon Errata (continued)

Errata	Name	Solution	Page
SSI			
ERR003778	SSI: In AC97, 16-bit mode, received data is shifted by 4-bit locations	No fix scheduled	123
ERR005764	SSI: AC97 receive data may be wrong when clock ratio between external clock to ipg is higher than 1:8	No fix scheduled	124
USB			
ERR004534	USB: Wrong HS disconnection may be generated after resume	No fix scheduled	125
ERR006281	USB: Incorrect DP/DN state when only VBUS is applied	No fix scheduled	126
ERR006308	USB: Host non-doubleword –aligned buffer address can cause host to hang on OUT Retry		127
ERR004535	USB: USB suspend and resume flow clarifications	No fix scheduled	128
VPU			
ERR004349	VPU: Cannot decode Sorenson Spark Version 0 bitstream	No fix scheduled	129
XTAL			
ERR005777	XTAL: In some cases, the 24 MHz oscillator start-up is slow	No fix scheduled	130

ERR005852 Analog: Transition from Deep Sleep Mode to LDO Bypass Mode may cause the slow response of the VDDARM_CAP output

Description:

Normally, the VDDARM_CAP supply takes only approximately 40 µs to raise to the correct voltage when exiting from Deep Sleep (DSM) mode, if the LDO is enabled. If the LDO bypass mode is selected, the VDDARM_CAP supply voltage will drop to approximately 0 V when entering and when exiting from DSM, even though the VDDARM_IN supply is already stable, the VDDARM_CAP supply will take about 2 ms to rise to the correct voltage.

Projected Impact:

ARM core may fail to resume.

Workarounds:

The software workaround to prevent this issue it to switch to analog bypass mode (0x1E), prior to entering DSM, and then, revert to the normal bypass mode, when exiting from DSM.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR003717 ARM: 740657—Global Timer can send two interrupts for the same event

Description:

The Global Timer can be programmed to generate an interrupt request to the processor when it reaches a given programmed value. Due to the erratum, when the Global Timer is programmed not to use the auto-increment feature, it may generate two interrupt requests instead of one.

Conditions:

The Global Timer Control register is programmed with the following settings:

- Bit[3] = 1'b0 – Global Timer is programmed in "single-shot" mode
- Bit[2] = 1'b1 – Global Timer IRQ generation is enabled
- Bit[1] = 1'b1 – Global Timer value comparison with Comparator registers is enabled
- Bit[0] = 1'b1 – Global Timer count is enabled

With these settings, an IRQ is generated to the processor when the Global Timer value reaches the value programmed in the Comparator registers.

The Interrupt Handler then performs the following sequence:

1. Read the ICCIAR (Interrupt Acknowledge) register
2. Clear the Global Timer flag
3. Modify the comparator value to set it to a higher value
4. Write the ICCEOIR (End of Interrupt) register

Under these conditions, due to the erratum, the Global Timer may generate a second (spurious) interrupt request to the processor at the end of this Interrupt Handler sequence.

Projected Impact:

The erratum creates spurious interrupt requests in the system.

Workarounds:

Because the erratum only happens when the Global Timer is programmed in "single-shot" mode, that is, when it does not use the auto-increment feature, a first possible workaround could be to program the Global Timer to use the auto-increment feature.

If this solution does not work, a second workaround could be to modify the Interrupt Handler to avoid the offending sequence. This is achieved by clearing the Global Timer flag after having incremented the Comparator register value.

Then, the correct code sequence for the Interrupt Handler should look as below:

1. Read the ICCIAR (Interrupt Acknowledge) register
2. Modify the comparator value to set it to a higher value
3. Clear the Global Timer flag

4. Clear the Pending Status information for Interrupt 27 (Global Timer interrupt) in the Distributor of the Interrupt Controller.
5. Write the ICCEOIR (End of Interrupt) register

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP. The BSP does not use ARM global timer.

ERR003718 ARM: 743622—Faulty logic in the Store Buffer may lead to data corruption

Description:

Under very rare conditions, a faulty optimization in the Cortex-A9 Store Buffer may lead to data corruption.

Conditions:

The code sequence which exhibits the failure requires at least five Cacheable writes in 64-bit data chunk:

- Three of the writes must be in the same cache line
- Another write must be in a different cache line
- All of the above four writes hit in the L1 data cache
- A fifth write is required in any of the above two cache lines that fully writes a 64-bit data chunk

With the above code sequence, under very rare circumstances, this fifth write may get corrupted, with the written data either being lost, or being written in another cache line.

The conditions under which the erratum can occur are extremely rare, and require the coincidence of multiple events and states in the Cortex-A9 micro-architecture.

As an example: let's assume A, A', A'', and A''' are all in the same cache line—B and B' are in another cache line. The following code sequence may trigger the erratum:

```
STR A
STR A′
STR A′′
STR B
STR A′′′ (or STR B′)
```

At the time where the first four STR are in the Cortex-A9 Store Buffer, and the fifth STR arrives at a very precise cycle in the Store Buffer input stage, then the fifth STR may not see its cache line dependency on the previous STR instructions. Because of this, in cases when the cache line A or B gets invalidated due to a coherent request from another CPU, the fifth STR may write in a faulty cache line, causing data corruption.

An alternative version of the erratum may happen even without a coherent request — In the case when the fifth STR is a 64-bit write in the same location as one of A, A', A'', then the erratum may also be exhibited. Note that this is a quite uncommon scenario because it requires a first write to a memory location that is immediately and fully overwritten.

Projected Impact:

When it occurs, this erratum creates a data corruption.

Workarounds:

A software workaround is available for this erratum that requires setting bit[6] in the undocumented Diagnostic Control register, placed in CP15 c15 0 c0 1.

The bit can be written in Secure state only, with the following Read/Modify/Write code sequence:

```
MRC p15,0,rt,c15,c0,1
ORR rt,rt,#0x40
MCR p15,0,rt,c15,c0,1
```

When this bit is set, the “fast lookup” optimization in the Store Buffer is disabled, which will prevent the failure to happen.

Setting this bit has no visible impact on the overall performance or power consumption of the processor.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR003719 ARM: 751469—Overflow in PMU counters may not be detected

Description:

Overflow detection logic in the Performance Monitor Counters is faulty, and under certain timing conditions, the overflow may remain undetected. In this case, the Overflow Flag Status Register (PMOVSR) is not updated as it should, and no interrupt is reported on the corresponding PMUIRQ line.

Projected Impact:

PMU overflow detection is not reliable.

Workarounds:

The workaround for this erratum involves setting two PMU counters to count the same event, and explicitly offset them by 1 at the start of the count. This can be achieved with the following sequence:

1. Disable PMU count
2. Set Counter0 to value N where N is an arbitrary count value
3. Set Counter1 to value (N+1)
4. Enable PMU count

This ensures that at least one of the two counters will detect the overflow. Most of the time, each of the two counters will trigger, so using this workaround requires the software to reset both counters when the first one triggers.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround to be applied in a future BSP release

ERR003720 ARM/MP: 751472—An interrupted ICIALLUIS operation may prevent the completion of a following broadcast operation

Description:

In an MPCore configuration with two or more processors working in SMP mode with maintenance operation broadcast enabled, if a processor is interrupted while executing an ICIALLUIS operation, and performs another broadcast maintenance operation during its Interrupt Service Routine, then this second operation may not be executed on other processors in the cluster.

Conditions:

The erratum requires an MPCore configuration with two or more CPUs working in SMP mode. One processor has interrupts enabled, and Cache and TLB maintenance broadcast enabled too (ACTLR.FW=1'b1). This processor executes an ICIALLUIS (invalidates all instruction caches Inner Shareable to Point of Unification). This instruction is executed on the processor, and also broadcast to other processors in the MPCore cluster. The processor then receives an interrupt (IRQ or FIQ), which interrupts the ICIALLUIS operation.

During the Interrupt Service Routine, the processor executes any other Cache or TLB maintenance operation which is also broadcast to other processors in the MPCore cluster. If the other processors in the cluster receive this second maintenance operation before having completed the first ICIALLUIS operation, then the erratum occurs, as the other processors will not execute the second maintenance operation. This is because there is no "stacking" mechanism for acknowledge answers between the processors, so that the acknowledge request sent to signify the completion of the ICIALLUIS will be interpreted by the originating processor as an acknowledge for the second maintenance operation.

Projected Impact:

Due to the erratum, the processor may end up with corrupted entries in the Cache or in the TLB, leading to possible failures in the system.

Workarounds:

A software workaround is available for this erratum that involves setting bit[11] in the undocumented Diagnostic Control register, placed in CP15 c15 0 c0 1.

This bit can be written in Secure state only, with the following Read/Modify/Write code sequence:

```
MRC p15,0,rt,c15,c0,1
ORR rt,rt,#0x800
MCR p15,0,rt,c15,c0,1
```

When it is set, this bit prevents CP15 maintenance operations to be interrupted.

Using this software workaround is not expected to cause any visible impact on the system.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR003721 ARM: 751473—Under very rare circumstances, Automatic Data prefetcher can lead to deadlock or data corruption

Description:

Under very rare timing circumstances, the automatic Data prefetcher may cause address hazard issues, possibly leading to a data corruption or a deadlock of the processor.

Conditions:

The erratum can only happen when the Data Cache and MMU are enabled in the following cases:

- On all memory regions marked as Write-Back Non-Shared, when the Data Prefetcher in L1 is enabled (ACTLR[2]=1’b1), regardless of the ACTLR.SMP bit.
- On all memory regions marked as Write-Back Shared, when the Data Prefetch Hint in L2 is enabled (ACTLR[1]=1’b1), and when the processor is in SMP mode (ACTLR.SMP=1’b1).

Projected Impact:

When the bug happens, a data corruption or a processor deadlock can happen.

Workarounds:

The workaround for this erratum requires not enabling the automatic Data Prefetcher by keeping ACTRL[2:1]=2’b00, which is the default value on exit from reset.

Although this feature may show significant performance gain on a few synthetic benchmarks, it usually has no impact on real systems. It means, this workaround is not expected to cause any visible impact on final products.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3. Linux BSP keeps ACTRL[2:1]=2’b00.

ERR003723 ARM: 751476—May miss a watchpoint on the second part of an unaligned access that crosses a page boundary

Description:

Under rare conditions, a watchpoint on the second part of an unaligned access that crosses a 4 KB page boundary and that is missed in the micro-TLB for the second part of its request may be undetected.

The erratum requires a previous conditional instruction that accesses the second 4 KB memory region (= where the watchpoint is set), is missed in the micro-TLB, and is condition failed. The erratum also requires that no other micro-TLB miss occurs between this conditional failed instruction and the unaligned access. This implies that the unaligned access must hit in the micro-TLB for the first part of its request.

Projected Impact:

A valid watchpoint trigger is missed.

Workarounds:

In case, a watchpoint is set on any of the first 3 bytes of a 4 KB memory region, and unaligned accesses are not being faulted, then the erratum may happen.

The workaround then requires setting a guard watchpoint on the last byte of the previous page, and dealing with any “false positive” matches as and when they occur.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003724 ARM: 754322—Possible faulty MMU translations following an ASID switch

Description:

A micro-TLB entry may be corrupted following an ASID switch, possibly corrupting subsequent MMU translations. The issue requires that a speculative explicit memory access is executed, but is speculative failed. The speculation fails if the memory access occurred under a mispredicted branch or in case it is conditional and condition failed.

This speculative memory access may miss in the TLB, and cause a Page Table Walk. The erratum occurs when the Page Table Walk starts, prior to the ASID switch code sequence, but completes afterwards.

In this case, the microTLB will get a new entry allocated with this new TLB entry, corresponding to the “old” ASID. The issue is that the micro-TLB does not register the ASID value, so that MMU translations which should happen with the new ASID following the ASID switch may hit in this stale micro-TLB entry, and get corrupted.

It is, however, important to note that there is no Trustzone Security risks because the Security state of the access is registered in the micro-TLB, and consequently, cannot be corrupted.

Projected Impact:

The errata may cause MMU translation corruptions.

Workarounds:

The workaround for this erratum involves adding a DSB in the ASID switch code sequence. The ARM architecture only mandates ISB before and after the ASID switch. Adding a DSB prior to the ASID switch ensures that the Page Table Walk completes prior to the ASID change, so that no stale entry can be allocated in the micro-TLB.

The examples in the ARM Architecture Reference Manual for synchronizing the change in the ASID and TTBR need to be changed as follows:

The sequence:

```
Change ASID to 0
ISB
Change Translation Table Base Register
ISB
Change ASID to new value
```

becomes

```
DSB
Change ASID to 0
ISB
Change Translation Table Base Register
ISB
DSB
Change ASID to new value
```

the sequence:

```
Change Translation Table Base Register to the global-only mappings
ISB
Change ASID to new value
ISB
Change Translation Table Base Register to new value
```

becomes

```
Change Translation Table Base Register to the global-only mappings
ISB
DSB
Change ASID to new value
ISB
Change Translation Table Base Register to new value
```

and the sequence:

```
Set TTBCR.PD0 = 1
ISB
Change ASID to new value
Change Translation Table Base Register to new value
ISB
Set TTBCR.PD0 = 0
```

becomes

```
Set TTBCR.PD0 = 1
ISB
DSB
Change ASID to new value
Change Translation Table Base Register to new value
ISB
Set TTBCR.PD0 = 0
```

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR003725 ARM: 725631—ISB is counted in Performance Monitor events 0x0C and 0x0D

Description:

The ISB is implemented as a branch in the Cortex-A9 micro-architecture. This implies that events 0x0C (software change of PC) and 0x0D (immediate branch) are asserted when an ISB occurs. This is not compliant with the ARM architecture.

Projected Impact:

The count of events 0x0C and 0x0D are not 100% precise when using the Performance Monitor counters, due to the ISB being counted in addition to the real software changes to PC (for 0x0C) and immediate branches (0x0D).

The erratum also causes the corresponding PMUEVENT bits to toggle in case an ISB is executed.

- PMUEVENT[13] relates to event 0x0C
- PMUEVENT[14] relates to event 0x0D

Workarounds:

Count ISB instructions along with event 0x90. The user should subtract this ISB count from the results obtained in events 0x0C and 0x0D, to obtain the precise count of software change of PC (0x0C) and immediate branches (0x0D).

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003726 ARM: 729817—MainID register alias addresses are not mapped on Debug APB interface

Description:

The ARM Debug Architecture specifies registers 838 and 839 as “Alias of the MainID register”. They should be accessible through the APB Debug interface at addresses 0xD18 and 0xD1C.

In Cortex-A9, the two alias addresses are not implemented. A read access at any of these two addresses returns 0, instead of the MIDR value.

Note that read accesses to these two registers through the internal CP14 interface are trapped to UNDEF, which is compliant with the ARM Debug architecture. So, the erratum only applies to the alias addresses through the external Debug APB interface.

Projected Impact:

If the debugger or any other external agent tries to read the MIDR register using the alias addresses, it will get a faulty answer (0x0), which can cause all sorts of malfunction in the debugger afterwards.

Workarounds:

The workaround for this erratum requires always accessing the MIDR at its original address, 0xD00, and not at any of its alias addresses.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003727 ARM: 729818—In debug state, next instruction is stalled when sdabort flag is set, instead of being discarded

Description:

When the processor is in debug state, an instruction written to the ITR after a Load/Store instruction that aborts gets executed on clearing the SDABORT_l, instead of being discarded.

Projected Impact:

Different failures can happen due to the instruction being executed when it should not. In most cases, it is expected that the failure will not cause any significant problem.

Workarounds:

There are a selection of workarounds with increasing complexity and decreasing impact. In each case, the impact is a loss of performance when debugging:

- Do not use stall mode
- Do not use stall mode when doing load/store operations
- Always check for a sticky abort after issuing a load/store operation in stall mode (the cost of this probably means the above second workaround is a preferred alternative)
- Always check for a sticky abort after issuing a load/store operation in stall mode, before issuing any further instructions that may corrupt important target state (such as, further load/store instructions, instructions that write to “live” registers [VFP, CP15, etc.])

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003728 ARM: 740661—Event 0x74 / PMUEVENT[38:37] may be inaccurate

Description:

Event 0x74 counts the total number of Neon instructions passing through the register rename pipeline stage. Due to the erratum, the “stall” information is not taken into account. So, one Neon instruction that remains for n cycles in the register rename stage is counted as n Neon instructions. As a consequence, the count of event 0x74 may be corrupted, and cannot be relied upon. The event is also reported externally on PMUEVENT[38:37], which suffers from the same inaccuracy.

Projected Impact:

The implication of this erratum is that Neon instructions cannot be counted reliably in the versions of the product that are affected by this erratum.

Workarounds:

No workaround is possible to achieve the required functionality of counting how many Neon instructions are executed (or renamed) in the processor.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003729 ARM: 740663—Event 0x68 / PMUEVENT[9:8] may be inaccurate

Description:

Event 0x68 counts the total number of instructions passing through the register rename pipeline stage. Under certain conditions, some branch-related instructions may pass through this pipeline stage without being counted. As a consequence, event 0x68 may be inaccurate, lower than expected. The event is also reported externally on PMUEVENT[9:8], which suffers from the same inaccuracy.

Conditions:

The erratum occurs when the following conditions are met:

- Events are enabled
- One of the PMU counters is programmed to count event 0x68 — number of instructions passing through the register rename stage. Alternatively, an external component counts, or relies on, PMUEVENT[9:8].
- A program, containing the following instructions, is executed:
  - A Branch immediate, without Link
  - An ISB instruction
  - An HB instruction, without Link and without parameter, in Thumb2EE state
  - An ENTERX or LEAVEX instruction, in Thumb2 or Thumb2EE state
- The program executed is causing some stalls in the processor pipeline

Under certain timing conditions specific to the Cortex-A9 micro-architecture, a cycle stall in the processor pipeline may “hide” the instructions mentioned above, thus ending with a corrupted count for event 0x68, or a corrupted value on PMUEVENT[9:8] during this given cycle. If the “hidden” instruction appears in a loop, the count difference can be significant.

As an example, let’s consider the following loop:

```
loop mcr 15, 0, r2, cr9, cr12, {4}
    adds r3, #1
    cmp.w r3, #loop_number
    bne.n loop
```

The loop contains four instructions; so, the final instruction count should (approximately) be four times the number of executed loops. In practice, the MCR is causing a pipeline stall that “hides” the branch instruction (bne.n); so, only three instructions are counted per loop, and the final count appears as three times the number of executed loops.

Projected Impact:

The implication of this erratum is that the values of event 0x68 and PMUEVENT[9:8] are imprecise, and cannot be relied upon.

Workarounds:

No workaround is possible to achieve the required functionality of counting how many instructions are precisely passing through the register rename pipeline stage.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003730 ARM: 743623—Bad interaction between a minimum of seven PLDs and one Non-Cacheable LDM can lead to a deadlock

Description:

Under very rare circumstances, a deadlock can happen in the processor when it is handling a minimum of seven PLD instructions, shortly followed by one LDM to an uncacheable memory location.

The LDM is treated as uncacheable in the following cases:

- The LDM is performed while the Data Cache is OFF
- The LDM is targeting a memory region marked as Strongly Ordered, Device, Normal Memory Non-Cacheable, or Normal Memory Write-Through
- The LDM is targeting a memory region marked as Shareable Normal Memory Write-Back, and the CPU is in AMP mode.

Conditions:

The code sequence that exhibits this erratum requires at least seven PLDs, shortly followed by one LDM, to an uncacheable memory region. The erratum happens when the LDM appears on the AXI bus before any of the seven PLDs. This can possibly happen if the first PLD is a miss in the micro-TLB, in that case, it needs to perform a TLB request which may not be serviced immediately because the mainTLB is already performing a Page Table Walk for another resource (for example, instruction side), or because the PLD request itself to the mainTLB is missing and causing a Page Table Walk.

Also note that the above conditions are not sufficient to recreate the failure, as additional rare conditions on the internal state of the processor are necessary to exhibit the errata.

Projected Impact:

The erratum may create a processor deadlock. However, the conditions that are required for this to occur are extremely unlikely to occur in real code sequences.

Workarounds:

The primary workaround might be to avoid the offending code sequence, that is, not to use uncacheable LDM when making intensive use of PLD instructions.

In case the above workaround cannot be done, another workaround for this erratum can be to set bit[20] in the undocumented Control register, which is placed in CP15 c15 0 c0 1.

This bit needs to be written with the following Read/Modify/Write code sequence:

```
MRC p15,0,r0,c15,c0,1
ORR r0,r0,#0x00100000
MCR p15,0,r0,c15,c0,1
```

Setting this bit causes all PLD instructions to be treated as NOPs, with the consequence that code sequences usually using the PLDs, such as the memcpy() routine, may suffer from a visible performance drop.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003731 ARM: 743626—An imprecise external abort, received while the processor enters WFI, may cause a processor deadlock

Description:

An imprecise external abort received while the processor is ready to enter into WFI state may cause a processor deadlock.

Explicit memory transactions can be completed by inserting a DSB before the WFI instruction. However, this does not prevent memory accesses generated by previously issued PLD instructions page table walks associated with previously issued PLD instructions or as a result of the PLE engine.

If an external abort is returned as a result of one of these memory accesses after executing a WFI instruction, the processor can cause a deadlock.

Projected Impact:

In case, the non-explicit memory request receives an external imprecise abort response while the processor is ready to enter into WFI state, the processor may cause a deadlock.

In practical systems, it is not expected that these memory transactions will generate an external abort, as external aborts are usually a sign of significant corruption in the system.

Workarounds:

A possible workaround for this erratum is to protect all memory regions that can return an imprecise external abort with the correct MMU settings, to prevent any external aborts.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003732 ARM: 751471—DBGPCSR format is incorrect

Description:

About the DBGPCSR register, the ARM architecture specifies that:

- DBGPCSR[31:2] contains sampled value of bits [31:2] of the PC.
  The sampled value is an instruction address plus an offset that depends on the processor instruction set state.
- DBGPCSR[1:0] contains the meaning of PC sample value, with the following permitted values:
  - 0b00 ((DBGPCSR[31:2] << 2) - 8) references an ARM state instruction
  - 0bx1 ((DBGPCSR[31:1] << 1) - 4) references a Thumb or ThumbEE state instruction
  - 0b10 IMPLEMENTATION DEFINED

This field encodes the processor instruction set state, so that the profiling tool can calculate the true instruction address by subtracting the appropriate offset from the value sampled in bits [31:2] of the register.

In Cortex-A9, the DBGPCSR samples the target address of executed branches (but possibly still speculative to data aborts), with the following encodings:

- DBGPCSR[31:2] contains the address of the target branch instruction, with no offset
- DBGPCSR[1:0] contains the execution state of the target branch instruction:
  - 0xb00 for an ARM state instruction
  - 0xb01 for a Thumb2 state instruction
  - 0xb10 for a Jazelle state instruction
  - 0xb11 for a Thumb2EE state instruction

Projected Impact:

The implication of this erratum is that the debugger tools neither rely on the architected description for the value of DBGPCSR[1:0], nor remove any offset from DBGPCSR[31:2], to obtain the expected PC value.

Subtracting 4 or 8 to the DBGPCSR[31:2] value would lead to an area of code that is unlikely to have been recently executed, or that could even not contain any executable code.

The same might be true for Thumb instructions at half-word boundaries, in which case PC[1]=1 but DBGPCSR[1]=0, or ThumbEE instructions at word boundaries, with PC[1]=0 and DBGPCSR[1]=1.

In Cortex-A9, because the DBGPCSR is always a branch target (= start of a basic block to the tool), the debugger should be able to spot many of these cases and attribute the sample to the right basic block.

Workarounds:

The debugger tools can find the expected PC value and instruction state by reading the DBGPCSR register, and consider it as described in the Description section.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003733 ARM: 751480—Conditional failed LDREXcc can set the exclusive monitor

Description:

A conditional LDREX may set the internal exclusive monitor of the Cortex-A9 even when its condition fails. So, any subsequent STREX that depends on this LDREXcc may succeed when it should not.

Projected Impact:

The implication of the erratum is that a subsequent STREX may succeed when it should not. So, the memory region protected by the exclusive mechanism can be corrupted if another agent is accessing it at the same time.

Workarounds:

The workaround for this erratum can be not to use conditional LDREX along with non-conditional STREX.

- If no conditional LDREX is used, the erratum cannot be triggered.
- If conditional LDREX is used, the associated STREX should be conditional too with the same condition, so that even if the exclusive monitor is set by the condition failed LDREX, the following STREX will not be executed because it will be condition failed too. For most situations this will naturally be the case anyway.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP. The BSP does not use conditional LDREX.

ERR003734 ARM: 752519—An imprecise abort may be reported twice on non-cacheable reads

Description:

When two outstanding read memory requests to device or non-cacheable normal memory regions are issued by the Cortex-A9, and the first one receives an imprecise external abort, then the second access may falsely report an imprecise external abort.

Conditions:

The erratum can only happen in systems which can generate imprecise external aborts on device or non-cacheable normal memory regions accesses.

Projected Impact:

When the erratum occurs, a second, spurious imprecise abort may be reported to the core when it should not.

In practice, the failure is not expected to cause any significant issues to the system because imprecise aborts are usually unrecoverable failures. Because the spurious abort can only happen following a first imprecise abort—either the first abort is ignored and the spurious abort is then ignored too, or it is acknowledged and probably generates a critical failure in the system.

Workarounds:

There is no practical software workaround for the failure.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003735 ARM: 754323—Repeated Store in the same cache line may delay the visibility of the Store

Description:

The Cortex-A9 implements a small counter that ensures the external visibility of all stores in a finite amount of time, causing an eventual drain of the Merging Store Buffer. This is to avoid an earlier issue, where written data could potentially remain indefinitely in the Store Buffer.

This Store Buffer has merging capabilities, and will continue merging data as long as the write accesses are performed in the same cache line. The issue that causes this erratum is that the draining counter is reset each time a new data merging is performed.

When a code sequence is looping, and keeps on writing data in the same cache line, then the external visibility of the written data may not be ensured.

A livelock situation may consequently occur in case any external agent is relying on the visibility of the written data, and that the writing processor cannot be interrupted while doing its writing loop.

Conditions:

The erratum can only happen on Normal Memory regions.

Two example scenario, which may trigger the erratum, are described below:

- The processor keeps on incrementing a counter: writing the same word at the same address. The external agent (possibly another processor) is polling on this address, waiting for any update of the counter value to proceed.
  The Store Buffer will keep on merging the updated value of the counter in its cache line, so that the external agent will never see any updated value, possibly leading to livelock.
- The processor writes a value in a given word to indicate completion of its task, then keeps on writing data in an adjacent word in the same cache line.
  The external agent keeps on polling the first word memory location to check when the processor has completed its task. The situation is the same as above, as the cache line may remain indefinitely in the merging Store Buffer, creating a possible livelock in the system.

Projected Impact:

This erratum may create performance issues, or worst case livelock scenario, in case the external agent relies on the automatic visibility of the written data in a finite amount of time.

Workarounds:

The recommended workaround for this erratum involves inserting a DMB operation after the faulty write operation in code sequences that may be affected by this erratum. This ensures visibility of the written data by any external agent.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR003736 ARM: 756421—Sticky Pipeline Advance bit cannot be cleared from debug APB accesses

Description:

The Sticky Pipeline Advance bit is bit[25] of the DBGDSCR register. This bit enables the debugger to detect whether the processor is idle. This bit is set to 1 every time the processor pipeline retires one instruction.

A write to DBGDRCR[3] clears this bit.

The erratum is that the Cortex-A9 does not implement any debug APB access to DBGDRCR[3].

Projected Impact:

Due to the erratum, the Sticky Pipeline Advance bit in the DBGDSCR cannot be cleared by the external debugger. In practice, this makes the Sticky Pipeline Advance bit concept unusable on Cortex-A9 processors.

Workarounds:

There is no practical workaround for this erratum. The only possible way to reset the Sticky Pipeline Advance bit is to assert the nDBGRESET input pin on the processor. This obviously has the side effect to reset all debug resources in the concerned processor, and any other additional Coresight components nDBGRESET is connected to.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003737 ARM: 757119—Some “Unallocated memory hint” instructions generate an UNDEFINED exception instead of being treated as NOP

Description:

The ARM architecture specifies that ARM opcodes of the form 11110 100x001 xxxx xxxx xxxx xxxx xxxx are “Unallocated memory hint (treat as NOP)” if the core supports the MP extensions, as the Cortex-A9 does.

The errata is that the Cortex-A9 generates an UNDEFINED exception when bits [15:12] of the instruction encoding are different from 4’b1111, instead of treating the instruction as a NOP.

Projected Impact:

Due to the erratum, an unexpected UNDEFINED exception may be generated. In practice, this erratum is not expected to cause any significant issue, as such instruction encodings are not supposed to be generated by any compiler, nor used by any handcrafted program.

Workarounds:

A possible workaround for this erratum is to modify the instruction encoding with bits[15:12]=4.b1111, so that the instruction is truly treated as a NOP by the Cortex-A9.

If the instruction encoding cannot be modified, the UNDEFINED exception handler has to cope with this case, and emulate the expected behavior of the instruction, that is, do nothing (NOP), before returning to normal program execution.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003738 ARM: 751475—Parity error may not be reported on full cache line access (eviction / coherent data transfer / cp15 clean operations)

Description:

In the Data Cache, parity error detection is faulty. Parity error may not be detected when the line exits from the Data Cache, due to a line replacement, or due to a coherent request from another processor or from the ACP, or because of a CP15 cache clean operation.

Projected Impact:

Due to the erratum, a corrupted line may be evicted or transferred from the processor without the parity error being detected and reported.

Workarounds:

There is no workaround for this erratum.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003739 ARM: 751470—Imprecise abort on the last data of a cache linefill may not be detected

Description:

Data linefills are returned as 4-beat bursts of 64-bit data on the AXI bus. When the first three beat of data are valid, and the fourth one aborts, then the abort is not detected by the processor logic and no abort exception is taken. The processor then behaves as if no abort is reported on the line. It can allocate the line in its Data Cache, and use the aborted data during its program flow.

Conditions:

The processor needs to work with Data Cache enabled, and access some cacheable memory regions (Write Back, either Shared or Non-Shared).

The memory system underneath the processor needs to be able to generate aborts in this memory region, and must be able to generate aborts with a granularity smaller than the cache line.

Projected Impact:

When the erratum triggers, the processor does not detect the abort, so it may use some invalid (aborted) data without entering the Data Abort exception handler as it should normally do.

Workarounds:

None

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003741 ARM/PL310: 729815—The “High Priority for SO and Dev reads” feature can cause Quality of Service issues to cacheable read transactions

Description:

The “High Priority for SO and Dev reads” feature can be enabled by setting the bit[10] of the PL310 Auxiliary Control Register to 1. When enabled, it gives priority to Strongly Ordered and Device reads over cacheable reads in the PL310 AXI master interfaces. When PL310 receives a continuous flow of SO or Device reads, this can prevent cacheable reads, which are misses in the L2 cache, from being issued to the L3 memory system.

Conditions:

The erratum occurs when the following conditions are met:

- The bit[10] “High Priority for SO and Dev reads enable” of the PL310 Auxiliary Control Register is set to 1
- PL310 receives a cacheable read that is a miss in the L2 cache
- PL310 receives a continuous flow of SO or Device reads that take all address slots in the master interface

Projected Impact:

When the above conditions are met, the linefill resulting from the L2 cache miss is not issued till the flow of SO/Device reads stops. Note that each PL310 master interface has four address slots, so that the Quality of Service issue only appears on the cacheable read, if the L1 is able to issue at least four outstanding SO/Device reads.

Workarounds:

A workaround is only necessary in systems that are able to issue a continuous flow of SO or Device reads. In such a case, the workaround is to disable the “High Priority for SO and Dev reads” feature. This is the behavior by default.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003743 ARM/PL310: 754670—A continuous write flow can stall a read targeting the same memory area

Description:

In the ARM L2 cache controller, PL310, hazard checking is done on bits [31:5] of the address.

When a read with Normal Memory (cacheable or not) attributes is received by PL310, hazard checking is performed with the active writes of the store buffer. If an address matching is detected, the read is stalled till the write completes.

Due to this erratum, a continuous flow of writes can stall a read targeting the same memory area.

Conditions:

The erratum occurs when the following conditions are met:

- PL310 receives a continuous write traffic targeting the same address marked with Normal Memory attributes
- While treating this flow, PL310 receives a read targeting the same 32-byte memory area

Projected Impact:

When the conditions above are met, the read might be stalled till the write flow stops.

Note that this erratum does not lead to any data corruption.

Note also that normal software code is not expected to contain long write sequence like the one causing this erratum to occur.

Workarounds:

There is no workaround for this erratum. A workaround is not expected to be necessary for this erratum either.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004324 ARM/MP: 761319—Ordering of read accesses to the same memory location may not be ensured

Description:

The ARM architecture, and general rules of coherency, requires reads to the same memory location to be observed in order.

Due to some internal replay path mechanisms, the Cortex-A9 can see a read access bypassed by a following read access to the same memory location, thus not observing the values in program order.

Conditions:

It can only happen on memory regions marked as Normal Memory Write-Back Shared.

Projected Impact:

The erratum leads to data coherency failure.

Workarounds:

The vast majority of multi-processing code is not written in a style which exposes the erratum. So, the erratum is expected to affect very specific areas of code which rely on this read ordering behavior.

A first workaround for this erratum consists in using LDREX instead of standard LDR in volatile memory places where a strict read ordering is required.

An alternative solution consists in inserting a DMB between the affected LDR which requires this strict ordering rule. This is the recommended workaround for tool chains integration.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR004325 ARM/MP: 764369—Data or unified cache line maintenance operation by MVA may not succeed on an Inner Shareable memory region

Description:

Under certain timing circumstances, a data or unified cache line maintenance operation by MVA targeting an Inner Shareable memory region may fail to proceed up to either the Point of Coherency or to the Point of Unification of the system. This is likely to affect self modifying code.

Conditions:

The erratum requires a Cortex-A9 MPCore configuration with two processors or more, working in SMP mode, with the broadcasting of CP15 maintenance operations enabled.

The following scenario illustrates how the erratum can happen:

- One CPU performs a data or unified cache line maintenance operation by MVA targeting a memory region that is locally dirty.
- A second CPU issues a memory request targeting this same memory location within the same time frame.

A race condition may occur, resulting in the cache operation not being performed up to the specified Point, either Coherency or Unification.

The following maintenance operations are affected:

- DCIMVAC: Invalidate data or unified cache line by MVA to PoC
- DCCMVAC: Clean data or unified cache line by MVA to PoC
- DCCMVAU: Clean data or unified cache line by MVA to PoU
- DCCIMVAC: Clean and invalidate data or unified cache line by MVA to PoC

The erratum may arise when the second CPU is performing either of:

- A read request resulting from any Load instruction; the Load can be a speculative one.
- A write request resulting from any Store instruction.
- A data prefetch resulting from a PLD instruction; the PLD may be a speculative one.

Projected Impact:

Since the cache maintenance operation is not ensured to be executed to either the Point of Unification or the Point of Coherence, stale data may remain in the data cache, and not become visible to other cache agents who should have gained visibility on it.

As such, self modifying code may fail, the new code sequence written into the Data Cache not having been made visible to the Instruction Cache.

Note that the data remains coherent on the L1 Data side. Any data read from another processor in the Cortex-A9 MPCore cluster, or from the ACP, would see the correct data. Identically, any write from another processor in the Cortex-A9 MPCore cluster, or from the ACP, on the same cache line, will not cause a data corruption resulting from a loss of either data.

Note that false sharing on a memory region used for self modifying code is extremely unlikely to exist. As such, the write operation targeting the same cache line than the cache operation occurring within the timing window, required to trigger this erratum, may not represent a real case. So, the erratum trigger in the case of self modifying code is probably restricted to read operations, being the consequence of either a speculative load, or a “blind” PLD instruction.

In addition, production of data to an agent external from the coherency domain may fail; particularly, the data target of the cache maintenance operation may not have been made visible to an external DMA engine when it completes. Again, false sharing on a memory region also accessed by an external agent, like a DMA engine, is extremely unlikely to exist. As such, the erratum trigger, when producing data for an external DMA agent, is probably restricted to read operations, being the consequence of either a speculative load, or a “blind” PLD instruction.

Workarounds:

To work around this erratum, ARM recommends to:

- Ensure there is no false sharing (on a cache line size alignment) for both self modifying code and data to be cleaned to an external agent, like a DMA engine.
- Set bit[0] in the undocumented SCU diagnostic control register located at offset 0x30 from the PERIPHBASE address. Setting this bit disables the “migratory bit” feature. This forces a dirty cache line to be evicted to the lower memory subsystem—which is both the point of coherency and the point of unification—when it is being read by another processor.
- Insert a DSB instruction in front of the cache maintenance operation. Note that if the cache maintenance operation is executed within a loop with no other memory operations, ARM only recommends adding a DSB prior to entering the loop.

Note that the atomicity between the DSB and the cache maintenance operation may not be ensured because an interrupt may still be taken between the two instructions. However, setting the “disable migratory line” bit and inserting the DSB in front of the cache maintenance operation will very significantly decrease the probability to trigger the erratum when false sharing for writes to either self-modifying code memory regions or DMA regions, on a cache line granularity, which is likely to be the case.

With these workarounds, the likely occurrence of this erratum is sufficiently low that the erratum does not limit or severely impair the intended use of specified features.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR004326 ARM/MP: 761321—MRC and MCR are not counted in event 0x68

Description:

Event 0x68 counts the total number of instructions passing through the Register rename pipeline stage. The erratum is that MRC and MCR instructions are not counted in this event.

The event is also reported externally on PMUEVENT[9:8], which suffers from the same defect.

Projected Impact:

The implication of this erratum is that the values of event 0x68 and PMUEVENT[9:8] are imprecise, omitting the number of MCR and MRC instructions. The inaccuracy of the total count depends on the rate of MRC and MCR instructions in the code.

Workarounds:

No workaround is possible to achieve the required functionality of counting how many instructions are precisely passing through the register rename pipeline stage, when the code contains some MRC or MCR instructions.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004327 ARM/MP: 764319—Read accesses to DBGPRSR and DBGOSLSR may generate an unexpected UNDEF

Description:

CP14 read accesses to the DBGPRSR and DBGOSLSR registers generate an unexpected UNDEFINED exception when the DBGSWENABLE external pin is set to 0, even when the CP14 accesses are performed from a privileged mode.

Projected Impact:

Due to the erratum, the DBGPRSR and DBGOSLSR registers are not accessible when DBGSWENABLE=0.

This is, however, not expected to cause any significant issue in Cortex-A9 based systems because these accesses are mainly intended to be used as part of debug over power-down sequences, which is not a feature supported by the Cortex-A9.

Workarounds:

The workaround for this erratum consists in temporarily setting the DBGSWENABLE bit to 1 so that the DBGPRSR and DBGOSLSR registers can be accessed as expected.

There is no other workaround for this erratum.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005175 ARM/MP: 771221—PLD instructions may allocate data in the Data Cache regardless of the Cache Enable bit value

Description:

Preload Data (PLD) instructions prefetch and allocate any data marked as Write-Back (either Write-Allocate or Non-Write-Allocate, Shared or Non-Shared), regardless of the processor configuration settings, including the Data Cache Enable bit value.

Projected Impact:

Due to this erratum, unexpected memory cacheability aliasing is created which may result in various data consistency issues.

In practice, this erratum is not expected to cause any significant issue. The Data Cache is expected to be enabled as soon as possible in most systems, and not dynamically modified. So, only boot-up code would possibly be impacted by this erratum, but such code is usually carefully controlled and not expected to contain any PLD instruction while Data Cache is not enabled.

Workarounds:

In the case where a system is impacted by this erratum, a software workaround is available which consists in setting bit [20] in the undocumented Control register, which is placed in CP15 c15 0 c0 1.

This bit needs to be written with the following Read/Modify/Write code sequence:

MRC p15,0,r0,c15,c0,1

ORR r0,r0,#0x00100000

MCR p15,0,r0,c15,c0,1

Setting this bit causes all PLD instructions to be treated as NOPs, with the consequence that code sequences usually using the PLDs, such as the memcpy() routine, may suffer from a visible performance drop. So, if this workaround is applied, ARM strongly recommends restricting its usage to periods of time where the Data Cache is disabled.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005183 ARM/MP: 771224—Visibility of Debug Enable access rights to enable/disable tracing is not ensured by an ISB

Description:

According to the ARM architecture, any change in the Authentication Status Register should be made visible to the processor after an exception entry or return, or an ISB.

Although this is correctly achieved for all debug-related features, the ISB is not sufficient to make the changes visible to the trace flow. As a consequence, the WPTTRACEPROHIBITEDn signal(s) remain stuck to their old value up to the next exception entry or return, or to the next serial branch, even when an ISB is executed.

A serial branch is one of the following:

- Data processing to PC with the S bit set (for example, MOVS pc, r14)
- LDM pc ^

Projected Impact:

Due to the erratum, the trace flow may not start or stop, as expected by the program.

Workarounds:

To work around the erratum, the ISB must be replaced by one of the events causing the change to be visible. In particular, replacing the ISB by a MOVS PC to the next instruction will achieve the correct functionality.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005185 ARM/MP: 771225—Speculative cacheable reads to aborting memory region clear the internal exclusive monitor, may lead to livelock

Description:

On Cortex-A9, when a cacheable read receives an external abort, the aborted line is allocated as invalid in the Data Cache, and any allocation in the Data Cache clears the internal exclusive monitor.

So, if a program executes a LDREX/STREX loop which keeps on receiving an abort answer in the middle of the LDREX/STREX sequence, then the LDREX/STREX sequence never succeeds, leading to a possible processor livelock.

As an example, the following code sequence may exhibit the erratum:

loop LDREX

...

DSB

STREX

CMP

BNE loop

....

LDR (into aborting region)

The LDREX/STREX does not succeed on the first pass of the loop, and the BNE is mispredicted, so, the LDR afterwards is speculatively executed.

So, the processor keeps on executing:

LDR to aborting region (this speculative LDR now appears “before” the LDREX and DSB)

LDREX

DSB

STREX

The LDR misses in L1, and never gets allocated as valid because it is aborting

The LDREX is executed, and sets the exclusive monitor

The DSB is executed. It waits for the LDR to complete, which aborts, causing an allocation (as invalid) in the Data Cache, which clears the exclusive monitor

The STREX is executed, but the exclusive monitor is now cleared, so the STREX fails

The BNE may be mispredicted again, so the LDR is speculatively executed again, and the code loops back on the same failing LDREX/STREX sequence.

Conditions:

The erratum happens in systems which may generate external aborts in answer to cacheable memory requests.

Projected Impact:

If the program reaches a stable state where the internal exclusive monitor keeps on being cleared in the middle of the LDREX/STREX sequence, then the processor may encounter a livelock situation.

In practice, this scenario seems very unlikely to happen because several conditions may prevent the erratum to happen:

- Usual LDREX/STREX code sequences do not contain any DSB, so that it is very unlikely that the system would return the abort answer precisely in the middle of the LDREX/STREX sequence on each iteration.
- Some external irritators (for example, interrupts) may happen and cause timing changes which may exit the processor from its livelock situation.
- Branch prediction is very usually enabled, so the final branch in the loop will usually be correctly predicted after a few iterations of the loop, preventing the speculative LDR to be issued, so that the next iteration of the LDREX/STREX sequence will succeed.

Workarounds:

The following two workarounds are available for this erratum:

- Turn on the branch prediction.
- Remove the DSB in the middle of the LDREX/STREX sequence. If a DSB is truly required, it is strongly recommended to place it before the LDREX/STREX sequence, and implement the LDREX/STREX sequence as recommended by the ARM architecture.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005187 ARM/MP: 771223—Parity errors on BTAC and GHB are reported on PARITYFAIL[7:6], regardless of the Parity Enable bit value

Description:

PARITYFAIL signal bits [7] and [6] are expected to report parity errors occurring on the BTAC and GHB RAMs, when the parity error detection logic is enabled (ACTLR[9]=1’b1).

The erratum is that the Parity Enable bit, ACTLR[9], is not taken into account by the logic driving PARITYFAIL[7:6]. As a consequence, any parity error on the BTAC or GHB RAM will be reported on PARITYFAIL[7] or [6], even when parity error detection is not enabled.

Conditions:

The erratum happens on all configurations that have implemented parity support on the BTAC or GHB RAMs when dynamic branch prediction is enabled (SCTLR[11]=1’b1).

Projected Impact:

Due to the erratum, unexpected parity errors may be reported when parity is not enabled, if any parity error happens on the BTAC or GHB RAMs.

Note that implementing parity error detection is not mandatory on the BTAC and GHB RAMs because such errors may cause a branch mispredict, but no functional failure.

In systems which are implementing parity error detection on the BTAC and GHB RAMs, the erratum is not expected to cause any significant issue because parity is likely to be enabled very soon in the boot process.

Workarounds:

Because parity errors on the BTAC and GHB RAMs are not reported when the dynamic branch prediction is not enabled, the workaround consists in enabling parity error detection (ACTLR[9]), prior to enabling dynamic branch prediction (SCTLR[11]).

In systems where branch prediction is enabled while parity error detection remains disabled, the workaround consists in ignoring any assertion on the PARITYFAIL[7:6] bits.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005198 ARM/PL310: 780370—DATAERR, TAGERR, and Tag parity errors are incorrectly sampled by the eviction buffer, leading to data corruption

Description:

The PL310 L2 cache controller implements error logic to indicate errors have occurred when accessing the L2 cache RAM array. The following error information is available when accessing the RAM array:

- DATAERR (or DATAERR[3:0] if data banking is implemented) from Data RAM
- TAGERR[7:0] (or TAGERR[15:0] if 16 ways are implemented) from Tag RAM
- Parity error on Tag or Data RAM if parity is implemented

This information is associated with each individual RAM access, and is only meant to be sampled by the PL310 internal access requestor at precise cycles, depending on the programmable latencies of the accessed RAM (see Technical Reference Manual (TRM) for more information on RAM latencies).

More specifically, when an eviction is handled by the PL310 eviction buffer, both Tag and Data RAMs are accessed to get the whole eviction information. When either DATAERR or TAGERR is asserted high, or a tag parity error is detected during that process, the error information is captured by the eviction buffer, which cancels the corresponding eviction as a result.

Due to this erratum, the eviction buffer can incorrectly sample error information. As a result, an eviction can be wrongly cancelled and dirty data can be lost, leading to data corruption.

Note that data parity error is not part of this erratum. The reason is that this type of error information is not taken into account by the eviction buffer. This means that an eviction is always sent to the L3 memory system, regardless of whether a Data parity error has been detected or not, when accessing its data in the L2 cache.

Conditions:

The erratum occurs when the following conditions are met:

- The L2 cache contains dirty cache lines
- The eviction buffer accesses Tag and Data RAMs to get dirty cache line information before replacement
- While the eviction buffer accesses the RAMs, a tag parity error is detected, or DATAERR or TAGERR are asserted HIGH, but this error information is not meant to be captured by the eviction buffer (it may be directed to another PL310 block or DATAERR may be transiently asserted high before the end of the Data RAM latency period)
- The eviction buffer incorrectly samples the error information and cancels the corresponding eviction

Projected Impact:

When the above conditions are met, dirty data can be lost, leading to data corruption.

The implications of this data corruption depend on the error information and the PL310 configuration. All cases listed below need to be carefully assessed to know the exact impact of the erratum on a particular system.

DATAERR

In a system where DATAERR is tied low, this erratum does not apply as far as DATAERR is concerned.

In a system not implementing banking on the Data RAM and not driving DATAERR constantly low, the eviction buffer can sample transient and unstable high values of DATAERR, even if there is actually no expected error reported to PL310. This case is the most serious consequence of this erratum because it leads to a silent data corruption without any actual data error.

In a system using DATAERR for indicating Data RAM error and implementing banking on the Data RAM, the eviction buffer can only sample a true error coming from the Data RAM. However, this error may actually target another PL310 sub-block or another eviction slot. The erratum can thus still lead to data corruption, but the latter must be put in perspective relative to the true data error the overall system is facing. This is up to the system to assess how serious the data corruption is compared to the RAM error.

TAGERR

In a system where TAGERR is tied low, this erratum does not apply as far as TAGERR is concerned.

In a system using TAGERR for indicating Tag RAM error, the eviction buffer can only sample a true error coming from the Tag RAM. However, this error may actually target another PL310 sub-block or another eviction slot. The erratum can thus still lead to data corruption, but the latter must be put in perspective relative to the true tag error the overall system is facing. This is up to the system to assess how serious the data corruption is compared to the RAM error.

Tag parity error

In a system not implementing parity configuration in PL310, this erratum does not apply as far as the tag parity error is concerned.

In a system implementing parity, the eviction buffer can only sample a true tag parity error detected by the PL310 parity logic. However, this error may actually target another PL310 sub-block or another eviction slot. The erratum can thus still lead to a data corruption, but the latter must be put in perspective relative to the true parity error the overall system is facing. This is up to the system to assess how serious the data corruption is compared to the error.

Workarounds:

The following two software workarounds are available for systems affected by this erratum:

- Use write-through memory attributes for all cacheable accesses targeting PL310.
- Disable the logic responsible for generating RAM errors. This can imply disabling parity in PL310 and/or disabling DATAERR and TAGERR generation in the RAM array, depending on the implementation.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005200 ARM/MP: 765569—Prefetcher can cross 4 KB boundary if offset is programmed with value 23

Description:

When prefetch feature is enabled (bits [29:28] of the Auxiliary or Prefetch Control Register set HIGH), the prefetch offset bits of the Prefetch Control Register (bits [4:0]) permits to configure the advance taken by the prefetcher compared to the current cache line. Refer to the TRM for more information. One requirement for the prefetcher is not to go beyond a 4 KB boundary. If the prefetch offset is set to 23 (5'b10111), this requirement is not fulfilled and the prefetcher can cross a 4 KB boundary.

This problem occurs when the following conditions are met:

1. One of the Prefetch Enable bits (bits [29:28] of the Auxiliary or Prefetch Control Register) is set HIGH.
2. The prefetch offset bits are programmed with value 23 (5'b10111).

Projected Impact:

When the conditions above are met, the prefetcher can issue linefills beyond a 4 KB boundary compared to original transaction. This can cause system issues because those linefills can target a new 4 KB page of memory space, regardless of page attribute settings in L1 MMU.

Workarounds:

A workaround for this erratum is to program the prefetch offset with any value except 23.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005382 ARM/MP: 775419—PMU event 0x0A (exception return) might count twice the LDM PC ^ instructions with base address register write-back

Description:

The LDM PC ^ instructions with base address register write-back might be counted twice in the PMU event 0x0A, which is counting the number of exception returns.

The associated PMUEVENT[11] signal is also affected by this erratum, and might be asserted twice by a single LDM PC ^ instruction with base address register write-back.

Projected Impact:

Due to the erratum, the count of exception returns is imprecise. The error rate depends on the ratio between exception returns of the form LDM PC ^ with base address register write-back and the total number of exceptions returns.

Workarounds:

There is no workaround to this erratum.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005383 ARM/MP: 775420—A data cache maintenance operation that aborts, followed by an ISB and without any DSB in-between, might lead to deadlock

Description:

Under certain micro-architectural circumstances, a data cache maintenance operation that aborts, followed by an ISB and with no DSB occurring between these events, might lead to processor deadlock.

Conditions:

The erratum occurs when the following conditions are met:

- Some write operations are handled by the processor, and take a long time to complete. The typical situation is when the write operation (STR, STM, …) has missed in the L1 Data Cache.
- No memory barrier (DMB or DSB) is inserted between the write operation and the data cache maintenance operation mentioned in condition 3.
- A data cache maintenance operation is performed, which aborts due to its MMU settings.
- No memory barrier (DMB or DSB) is inserted between the data cache maintenance operation in previous condition and the ISB in next condition. Any other kind of code can be executed here, starting with the abort exception handler, following the aborted cache maintenance operation.
- An ISB instruction is executed by the processor.
- No memory barrier (DMB or DSB) is inserted between the ISB in previous condition and the read or write operation in next condition.
- A read or write operation is executed.

With the above conditions, an internal “Data Side drain request” signal might remain sticky, causing the ISB to wait for the Data Side to be empty, which never happens because the last read or write operation waits for the ISB to complete.

Projected Impact:

The erratum can lead to processor deadlock.

Workarounds:

A simple workaround for this erratum is to add a DSB at the beginning of the abort exception handler.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005385 ARM/MP: 782772—A write to Strongly Ordered memory region, followed by a condition-failed LDREX, might deadlock the processor

Description:

Under certain timing circumstances specific to the Cortex-A9 micro-architecture, a processor might deadlock when the execution of a write to a Strongly Ordered memory region is followed by the execution of a conditional, LDREX instruction that fails its condition code check.

Conditions:

The erratum occurs when the following conditions are met:

- The processor executes a write instruction to a Strongly Ordered memory region.
- The processor executes a conditional LDREX instruction.
- The instruction fails its condition code check.

The erratum also requires additional timing conditions to reach the point of failure, which are specific to the Cortex-A9 micro-architecture and cannot directly be controlled by software.

Projected Impact:

The erratum causes processor deadlock.

Workarounds:

The recommended workaround for this erratum is to add a DMB or DSB instruction between the write to the Strongly Ordered memory region and the conditional LDREX.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP. No conditional usage of LDREX instructions in the BSP.

ERR005386 ARM/MP: 782773—Updating a translation entry to move a page mapping might erroneously cause an unexpected translation fault

Description:

Under certain conditions specific to the Cortex-A9 micro-architecture, a write operation that updates a Cacheable translation table entry might cause both the old and the new translation entry to be temporarily invisible to translation table walks, thus erroneously causing a translation fault.

Conditions:

The erratum occurs when the following conditions are met:

- The processor has its Data Cache and MMU enabled.
- The TTB registers are set to work on Cacheable descriptors memory regions.
- The processor is updating an existing Cacheable translation table entry, and this write operation hits in the L1 Data Cache.
- A hardware translation table walk is attempted. The hardware translation table walk can be either due to an Instruction fetch, or due to any other instruction execution that requires an address translation, including any load or store operation. This hardware translation walk must attempt to access the entry being updated in condition 2, and that access must hit in the L1 Data Cache.

In practice, this scenario can happen when an operating system (OS) is changing the mapping of a physical page. The OS might have an existing mapping to a physical page (the old mapping), but wants to move the mapping to a new page (the new mapping). To do this, the OS might:

1. Write a new translation entry, without cancelling the old one. At this point the physical page is accessible using either the old mapping or the new mapping.
2. Execute a DSB instruction followed by an ISB instruction pair, to ensure that the new translation entry is fully visible.
3. Remove the old entry.

Due to the erratum, this sequence might fail because it can happen that neither the new mapping, nor the old mapping, is visible after the new entry is written, causing a Translation fault.

Projected Impact:

The erratum causes a Translation fault.

Workarounds:

The recommended workaround is to perform a clean and invalidate operation on the cache line that contains the translation entry before updating the entry, to ensure that the write operation misses in the Data Cache. This workaround prevents the micro-architectural conditions for the erratum from happening. Interrupts must be temporarily disabled so that no interrupt can be taken between the maintenance operation and the translation entry update. This avoids the possibility of the interrupt service routine bringing the cache line back in the cache.

Another possible workaround is to place the translation table entries in Non-Cacheable memory areas, but this workaround is likely to have a noticeable performance penalty.

Note that inserting a DSB instruction immediately after writing the new translation table entry significantly reduces the probability of hitting the erratum, but it is not a complete workaround.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005387 ARM/MP: 782774—A spurious event 0x63, “STREX passed,” can be reported on an LDREX that is preceded by a write to Strongly Ordered memory region

Description:

A write to Strongly Ordered memory region, followed by the execution of an LDREX instruction, can cause the “STREX passed” event to be signaled even if no STREX instruction is executed.

As a result, the event 0x63 count might be faulty, reporting too many “STREX passed” events.

This erratum also affects the associated PMUEVENT[27] signal. This signal will report the same spurious events.

Conditions:

The erratum occurs when the following conditions are met:

- The processor executes a write instruction to a Strongly Ordered memory region.
- The processor executes an LDREX instruction.
- No DSB instruction is executed, and there is no exception call or exception return, between the write and the STREX instructions.

Under these conditions, if the write instruction to Strongly Ordered memory region receives its acknowledge (BRESP response on AXI) while the LDREX is being executed, the erratum can happen.

Projected Impact:

The erratum leads to a faulty count of event 0x63, or incorrect signaling of PMUEVENT[27].

Workarounds:

The workaround for this erratum is to insert a DMB or DSB instruction between the write to Strongly Ordered memory region and the LDREX instruction.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005391 ARM: Debug CTI interrupt can cause a system deadlock when power gating the core

Description:

The Cross Trigger Interface (CTI) provides an interface for trigger inputs and outputs to the device-wide cross triggering system through the cross trigger matrix (CTM). The triggers are coupled to debug-centric signals associated with the ARM platform and are mapped to a CTI interrupt. Due to an issue with the implementation logic, using this CTI interrupt when power gating the core can cause a system deadlock.

Projected Impact:

System deadlock if the CTI interrupt is enabled when power gating the core.

Workarounds:

To prevent this issue from occurring, software should mask the CTI interrupt before power gating the core.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR006259 ARM: Debug/trace functions (PMU, PTM and ETB) are disabled with absence of JTAG_TCK clock after POR

Description:

When JTAG_TCK is not toggling after power-on reset (POR), the ARM PMU, PTM, and ETB stay in their disabled states so various debug and trace functions are not available.

Projected Impact:

Limited debug/trace capability

Workarounds:

Provide at least 4 JTAG_TCK clock cycles following POR if the PMU, PTM and ETB functions will be used. A free-running JTAG_TCK can also be used.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005766 CAAM: CAAM cannot handle interleaved READ data “beats” returned by two different slaves in the system, in reply to two different AXI-ID accesses

Description:

The CAAM can issue several transactions with different AXI-IDs but its AXI master port does not handle interleaved data properly. The faulty behavior is expected to occur when working in DDR interleaving mode. For example, one access with ID X is directed to DDR0, while almost simultaneously, another access with different AXI-ID is passed to the second DDR controller. This way the data “beats” of the two AXI-IDs may be replied interleaved.

CAAM has two sources of transactions—first, the Job Queue controller, which fetches jobs and prepares descriptors to be run, and second, the DECO, which executes the descriptors. With a single DECO, there are less chances of the Job Queue controller and DECO to overlap while performing AXI read requests.

Projected Impact:

CAAM may read wrong data.

Workarounds:

There are two workarounds for this issue. They both prevent CAAM from issuing multiple AXI read transactions with different AXI-IDs. The workarounds are as follows:

- Workaround 1: The first workaround is to only issue a single descriptor to CAAM at a time. CAAM will not pre-fetch a second descriptor, as there is no second descriptor. HAB uses this approach. HAB in i.MX 6 Series only issues one descriptor at a time.
- Workaround 2: The second workaround is for cases where multiple descriptors will be issued to CAAM, (for example, a Linux device driver). In this case, CAAM can be configured to only issue one AXI transaction at at time by setting the CAAM AXI pipeline depth to 1. This will prevent multiple outstanding transactions, and thus multiple transactions with different AXI-IDs. This is done by setting the AXIPIPE field of the CAAM Master Configuration Register (MCFGR) to 1. The workaround seems to have minimal impact on the performance.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR006223 CCM: Failure to resume from Wait/Stop mode with power gating

Description:

When entering Wait/Stop mode with power gating of the ARM core(s), if an interrupt arrives during the power-down sequence, the system could enter an unexpected state and fail to resume.

Projected Impact:

Device may fail to resume from low-power state.

Workarounds:

Use REG_BYPASS_COUNTER (RBC) to hold off interrupts when the PGC unit is in the middle of the power-down sequence. The counter needs to be set/cleared only when there are no interrupts pending. The counter needs to be enabled as close to the WFI (Wait For Interrupt) state as possible.

The following equation can be used to aid determination of the RBC counter value:

$RBC_COUNT \times (1/32K\ RTC\ Frequency) \geq (25 + PDNSCR_SW2ISO) \times (1/IPG_CLK\ Frequency)$

PDNSCR_ISO2SW = PDNSCR_ISO = 1 (counts in IPG_CLK clock domain)

Proposed Solution:

No fix scheduled.

Linux BSP Status:

Implemented in BSP version GA L3.0.35_3.0.0

ERR004446 EIM: AUS mode is nonfunctional for devices larger than 32 MB

Description:

When the AUS bit is set, the address lines of the EIM are unshifted. By default, the AUS bit is cleared and address lines are shifted according to port size (8, 16 or 32 bits). Due to an error, the address bits 27:24 are shifted when AUS=1. For example, CPU address 0xBD00_0000 ([A27:20]=1101 0000 becomes 0xB600_0000 ([A27:20]=0110 0000) on the EIM bus, because A[27:25] is shifted to [A26:24] and A[23:0] is not shifted. As a result A[24] is missed.

Projected Impact:

If the memory used does not exceed 32 MB, there is no impact.

This mode is related to a unique memory configuration that is not often used. Most systems can work in the default mode (AUS=0). Board designers should connect the EIM address bus without a shift (for example, A0→A0 and A1→A1), while working in AUS=0 mode.

Workarounds:

- Use the AUS = 0 mode (default) while connecting the address signals without a shift (for example, A0→A0 and A1→A1).
- For AUS=1, for devices larger than 32 MB, it is necessary to build a memory map that takes this shifting into consideration and does not include A[24] line.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004512 ENET: 1 Gb Ethernet MAC (ENET) system limitation

Description:

The theoretical maximum performance of 1 Gbps ENET is limited to 470 Mbps (total for Tx and Rx). The actual measured performance in optimized environment is up to 400 Mbps.

Projected Impact:

Minor. Limitation of ENET throughput to around 400 Mbps. ENET remains fully compatible to 1Gb standard in terms of protocol and physical signaling.

Workarounds:

No workaround.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005783 ENET: ENET Status FIFO may overflow due to consecutive short frames

Description:

When the MAC receives shorter frames (size 64 bytes) at a rate exceeding the average line-rate burst traffic of 400 Mbps the DMA is able to absorb, the receiver may drop incoming frames before a Pause frame is issued.

Projected Impact:

No malfunction will result aside from the frame drops.

Workarounds:

The application may want to implement some flow control to ensure the line-rate burst traffic is below 400 Mbps if it only uses consecutive small frames with minimal (96 bit times) or short Inter-frame gap (IFG) time following large frames at such a high rate. The limit does not exist for frames of size larger than 800 bytes.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005895 ENET: ENET 1588 channel 2 event capture mode not functional

Description:

The ENET module provides a 4-channel IEEE 1588 compliant timer that supports event input capture and output compare mode. The capture/compare feature requires the ENET 1588 clock to latch in the correct IEEE 1588 counter value to the Timer Compare Capture Register (ENET_TCCRn). Due to an integration issue, the ENET 1588 clock and Channel 2 event capture/compare signal are both connected to the same GPIO16 pin.

Projected Impact:

ENET 1588 channel 2 event capture/compare mode cannot be used.

Workarounds:

None. Channels 1, 3, and 4 can be used for the event capture instead.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR006358 ENET: Write to Transmit Descriptor Active Register (ENET_TDAR) is ignored

Description:

If the ready bit in the transmit buffer descriptor (TxBD[R]) is previously detected as not set during a prior frame transmission, then the ENET_TDAR[TDAR] bit is cleared at a later time, even if additional TxBDs were added to the ring and the ENET_TDAR[TDAR] bit is set. This results in frames not being transmitted until there is a 0-to-1 transition on ENET_TDAR[TDAR].

Projected Impact:

Reduced ENET performance due to delayed servicing of interrupts.

Workarounds:

Code can use the transmit frame interrupt flag (ENET_EIR[TXF]) as a method to detect whether the ENET has completed transmission and the ENET_TDAR[TDAR] has been cleared. If ENET_TDAR[TDAR] is detected as cleared when packets are queued and waiting for transmit, then a write to the TDAR bit will restart TxBD processing.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in Linux BSP release L3.0.35_4.0.0

ERR006687 ENET: Only the ENET wake-up interrupt request can wake the system from Wait mode.

Description:

The ENET block generates many interrupts. Only one of these interrupt lines is connected to the General Power Controller (GPC) block, but a logical OR of all of the ENET interrupts is connected to the General Interrupt Controller (GIC). When the system enters Wait mode, a normal RX Done or TX Done does not wake up the system because the GPC cannot see this interrupt. This impacts performance of the ENET block because its interrupts are serviced only when the chip exits Wait mode due to an interrupt from some other wake-up source.

Projected Impact:

Reduced ENET performance due to delayed servicing of interrupts.

Workarounds:

All of the interrupts can be selected by MUX and output to pad GPIO6. If GPIO6 is selected to output ENET interrupts and GPIO6 SION is set, the resulting GPIO interrupt will wake the system from Wait mode.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in Linux BSP version L3.0.35_4.0.0

ERR004573 EPDC: Collision status must be read before clearing IRQ

Description:

When an interrupt event occurs, the appropriate bit within the EPDC_IRQ register is set by the EPDC. Once the interrupt has been handled, software clears the interrupt source by writing to the CLEAR address. However, when the EPDC interrupt is cleared, the Look Up Table (LUT) status of LUTs, 16–63, will be cleared and lost before software can capture it.

Projected Impact:

Collision status of LUTs, 16–63, gets cleared incorrectly, when EPDC interrupt is cleared.

Workarounds:

When handling an interrupt, the software must capture the LUT status by reading the EPDC_STATUS_LUTS1 and EPDC_STATUS_LUTS2 registers, before clearing the interrupt.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR005313 EPDC: Incorrect data fetched when the buffer update width is 2048 pixels or greater

Description:

When sending an image update to a high resolution panel with an image buffer width ≥ 2048 pixels, the EPDC module fetches incorrect data because an internal address register is incorrectly truncated.

Projected Impact:

EPDC may fetch incorrect data. The INIT update with FIXNP is not affected by this issue.

Workarounds:

Split the update down to < 2048 pixels. If group updates are supported, send updates in a group, so all of them get scanned together exactly like a single update.

If the group update feature is not available, send updates sequentially (one after another immediately after Working Buffer (WB) is done, no need to wait for FRAME scan to complete). One frame difference on scan time is the theoretical worst case; however, this was not visually observed during testing by Freescale.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround to be applied in a future BSP release

ERR005991 EPDC: Memory access may lock up when two continuous unaligned burst accesses return data back to back

Description:

Due to a logic issue, the EPDC module can potentially miss one cycle of data when two continuous unaligned burst accesses return data back to back. The bus data counter keeps indefinitely waiting for the missed data and can, therefore, hang up the module.

Projected Impact:

If you set update buffer to unaligned address, it may hang up bus for this corner case. Note here that unaligned access is not allowed in previous products.

Workarounds:

To prevent the issue, software should ensure that only aligned burst accesses are made to the EPDC memory.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround to be applied in a future BSP release

ERR005992 EPDC: EPDC may not detect collision correctly on update buffer boundary when using >16 LUTs AND with unaligned update buffer AND have different LUT updating adjacent area

Description:

EPDC reads Look Up Table (LUT) number from Working Buffer (WB) and determines its status for collision calculation, but in this corner case, the LUT number from WB is incorrect.

Projected Impact:

EPDC may generate incorrect collision status for boundary pixels. However, this issue was not observed during Freescale testing.

Workarounds:

To prevent the issue, use <16 LUTs.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround to be applied in a future BSP release

ERR005828 EXSC: Protecting the EIM memory map region causes unpredictable behavior

Description:

If a write access to the EIM address region is denied due to the CSU access control policy, then, all subsequent write accesses to the EIM region will write unintended data.

Projected Impact:

Blocking write accesses to the EIM region through Trustzone is not supported.

Workarounds:

To prevent unpredictable behavior, prior to accessing the EIM region, set all bits in the EIM CSU_CSL field to 1 so that all accesses are allowed. Specifically:

EIM: CSU_CSL31[23:16] = 0xff

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP. The BSP does not use CSU.

ERR005829 FlexCAN: FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment during the arbitration process

Description:

FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment during the arbitration process. The following conditions are necessary for the issue to occur:

- Only one message buffer is configured to be transmitted
- The write which enables the message buffer to be transmitted (write on Control/Status word) happens during a specific clock during the arbitration process.
- After this arbitration process occurs, the bus goes to the Idle state and no new message is received on the bus.

For example:

1. Message buffer 13 is deactivated on RxIntermission (write 0x0 to the CODE field from the Control/Status word) [First write to CODE]
2. Reconfigure the ID and data fields
3. Enable the message buffer 13 to be transmitted on BusIdle (write 0xC on CODE field) [Second write to CODE]
4. CAN bus keeps in Idle state
5. No write on the Control/Status from any message buffer happens.

During the second write to CODE (step 3), the write must happen one clock before the current message buffer 13 to be scanned by arbitration process. In this case, it does not detect the new code (0xC) and no new arbitration is scheduled.

The problem can be detected only if the message traffic ceases and the CAN bus enters into Idle state after the described sequence of events.

There is no issue if any of the conditions below holds:

- Any message buffer (either Tx or Rx) is reconfigured (by writing to its CS field) just after the Intermission field.
- There are other configured message buffers to be transmitted
- A new incoming message sent by any external node starts just after the Intermission field.

Projected Impact:

FlexCAN does not transmit a message that is enabled to be transmitted in a specific moment.

Workarounds:

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following standard 5-step procedure:

1. Check if the respective interrupt bit is set and clear it.

2. If the message buffer is active (transmission pending), write the ABORT code (0b1001) to the CODE field of the Control/Status word to request an abortion of the transmission. Wait for the corresponding IFLAG to be asserted by polling the IFLAG register or by the interrupt request if enabled by the respective IMASK. Then read back the CODE field to check if the transmission was aborted or transmitted. If backwards compatibility is desired (MCR[AEN] bit negated), just write the INACTIVE code (0b1000) to the CODE field to inactivate the message buffer, but then the pending frame may be transmitted without notification.
3. Write the ID word.
4. Write the data bytes.
5. Write the DLC, Control and CODE fields of the Control/Status word to activate the message buffer.
6. The workaround consists of executing two extra steps:
7. Reserve the first valid mailbox as an inactive mailbox (CODE=0b1000). If RX FIFO is disabled, this mailbox must be message buffer 0. Otherwise, the first valid mailbox can be found using the "RX FIFO filters" table in the FlexCAN chapter of the chip reference manual.
8. Write twice INACTIVE code (0b1000) into the first valid mailbox.

NOTE

The first mailbox cannot be used for reception or transmission process.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004341 GPU2D: Accessing GPU2D when it is power-gated will cause a deadlock in the system

Description:

Accessing GPU2D when it is power-gated will cause a deadlock in the system.

Projected Impact:

Deadlock in the system.

Workarounds:

GPU2D should not be mistakenly accessed by software when power-gated.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005908 GPU2D: Image quality degradation observed for stretch blits when the stretch factor is exactly an integer

Description:

GPU2D supports BLIT acceleration by using the Graphics Device Interface (GDI) API. When using the stretch blit GDI API, if the stretch factor is exactly an integer, the resulting image has rendering errors.

Projected Impact:

Minor visual impact in image quality when using the stretch blit for hardware acceleration. The rendering result using the BLIT hardware acceleration will not be the same as the software rendered image.

Workarounds:

There are no software workarounds that completely resolve the issue. The filter blit API can be used instead of the stretch blit for BLIT acceleration; however, there will be a performance impact and may not be suitable for all applications. The issue is not observed when the stretch blit for BLIT acceleration is not used.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004300 GPU3D: L1 cache performance drop

Description:

The GPU3D L1 cache assumes that all memory requests are 16 bytes. If a request is 16 bytes, there are no issues since the data boundary lines up evenly. If a request is not aligned to 16 bytes, the memory controller will split those unaligned requests into two requests, doubling the number of requests processed internally in L1 cache.

Projected Impact:

Application performance is reduced when L1 cache is present and data requests are unaligned to 16 bytes.

Workarounds:

Tune applications to access L1 cache and memory requests at 16 byte boundaries to prevent this issue (slight performance impact).

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004484 GPU3D: L1 cache “Write Address Data” pairing error

Description:

This issue causes a data alignment error under the following two corner case conditions:

- The last 16 bytes of the cache line are being sent to the memory controller when it is not ready
- The memory controller’s “Ready” signal is asserted for one cycle. It then reads 8 bytes of data and then the “Ready” signal becomes de-asserted again.

In the design, the memory controller uses the address of the last 8 bytes as the address of the entire cache line. When either of these conditions happens, the last 8 bytes of data are paired with the address of the subsequent cache line, and the entire cache line gets written to the wrong location. The likelihood of hitting this corner case is significantly reduced if the GPU3D core and shader clocks run at the same frequency.

Projected Impact:

This issue only affects OpenCL applications by causing data corruption (incorrect data calculations). This will not cause a system hang or screen corruption in 3D applications.

Workarounds:

None

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005216 GPU3D: Black texels in Android App Singularity 3D

Description:

Texture attributes may be incorrectly reported by the Setup Engine, when the maximum X and/or Y vertex is very large (X or Y vertex greater than 8 million pixels), and consequently the Texture Engine may sample black texels.

Projected Impact:

Visual impact will vary depending on whether the application texture is clamped or wrapped.

Workarounds:

No workaround at this time.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004366 HDMI: 9000482480—ARM core read operation returns incorrect data

Description:

When ARM core performs a read access operation on the register bank, it samples the data an SFR clock cycle earlier than required, hence the data returned may be invalid.

Projected Impact:

Read invalid data by ARM core.

Workarounds:

For reliable read operations, use the ARM core read command twice targeting the same address, discard the first data read value, and use the second read value.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR005172 HDMI: Under certain circumstances, the HDCP may transmit incorrect Ainfo value, causing a failure on the receiver side

Description:

The HDCP specification requires that a feature support search procedure be performed to enable an HDCP link with Features 1.1 active. The HDCP transmitter has to read the HDCP receiver BCAPS to check if it supports Features 1.1, and if this is the case and the HDCP transmitter desires to use Features 1.1, it must enable them on the HDCP receiver by writing 0x02 in the HDCP I2C Ainfo register. It has been found that the HDCP transmitter is using the local configuration register (A_HDCPCFG0.en11 feature bit field register) and sending that data on the Ainfo register, ignoring that the remote HDCP Features 1.1 support has been indicated on the I2S BCAPS register.

Projected Impact:

HDCP may transmit incorrect Ainfo value, causing a failure on the receiver side. A failure can only occur if the HDCP receiver indicates that it does not support Features 1.1 in the I2C BCAPS register, and then acts upon the incorrect Ainfo information.

Workarounds:

To avoid this problem, the software should previously check BCAPS by reading A_HDCPOBS3.FEATURES_1_1 to ensure whether the receiver can support features 1.1 or not. If it does not support, then it should not set A_HDCPCFG0.en11 feature.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR004307 I/O: MIPI_HSI, USB_HSIC, and ENET I/O interfaces should not be configured to Differential input mode

Description:

DDR3, LPDDR2, MIPI_HSI, USB_HSIC, and ENET I/O interfaces are of the DDR I/O type, thus having the option to work in DDR input mode. This mode requires setting the DRAM_VREF to half the I/O voltage. This reference pad is used in all DDR type I/O interfaces. Since all I/O interfaces do not have a common voltage, configuring more than two I/O interfaces to DDR input mode may not work.

Conditions:

De-assertion of POR_B when the SoC is powered-up.

Projected Impact:

DDR3, LPDDR2, MIPI_HSI, USB_HSIC, and ENET I/O interfaces may not work together.

Workarounds:

Configure the DDR_INPUT bit in IOMUXC for MIPI_HSI, USB_HSIC, and ENET to “0,” that is, CMOS input type.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP. The BSP ensures that the DDR_INPUT bit is set to CMOS input type.

ERR005190 MIPI: CSI2 Data lanes are activated before the HS clock from the CSI Tx side (camera) starts

Description:

The MIPI CSI2 circuit is enabled by default with all the D-PHY data lanes active and will only disable the lanes that are not required when HS clock is available.

Projected Impact:

Affects the non-active data lanes status register value and adds some minor power consumption (1–2 mA); however, there will not be any packet loss. Once the HS clock is detected, the data lanes will be disabled.

Workarounds:

None.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005191 MIPI: Corruption of short command packets with Word Count (WC) greater than 16’hFFEE, during video mode transmission by the MIPI Generic Interface

Description:

On short packets, the WC[15:0] header field delivers the actual data payload of the packet. However, In long packets, the same field is used to indicate the size of the packet’s payload. Video Mode packet scheduler prevents Generic long packets to be generated with a size higher than 16’hFFEE. This size limit is imposed by the video mode packet scheduler since the maximum line size is 16’hFFFF minus additional security margins. The core is incorrectly filtering the short packets with WC field higher than 16’hFFEE since the size protection is applied without considering that this field now contains data and not packet size. Short packet commands are erroneously transmitted in DSI link with WC field equal to 16’hFFEE when this value is higher than 16’hFFEE.

Projected Impact:

Error in transmitting the package.

Workarounds:

If a generic short packet with packet data (WC fields) higher that 16’hFFEE is required, the application should disable the video mode transmission and use command mode transmission to issue the command.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005192 MIPI: Reverse direction long packets with no payload incorrectly issue a CRC error for MIPI DSI

Description:

The issue appears when a DSI device, which is in reverse mode, sends a long packet with no payload. When receiving this packet, the DSI host controller checks the 16bit CRC field of the packet and incorrectly issues an error.Although there is no apparent reason for a device to send a long packet with no payload, there is no restriction in the DSI specification that forbids this. Also, there is no data loss resulting from this bug, since a long packet with no payload carries no data. The only inconvenience is that a CRC error is asserted in the bit crc_err of the register ERROR_ST1.

Projected Impact:

The CRC error is asserted in the bit crc_err of the register ERROR_ST1, when DSI received a long packet with no payload. This errata does not affect the correct functionality.

Workarounds:

To avoid the assertion of the CRC error, disable verification of CRC reception errors in bit en_CRC_rx of the register PCKHDL_CFG. When disabling the CRC verification on the receive path, users should be aware that the CRC verification will be disabled for all reverse packets and not limited just to the long packets with no payload.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR005193 MIPI: The bits for setting the MIPI DSI video mode cannot be changed on the fly

Description:

The bits for setting video mode and command mode are assumed to be static during use and have no synchronization mechanism. These correspond to bit 0 of VID_MODE_CFG (address 0x1C) and bit 0 of register CMD_MODE_CFG register (address 0x24). these bits should only be changed while the digital core is in reset.

Projected Impact:

Will lead to corrupted video display or fail the command send.

Workarounds:

Setting PWR_UP register (address 0x04) to 0x00 keeps the controller under reset so that en_video_mode and en_cmd_mode can be changed free of any timing violations resulting from Clock-Domain Crossing. After those changes, PWR_UP can be set to 0x01 again, leading the controller to start working with the new configuration.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR005194 MIPI: On MIPI DSI, there is a possible corruption of the video packets caused by overlapping of the current line over the next line, if the configuration is programmed incorrectly when using the DPI interface

Description:

For an incorrectly programmed configuration that enables the Null Packets and disables the Multiple Packets, the delay calculation is incorrectly done. Calculation of the delay time applied to the synchronization events when the Null Packets are enabled does not consider that the delay should only be applied when Multiple Packets are also enabled. This inaccuracy in the delay time may lead to an eventual overlap of current line with next line transmission resulting in the corruption of the packets.

Projected Impact:

It will lead to the overlapping of two adjacent lines display data.

Workarounds:

This problem only occurs when an incorrect configuration is applied to the controller. The problem can be avoided only by activating the Null Packets, if Multiple Packets are also enabled.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005195 MIPI: Incorrect blanking packet may be sent by the MIPI DSI interface

Description:

When the HBP programmed timing is shorter than the time required to transmit the smallest blanking packet (6 bytes long packet), the controller incorrectly sends a blanking packet. This incorrect behavior models the HBP for a longer period than expected while the core should decide not to send any blanking packet. The transmission of blanking packet under these conditions can only be observed in Video Synchronous Mode with pulses.

Projected Impact:

Incorrect video stream in some conditions.

Workarounds:

HSA and HBP should be programmed with values higher than 10 lane byte cycles. The 10 lane byte clock corresponds to the transmission of a Horizontal Sync Start packet (4 bytes) followed by the smallest blanking packet (6 bytes).

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005196 MIPI: Error Interrupt generated by the MIPI CSI interface for certain legal packet types

Description:

Data types from 0x13 to 0x17 are reserved but not considered invalid in the CSI-2 specification. However, the MIPI CSI controller raises an interrupt due to an err_id* being flagged when a packet with one of these data types is received. Data types from 0x13 to 0x17 should be processed without an error notification of this kind.

Projected Impact:

Erroneously generated error interrupt.

Workarounds:

To avoid the interrupt, err_id* can be masked by setting the bits 12 to 15 of the MASK2 register. Bits 12 to 15 of the ERR2 register also should be ignored when reading. But, this procedure hides the occurrence of an err_id* error caused by the reception of other unidentified or unimplemented data type.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP. The BSP does not support data types from 0x13 to 0x17.

ERR005197 MIPI: Null and Blanking data packets activate ‘dvalid’ signal

Description:

The databook mentions that the data sent through Null or Blanking data packets do not activate ‘dvalid’ signal. CSI-2 Host controller implementation currently activates ‘dvalid’ for payload of any kind of long packet. The IP should match what is described in the databook and ‘dvalid’ should not be activated by Null and Blanking data.

Projected Impact:

None.

Workarounds:

The ‘dvalid’ signal can be filtered by configuring the following IPU data type registers:

- IPU_CSI0_DI__CSI0_MIPI_DI1
- IPU_CSI0_DI__CSI0_MIPI_DI2
- IPU_CSI0_DI__CSI0_MIPI_DI3

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005778 MMDC: DDR Controller's measure unit may return an incorrect value when operating below 100 MHz

Description:

The measure unit counts cycles of an internal ring oscillator. The measure unit readout is used to fine tune the delay lines for temperature/voltage changes for both DDR3 and LPDDR2 interfaces. When operating at low frequencies (below 100 MHz), the measure unit counter may overflow due to an issue in the overflow protection logic. As a result, an incorrect measure value will be read.

Projected Impact:

This may cause a rare issue if the measure unit counter stops within a small range of values that translate to a delay that tunes the system incorrectly. This issue may not manifest in the application because it is dependent on a combination of DDR frequencies coupled with specific Process, Voltage, and Temperature conditions.

Workarounds:

To workaround this issue, following steps should be performed by software:

1. Prior to reducing the DDR frequency (400 MHz), read the measure unit count bits (MU_UNIT_DEL_NUM).
2. Bypass the automatic measure unit when below 100 MHz, by setting the measure unit bypass enable bit (MU_BYP_EN).
3. Double the measure unit count value read in step 1 and program it in the measure unit bypass bit (MU_BYP_VAL) of the MMDC PHY Measure Unit Register, for the reduced frequency operation below 100 MHz.

Software should re-enable the measure unit when operating at the higher frequencies, by clearing the measure unit bypass enable bit (MU_BYP_EN). This code should be executed out of Internal RAM or a non-DDR based external memory.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER4

ERR003747 PCIe: 9000436491—Reading the Segmented Buffer Depth Port Logic registers returns all zeros

Description:

When disabling the Dynamic Q Depth Adjustment, DBI reads to the Segmented Buffer Depth Port Logic registers return all zeros versus returning the hardwired default value. Internally the DBI read access clears these registers, overwriting the default value with all zeros. Clearing these registers results in all zeros being returned for subsequent PCIe Cfg reads.

Following is an example scenario for this erratum:

1. Issue a PCIe Cfg read to any Port Logic Segmented Buffer Depth register.
   The read data value returned to the requester is the hardwired default value.
2. Issue a DBI read to same Port Logic Segmented Buffer Depth register.
   The read data value returned to the requester is all zeros.
3. Issue a PCIe Cfg read to same Port Logic Segmented Buffer Depth register.
   The read data value returned to the requester is all zeros.

Projected Impact:

The default Segmented Buffer Depth register values cannot be read when a DBI read access is performed with CX_DYNAMIC_SEG_SIZE = 0.

Workarounds:

PCIe Cfg, instead of DBI, should be used for reading the Segmented Buffer Depth Port Logic registers.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003748 PCIe: 9000427578—Root ports with address translation drop inbound requests, without reporting an error

Description:

Root ports which have address bus widths < 64 drop inbound memory requests, when the address of the request is greater than the implemented address bus width. This feature is to prevent address aliasing when requests with addresses above 4 GB are received. When this feature is used in conjunction with address translation (iATU or xATU), inbound memory TLPs that violate this address check rule are dropped but no error is reported.

Conditions:

Issue an inbound memory TLP that has an address larger than the AMBA or RTRGT1 address bus width.

Projected Impact:

When the requirements specified in the Conditions section are met, the inbound TLP is dropped but no error is reported on the port.

In case of MemRd TLPs, a completion with UR status is issued, but no error bits will be set on the root port.

Workarounds:

None

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003749 PCIe: 9000426180—MSI Interrupt Controller Status Register bit not cleared after being written by software

Description:

Each Interrupt Status Register contains 32-bit status bits, allowing for the status of 32 individual interrupt vectors to be reported. The status bits are RW1C bits and are set when an MSI Interrupt vector is received, and are cleared by software writing a 1 to the bit.

The setting of a status bit takes precedence over the clearing of a status bit. The precedence given to setting of the status bits resulted in the setting of a single status bit in the register, preventing any other status bits from being cleared at the same time. As a result, if an MSI interrupt is being logged in a status bit, and during the same clock cycle, software also attempts to clear another status bit in the same Status Register, then the status bit corresponding to the MSI interrupt is set but the status bit being written by software is not cleared and remains set. As a result, even though software has written a 1 to the status bit, the status bits remains set, reporting that an MSI interrupt has been received, even though software has serviced the Interrupt Request.

This issue only occurs if the setting of a status bit and the clearing of another status bit within the same Interrupt Status Register happens during the same clock cycle.

Projected Impact:

A Status bit that has been cleared by Software remains set.

Workarounds:

Read and clear status bit until it is read as cleared.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003751 PCIe: 9000413207—PME Requester ID overwritten when two PMEs are received consecutively

Description:

When multiple PM_PME TLPs are received by an RC, the PME Requester ID of the first received PM_PME is overwritten with the Requester ID of the subsequent PM_PME in the PME Requester ID field of the Root Status register. The correct operation is to store the Requester ID of the initial PM_PME and only update the PME Requester ID field, once software has cleared the PME Status bit to acknowledge the receipt of the initial PM_PME TLP.

Following is an example scenario for this erratum:

1. Send two PM_PME TLPs with different Requester IDs upstream to the core.
2. Read back the contents of the Root Status Register. The PME Status and PME Pending bits should both be set to 1. The Requester ID in the PME Requester ID field corresponds to the second PM_PME and not the initial PM_PME received by the core.

Projected Impact:

The Requester ID of the initial PM_PME is lost and not correctly stored in the Root Status register.

Workarounds:

If multiple devices requested a power mode state change, possibly some of them would not get served if the requester ID is overwritten. However, according to Section 5.3.3.3 of PCIe Specification, all agents that are capable of generating PM_PME must implement a PME Service Timeout mechanism to ensure that their PME requests are serviced within a reasonable amount of time.

If there is a time-out, the PM_PME TLP should be re-sent. So, this should not be an issue.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003753 PCIe: 9000405932—AXI/AHB Bridge Slave does not return a response to an outbound non-posted request

Description:

The completion timeout mechanism defined in Section 2.8 of PCIe Base Specification version 2.1 describes a method to allow a requester to recover from a scenario where it does not receive all completions to a non-posted request. In the AXI or AHB bridge module, this triggers an error response to the original request on the slave interface.

There is a separate completion timeout interface that passes information about the request that has been timed out from the PCIe Core to the AXI or AHB Bridge. The Bridge cannot process valid completions and completion timeouts in parallel and so, if a completion timeout is received at the same time as a valid completion is passed into the bridge, it must be stored and processed later.

There is only a single storage element available for a completion timeout in the bridge. If a second completion timeout is passed into the bridge before it processes the first, the second timeout will overwrite the first completion timeout in the storage element. This results in the information associated with the first timeout being lost and no response is returned on the AHB or AXI slave interface for the original request.

Following is an example scenario for this erratum:

1. Issue a continuous stream of outbound MemRd requests targeting the AXI or AHB Slave interface.
2. Correctly return completions back to back for all but two of these requests.
3. Wait for the timeout mechanism to trigger for the two requests that do not receive completions.

Projected Impact:

If all the steps of the example scenario, given in the Description section, are performed, then there will be no response on the AHB or AXI slave interface for the first non-posted request that triggers a completion timeout. The second request to trigger a completion timeout will correctly receive an ERROR response on the AXI or AHB slave interface.

Workarounds:

None

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003754 PCIe: 9000403702—AHB/AXI Bridge Master responds with UR status instead of CA status for inbound MRd requesting greater than CX_REMOTE_RD_REQ_SIZE

Description:

The AHB/AXI Bridge RAM is sized at configuration time to support inbound read requests with a maximum size of CX_REMOTE_RD_REQ_SIZE. When this limit is violated the core responds with UR status, when it should respond with CA status.

Projected Impact:

Software gets the wrong status.

Workarounds:

None.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003755 PCIe: 9000402443—Uncorrectable Internal Error Severity register bit has incorrect default value

Description:

The PCI Express AER Capability register ‘Uncorrectable Error Severity’ (at offset 0x0C) has the wrong default value for the ‘Uncorrectable Internal Error’ bit. It should be 1’b1.

Uncorrectable Internal Error is an optional feature that the PCI Express block does not support, but it must default to 1’b1 anyway.

Projected Impact:

If user chooses to implement Uncorrectable Error, it is not supported and is not compliant.

Workarounds:

None.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003756 PCIe: 9000387484—LTSSM: Software-initiated transitions to Disabled, Hot Reset, Configuration, or Loopback states sometimes take longer than expected

Description:

The PCI Express Specification is unclear regarding the transmission of Idle Symbols when a directed state transition occurs in the Recovery.Idle state. This can sometimes result in temporary loss of synchronization between link partners when transitioning from L0 to Detect, through the Disabled, Hot Reset, Configuration, or Loopback states.

Section 4.2.6.4.4 of the PCI Express Specification states that Recovery.Idle Transmitter sends Idle data on all configured Lanes. Note: If directed to other states, Idle Symbols do not have to be sent before transitioning to the other states (that is, Disable, Hot Reset, Configuration, or Loopback). The PCI Express block chooses to send Idle symbols, as the specification does not prohibit the sending of Idle symbols.

Projected Impact:

The device that initiates the state transition moves from Recovery.Idle through the requested state and back to Detect. The remote partner moves from Recovery.Idle back to L0 state, without going through the required state transition. The link partners lose synchronization. After a timeout period, the link partner moves through the correct state transition, and the link resynchronizes.

The probability of occurrence of this issue depends on the link latency.

Workarounds:

None.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003757 PCIe: 9000448152—Internal Address Translation Unit (iATU): Inbound Vendor Defined Message (VDM) 'ID Match Mode' is not functional

Description:

The VDM 'ID Match Mode' of the iATU allows inbound ID-routed VDMs to be translated without explicit knowledge of the Bus, Device, or Function number of the target function. ID-routed VDMs contain the destination ID in bits [31:16] of Header DWORD3.

This mode is not functional and the iATU requires the actual ID of the destination Bus, Device, and function to be known and programmed as bits [63:48] of the iATU region Base Address.

Following is an example scenario for this erratum:

1. Setup an inbound iATU region with the type field set to match messages and the vendor ID match mode bit set to 1.
2. Send a message that matches the bits [47:ATU_REG_WD] of the region base, but that does not match bits [63:48]. The message will not be translated.

Projected Impact:

Message will not be translated.

Workarounds:

Program the required destination ID in bits [31:16] of the region Upper Base Address Register.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003758 PCIe: 9000441819—Upstream Port does not transition to Recovery after receiving TS OSs during “ENTER_L2 negotiation”

Description:

Following is an example scenario for this erratum:

1. Downstream Port sends PME_Turn_Off message, Upstream Port sends PM_Enter_L23 DLLPs.
2. Downstream Port sends PM_Request_Ack DLLPs and does not move to Electrical Idle.
3. Upstream Port moves into Transmitter Electrical Idle and is waiting for Receiver Electrical Idle.
4. Downstream Port moves to Recovery and sends TS OSs.

Projected Impact:

Upstream Port receives TS OSs and transitions to L2_IDLE state. This causes Upstream Port to lose synchronization with Downstream Port. This is not a problem for the Downstream Port because the Downstream Port experiences a Fundamental Reset after entering L2_IDLE state, according to PCIe_CARD_ELECTROMECHANICAL specification.

Workarounds:

None

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003759 PCIe: 9000439510—Internal Address Translation Unit (iATU) can sometimes overwrite Outbound (Tx) Vendor Messages and MSIs

Description:

Outbound TLPs created at the vendor message interface (VMI) or the MSI interface are always subject to translation by the iATU. In PCI Express block configurations with an AHB/AXI interface and a 32-bit slave address bus width, the iATU incorrectly only considers the lower 32 bits of the 64-bit VMI or MSI address, when determining whether to translate the outbound TLP or not.

The VMI always uses 64 bits of the vend_msg_data input. These 64 bits of data are placed in DW3 and DW4 of the message TLP header and are treated by the iATU as an address.

When the lower 32 bits on vend_msg_data match an enabled iATU region, then the resulting TLP is incorrectly translated, regardless of the upper 32 bits. All of the 64-bits should have been checked in the iATU.

Following is an example scenario for this erratum with respect to VMI interface:

1. Setup any outbound iATU region (any type, any target address)
2. Send a message using VMI where the lower 32 bits of the message match the iATU region
3. The resulting Vendor Message TLP will be translated by the iATU regardless of the value of the upper 32 bits on vend_msg_data

Following is an example scenario for this erratum with respect to MSI interface:

1. Setup any outbound iATU region of any type where the lower 32 bits of the base address of the region match the lower 32 bits of the MSI address for any function within the device
2. Stimulate a MSI request
3. The resulting TLP will match the iATU region target specification and not the MSI address of the function

Projected Impact:

Messages are overwritten

Workarounds:

Program an iATU region to generate the MSI and then write to that region to generate the message.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR003760 PCIe: 9000439175—Poisoned Atomic Op requests targeting RTRGT0 receive UR response instead of CA response

Description:

The core does not support Atomic Ops that are targeted towards the RTRGT0, because RTRGT0 can only process one DWORD requests. Therefore, any Atomic Op request targeting the RTRGT0 interface should receive a CPL with CA completion status.

There is an issue when the core receives an Atomic Op request that is poisoned (EP bit is set to 1) and the request is targeting the RTRGT0 interface. The core correctly disregards the poisoned status as the CA response is a high priority error. However, the poisoned bit causes the internal filter to treat the request as UR instead of CA.

Projected Impact:

The core interprets the request as an unsupported request (UR), and updates the UR status register, instead of the CA status register.

Workarounds:

None.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004297 PCIe: 9000336356—Link configuration sometimes proceeds when incorrect TS Ordered Sets are received

Description:

The core moves ahead even when it does not receive the same non-PAD lane number in two consecutive TS Ordered Sets (OS) in the Link configuration process.

Scenario Setup:

- The link is in the link training phase.
- Remote partner sends TS OS without the same non-PAD lane number in any two consecutive TS OS on any active lane.
- The core moves ahead regardless of the non-PAD lane number not being the same in two consecutive TS OS.

Projected Impact:

This violates PCIe Base Specification “two consecutive TS Ordered Sets are received”. The core moves ahead without robustness to the Link configuration process.

Workarounds:

None. This issue will not lead to any compliance failures.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004298 PCIe: 9000471173—Bad DLLP error status checking is too strict

Description:

Figure 3-15 in Section 3.5.2.2. “Handling of Received DLLPs” of the PCI Express base Specification 3.0, indicates when a bad DLLP error should be reported. It should occur when the calculated CRC is not equal to the received value. The core correctly reports a bad DLLP error under this scenario. However, it also sets it if the Physical Layer reports a packet error during reception of the DLLP or if the DLLP ends with an ENDB symbol and not an END symbol. These extra conditions should not result in the reporting of a bad DLLP error.

Projected Impact:

A bad DLLP error is reported when it should not be. However, in this scenario, the core has received a bad DLLP. There are no adverse side effects.

Workarounds:

None required, there are no adverse side effects.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004299 PCIe: 9000493959—L1 ASPM incorrectly entered after link down event during L1 ASPM entry negotiation

Description:

Upstream ports are responsible for initiating entry into the L1 low-power state. The core implements an idle timer mechanism to trigger entry into the L1 state when L1 ASPM is enabled. If this timer has triggered but the port has not yet negotiated entry into the L1 low power state, and a link down event occurs, then the port will attempt to enter L1 again as soon as the link has resumed operation. This attempted L1 entry occurs, even though L1 ASPM is no longer enabled for the link (because of the link down reset).

Conditions:

Scenario setup:

- After link-up, enable L1 ASPM.
- Allow the link to go idle. Eventually, the port begins to request L1 entry by sending PM_Active_State_Request_L1 TLPs.
- Do not acknowledge the PM_Active_State_Request_L1 TLPs, but bring the link down by forcing the remote partner into the detect state.
- Allow the link to retrain to L0.
- After the link has retrained, the port will again attempt entry into the L1 ASPM state, even though L1 ASPM is now disabled.

Projected Impact:

As the remote partner is required to process the request, there should be no adverse affects. It cannot assume the state of the ASPM enable on the other side of the link. It should acknowledge the request either positively or negatively and normal operation can resume.

Workarounds:

None, but a graceful resumption of normal operation is expected.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004321 PCIe: 9000470913—Power Management Control: Core might enter L0s/L1 before Retry buffer is empty

Description:

The PCIe base specification states that before the L1 state can be entered, the Retry buffer must be empty.

For PM Directed L1 Entry

5.3.2.1. Entry into the L1 State

The Downstream component then waits until it receives a Link Layer acknowledgement for the PMCSR Write Completion, and any other TLPs it had previously sent. The component must retransmit a TLP out of its Data Link Layer Retry buffer if required to do so by Data Link Layer 15 rules.

For ASPM L1 Entry

5.4.1.2.1. Entry into the L1 State

The Downstream component must wait until it receives a Link Layer acknowledgement for the last TLP it had previously sent (the retry buffer is empty). The component must retransmit 30 a TLP out of its Data Link Layer Retry buffer if required by the Data Link Layer rules.

In Addition For Entry into The L0s State

5.4.1.1.1. Entry into the L0s State

No TLP is pending to transmit over the Link, or no FC credits are available to transmit any TLPs.

This can be interpreted as meaning the retry buffer should be empty. This is because it may be necessary to retransmit a TLP over the link, until a TLP has been acknowledged. The core does not wait for the retry buffer to be empty before commencing L0s or L1 entry.

Conditions:

Scenario Setup:

1. Transmit a TLP from the DWC_pcie core
2. Suppress Ack/Nak transmission from the Link Partner.
3. Initiate PM directed L1, ASPM L0s or ASPM L1 entry.
4. The core will enter the appropriate low power state, even though it still has TLPs in the retry buffer.

Projected Impact:

Low power states are entered inappropriately.

Workarounds:

This defect has no adverse side effects, rather a strict interpretation of the specification. The only consequence is that L0S (PCI Express Link Power State) will be exited prematurely if this condition is hit however will re-enter if the condition to enter prevails. So, just an early exit out of L0S but not functional problems or data corruption or compliance failure.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR004374 PCIe: 9000487440—TLP sometimes unnecessarily replayed

Description:

A DLLP Ack can be missed by the core on the receive path when it is immediately followed by EIOS.

Conditions:

After the Ack, two EIOS are seen on the PIPE interface. In this scenario, the Ack is missed by the RX logic, causing the corresponding TLP to be re-transmitted from the Tx replay buffer. Eventually, the link recovers from this event, as the receiver on the other side drops the re-transmitted TLP as a duplicate TLP. If the missed frame is a TLP, no ACK will be sent to the link partner, resulting in re-transmission of the TLP from the link partner.

Projected Impact:

Unnecessary re-transmission of a TLP. This is not a fatal error.

Workarounds:

No workaround. Also should not cause any compliance issues.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004489 PCIe: 9000505660—PCIe2 receiver equalizer settings

Description:

In the PCIe2 PHY, the rx0_eq[2:0] pins are controlled by the PCS, which currently drives these to a fixed value of 3’b000. This removes the capability to change RX equalization if any compliance issues are encountered.

Projected Impact:

RX equalization cannot be modified.

Workarounds:

The workaround is to override the RX_EQ settings, accordingly, using control registers inside the PHY:

```
RX_OVRD_IN_HI - 0x1006
10:8 - RX_EQ[2:0]
11 - RX_EQ_OVRD
```

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR004490 PCIe: 9000514662—LTSSM delay when moving from L0 to recovery upon receipt of insufficient TS1 Ordered Sets

Description:

When the remote link partner enters Recovery.rcvrlock from the L0 state and transmits only two TS1 Ordered Sets (OS), the core can sometimes miss the second TS1 OS and therefore, delay its entry into Recovery.rcvrlock.

Conditions:

Scenario Setup:

- The remote link partner enters Recovery.rcvrlock from the L0 state and transmits only two TS1 OSs
- The remote link partner then unusually moves to ElecIdle and de-asserts the PIPE signal rxvalid in Recovery.RcvrLock
- The expected response from the core is that it will transition to Recovery.rcvrlock on receipt of the two TS1 OSs
- The core receives a SKP OS or EIEOS that was inserted between the two TS1 Ordered Sets.

Projected Impact:

Consequences:

- The core might miss the second TS1 OS
- Because the remote partner only sent two TS1 OSs, the core will not receive a second TS1 OS and therefore, stays in L0
- The PHY detects a decode error and passes it the to core
- The core sends the error message to the remote link partner
- The core does not get a response from the remote partner and replays the message three times
- The replay timer rolls over (caused by unacknowledged ERR_CORR messages) and a link retrain is requested.
- The core moves to recovery.

Workarounds:

None. This is an unusual verification setup, and in a real system the remote partner must keep sending TS1 OSs in Recovery.RcvrLock and then the core will move to Recovery after receiving 2 TS1 OSs.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR004491 PCIe: 9000507633—TLP might be replayed an extra time before core enters recovery

Description:

The PCI Express base specification states in section 3.5.2.1 “If REPLAY_NUM rolls over from 11b to 00b, the Transmitter signals the Physical Layer to retrain the Link, and waits for the completion of retraining before proceeding with the replay.”

In the core, there are scenarios where the first TLP to be replayed might be replayed a fourth time before the link is retrained. This happens because the replay buffer logic requests the link to retrain at the same time that it begins a replay. If the link does not begin to retrain quickly enough, the first TLP of the replay may be transmitted again prior to link retraining.

Conditions:

Scenario Setup:

1. Transmit a series of TLPs from the core.
2. Send a Nak DLLP for the first TLP to initiate a replay.
3. Wait for the replay to begin.
4. Send a Nak DLLP for the first TLP to again initiate a replay.
5. Repeat steps (3) and (4) two more times.
6. After sending the fourth NAK, in some circumstances the first replayed TLP might be seen before the link begins to retrain.

Projected Impact:

A TLP is replayed before link retraining begins. However, the link recovers gracefully. After retraining, the TLP will be replayed again as necessary, until successful transmission is achieved.

Workarounds:

None, the link recovers gracefully.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005186 PCIe: The PCIe Controller Core Does Not Send Enough TS2 Ordered Sets During Link Retrain And Speed Change

Description:

The PCIe core might send less than 32 TS2 ordered sets during link retraining and speed changing if the remote partner sends more TS1 ordered sets than expected. This occurs when the “Extended Synch” bit cleared in PCIe and set at the remote partner.

Scenario Setup:

- Link partners agree to do speed change negotiation and move to Recovery.
- The remote partner stays in Recovery.RcvrLock longer and sends more TS1s (for example, Extended Synch bit is set).
- In Recovery.RcvrCfg state core will send less than 32 TS2s before transition to Recovery.Speed

Projected Impact:

The speed change negotiation may fail.

Workarounds:

In current PCIe core, there is no signal indicating that remote partner has “Extended Synch” bit set per PCIe base spec. The workaround is to know in advance if the link partner has the “Extended Synch” bit set and in that case set that bit in the PCIe also. The “Extended Synch” is intended for a logic analyzer. It may be used if there are repeaters on the link.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005188 PCIe: The PCIe Controller cannot exit successfully L1 state of LTSSM when the Core Clock is removed

Description:

Under the condition where the core enters L1 and is then directed to immediately exit due to a pending TLP transmission, the LTSSM misses the PhyStatus pulse because of the gated core_clk in clk_rst.v.

Scenario Setup:

- LTSSM enters L1 and indicates to the PHY to change Powerdown to P1.
- Core immediately gets a wake-up event and wants to exit from L1. To make the transition into Recovery, the core needs to receive PhyStatus back from the PHY
- The PHY changes Powerdown to P1 and asserts PhyStatus back to Core.
- LTSSM moves to Recovery state and indicates to the PHY to change Powerdown to P0
- core_clk is gated off immediately after LTSSM enters Recovery. This can happen as a result of the logic in the clk_rst module that is performing glitch-less clock switch on aux_clk
- The PHY changes Powerdown to P0 and asserts PhyStatus back to core.
- LTSSM misses the PhyStatus because core_clk is still gated off by the clk_rst module.

Projected Impact:

The core's LTSSM does not proceed to exit from Recovery and is awaiting a Powerdown indication from the PHY. The LTSSM will eventually timeout and move to Detect state and resume operation by initiating receiver detection.

Workarounds:

Increase the programming of the "Low Power Entrance Count" field of the "Port Force Link Register" (maximum is 255). This delays the entry into L1 and prevents the problem from occurring. The "Low Power Entrance Count" field is for Power Management state to wait for these many clock cycles for the associated completion of a configuration write to D-state register to go low power. The longer delay is to ensure that the completion TLP can be sent by the core to avoid immediate waking-up after entering L1.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005189 PCIe: PCIe Gen2/Gen3 Hardware Autonomous Speed Disable Bit In Configuration Register is not sticky

Description:

Hardware Autonomous Speed Disable bit Attributes in *PCIe Base Specification Rev 2.0* were RW/RsvdP. In *Rev 3.0* it is changed to RWS/RsvdP.

Scenario Setup:

- "No soft reset" bit is programmed to 0 on any function.
- "Hardware Autonomous Speed Disable" bit is programmed to 1, if the component does not want to adjust the Link speed autonomously.
- Train the Link to Hot Reset.
- Core will have a Link down reset which causes non-sticky reset.

Projected Impact:

The component is permitted to autonomously adjust the Link speed.

Workarounds:

No workaround, no effect on software but non compliance with standard.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005723 PCIe: PCIe does not support L2 power down

Description:

When PCIe works as Root Complex, it can exit L2 mode only through reset. Since PCIe does not have a dedicated reset control bit, it cannot exit L2 mode.

Projected Impact:

PCIe does not support L2 power down.

Workarounds:

The PCIe can be put into PDDQ mode to save PCIe PHY power and wake up only by the OOB (Out of Band) wakeup signal (since wakeup by a beacon from link partner is not supported) driven from the link partner (End Point). This signal could be used as a GPIO interrupt to exit this mode.

The limitation of this workaround is that the link partner cannot be put into L2 mode.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005645 ROM: Normal SD clock speed (SDR12) not selectable in SD/SDXC boot mode

Description:

When booting in SD/SDXC boot mode, users cannot set the SD clock speed to Normal mode (SDR12). Selecting the SDR12 boot switch setting for BOOT_CFG1[3:2] in the fuse table will default to the High Speed mode (SDR25) due to an incorrect mapping in the boot ROM.

Projected Impact:

Due to this mapping issue, the user cannot select Normal SD clock speed at boot time; however, this will not cause any issues. For an older SD device that does not support high-speed mode, ROM will first attempt high speed mode and when this mode fails will automatically switch to normal speed and continue normally with the boot process. This mapping issue does not impact MMC boot.

Workarounds:

None. The minimum SD clock speed supported is high-speed mode (SDR25) for initial booting in SD/SDXC boot mode. When booting with an SD card that only supports SD clock speed in Normal mode (SDR12), users need to be aware of the revised SD/SDXC boot mode switch settings for BOOT_CFG1[3:2] given in Table 3. The automatic switch from high speed to normal speed is transparent to the user.

Table 3. Revised SD/SDXC Boot Mode Switch Settings for BOOT_CFG1[3:2]

BOOT_CFG1[3:2]	SD/SDXC Boot Speed
0x	SDR25
10	SDR50
11	SDR104

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR005768 ROM: In rare cases, secondary image boot flow may not work due to mis-sampling of the WDOG reset

Description:

In case the primary image authentication fails, ROM will try to perform a WDOG reset and boot with the secondary image. However ROM does not set the SRE bit of watchdog control register which may cause a WDOG reset failure occasionally and result in ROM staying in an endless loop.

Projected Impact:

The secondary boot may not work in the first attempt.

Workarounds:

There are no software workarounds for this issue, instead the user will need to reboot the IC, which will force a second iteration of the secondary boot.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR003778 SSI: In AC97, 16-bit mode, received data is shifted by 4-bit locations

Description:

When the SSI is configured in AC97, 16-bit mode, the Rx data is received in bits [19:4] of the RxFIFO, instead of [15:0] bits.

Projected Impact:

The SDMA script should be updated accordingly to perform the shift to the right location on the fly during data transfer. If the data register is accessed directly by software, it should account for the shifted data and perform shifting to the right location.

Workarounds:

The data should be shifted to the right location by the SDMA script or by the software in case of direct access to the register.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3. Implemented by using the latter method mentioned in the Workarounds section.

ERR005764 SSI: AC97 receive data may be wrong when clock ratio between external clock to ipg is higher than 1:8

Description:

The data in SSI gets corrupted in the following configuration:

- SSI is configured to AC_97 mode
- Transmitter and receiver are enabled
- The IPG_CLK and external clock ratio is higher than 1:8

The internal “ignore_time_slot” signal may de-assert for 1 cycle between frames. This may result in ambiguous behavior where the synchronization and identification of “ignore_time_slot” requires 4 ipg_clk cycles to fit in a half cycle of the external clock.

Projected Impact:

Data corruption in the specific configuration.

Workarounds:

Do not use the following configuration:

- SSI is configured to AC_97 mode
- Transmitter and receiver are enabled
- The IPG_CLK and external clock ratio is higher than 1:8

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR004534 USB: Wrong HS disconnection may be generated after resume

Description:

When the IC works as a USB host and one High Speed device is connected, software can put it into Suspend mode and it can wake up by a Host Resume or a remote wakeup.

The UTM block drives FS-K during resume and drives SE0 as the end of the resume. Meanwhile UTM bypasses the DP/DN lines to USB controller. Once the controller detects the SE0, it will switches to High Speed. Once UTM detects it switches to High Speed, it will stop driving SE0. After that, the controller starts to send SOF through UTM.

If the controller sends the SOF too fast, while the external device may still be in Full Speed mode, the SOF signal level will be 800mV which will be recognized as a High Speed disconnection.

The USB controller may send the SOF packet during that period, but according to USB2.0 spec, DP/DN should keep in IDLE (SE0 state) for 1.333 μs after resume to avoid this issue (the device must switch to High Speed in 1.333 μs).

Projected Impact:

HS disconnection after resume with some devices.

Workarounds:

The UTM block has a configurable bit (HW_USBPHY_CTRL.ENHOSTDISCONDETECT) to enable/disable the High Speed disconnection detection circuit. This bit should be used to disable this in Suspend, and enable after resume.

Proposed Solution:

No fix scheduled

Linux BSP Solution:

Software workaround implemented in BSP version ER3

ERR006281 USB: Incorrect DP/DN state when only VBUS is applied

Description:

When VBUS is applied without any other supplies, incorrect communication states are possible on the data (DP/DN) signals. If VDDHIGH_IN is supplied, the problem is removed.

Projected Impact:

This issue primarily impacts applications using charger detection to signal power modes to a PMIC in an undercharged battery scenario where the standard USB current allotment is not sufficient to boot the system.

Workarounds:

Apply VDDHIGH_IN if battery charge detection is needed. Otherwise, disable charger detection by setting the EN_B bit in USB_ANALOG_USBx_CHRG_DETECTn to 1.

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround not applicable to the BSP

ERR006308 USB: Host non-doubleword –aligned buffer address can cause host to hang on OUT Retry

Description:

The USB host core operating in streaming mode may underrun while sending the data packet of an OUT transaction. The host then retries the OUT transaction according to the USB specification.

This issue occurs during the OUT retry. The USB host may hang on OUT retry if the data buffer start address is not 4-byte aligned.

This applies to both the host controller and the OTG controller in host mode.

Projected Impact:

Host controller only transmits SOF packets. All other traffic is blocked.

Workarounds:

- Set the host TXFIFO threshold to a large value (TXFIFOTHRES in the TXFILLTUNING register). This increases the tolerance to bus latency and avoids a FIFO underrun.
- Set the Stream Disable bit (SDIS) to 1 in the USBMODE register. This forces the controller to load an entire packet in the FIFO before starting to transmit on the USB bus. Hence, the FIFO never underruns. This somewhat reduces the maximum bandwidth of the USB, because there is idle time when the the controller waits for the entire packet to be loaded.

Proposed Solution:

No fix scheduled.

Linux BSP Status:

Software workaround implemented in Linux BSP version L3.0.35_4.0.0

ERR004535 USB: USB suspend and resume flow clarifications

Description:

In device mode, The PHY can be put into Low Power Suspend when the device is not running or the host has signaled suspend. The PHY Low power suspend bit (PORTSC1.PHCD) will be cleared automatically when the host initials resume. Before forcing a resume from the device, the device controller driver must clear this bit. In host mode, the PHY can be put into Low Power Suspend when the downstream device has been placed into suspend mode (PORTSC1.SUSP) or when no downstream device is connected. Low power suspend is completely under the control of software.

To place the PHY into Low power mode, software needs to set PORTSC1.PHCD bit, set all bits in USBPHY_PWD register and set the USBPHY_CTRL.CLKGATE bit.

When a remote wakeup occurs after the Suspend (SUSP) bit is set while the PHY Low power suspend bit (PHCD) is cleared, a USB interrupt (USBSTS.PCI) will be generated. In this case, the PHCD bit will NOT be set because of the interrupt. However, if a remote wakeup occurs after the PHCD bit is set while the USB PHY Power-Down Register (USBPHY_PWD) and the UTMI clock gate (USBPHY_CTRL.CLKGATE) bit is cleared, a remote wakeup interrupt will be generated. In this case, all the bits in the HW_USBPHY_PWD register and the USBPHY_CTRL.CLKGATE bit will be set, even after the remote wakeup interrupt is generated, which is incorrect.

Projected Impact:

Resume error, if the correct flow is not adhered to.

Workarounds:

To place the USB PHY into low power suspend mode, the following sequence should be performed in an atomic operation (interrupts should be disabled during these three steps):

1. Set the PORTSC1.PHCD bit
2. Set all bits in the USBPHY_PWD register
3. Set the USBPHY_CTRL.CLKGATE bit

Proposed Solution:

No fix scheduled

Linux BSP Status:

Software workaround implemented in BSP version ER3

ERR004349 VPU: Cannot decode Sorenson Spark Version 0 bitstream

Description:

The bitstream of Sorenson Spark codec has two versions: Version 0 and Version 1. This issue causes the VPU to fail to decode Version 0 bitstream (sequence initialization error) because the VPU cannot find the start code and returns the SEQ_INIT error for decoding a Version 0 bitstream. The VPU can decode a Version 1 bitstream.

Projected Impact:

Version 0 of Sorenson Spark bitstream cannot be decoded.

Workarounds:

No software workaround.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

ERR005777 XTAL: In some cases, the 24 MHz oscillator start-up is slow

Description:

In some cases, the 24 MHz oscillator start-up is slow.

Projected Impact:

Adds 1.5 ms to the startup sequence.

Workarounds:

The addition of a 2.2 MΩ external resistor from the XTALI pin to ground will ensure that the 24 MHz oscillator start-up is not delayed.

Proposed Solution:

No fix scheduled

Linux BSP Status:

No software workaround available

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

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

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 that may be provided in Freescale 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 does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale and the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. ARM is the registered trademarks of ARM Limited.
© 2012, 2013 Freescale Semiconductor, Inc. All rights reserved.

Document Number: IMX6SDLCE
Rev. 2
5/2013