Arm Cortex-A35 (MP060)

Software Developer Errata Notice

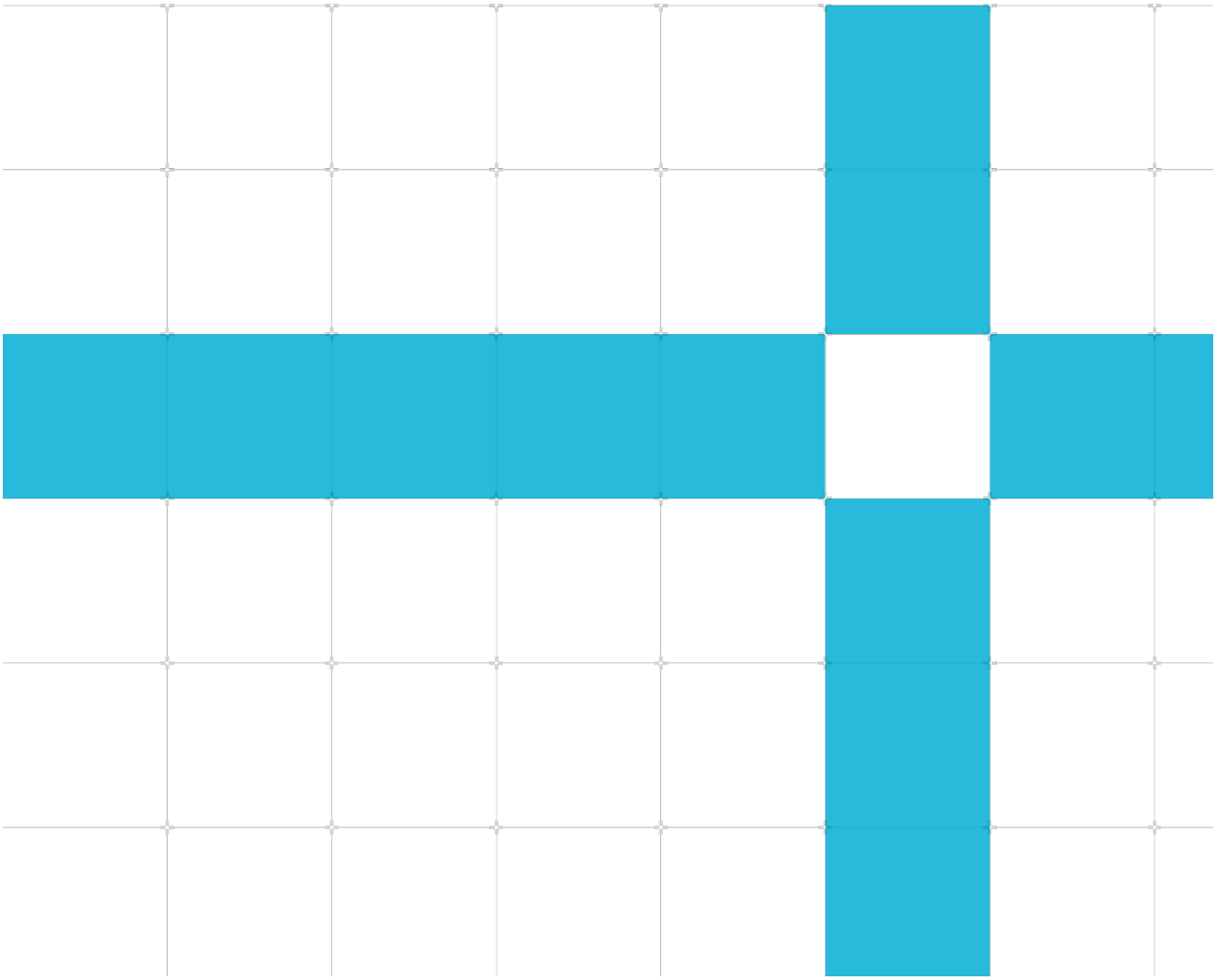
Date of issue: 10-Feb-2023

Non-Confidential

Document version: v6.0

Document ID: SDEN-843627

This document contains all known errata since the r0p0 release of the product.



Non-confidential proprietary notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on Arm Cortex-A35 (MP060), create a ticket on **https://support.developer.arm.com**.

To provide feedback on the document, fill the following survey: **https://developer.arm.com/documentation-feedback-survey**.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

If you find offensive language in this document, please email **terms@arm.com**.

Contents

Introduction

Scope

This document describes errata categorized by level of severity. Each description includes:

- The current status of the erratum.
- Where the implementation deviates from the specification and the conditions required for erroneous behavior to occur.
- The implications of the erratum with respect to typical applications.
- The application and limitations of a workaround where possible.

Categorization of errata

Errata are split into three levels of severity and further qualified as common or rare:

Category A	A critical error. No workaround is available or workarounds are impactful. The error is likely to be common for many systems and applications.
Category A (Rare)	A critical error. No workaround is available or workarounds are impactful. The error is likely to be rare for most systems and applications. Rare is determined by analysis, verification and usage.
Category B	A significant error or a critical error with an acceptable workaround. The error is likely to be common for many systems and applications.
Category B (Rare)	A significant error or a critical error with an acceptable workaround. The error is likely to be rare for most systems and applications. Rare is determined by analysis, verification and usage.
Category C	A minor error.

Change Control

Errata are listed in this section if they are new to the document, or marked as "updated" if there has been any change to the erratum text. Fixed errata are not shown as updated unless the erratum text has changed. The **errata summary table** identifies errata that have been fixed in each product revision.

10-Feb-2023: Changes in document version v6.0

ID	Status	Area	Category	Summary
2252746	New	Programmer	Category C	ATB flush response may be delayed

23-Oct-2019: Changes in document version v5.0

ID	Status	Area	Category	Summary
1608096	New	Programmer	Category B	Speculative AT instruction using out-of-context translation regime could cause subsequent request to generate an incorrect translation

28-Feb-2019: Changes in document version v4.0

ID	Status	Area	Category	Summary
1010162	New	Programmer	Category C	PMU counter might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST

29-Mar-2017: Changes in document version v3.0

ID	Status	Area	Category	Summary
743488	New	Programmer	Category B	Clearing TCR_ELx.TBI from inside a tagged address region might cause a Translation fault or an Address size fault
799764	New	Programmer	Category B	Some AT instructions executed from EL3 might incorrectly report a domain fault
676796	New	Programmer	Category C	ROM table entries for cores 1, 2, and 3 might be ignored
689935	New	Programmer	Category C	ETM might output an incorrect exception return address
702094	New	Programmer	Category C	WFx can cause the PC to jump from lower VA subrange to upper VA subrange
705778	New	Programmer	Category C	Accessing EDPCSR has side-effects when OS Lock is locked
707223	New	Programmer	Category C	PMU counter value might be inaccurate when monitoring EXC_RETURN
755748	New	Programmer	Category C	ETM might assert AFREADY before all trace has been output
755766	New	Programmer	Category C	ETM might not generate an event packet and ATB trigger
776025	New	Programmer	Category C	Instruction cache parity error might cause an incorrect abort to be taken
797529	New	Programmer	Category C	Incorrect fault status codes used for reporting synchronous uncorrectable ECC aborts in the IFSR
798161	New	Programmer	Category C	ETM trace reports incorrect branch target
801757	New	Programmer	Category C	ATS12NSOPR instruction might incorrectly translate when the HCR.TGE bit is set
818510	New	Programmer	Category C	Debug not entering Memory Access mode without setting EDSCR.ERR
857487	New	Programmer	Category C	Mismatch between EDPRSR.SR and EDPRSR.R

10-Mar-2016: Changes in document version v2.0

ID	Status	Area	Category	Summary
607615	New	Programmer	Category B	ETM does not report IDLE state when disabled using OSLOCK
607813	New	Programmer	Category B	An eviction might overtake a cache clean operation
603177	New	Programmer	Category B (rare)	A Store-Exclusive instruction might pass when it should fail

04-Dec-2015: Changes in document version v1.0

No errata in this document version.

Errata summary table

The errata associated with this product affect the product versions described in the following table.

ID	Area	Category	Summary	Found in versions	Fixed in version
1608096	Programmer	Category B	Speculative AT instruction using out-of-context translation regime could cause subsequent request to generate an incorrect translation	r0p0, r0p1, r0p2, r1p0	Open
799764	Programmer	Category B	Some AT instructions executed from EL3 might incorrectly report a domain fault	r0p0, r0p1, r0p2, r1p0	Open
743488	Programmer	Category B	Clearing TCR_ELx.TBI from inside a tagged address region might cause a Translation fault or an Address size fault	r0p0, r0p1	r0p2
607813	Programmer	Category B	An eviction might overtake a cache clean operation	r0p0	r0p1
607615	Programmer	Category B	ETM does not report IDLE state when disabled using OSLOCK	r0p0	r0p1
603177	Programmer	Category B (rare)	A Store-Exclusive instruction might pass when it should fail	r0p0	r0p1
2252746	Programmer	Category C	ATB flush response may be delayed	r0p0, r0p1, r0p2, r1p0	Open
1010162	Programmer	Category C	PMU counter might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST	r0p0, r0p1, r0p2, r1p0	Open
857487	Programmer	Category C	Mismatch between EDPRSR.SR and EDPRSR.R	r0p0, r0p1, r0p2, r1p0	Open
818510	Programmer	Category C	Debug not entering Memory Access mode without setting EDSCR.ERR	r0p0, r0p1	r0p2
801757	Programmer	Category C	ATS12NSOPR instruction might incorrectly translate when the HCR.TGE bit is set	r0p0, r0p1, r0p2, r1p0	Open
798161	Programmer	Category C	ETM trace reports incorrect branch target	r0p0, r0p1	r0p2
797529	Programmer	Category C	Incorrect fault status codes used for reporting synchronous uncorrectable ECC aborts in the IFSR	r0p0, r0p1	r0p2
776025	Programmer	Category C	Instruction cache parity error might cause an incorrect abort to be taken	r0p0, r0p1	r0p2
755766	Programmer	Category C	ETM might not generate an event packet and ATB trigger	r0p0, r0p1	r0p2

ID	Area	Category	Summary	Found in versions	Fixed in version
755748	Programmer	Category C	ETM might assert AFREADY before all trace has been output	r0p0, r0p1	r0p2
707223	Programmer	Category C	PMU counter value might be inaccurate when monitoring EXC_RETURN	r0p0, r0p1	r0p2
705778	Programmer	Category C	Accessing EDPCSR has side-effects when OS Lock is locked	r0p0, r0p1	r0p2
702094	Programmer	Category C	WFx can cause the PC to jump from lower VA subrange to upper VA subrange	r0p0, r0p1	r0p2
689935	Programmer	Category C	ETM might output an incorrect exception return address	r0p0, r0p1	r0p2
676796	Programmer	Category C	ROM table entries for cores 1, 2, and 3 might be ignored	r0p0, r0p1	r0p2

Errata descriptions

Category A

There are no errata in this category.

Category A (rare)

There are no errata in this category.

Category B

1608096
Speculative AT instruction using out-of-context translation regime could cause subsequent request to generate an incorrect translation

Status

Affects: Cortex-A35
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Open.

Description

A speculative Address Translation (AT) instruction translates using registers that are associated with an out-of-context translation regime and caches the resulting translation in the TLB. A subsequent translation request that is generated when the out-of-context translation regime is current uses the previous cached TLB entry producing an incorrect virtual to physical mapping.

Configurations Affected

All configurations are affected.

Conditions

1. A speculative AT instruction performs a table walk, translating a virtual address to a physical address using registers associated with an out-of-context translation regime.
2. Address translation data that is generated during the walk is cached in the TLB.
3. The out-of-context translation regime becomes current and a subsequent memory access is translated using previously cached address translation data in the TLB, resulting in an incorrect virtual to physical mapping.

Implications

If the above conditions are met, the resulting translation would be incorrect.

Workaround

When context-switching the register state for an out-of-context translation regime, system software at EL2 or above must ensure that all intermediate states during the context-switch would report a level 0 translation fault in response to an AT instruction targeting the out-of-context translation regime. A workaround is only required if the system software contains an AT instruction as part of an executable page.

799764
Some AT instructions executed from EL3 might incorrectly report a domain fault

Status

Affects: Cortex-A35
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Open.

Description

Address translation instructions executed from EL3 and targeting EL1 or EL0 might report an incorrect result in the PAR when the HCR_EL2.DC bit is set.

Configurations Affected

This erratum affects all configurations of the processor.

Conditions

1. The core is executing at Exception level 3 in AArch64.
2. The core executes one of the following address translation instructions:
   - AT S1E0R, AT S1E0W
   - AT S1E1R, AT S1E1W
   - AT S12E0R, AT S12E0W
   - AT S12E1R, AT S12E1W
3. The Exception level targeted by the address translation instruction is Non-secure and AArch32.
4. HCR_EL2.DC is set to 1.
5. The DACR is programmed so that domain 0 would cause a domain fault if the HCR_EL2.DC bit had not been set. Note that this is the default value out of reset.

Implications

The PAR register will incorrectly report that a domain fault occurred.

Workaround

Secure software can set the DACR[1:0] to 0b01 before executing the address translation instruction. It should restore the previous DACR value before returning to a lower Exception level.

743488
Clearing TCR_ELx.TBI from inside a tagged address region might cause a Translation fault or an Address size fault

Status

Affects: Cortex-A35
Fault Type: Programmer Category B
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The ARMv8 architecture supports address tagging in the AArch64 execution state. When address tagging is used, the top byte of the virtual address is ignored during address translation. If a branch is executed, tag bits should be removed from the target address when the program counter is updated. Because of this erratum, the tag bits might propagate to the program counter. If the Top Byte Ignored bit is subsequently cleared in the relevant Translation Control Register, then the processor might report a fault because the program counter is out of range.

Configurations Affected

All configurations are affected.

Conditions

1. A core is executing in AArch64.
2. The TBI bit in the Translation Control Register for the current Exception level is set to 1.
3. The core executes an indirect branch to an address where the top byte is not sign-extended.
4. The branch was incorrectly predicted not-taken before its execution.

Implications

If the above conditions are met, then the address tag might propagate to an internal copy of the program counter. Reads of the PC register are not affected.

If the TBI bit remains set to 1, then the address tag will be removed from the program counter at the next context synchronization event. Address translations for instruction fetches will continue to ignore the top byte of the virtual address and no unexpected behavior will occur.

If the TBI bit is cleared to 0 by the code in the address-tagged region, and if the change takes effect before or as part of the next context synchronization event, then the core might report a fault because the program counter is out of range.

For example, if an MSR instruction to clear the TBI bit is followed by an ISB instruction, then successful execution of the ISB will resolve the issue. However, if the change in TBI bit takes effect before the ISB is executed then the address translation that is performed to fetch the ISB instruction might trigger a Translation fault (if the translation system is enabled) or an Address size fault (if the translation system is not enabled).

Workaround

The issue can be avoided by inserting an ISB instruction before the instruction that clears the TBI bit.

607813
An eviction might overtake a cache clean operation

Status

Affects: Cortex-A35
Fault Type: Programmer Category B
Fault Status: Present in r0p0, Fixed in r0p1

Description

The Cortex-A35 processor supports instructions for cache clean operations. To avoid data corruption, the processor must ensure correct ordering between evictions and cache clean operations for the same address.

Because of this erratum, the processor might issue an eviction and an L2 cache clean operation to the interconnect in the wrong order. The processor might also issue the transactions such that they are outstanding in the interconnect at the same time. This violates the ACE protocol specification and might cause the transactions to be erroneously re-ordered in the interconnect.

Configurations Affected

To be affected by this erratum, the following must be true:

- The processor is configured with an ACE bus interface.
- The processor is configured with an L2 cache.

Conditions

The erratum can be hit if the following conditions are met under specific timing conditions.

1. One or both of the following are true:
   a. L2ACTLR[14] is set to 1. This enables sending of WriteEvict transactions on the ACE interface when the processor evicts data that it holds in the UniqueClean state.
   b. L2ACTLR[3] is set to 0. This enables sending of Evict transactions on the ACE interface when the processor evicts clean data.
2. A core executes a cache clean by address operation for a line that is present and dirty in the L2 cache.
3. A core performs a memory access to the same set. This could be any type of memory access including a pagewalk, an instruction fetch, a cache maintenance operation, or a data access.
4. The instruction in condition (3) triggers an L2 cache eviction.
5. The line chosen for eviction from the L2 cache is the same line that was targeted by the cache clean operation in condition (2).

Implications

If the processor is connected to an interconnect that has a system cache or a snoop filter then this erratum might cause data corruption.

Workaround

The erratum can be avoided by upgrading cache clean by address operations to cache clean and invalidate operations. This can be achieved by setting CPUACTLR.ENDCCASCI to 1.

607615
ETM does not report IDLE state when disabled using OSLOCK

Status

Affects: Cortex-A35
Fault Type: Programmer Category B
Fault: Status: Present in r0p0, Fixed in r0p1

Description

The OS Lock feature in the ETM allows software running on a processor to disable external debug access then save the register state before powering down the ETM. There is a defined sequence which must be followed to ensure that the register state is stable and that all trace has been output before the system is powered-down. Because of this erratum, when the OS Lock mechanism is used, the ETM will never indicate that it is safe to be powered-off.

Configurations Affected

All configurations are affected.

Conditions

1. The ETM is enabled using TRCPRGCTLR.EN == 1
2. The OS Lock feature is used to disable the ETM using TRCOSLAR.OSLK==1

Implications

Software which follows the defined save and restore sequence will poll TRCSTATR.IDLE, but this will remain HIGH even though the ETM will be disabled and will drain. If the ETM was already disabled with TRCPRGCTLR.EN == 0 then the sequence will behave correctly.

Workaround

After the ETM has been disabled using TRCOSLAR.OSLK and the state of TRCPRGCTLR.EN has been recorded, TRCPRGCTLR.EN can be written LOW. This will allow the disable sequence to complete correctly.

Category B (rare)

603177
A Store-Exclusive instruction might pass when it should fail

Status

Affects: Cortex-A35
Fault Type: Programmer Category B (Rare)
Fault Status: Present in r0p0, Fixed in r0p1

Description

The processor implements an internal exclusive monitor to manage Load-Exclusive, Store-Exclusive, and Clear-Exclusive instructions. Because of this erratum, a Load-Exclusive instruction to cacheable memory might set the monitor to the exclusive state when the processor does not have exclusive access to the line. A subsequent Store-Exclusive instruction might pass when it should fail.

The erratum affects all Load-Exclusive and Store-Exclusive instructions, including Load-Acquire Exclusive and Store-Release Exclusive instructions.

Configurations Affected

All configurations are affected.

Conditions

1. A core executes a store to memory that is marked as both inner-writeback and outer-writeback.
2. The store is not a Store-Exclusive (or a Store-Release Exclusive) or a Store-Release instruction.
3. The store is not followed by a DMB SY or DSB.
4. The store misses in the L1 data cache.
5. The store does not trigger a linefill. This requires one or more of the following to be true:
   - The core is in read-allocate mode.
   - The memory is marked as no-write-allocate.
   - The memory is marked as transient.
   - The store is a STNP instruction.
   - The store is triggered by a DC ZVA instruction.
6. The core starts a linefill to the same address as the store. The linefill is started for one of the following:
   - A PRFM, PLD, or PLDW instruction.
   - An automatic data prefetch.
   - A pagewalk.
7. The core executes a Load-Exclusive (or a Load-Acquire Exclusive) instruction to the same address as the store.

8. The store data is forwarded to the Load-Exclusive instruction.
9. The Load-Exclusive instruction retires before the linefill in condition (6) is serialised.

If the above conditions are met then the processor might set the internal exclusive monitor. This is not correct because the processor is not guaranteed to have exclusive access to the line.

Implications

If another core or master executes a Load-Exclusive instruction to the same address then both cores or masters might gain access to an exclusive region of code at the same time.

Most of the code sequences that can hit this erratum are not expected to exist commonly in real code. The scenario involving read-allocate mode in condition (5) is the most plausible.

The erratum also affects Cortex-A53 and has not been observed in the field. This indicates that the code sequences and timing conditions required to hit the erratum are rare.

Workaround

If a workaround is required then the only workaround is to avoid the conditions described. Disabling read-allocate mode by setting CPUACTLR.RADIS to 0b11 degrades write-stream performance. Therefore, the preferred workaround is to avoid conditions (2) or (3) using an appropriate Store-Release or DMB instruction.

Category C

2252746
ATB flush response may be delayed

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, and r1p0. Open.

Description

The *Embedded Trace Macrocell* (ETM) supports an external flush request for each ATB bus. Under certain timing conditions, an AFREADY response from the processor may be delayed until a new ATB transfer is generated by the processor.

Configurations Affected

This erratum affects all processor configurations.

Conditions

The erratum occurs if the following sequence of conditions is met:

1. The ETM is enabled
2. Trace data is generated on the ATB bus
3. An external flush request is generated
4. Trace data stops being generated due to filtering in the ETM or the ETM being disabled

Implications

External trace infrastructure which is waiting for trace to be captured may stall forever (for example in a 'flush and stop' scenario). All of the trace data will be captured, but it is not possible to identify this by polling the Trace Capture Device. If the flush is acknowledged, this can be treated as a reliable indication.

If the ETM is enabled again after the erratum has been triggered, the flush logic should become active again. If a flush is generated while trace is being generated, the only effect will be a delay in acknowledging the flush. This should not have any observable impact.

In a system with multiple trace sources, the delayed flush response may prevent other trace sources from accessing the ATB bus if they are generating trace while the flush is in progress. This could cause trace to be lost from these other sources.

Workaround

There is no workaround to reliably avoid this erratum. As an alternative to waiting for the flush to be acknowledged, a sufficiently long timeout can be used if it is likely that trace generation has stopped. If ATB upsizers are present in the system, this workaround will not be effective.

1010162
PMU counter might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST

Status

Affects: Cortex-A35 MPCore
Fault type: Programmer Category C
Fault status: Present in r0p0, r0p1, r0p2, r1p0. Open.

Description

The Cortex-A35 processor implements a Performance Monitor Unit (PMU). The PMU allows programmers to gather statistics on the operation of the processor during runtime. Because of this erratum, the PMU counter values might be inaccurate when monitoring BUS_ACCESS and BUS_ACCESS_ST events.

Configurations Affected

To be affected by this erratum, one or more of the following must be true:

- The processor is configured with an ACE or CHI bus interface
- The processor is configured with more than one core
- The processor is configured with an L2 cache
- The processor is configured with CPU cache protection

Conditions

1. A performance counter is enabled and configured to count BUS_ACCESS or BUS_ACCESS_ST events.
2. A write or eviction occurs on the bus.

Implications

The PMU counter will erroneously increment or erroneously fail to increment. The inaccuracy varies between cores. Some bus accesses for core 0 might be attributed to core 1. Some bus accesses for core 1 might be attributed to core 2 or core 3. Bus accesses for cores 2 and 3 will not be attributed to any core. The number of cores present in the configuration does not affect how the bus accesses are attributed. For example, some core 1 accesses might be attributed to cores 2 and 3 even if the configuration has only two cores. The impact on the final counter value in each core is high.

This might lead to inaccurate results when using the PMU to debug or profile code.

Workaround

The BUS_ACCESS and BUS_ACCESS_ST events can be counted accurately for core 0 by enabling the counter on both core 0 and core 1 and taking the total. This workaround requires core 1 to be present in the configuration and idle during testing. Similarly, the total count for core 0 and core 1 can be found by enabling the counter on all four cores and taking the total. This workaround requires cores 2 and 3 to be present in the configuration and idle during testing. In a configuration with four cores, the events can be used with an intensive memory test that runs on two cores to give a reasonable indication of maximum bandwidth for the cluster.

The event L2D_CACHE_WB counts write-backs from the L2 cache that are attributable to a core. For a test focused on cacheable memory bandwidth, this might be a suitable replacement for BUS_ACCESS_ST. This event includes:

- Write-backs as a result of evictions and cache maintenance instructions.
- Writes in read allocate mode (write-streaming mode).

L2D_CACHE_WB does not include:

- Write-backs as a result of snoop requests from outside of the cluster.
- Write-backs as a result of ACP interface accesses.

857487
Mismatch between EDPRSR.SR and EDPRSR.R

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Open.

Description

The processor provides access to the EDPRSR through the APB interface. If this access is done at the same time as the core leaves a Warm reset, then a subsequent read of the same register will read an incorrect value of the SR field.

Configurations affected

This erratum affects all configurations of the processor.

Conditions

1. The core is in Warm reset.
2. A debugger reads the EDPRSR register over the APB interface.
3. The core comes out of Warm reset during the APB read.
4. A second APB read is made to the EDPRSR register.

Implications

The first read of the EDPRSR will read the SR field and R field both as 0b1.
The second read of the EDPRSR.SR field will read 0b0 whereas the previous read of the EDPRSR.SR was 0b1 while in Warm reset. Because the first read took place while in Warm reset, the sticky bit should still be set on the second read.

Workaround

If the debugger reads the EDPRSR and sees both the SR and R fields set, then it must remember this result and on the next EDPRSR read treat the SR bit as if it was set.

818510
Debug not entering Memory Access mode without setting EDSCR.ERR

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The processor supports entering debug Memory Access (MA) mode through the APB interface. This might fail when an instruction previously executed through the APB interface has not yet completed.

Configurations affected

This erratum affects all configurations of the processor.

Conditions

1. The debugger inserts an instruction through the APB interface by writing the EDITR.
2. The debugger writes to the EDSCR to enter MA mode.
3. The debugger reads the DTRTX at the same time as the instruction completes.

Implications

The APB read of the DTRTX does not start up the MA state machine, but EDSCR.TXU and EDSCR.ERR are not set.

Workaround

Before executing the instruction, the debugger should write 1 to the EDRCR.CSPA. After executing the instruction, the debugger should poll the EDSCR.PipeAdv bit to determine when the instruction has completed, and only attempt to enter MA mode after the PipeAdv bit is set.

801757
ATS12NSOPR instruction might incorrectly translate when the HCR.TGE bit is set

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1, r0p2, r1p0. Open.

Description

An ATS12NSOPR address translation instruction executed from EL3 might report an incorrect result in the PAR when both the HCR.TGE bit is set and the SCTLR.M is set.

Configurations Affected

This erratum affects all configurations of the processor.

Conditions

1. The core is executing at Exception level 3 in AArch32.
2. SCR.NS = 0
3. HCR.TGE = 1
4. SCTLR(ns).M = 1
5. The core executes an ATS12NSOPR instruction

Implications

The PAR register will incorrectly report the translation as if the stage 1 MMU was enabled.
Note that the combination of both HCR.TGE and SCTLR.M bits being set was UNPREDICTABLE in earlier versions of the architecture, and was only given a defined behavior to reduce the UNPREDICTABLE space. It is not expected to be a useful combination for software.

Workaround

Secure software can clear the SCTLR(ns).M bit before executing the address translation instruction, if the HCR.TGE bit is set. It should restore the previous SCTLR(ns).M value before returning to a lower Exception level.

798161
ETM trace reports incorrect branch target

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

When executing a branch that targets an out of range virtual address, the ETM might trace the target address incorrectly.

Configurations Affected

This erratum affects all configurations of the processor.

Conditions

1. ETM trace is enabled and not prohibited.
2. The core is executing code located in the virtual address range 0hFFFF000000000000-0hFFFFFFFFFFFFFFFF, and bit[48] of the PC is set. If the relevant TCR.TBI bit is set, then bits[63:56] of the virtual address can be any value.
3. The core executes a branch instruction with the target virtual address in the range 0h0002000000000000-0hFFFEFFFFFFFFFFFF, with bit[48] of the target address clear. This could be any kind of instruction that alters the program flow, including immediate branches, register based branches, and returns.

Implications

The target of the branch will take a translation fault, because it is not in a valid address range, but because the ETM trace decompression will report the branch target incorrectly the reason for the fault might be harder to determine when debugging. Bits[47:0] of the branch target will still be correct, but bits[63:48] will be incorrectly reported as all zero.

Workaround

There is no workaround for this erratum.

797529
Incorrect fault status codes used for reporting synchronous uncorrectable ECC aborts in the IFSR

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

When reporting a synchronous uncorrectable ECC error, an incorrect encoding is written to the fault status code field of the Instruction Fault Status Register.

Configurations Affected

This erratum affects configurations of the processor that have CPU_CACHE_PROTECTION or SCU_CACHE_PROTECTION set to TRUE.

Conditions

1. A core is in the AArch32 execution state.
2. The core is executing in EL0, EL1, or EL3.
3. The core is using the Short-descriptor page format.
4. An uncorrectable ECC error occurs in one of the following caches:
    - The L1 data cache of the core.
    - The L1 data cache of another core in the processor.
    - The L2 cache.
5. An instruction fetch or a translation table walk for an instruction fetch hits the cache line containing the ECC error, triggering a prefetch abort.

If these conditions are met, then IFSR.FS[4] is set to 0 instead of 1.

Implications

When the short descriptor page format is in use, software cannot make a distinction between a Domain fault, level 1, or a synchronous parity or ECC error on memory access. Both will be reported in the IFSR as 01001.

Similarly, software cannot make a distinction between synchronous external abort on translation table walk and synchronous parity or ECC error on translation table walk. Both will be reported in the IFSR as synchronous external abort on translation table walk (IFSR.FS[4:0] will be 01100 for translation level 1 and 01110 for translation level 2). This is valid behavior because it is IMPLEMENTATION DEFINED whether ECC errors are reported using the assigned fault code or using another appropriate encoding. However, the behavior is inconsistent with fault code usage in other fault status registers, and in the IFSR when using the Long-descriptor page format.

Workaround

When reading the fault status code from the IFSR, software will have to be aware that the value reported might not be accurate. If software reads 01001 for IFSR.FS[4:0], then software will have to determine if a Domain fault could have occurred for this address.

776025
Instruction cache parity error might cause an incorrect abort to be taken

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The core cache protection option allows the processor to detect and correct a 1-bit error in the instruction cache RAMs. If an error is detected on an instruction at the end of a page, and the following page would generate a fault if executed, then under some conditions it is possible for a fault to be taken when no instruction from the faulting page was architecturally executed.

Configurations affected

This erratum affects configurations of the processor with CPU_CACHE_PROTECTION enabled.

Conditions

1. The core is executing in AArch32 T32 instruction state.
2. The program counter points to a 16-bit length instruction which is at the end of a page.
3. The related instruction is stored in the L1 instruction cache.
4. There is a single bit error on one specific bit of the instruction, which causes the instruction to be incorrectly interpreted as a 32-bit instruction.
5. The page which is immediately after the address range of the previous 16-bit instruction would cause a translation fault, access flag fault, or permission fault if it was accessed.

Implications

If the above conditions are met, then the translation, permission, or access flag fault is incorrectly taken, instead of correcting the error. There is still substantial benefit being gained from the parity logic. There might be a negligible increase in overall system failure rate due to this erratum.

Workaround

No workaround is required.

755766
ETM might not generate an event packet and ATB trigger

Status

Affects: Cortex-A35
Fault Type: Programmer Cat-C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The ETM contains four resources to generate events based on core activity. When the ETM generates an event, it is reported to the CTI on the external outputs bus, an ATB trigger is generated, and an event packet is inserted into the trace stream.

Because of this errata, when the ETM is becoming idle, there is a one cycle window where an event might get indicated to the CTI, but would not generate an ATB trigger and would not generate an event packet.

Configurations Affected

To be affected by this erratum, the processor must be configured with ETMs.

Conditions

1. The ETM is becoming idle due to any of the following
   - The ETM has been disabled due to TRCPRGCTLR.EN = 0 or TRCOSLAR.OSLK = 1
   - The core has started to execute a WFI or WFE and is going to enter sleep
   - **DBGEN**/**NIDEN** have changed and trace is no longer permitted
2. The resources are configured to generate events.
3. An event is generated because of a PMU event or CTI trigger on the last cycle in which the ETM could generate an event.

Implications

If the stimulus that caused the event to be generated occurred one cycle later, then the event would not have been generated at all. Therefore, this errata will only be noticed when trying to correlate the CTI behaviour with the trace stream.

Workaround

There is no workaround for this erratum.

755748
ETM might assert AFREADY before all trace has been output

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

When the **AFVALID** signal on the ATB interface is asserted, the ETM should immediately start outputting all buffered trace. It should assert the **AFREADY** output one cycle after all trace that was buffered on the cycle in which **AFVALID** was first asserted has been output.

Because of this erratum, the **AFREADY** signal might be asserted before all the necessary trace has been output.

Configurations Affected

To be affected by this erratum, the processor must be configured with ETMs.

Conditions

1. The ETM must contain buffered trace.
2. **ATVALID** must be LOW on the first cycle **AFVALID** is HIGH.

Implications

This might result in the ETM containing trace that was generated before the flush request when the rest of the system expects this trace to have been output.

Workaround

The system can ensure that all trace has been drained from the ETM by disabling it. The ETM can be disabled by setting TRCPRGCTLR.EN to 0. The system should then poll the TRCSTATR.IDLE bit. When it reads as 1, the ETM is idle and all trace that was generated before the write to TRCPRGCTLR has been output.

707223
PMU counter values might be inaccurate when monitoring EXC_RETURN

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The Cortex-A35 processor implements a Performance Monitor Unit (PMU). The PMU allows programmers to gather statistics on the operation of the processor during runtime. Because of this erratum, a PMU counter value might be inaccurate when monitoring EXC_RETURN events.

Configurations Affected

All configurations are affected.

Conditions

1. A performance counter is enabled and configured to count EXC_RETURN events.
2. The Secure EL3 filtering bit, PMEVTYPER<n>_EL0.M, is set to 1.
3. The core returns from an exception in EL3 to a lower Exception level.
4. EL3 uses AArch64 and the lower Exception level uses AArch32.

Implications

If the erratum conditions are met, the exception return from EL3 event will be filtered using the EL1 modes filtering bit, PMEVTYPER<n>(_EL0).P, instead of a combination of the EL1 modes filtering bit and the Secure EL3 filtering bit. The expected behavior and actual behavior are shown below.

P	M	Expected Behavior	Actual behavior
0	0	Count event in EL1 and EL3	As expected
0	1	Count event in EL1 but not EL3	Count event in EL1 and EL3
1	0	Do not count event in EL1 or EL3	As expected
1	1	Do not count event in EL1 but do count in EL3	Do not count event in EL1 or EL3

If the exception return from EL3 is to a lower Exception level that uses AArch64 then the event will be counted, or not counted, correctly.

This means:

- Exception returns from EL3 are not consistently counted or excluded.
  - If counting exception returns from EL3 is intended, the PMU counter value might be lower than expected.
  - If counting exception returns from EL3 is not intended, the PMU counter value might be higher than expected.
- Exception returns from EL1 and EL3 cannot be counted separately.

Workaround

Exception returns from EL3 can be consistently counted or excluded by setting the Secure EL3 filtering bit, PMEVTYPER<n>_EL0.M, to 0.

There is no workaround to allow exception returns from EL1 and EL3 to be counted separately.

705778
Accessing EDPCSR has side-effects when OS Lock is locked

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in: r0p0, r0p1. Fixed in r0p2.

Description

Access to the External Debug Program Counter Sample Register, EDPCSR, should not have side-effects when the OS lock is locked or when the OS Double Lock is locked. Because of this erratum, the side-effects are only disabled when OS Double Lock is locked.

Configurations Affected

All configurations are affected.

Conditions

1. The OS Lock is locked (EDPRSR.OSLK == 1).
2. The OS Double Lock is unlocked (EDPRSR.DLK == 0).
3. An external debugger reads EDPCSR[31:0].

Implications

If the erratum conditions are met then EDCIDSR, EDVIDSR, and EDPCSR[63:32] will be updated as if the OS Lock were unlocked.

Workaround

There is no workaround.

702094
WFx can cause the PC to jump from lower VA subrange to upper VA subrange

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

If a core executes a WFI or WFE instruction at VA 0x0000_ffff_ffff_fffc in the AArch64 execution state then, when the core wakes up, it should resume execution from VA 0x0001_0000_0000_0000. This address is outside of the implemented virtual address space and will trigger a Translation fault if accessed.

Because of this erratum, the core will resume execution from VA 0xffff_0000_0000_0000 instead of 0x0001_0000_0000_0000. Address 0xffff_0000_0000_0000 is a valid address in the upper VA subrange for EL0/EL1 translations if TCR_EL1.T1SZ is set to 16.

Configurations Affected

All configurations.

Conditions

1. A core is in the AArch64 execution state.
2. The core executes a WFI or WFE instruction at VA 0x0000_ffff_ffff_fffc.

Implications

Real code is not expected to include a WFI or WFE instruction at VA 0x0000_ffff_ffff_fffc because, if this erratum did not exist, wake-up from the WFx would always trigger an access to a VA that is outside of the implemented virtual address space.

Workaround

If a workaround is required, code could be inserted at VA 0xffff_0000_0000_0000 to catch unexpected accesses to this address.

689935
ETM might output an incorrect exception return address

Status

Affects: Cortex-A35
Fault Type: Programmer Cat-C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The processor supports instruction trace using a per-core ETM. If tracing is enabled, a core's ETM generates an Exception element when the core takes an exception. The Exception element contains an indication that an exception has occurred, the type of exception, and the exception return address. Because of this erratum, the exception return address that is output by the ETM might have bits [63:49] set to 1 when they should be set to 0.

Configurations Affected

This erratum affects configurations of the processor with a per-core ETM.

Conditions

1. Trace is enabled.
2. Non-invasive debug is permitted.
3. A core is in the AArch64 execution state.
4. The core executes one of the following instructions at VA 0x0000_ffff_ffff_fffc
    - SVC
    - HVC
    - SMC

Implications

The exception return address reported by the ETM might be incorrect. For example, an SMC instruction at VA 0x0000_ffff_ffff_fffc might cause the ETM to output an exception return address of 0xffff_0000_0000_0000 instead of 0x0001_0000_0000_0000.

Real code is not expected to meet the conditions in this erratum because, if this erratum did not exist, the core would access a VA that is outside of the implemented virtual address space.

Workaround

There is no workaround for this erratum.

676796
ROM table entries for cores 1, 2, and 3 might be ignored

Status

Affects: Cortex-A35
Fault Type: Programmer Category C
Fault Status: Present in r0p0, r0p1. Fixed in r0p2.

Description

The processor includes a ROM table that allows external debuggers to determine which debug components are implemented inside the processor. The end of the ROM table is specified by an end-marker that reads 0x00000000.

Because of this erratum, the ROM table might include an end-marker before the end of the ROM table.

Configurations Affected

To be affected by this erratum, all the following must be true:

- The processor is configured with more than one core.
- The processor is configured with the v8 debug map (not the legacy v7 debug map).
- The processor is configured without ETMs.

Conditions

1. An external debugger reads the ROM entry register for the core 0 ETM component.
2. The external debugger interprets the read value of 0x00000000 as the ROM table end-marker for all cores in the processor.

Implications

The external debugger might report core 1, core 2, and core 3 debug components as not present even if the cores are present. This might prevent the user from accessing those components with the debugger.

Workaround

The external debugger must read the ROM entry registers for a core even if it reads an end-marker in the ROM entry register for the previous core's ETM component.

